Folien zur Vorlesung ”Mathematische Grundlagen der Computerlinguistik”

Susanne Schacht
Arbeitsgruppe Computerlinguistik, Universität Freiburg
Friedrichstr. 50, D-79098 Freiburg, Germany
e-mail: schacht@coling.uni-freiburg.de

WS 1997/98

Die Vorlesung spricht Themen sowohl der mathematischen Grundlagen der Linguistik als auch der formalen Grundlagen der maschinellen Sprachverarbeitung an. Die Veranstaltung besteht aus den vier Blöcken Algebra, Logik, formale Sprachen und Komplexität, sowie Statistik. Die Betonung liegt dabei jeweils auf möglichen Anwendungen für die Computerlinguistik.

1 Inhalt

1. Mengentheorie und Algebra
   (a) Einführung;Mengen und Relationen
   (b) Funktionen
   (c) Algebraische Strukturen: Gruppen, Partiell geordnete Mengen, Verbände
2. Logik, formale Systeme
   (a) Aussagenlogik
   (b) Prädikatenlogik
   (c) Formale Systeme, Inferenz
   (d) Nicht-Standard-Logiken
   (e) Merkmalslogik, Unifikation
   (f) Typtheorie, Lambda-Kalkül, Montague-Semantik
3. Theoretische Informatik
   (a) Formale Grammatiken, Chomsky-Hierarchie
   (b) Sprachen und Automaten
   (c) Berechenbarkeit und Komplexität
4. Statistik
   (a) Grundbegriffe

(b) Wahrscheinlichkeitstheorie
(c) Hidden Markov Modelle
(d) Anwendung: Tagging
(e) Anwendung: Parsing
(f) Bits and Pieces

2 Literatur

2.1 Lehrbuch/Script

1. Barbara H. Partee, Alice ter Meulen & Robert Wall: Mathematical Methods in Linguistics, 1990 (Partee et al., 1990).

   Das Buch deckt ca. 3/4 der Vorlesung ab. Es setzt keine mathematischen Grundkenntnisse voraus und betont die Anwendungen der vorgestellten Formalismen.

2. Brigitte Krenn und Christer Samuelsson: The Linguist's Guide to Statistics (Krenn & Samuelsson, 1997).

   http://www.coli.uni-sb.de/ christer/stat_cl.ps

   Sehr gründliche Einführung in Wahrscheinlichkeitstheorie und alle möglichen Anwendungen von Statistik in der Linguistik. Leider noch Lücken an einigen Stellen, ausführliche Literaturliste.

3. Eugene Charniak: Statistical Language Learning, 1993 (Charniak, 1993)

   Nur bedingt empfehlenswert, mathematisch eher dünn, als Anwendungen nur Tagging, Parsing und Grammatik-Inferenz.

2.2 zusätzliche Literatur zu den einzelnen Themen

1. Algebra

2. Logik

   Zur Einführung in Aussagen- und Prädikatenlogik: (Schöning, 1987)

   Merkmalslogik, Unifikation: (Shieber, 1986)

   Anwendung Semantik: (Lohnstein, 1997), Montague-Semantik gründlich:

3. formale Sprachen, Berechenbarkeit Komplexität

   Einführung in die Theoretische Informatik: (U. Schöning, 1997)

   Komplexität und Natürliche Sprache: (Barton et al., 1987), Komplexität einzelner Grammatikformalismen und Parsingalgorithmen: (Barton, 1985), (Johnson, 1989), (Neuhaus & Bröker, 1997)

4. Wahrscheinlichkeitsrechnung und Statistik

   Zur Motivation von statistischen Methoden: (Abney, 1996a)

   Stochastische Grammatiken: TAGs: (Schabes, 1992) HPSG: (Brew, 1995) Merkmalsgrammatiken: (Abney, 1996b)

   Sparse Data: (Samuelsson, 1996) Semantik: (Bonnema et al., 1997)

Literatur

Abney, Steven (1996a). Statistical methods and linguistics. In Judith Klavans & Philip Resnik (Eds.), *The Balancing Act*. MIT Press, Cambridge, MA.

Abney, Steven (1996b). *Stochastic Attribute-Value Grammars*. unpublished.

Barton, Edward G., Robert C. Berwick & Eric Sven Ristad (1987). *Computational Complexity and Natural Language*. Cambridge, MA, London: Bradford, MIT Press.

Barton, G. Edward (1985). On the complexity of id/lp parsing. *Computational Linguistics*, 11(4):205–218.

Bonnema, Remko, Rens Bod & Remko Scha (1997). A DOP model for semantic interpretation. In *ACL/EACL 1997 Proceedings*, pp. 159 – 167. Madrid, Spain: ACL. http://earth.let.uva.nl/ bonnema/acl-DOP-sem/acl-DOP-sem.html.

Brew, Chris (1995). Stochastic hpsg. In *ACL'95 – 33rd Annual Meeting of the Association for Computational Linguistics*.

Charniak, Eugene (1993). *Statistical Language Learning*. Cambridge, MA: MIT Press.

Johnson, M. (1989). The computational complexity of Tomita's algorithm. In *Proceedings of the 1st Int'l. Workshop on Parsing Technologies (IWPT'89). Carnegie-Mellon University, Pittsburgh, PA, 1989, pp.203-208*.

Krenn, Brigitte & Christer Samuelsson (1997). *The Linguist's Guide to Statistics*. http://www.coli.uni-sb.de/christer/stat_cl.ps.

Lohnstein, Horts (1997). *Formale Semantik und Natürliche Sprache*. Westdeutscher Verlag.

Neuhaus, P. & N. Bröker (1997). The complexity of recognition of linguistically adequate dependency grammars. In *ACL-97/EACL-97 – Proc. of the 35rd Annual Meeting of the Association for Computational Linguistics and the 8th Conf. of the European Chapter of the Association for Computational Linguistics*.

Partee, B. H., A. ter Meulen & R. E. Wall (1990). *Mathematical Methods in Linguistics*, Vol. 30. Studies in Linguistics and Philosophy. Dordrecht, NL: Kluwer Academic Press.

Samuelsson, Christer (1996). Handling sparse data by successive abstraction. In *Proceedings of the 16th International Conference on Computational Linguistics, ICCL 1996*.

Schabes, Yves (1992). Stochastic lexicalized tree-adjoining grammars. In *COLING '92 - Proceedings 15th Int'l. Conference on Computational Linguistics*, pp. 426–432.

Schöning, Uwe (1987). *Logik für Informatiker*. Mannheim: BI Wissenschaftsverlag.

Shieber, Stuart M. (1986). *An introduction to unification-based approaches to grammar*. Lecture Notes 4. Stanford, CA: CLSI.

U. Schöning (1997). *Theoretische Informatik kurz gefasst* (3 ed.). Heidelberg: Spektrum Akademischer Verlag.

①

Warum?

was?	Linguistik	Computerlinguistik
Algebra u. Logik	exakte formale Beschreibungen von: Syntax u. Semantik	 zusätzlich Weltwissen
Theoretische Informatik		Aufwandsabschätzungen, Berechenbarkeit
Statistik	quantitative Linguistik	“Performanz”: kogn. Modellierung techn. P.-Steigerung

Mengen

Eine *Menge* ist eine (ungeordnete) Sammlung von unterscheidbaren Objekten, ihren *Elementen*.
– das können auch Mengen sein!

Eine Menge ist wohl-definiert, wenn nach einem klaren Prinzip entschieden werden kann, ob ein Objekt Element einer Menge ist oder nicht.

Schreibweise:

Grossbuchstaben $A, B, C, ...$ für Mengen,

Kleinbuchstaben $a, b, c, ...x, y, ...$ für Elemente.

leere Menge: $\emptyset$

x ist / ist nicht Element von M :

$x \in M$ bzw. $x \notin M$

③

Beschreibungen für Mengen

1. Listen-Notation, Aufzählung:
   $\{1, 2, 3\}$
2. Charakteristische Eigenschaft, Beschreibung:
   $\{x \mid x$ gerade$\} \rightarrow$ Russells Paradox...
3. Verfahren zur Generierung der Elemente:
   1. 4 ist Element der Menge M.
   2. wenn x Element der Menge M ist,
   dann auch x + 3.
   3. nichts sonst ist Element der Menge M.

Identität:

$M = N$ genau dann, wenn alle x in M auch Element von N sind und alle y in N auch in M.

Kardinalität:

Anzahl der Elemente einer Menge:

$\mid A \mid$ oder # (A)

Wenn $\mid A \mid$ eine natürliche Zahl ist, heisst A *endlich*, sonst *unendlich*. Beispiel?

④

Mengenoperationen

Teilmenge: $A \subseteq B$ genau dann, wenn alle Elemente von A auch Elemente von B sind.

Für alle Mengen A gilt:

$$A \subseteq A \text{ und } \emptyset \subseteq A.$$

echte Teilmenge:

$$A \subset B \text{ gdw. } A \subseteq B \text{ und } \mid A \mid \; < \; \mid B \mid .$$

disjunkte Mengen: Zwei (mehrere) Mengen heissen *disjunkt*, wenn kein x Element von A <u>und</u> B ist.

Potenzmenge: 2^M, Menge aller Teilmengen von M.

Vereinigung: $A \cup B$, alle Elemente von A <u>oder</u> B.

Schnitt: $A \cap B$, Menge aller Elemente von A <u>und</u> B.

Differenz: $A - B, A \setminus B ::= \{x \mid x \in A \text{ and } x \notin B\}$

Komplement: $A' ::= \{x \mid x \notin A\}$
immer in Bezug auf Grundmenge / Universum U:

$$A' = U \setminus A$$

Verknüpfungseigenschaften

- Idempotenz
  (a) $X \cup X = X$ (b) $X \cap X = X$
- Kommutativität
  (a) $X \cup Y = Y \cup X$ (b) $X \cap Y = Y \cap X$
- Assoziativität
  (a) $(X \cup Y) \cup Z = X \cup (Y \cup Z)$
  (b) $(X \cap Y) \cap Z = X \cap (Y \cap Z)$
- Distributivität
  (a) $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$
  (b) $X \cap (Y \cup Z) = (X \cap Y) \cup (X \cap Z)$
- Identitätsgesetze
  (a) $X \cup \emptyset = X$ (b) $X \cap \emptyset = \emptyset$
  (c) $X \cup U = U$ (d) $X \cap U = X$
- Komplementgesetze
  (a) $X \cup X' = U$ (b) $X \cap X' = \emptyset$
  (c) $(X')' = X$ (d) $X \backslash Y = X \cap Y'$
- DeMorgans Gesetze
  (a) $(X \cup Y)' = X' \cap Y'$ (b) $(X \cap Y)' = X' \cup Y'$
- Konsistenz
  (a) $X \subseteq Y$ gdw. $X \cup Y = Y$
  (b) $X \subseteq Y$ gdw. $X \cap Y = X$

Tupel

geordnetes Paar, mengentheoretische Definition:

$$\langle a, b\rangle ::= \{\{a\}, \{a, b\}\}$$

mit der Eigenschaft $\langle a, b\rangle \neq \langle b, a\rangle$.

Kartesisches Produkt:

$$A \times B ::= \{\langle a, b\rangle \mid a \in A \text{ und } b \in B\}$$

Projektion auf erste / zweite Koordinate von $C \subseteq A \times B$:

$$M = \{a \mid \langle a, b\rangle \in C\} \text{ bzw. } N = \{b \mid \langle a, b\rangle \in C\}$$

Erweiterung auf *n-Tupel* möglich...

Relationen

(zweistellige) Relationen: Beziehungen zwischen (zwei) Objekten.

Relation $R \subseteq A \times B$ <u>von A nach B</u>;

falls $A = B$, <u>in A</u>.

Definitionsbereich, domain:

$\{a \mid a \in A$ und es gibt ein $b \in B$ so, dass $\langle a, b\rangle\ inR\}$

Wertebereich, range:

$\{b \mid b \in B$ und es gibt ein $a \in A$ so, dass $\langle a, b\rangle\ inR\}$

Schreibweise: aRb oder Rab.

$$R = \{\langle a, b\rangle \mid aRb\}$$

Komplement von $R \subseteq M \times N$:

$$R' ::= (M \times N) \setminus R$$

Inverse:

$$R^{-1} ::= \{\langle b, a\rangle \mid aRb\}$$

identische Abbildung in A:

$$id_A ::= \{\langle a, a\rangle \mid a \in A\}$$

Eigenschaften von Relationen $R \subseteq A \times A$

- *Reflexivität*:
  für alle $x \in A : \langle x, x \rangle \in R$, d.h. $id_A \subseteq R$.
  - *nicht reflexiv*: nicht alle $\langle x, x \rangle \in R$, $id_A \not\subseteq$ von R.
  - *irreflexiv*: kein $\langle x, x \rangle \in R$, $id_A \cap R = \emptyset$.

- *Symmetrie*:
  für alle $\langle x, y \rangle \in R : \langle y, x \rangle \in R$.
  - *nicht symmetrisch*: nicht für alle $\langle x, y \rangle \in R$ auch $\langle y, x \rangle \in R$.
  - *asymmetrisch*: nie sowohl $\langle x, y \rangle \in R$ als auch $\langle y, x \rangle \in R$ (also auch irreflexiv).
  - *antisymmetrisch*: wenn $\langle x, y \rangle \in R$ und $\langle y, x \rangle \in R$, dann $x = y$.

- *Transitivität*:
  für alle $\langle x, y \rangle, \langle y, z \rangle \in R : \langle x, z \rangle \in R$.
  - *nicht transitiv*: nicht für alle ...
  - *intransitiv*: für keine ...

Beispiel

Relation K in der Menge W der Artikel (Relativpronomen) und Substantive:

$\langle w_1, w_2 \rangle \in K$ gdw. eines der beiden ist Artikel, das andere Substantiv und w_1 und w_2 kongruieren in Kasus, Numerus und Genus

- irreflexiv
- symmetrisch
- nicht transitiv

Überprüfung durch Relationstafel:

zur Reflexivität betrachte die Diagonale...

zur Symmetrie: sind die Einträge an der Diagonale gespiegelt?

zur Transitivität: nachrechnen...

Äquivalenzrelationen

Relationen, die

- reflexiv
- symmetrisch
- transitiv

sind, heissen *Äquivalenzrelationen*. Sie *zerlegen* ihren Definitionsbereich in disjunkte Teilmengen, die *Äquivalenzklassen*.

Schreibweise:

Äquivalenzklasse von x:

$$[\![x]\!] ::= \{y \mid \langle x, y\rangle \in R\}$$

x ist *äquivalent* zu y : $x \sim y$ (auch: *kongruent*)

Beispiel: Relation R in der Menge N der natürlichen Zahlen (mit 0):

$\langle n, m\rangle \in R$ gdw. m und n hinterlassen bei der Division durch 4 den gleichen Rest.

$$R = \{\langle 0,0\rangle, \langle 0,4\rangle, ..., \langle 1,1\rangle, \langle 1,5\rangle, ..., \langle 4,0\rangle, ...\}$$

Ordnungsrelationen

Transitive Relationen heissen *Ordnungsrelationen*.

- *partielle Ordnung,* $\leq$:
  - transitiv, d.h. wenn $x \leq y \wedge y \leq z$, dann $x \leq z$.
  - reflexiv, d.h. für alle $x : x \leq x$.
  - antisymmetrisch, d.h. wenn $x \leq y \wedge y \leq x$, dann $x = y$.

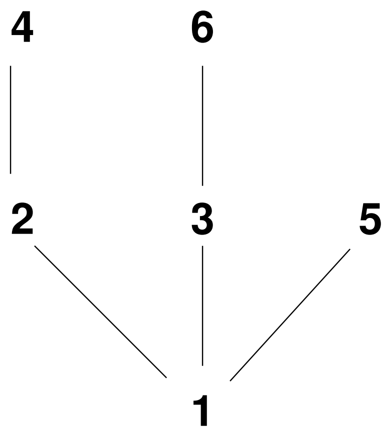
- *totale Ordnung*:
  alle x, y vergleichbar: $x \leq y$ oder $y \leq x$.

- *strikte Ordnung,* $<$
  - transitiv
  - irreflexiv
  - asymmetrisch

Beispiele

Partielle Ordnungsrelationen lassen sich in sog. Hasse-Diagrammen darstellen:

$$R = \{\langle a, b\rangle \mid a, b \in \{1, 2, 3, 4, 5, 6\} \text{ und } a \text{ teilt } b\}$$



Totale Ordnungen bilden eine Kette, betrachte Hasse-Diagramm von

$$R = \{\langle a, b\rangle \mid a, b \in \{1, 2, 4, 8\} \text{ und } a \text{ teilt } b\}$$

Sei eine Ordnungsrelation R in A gegeben.

$x \in A$ heisst *untere Grenze*, wenn es kein $y \in A$ gibt mit $\langle y, x\rangle \in R$.

x ist *kleinstes* Element von A, wenn für alle $y \in A$ gilt: $\langle x, y\rangle \in R$.

obere Grenze und *grösstes* Element entsprechend...

wohlgeordnet: jede Teilmenge hat kleinstes Element...

Übung: Teilmengen

A, B seien Mengen.

Zu zeigen ist: wenn $A \subseteq B$ dann auch $2^A \subseteq 2^B$

Beweis:

Für alle Elemente m einer Menge M, die Teilmenge einer Menge N ist, gilt, dass m auch Element von N ist. (Definition der Teilmenge)

$$\text{für alle } a \in A : a \in B$$

Dies gilt auch für alle Elemente in Teilmengen von A, somit sind alle Teilmengen von A auch Teilmengen von B,

$$\text{für alle } C \subseteq A : C \subseteq B$$

und damit Element von 2^B. (Definition der Potenzmenge)

$$\text{für alle } C \subseteq 2^B$$

Damit sind alle Elemente von 2^A auch Element von 2^B, also gilt:

$$2^A \subseteq 2^B \blacksquare$$

Gilt auch die Umkehrung?

Funktionen (= Abbildungen)

Eine Relation F von M nach N heisst *partielle Funktion* gdw.

wenn $\langle x, y \rangle \in F$ und $\langle x, z \rangle \in F$, dann $y = z$.

D.h., ein $x \in M$ wird auf höchstens ein $y \in N$ abgebildet.

Schreibweise: $F : A \to B$; $F(a) = b$

$a \in A$ ***Argumente***, $b \in B$ ***Werte***.

A heisst auch *Urbild*,

$\{b \mid b \in B$ und es gibt a mit $F(a) = b\}$ *Bild* von F,

a Urbild von $F(a)$.

oft Kleinbuchstaben $f, g, h, ...$ für Funktionen

(*totale*) Funktion von M nach N: jedes $x \in M$ ...

Beispiel: Sei L eine Menge von lexikalischen Schlüsseln, E die Menge der Einträge.

Ist $f : L \mapsto E$ mit $f(l) =$ Eintrag zu l eine Funktion?

Eigenschaften von Funktionen

surjektiv:
$\{F(a) \mid a \in A\} = B$
“jedem Element von B ist ein Urbild zugeordnet”

injektiv: Eins-zu-eins-Abbildung, d.h.:
wenn $F(a) = F(b)$, dann $a = b$.
“verschiedene Argumente haben verschiedene Funktionswerte.”

bijektiv: injektiv und surjektiv
“jedem Element von B ist genau ein Urbild zugeordnet”

Übungen zu Funktionseigenschaften

Sei $f : A \to B$. Sind die folgenden Aussagen wahr oder falsch?

1. f ist genau dann bijektiv, wenn f umkehrbar ist.
2. Ist f surjektiv, so hat jedes $a \in A$ mindestens ein Bild $f(a) \in B$.
3. Hat jedes $a \in A$ mindestens ein Bild $f(a) \in B$, so ist f surjektiv.
4. Gibt es zu jedem $a \in A$ höchstens ein $b \in B$ mit $f(a) = b$, so ist f injektiv.
5. Gibt es zu jedem $b \in B$ ein $a \in A$ mit $f(a) = b$, so ist f injektiv.
6. f ist genau dann bijektiv, wenn zu jedem $a \in A$ genau ein $b \in B$ existiert mit $f(a) = b$.
7. f ist genau dann bijektiv, wenn zu jedem $b \in B$ genau ein $a \in A$ existiert mit $f(a) = b$.
8. Ist f bijektiv, so gibt es zu jedem $b \in B$ höchstens ein $a \in A$ mit $f(a) = b$.

1, 2, 5, 7, 8 sind wahr, die anderen falsch.

1. Ist f bijektiv, so gibt es zu jedem $a \in A$ höchstens ein $b \in B$ mit $f(a) = b$.
2. Folgt aus $f(a_1) = f(a_2)$ für alle $a_1, a_2 \in A$, dass $a_1 = a_2$, dann ist f injektiv.
3. Sind A und B endlich und ist f injektiv, dann ist f auch bijektiv.
4. $f : A \to f(A)$ ist stets surjektiv.
5. Gilt $B \subset A$, so kann f nicht bijektiv sein.
6. Ist f surjektiv und die Umkehrrelation von f eine injektive Abbildung, so ist f auch injektiv.
7. Ist f injektiv und die Umkehrrelation von f eine surjektive Abbildung, so ist f bijektiv.
8. Ist f surjektiv und nicht injektiv, so ist die Umkehrrelation eine surjektive, nicht injektive Funktion.
9. Ist $A = B$, so ist jede Injektion auch eine Bijektion
10. Jede Injektion lässt sich durch Einschränkung des Wertebereichs zu einer Bijektion machen.

1, 2, 4, 5, 6, 7, 10 sind wahr, die anderen falsch.

Verknüpfung von Funktionen

Komposition, Verknüpfung

Gegeben: $F : A \to B, G : B \to C,$

dann: $G \circ F : A \to C$ mit $(G \circ F)(a) = G(F(a))$

(Achtung: Schreibrichtung...)

Sei $F : A \to B$ gegeben. Dann gilt:

$$id_B \circ F = F = F \circ id_A$$

und für bijektive F:

$$F^{-1} \circ F = id_A \text{ und } F \circ F^{-1} = id_B$$

mehrstellige Funktionen: mit Hilfe von Tupeln...

Beispiel Komposition

Gegeben: Mengen von Sätzen in natürlichen Sprachen: E englisch, D deutsch, S spanisch.

Ausserdem Funktionen:

$f_1 : E \to D$,

$f_2 : S \to E$,

$f_3 : D \to S$,

Wie übersetze ich einen deutschen Satz s in einen englischen?

zuerst deutsch nach spanisch, dann spanisch nach englisch:

$$f_2(f_3(s)) = (f_2 \circ f_3)(s)$$

Wie viele einzelne Übersetzungsfunktionen braucht man, um Übersetzer für n Sprachen zu konstruieren?

Wenn nicht mehr als zwei Schritte vorgenommen werden sollen?

Mengen mit unendlicher Kardinalität

Zwei (endliche) Mengen heissen äquivalent, wenn sie die gleiche *Anzahl* von Elementen haben : $A \sim B$ gdw. $|A| = |B|$

Zwei Mengen sind äquivalent, wenn zwischen ihnen eine bijektive Abbildung existiert.

Eine Menge ist unendlich, wenn sie zu einer ihrer echten Teilmengen äquivalent ist.

Beispiel: Abbildung von **N** $= \{0, 1, 2, 3, ...\}$ auf $\{0, 2, 4, 6, ...\}$? Auf die Menge der ganzen Zahlen?

Mengen, die äquivalent sind zur Menge der Natürlichen Zahlen **N**, heissen aufzählbar.

Kardinalität dieser Mengen: $\aleph_0$ (Aleph)

Noch "grössere" Mengen?

Prinzip der vollständigen Induktion

Ziel: eine allgemeine Aussage über Elemente aufzählbarer Mengen zu beweisen.

Vorgehensweise:

1. A(1), Induktionsanfang: Beweis für das erste Element.
2. A(k) $\rightarrow$ A(k+1), Induktionsschritt:
   (a) Annahme: Aussage gelte für die ersten k Elemente.
   (b) Zeige, dass daraus Aussage über $k+1$. Element folgt.
3. Induktionsschluss: Daraus folgt, dass die Aussage für alle Elemente gilt!

Beispiel

Behauptung:

Die Anzahl der Elemente der Potenzmenge einer endlichen Menge A ist gleich 2 hoch der Anzahl der Elemente von A

Beweis durch Induktion über die Anzahl der Elemente von A:

1. Induktionsanfang: A habe genau ein Element, e_1. Dann ist $2^A = \{\emptyset, \{e_1\}\}$, und hat zwei Elemente, $2 = 2^1$
2. Induktionsschritt:
   (a) Annahme: Sei $A = \{e_1, ..., e_k\}$, $\mid 2^A \mid = 2^{\mid A \mid}$
   (b) Sei $A' ::= A \cup \{e_{k+1}\}$.
   Dann gilt: $2^{A'} = 2^A \cup \{A'\} \cup ... \cup \{e_1, e_{k+1}\} \cup \{e_{k+1}\}$,
   d.h. alle Teilmengen von A erweitert um $\{e_{k+1}\}$,
   d.h. genau $2^{\mid A \mid}$ mehr Mengen.
   Daraus folgt: $\mid 2^{A'} \mid = 2^{\mid A \mid} + 2^{\mid A \mid} = 2^{\mid A \mid + 1}$
3. Induktionsschluss:
   Daraus folgt, dass $\mid 2^A \mid = 2^{\mid A \mid}$ für alle <u>endlichen</u> Mengen gilt.

überaufzählbare Mengen

Theorem von Cantor:

die Potenzmenge einer Menge hat eine grössere Kardinalität als die Menge selbst.

$$|\, A \,| \; < \; |\, 2^A \,|$$

Beweis durch Widerspruch:

Annahme: es existiert bijektive $F : A \to 2^A$.

Sei $B = \{x \in A \,|\, x \notin F(x)\}$.

Es muss $y \in A$ mit $F(y) = B$ geben.

Ist y Element von B?

Daraus folgt: $|\, \mathbf{N} \,| \; < \; |\, 2^{\mathbf{N}} \,|$

Diagonalisierungsargument (Gödel):

Wenn $\mathbf{N} \sim 2^{\mathbf{N}}$, dann muss es eine Möglichkeit geben, $2^{\mathbf{N}}$ systematisch hinzuschreiben.

Sei diese Folge von Mengen $S_0, S_1, S_2, ...$

Sei $S* ::= \{n \,|\, n \notin S_n\}$.

Es gibt kein n mit $S* = S_n$, $S*$ kann nicht in der Liste stehen, also kann es diese Liste gar nicht geben.

Algebren

Eine *algebraische Struktur* oder *Algebra* **A** ist eine Menge A zusammen mit einer oder mehreren Operationen f_i ($+, \circ, \times$ für f_i):

$$\mathbf{A} = \langle A, f_1, f_2, ..f_n \rangle$$

Beschränkungen: die Stelligkeit jeder Operation muss endlich sein; jede Anwendung der Operationen ergibt genau ein Element aus A.

Gegeben: $\langle A, \circ \rangle$

Axiom 1 Abgeschlossenheit:

A ist unter $\circ$ abgeschlossen, d.h. für alle $a, b \in A$ existiert ein $c \in A$ mit $a \circ b = c$.

Axiom 2 Eindeutigkeit:

Wenn $a = a'$ und $b = b'$, dann $a \circ b = a' \circ b'$

$\langle B, \circ \rangle$ ist eine *Unterstruktur / Subalgebra* vom $\langle A, \circ \rangle$ gdw. $B \subset A$ und B abgeschlossen bez. $\circ$

Eigenschaften von Operationen

Gegeben: eine Operation $\circ$ in A.

Für alle $a, b, c \in A$:

- *Assoziativität*

  $$(a \circ b) \circ c = a \circ (b \circ c)$$

  assoziativ: Addition, Multiplikation; nicht-assoziativ: Subtraktion, Division

- *Kommutativität*

  $$a \circ b = b \circ a$$

  kommutativ: Multiplikation, ...; nicht kommutativ: Subtraktion, ...

- *Idempotenz*

  $$a \circ a = a$$

  idempotent: Schnittmengenbildung, ...

- *Distributivität* zusätzlich: zweite Operation $\diamond$

  $$a \circ (b \diamond c) = (a \circ b) \diamond (a \circ c)$$

  Multiplikation distribuiert über Addition, aber nicht umgekehrt

spezielle Elemente

- *links- bzw. rechtsneutrales Element, neutrales Element*

  $$e_l \circ a = a \text{ bzw. } a \circ e_r = a$$

  Komposition von Funktionen $F : M \to N$: identische Abbildung id_M, bzw. id_N; 0 ist rechtsneutral für Subtraktion ...

  Bei kommutativen Operationen gilt $e_l = e_r = e$, neutrales Element. (Kommutativität hinreichend, nicht notwendig)

  Wenn ein neutrales Element existiert, ist es eindeutig! Beweis...

- *inverse Elemente*

  Gegeben neutrales Element e.

  Für a existiert rechts-(links-)inverses El. $a_r(a_l)$, wenn

  $$a \circ a_r = e \text{ bzw. } a_l \circ a = e$$

  $a^{-1} = a_l = a_r$ heisst *inverses* Element von a.

  Wenn a invers zu b, dann auch b zu a!

  Beispiele?

Verknüpfungstafeln

Zeigen von Eigenschaften: “durchrechnen”
z.B. mit Verknüpfungstafeln:

Bsp: Addition modulo 4 in $\{0, 1, 2, 3\}$

$+_{mod4}$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

abgeschlossen, kommutativ, inverse Elemente, neutrales Element...

Morphismen

Abbildungen zwischen Algebren, $\varphi : \mathbf{A} \rightarrow \mathbf{B}$. Gleiche Anzahl der Operationen!

Sei $\mathbf{A} = \langle A, \circ \rangle$, $\mathbf{B} = \langle B, \diamond \rangle$

- *Morphismus*: Funktion φ mit
$$\varphi(a_1 \circ a_2) = \varphi(a_1) \diamond \varphi(a_2)$$
, φ heisst *verknüpfungstreu*.

- *Isomorphismus:*
verknüpfungstreue, bijektive Funktion φ
A und **B** heissen *isomorph*, wenn es einen Isomorphismus zwischen ihnen gibt.

- *Automorphismus:*
Isomorphismus von **A** auf sich selbst,
z.B. identische Abbildung

Gruppen

Eine Struktur **G** $= \langle G, \circ \rangle$ ist eine *Gruppe*, wenn gilt:

G1: **G** ist eine Algebra (d.h. $\circ$ ist vollständig definiert und G ist unter $\circ$ abgeschlossen)

G2: $\circ$ ist assoziativ

G3: G enthält ein neutrales Element

G4: Jedes Element in G hat ein inverses.

Bsp: positive rationale Zahlen und Multiplikation, Addition modulo 4 in $\{0, 1, 2, 3\}$, gerade Zahlen und Addition.

Abelsche Gruppe: zusätzlich kommutativ.

Gruppen II

Untergruppe: Subalgebra und selbst Gruppe

Ordnung einer Struktur: Anzahl der Elemente der Menge

• Die Ordnung jeder Untergruppe einer endlichen Gruppe **G** teilt die Ordnung von **G**.

Beispiel: Addition modulo 4 in $\{0, 1, 2, 3\}$,

$\mathbf{G}_{+_{mod4}} = \langle\{0, 1, 2, 3\}, +_{mod4}\rangle$

Untergruppen:

$\langle\{0, 1, 2, 3\}, +_{mod4}\rangle$

$\langle\{0, 2\}, +_{mod4}\rangle$

$\langle\{0\}, +_{mod4}\rangle$

• Alle Subalgebren von endlichen Gruppen sind ebenfalls Gruppen, d.h., nur Abgeschlossenheit muss gezeigt werden...

• Der Schnitt zweier Untergruppen ist wieder eine Untergruppe.

Halbgruppen: nur G1 + G2

Monoide: G1, G2 + G3

Abelsches Monoid: kommutatives Monoid

Integritätsringe

D $= \langle D, +, \cdot \rangle$ ist ein Integritätsring, wenn gilt:

$\langle D, + \rangle$ ist Abelsche Gruppe mit neutralem Element 0

$\langle D, \cdot \rangle$ ist Abelsches Monoid mit neutralem Element $1, 1 \neq 0$

(Distributivgesetz)

Für alle $a, b, c \in D, a \cdot (b + c) = (a \cdot b) + (a \cdot c)$.

(Kürzungsregel)

Wenn $c \neq 0$ und $c \cdot a = c \cdot b$, dann $a = b$.

Beispiel: Ganze Zahlen, Addition und Multiplikation

$\leq$ Ordnungsrelation für Integritätsring **D**, gdw.

für alle $a, b, c, d \in$ **D**:

1. Additionsgesetz:
   wenn $a \leq b$ und $c \leq d$, dann $a + c \leq b + d$.
2. Multiplikationsgesetz:
   wenn $a \leq b$ und $0 \leq d$, dann $a \cdot c \leq b \cdot c$.
3. Trichotomiegesetz:
   entweder $a < b$, $a = b$ oder $a > b$.

wohlgeordnet: wenn jede nicht-leere Teilmenge ein kleinstes Element enthält.

Posets

Partiell geordnete Menge (poset): $\langle A, \leq \rangle$

eine Menge A und eine reflexive, antisymmetrische und transitive Relation $\leq$ (partielle Ordnungsrelation) darin.

Kette: vollständig (total) geordnete Menge $a \leq b$ oder $b \leq a$ für alle a, b.

Sei $B \subseteq A$.

$x \in A$ heisst *untere Grenze* von B, wenn es kein $y \in B$ gibt mit $y \leq x$.

x ist *Infimum (greatest lower bound)* von B, inf$B$, wenn für alle $y \in B$ gilt: $x \leq y$.

Umkehrung der Relation erhält die Eigenschaften des Posets.

$x \in A$ heisst *obere Grenze von* B, wenn es kein $y \in B$ gibt mit $x \leq y$.

x ist *Supremum (least upper bound)* von B, sup$B$, wenn für alle $y \in B$ gilt: $y \leq x$.

Supremum und Infimum sind immer eindeutig!

Verbände

Zwei Hinleitungen:

1. durch Posets:

Ein Poset $\langle A, \leq \rangle$ ist ein Verband, wenn sup$\{a, b\}$ und inf$\{a, b\}$ für alle $a, b \in A$ existieren.

2. als algebraische Struktur:

Zwei neue (binäre) Operationen auf A:

$a \wedge b =$ inf$\{a, b\}$ (*meet*) und $a \vee b =$ sup$\{a, b\}$ (*join*).

Eigenschaften eines Verbandes:

Idempotenz

$$a \vee a = a, \; a \wedge a = a$$

Kommutativität

$$a \vee b = b \vee a, \; a \wedge b = b \wedge a$$

Assoziativität

$$(a \vee b) \vee c = a \vee (b \vee c), \; (a \wedge b) \wedge c = a \wedge (b \wedge c)$$

Absorption

$$a \wedge (a \vee b) = a, \; a \vee (a \wedge b) = a$$

Aussagen

Atomare Aussagen, die wahr oder falsch sein können:

“ Paris liegt in Frankreich ”

“ Essig schmeckt sauer ”

nicht: “ Paris ” oder “ in Frankreich ”

Syntax der Aussagenlogik

Atomare Formeln: $A_1, A_2, ...$;

Formeln induktiv definiert:

1. Alle atomaren Formeln sind Formeln.
2. Für alle Formeln F und G sind $(F \wedge G)$ (*Konjunktion*) und $(F \vee G)$ (*Disjunktion*) Formeln.
3. Für jede Formel F ist $\neg F$ (*Negation*) eine Formel.

Abkürzungen:

$A, B, C, ...$ für $A_1, A_2, A_3, ...,$

$(F \rightarrow G)$ für $(\neg F \vee G)$,

$(F \leftrightarrow G)$ für $((F \wedge G) \vee (\neg F \wedge \neg G))$

Notationsvarianten:

$\sim A$, $\overline{A}$ für $\neg A$,

& für $\wedge$

Bedeutung

Zuordnung einer "Bedeutung", Interpretation:

Semantik der Aussagenlogik

Menge der *Wahrheitswerte*: $\{0, 1\}$,

D Teilmenge der atomaren Formeln,

Belegung: $\mathcal{A}$ **D** $\rightarrow \{0, 1\}$,

1. $\mathcal{A}((F \wedge G)) = \begin{cases} 1, & \text{falls } \mathcal{A}(F) = 1 \text{ und } \mathcal{A}(G) = 1 \\ 0, & \text{sonst} \end{cases}$

2. $\mathcal{A}((F \vee G)) = \begin{cases} 1, & \text{falls } \mathcal{A}(F) = 1 \text{ oder } \mathcal{A}(G) = 1 \\ 0, & \text{sonst} \end{cases}$

3. $\mathcal{A}(\neg F) = \begin{cases} 1, & \text{falls } \mathcal{A}(F) = 0 \\ 0, & \text{sonst} \end{cases}$

$\wedge$ entspricht "und",
$\vee$ dem einschliesslichen "oder",
$\rightarrow$ "wenn ..., dann ...",
$\leftrightarrow$ " genau dann ..., wenn ...".

Wahrheitstafeln

$\mathcal{A}(F)$	$\mathcal{A}(G)$	$\mathcal{A}((F \wedge G))$
0	0	0
0	1	0
1	0	0
1	1	1

$\mathcal{A}(F)$	$\mathcal{A}(G)$	$\mathcal{A}((F \vee G))$
0	0	0
0	1	1
1	0	1
1	1	1

$\mathcal{A}(F)$	$\mathcal{A}(G)$	$\mathcal{A}((F \rightarrow G))$
0	0	1
0	1	1
1	0	0
1	1	1

$\mathcal{A}(F)$	$\mathcal{A}(G)$	$\mathcal{A}((F \leftrightarrow G))$
0	0	1
0	1	0
1	0	0
1	1	1

Beispiel

Sei $\mathcal{A}(A) = 1, \mathcal{A}(B) = 1, \mathcal{A}(C) = 0$.

Dann:

$$\begin{aligned}
\mathcal{A}(\neg((A \wedge B) \vee C)) &= \begin{cases} 1, & \text{falls } \mathcal{A}(((A \wedge B)) \vee C)) = 0 \\ 0, & \text{sonst} \end{cases} \\
&= \begin{cases} 0, & \text{falls } \mathcal{A}(((A \wedge B)) \vee C)) = 1 \\ 1, & \text{sonst} \end{cases} \\
&= \begin{cases} 0, & \text{falls } \mathcal{A}((A \wedge B)) = 1 \text{ oder } \mathcal{A}(C) = 1 \\ 1, & \text{sonst} \end{cases} \\
&= \begin{cases} 0, & \text{falls } \mathcal{A}((A \wedge B)) = 1 \text{ (da } \mathcal{A}(C) = 0) \\ 1, & \text{sonst} \end{cases} \\
&= \begin{cases} 0, & \text{falls } \mathcal{A}(A) = 1 \text{ und } \mathcal{A}(B) = 1 \\ 1, & \text{sonst} \end{cases} \\
&= 0
\end{aligned}$$

Definitionen

Sei F eine Formel und $\mathcal{A}$ eine Belegung.

$\mathcal{A}$ heisst *zu F passend*,

falls $\mathcal{A}$ für alle in F vorkommenden Formeln definiert ist.

$\mathcal{A}$ heisst *Modell für F*, geschrieben $\mathcal{A} \models F$,

falls $\mathcal{A}(F) = 1$.

F heisst *erfüllbar*,

falls es mindestens ein Modell für F gibt (sonst ist F unerfüllbar).

F heisst *gültig* oder *Tautologie*, geschrieben $\models F$,

falls jede zu F passende Belegung ein Modell für F ist.

G *folgt* aus F, wenn für jede Belegung $\mathcal{A}$, die zu F und G passt, gilt:

wenn $\mathcal{A}$ Modell für F ist, ist $\mathcal{A}$ auch Modell für G.

Bemerkung: dies ist die *semantische* Entsprechung der Implikation! G *folgt* aus F genau dann, wenn $(F \to G)$ eine Tautologie ist.

Satz, Beweis,...

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

Beweis:

F ist gültig gdw. jede passende Belegung ist Modell für F

gdw. jede dieser Belegungen ist kein Modell für $\neg F$

gdw. F hat kein Modell, ist also unerfüllbar.

Für Belegungen $\mathcal{A}$ und $\mathcal{A}'$ gilt: $\mathcal{A}(F) = \mathcal{A}'(F)$ gdw. $\mathcal{A}$ und $\mathcal{A}'$ auf allen atomaren Formeln in F übereinstimmen.

Beweis durch Induktion über den Formelaufbau:

Behauptung gelte für F und G, zu zeigen: Behauptung gilt auch für $\neg F$, $(F \wedge G)$ und $(F \vee G)$

...

Äquivalenz, Ersetzbarkeit

Zwei Formeln F und G heissen *äquivalent*, $F \equiv G$, falls für alle zu F und G passenden Belegungen $\mathcal{A}$ gilt: $\mathcal{A}(F) = \mathcal{A}(G)$

Ersetzbarkeitstheorem

Seien zwei Formeln F und G äquivalent, H eine Formel mit mindestens einem Vorkommen der Teilformel F. Dann ist H äquivalent zu H', wenn H' aus H gebildet wird, indem F darin (mindestens einmal) durch G ersetzt wird.

(Beweis durch Induktion)

Äquivalenzregeln und Ersetzbarkeitstheorem ermöglichen Umformung von Formeln in Normalformen...

Äquivalenzregeln

- Idempotenz
  (a) $(F \vee F) \equiv F$ (b) $(F \wedge F) \equiv F$
- Kommutativität
  (a) $(F \vee G) \equiv (G \vee F)$ (b) $(F \wedge G) \equiv (G \wedge F)$
- Assoziativität
  (a) $((F \vee G) \vee Z) \equiv (F \vee (G \vee Z))$
  (b) $((F \wedge G) \wedge Z) \equiv F \wedge (G \wedge Z))$
- Distributivität
  (a) $(F \vee (G \wedge Z)) \equiv ((F \vee G) \wedge (F \vee Z))$
  (b) $(F \wedge (G \vee Z)) \equiv ((F \wedge G) \vee (F \wedge Z))$
- Absorption
  (a) $(F \wedge (F \vee G)) \equiv F$ (b) $(F \vee (F \wedge G)) \equiv F$
- Doppelnegation
  $\neg\neg F \equiv F$
- DeMorgans Gesetze
  (a) $\neg(F \vee G) \equiv (\neg F \wedge \neg G)$ (b) $\neg(F \wedge G) \equiv (\neg F \vee \neg G)$
- Tautologieregeln
  (a) $(F \vee G) \equiv F$, falls F Tautologie
  (b) $(F \wedge G) \equiv G$, falls F Tautologie
- Unerfüllbarkeitsregeln
  (a) $(F \vee G) \equiv G$, falls F unerfüllbar
  (b) $(F \wedge G) \equiv F$, falls F unerfüllbar

Beispiele

“Wenn die Sonne scheint, gehe ich schwimmen. Da es regnet, gehe ich nicht zum Schwimmen”

$$(So \to Schw) \equiv (\neg So \vee Schw)$$
$$\not\equiv (\neg So \to \neg Schw) \equiv (So \vee \neg Schw)$$

“Wenn das Kind Fieber oder Husten hat und wir erreichen den Arzt, dann rufen wir ihn”

$$\begin{array}{lll}
 & (((F \vee H) \wedge A) \to R) & \\
\equiv & (\neg((F \vee H) \wedge A) \vee R) & \text{Auflösung von } \to \\
\equiv & ((\neg(F \vee H) \vee \neg A) \vee R) & \text{deMorgan} \\
\equiv & (((\neg F \wedge \neg H) \vee \neg A) \vee R) & \text{deMorgan} \\
\equiv & (((\neg F \vee \neg A) \wedge (\neg H \vee \neg A)) \vee R) & \text{Distr.Ges.} \\
\equiv & ((\neg F \vee \neg A \vee R) \wedge (\neg H \vee \neg A \vee R)) & \text{Distr.Ges. + Ass.Ges.} \\
\equiv & ((\neg A \vee \neg F \vee R) \wedge (\neg A \vee \neg H \vee R)) & \text{Komm.Ges.}
\end{array}$$

“Wenn das Kind Fieber hat, dann rufen wir den Arzt, wenn wir ihn erreichen, und, wenn wir ihn erreichen, dann rufen wir ihn, wenn das Kind hustet”

$$((A \to (F \to R)) \wedge (A \to (H \to R)))$$
$$\equiv ((\neg A \vee (F \to R)) \wedge (\neg A \vee (H \to R)))$$
$$\equiv ((\neg A \vee (\neg F \vee R)) \wedge (\neg A \vee (\neg H \vee R)))$$
$$\equiv ((\neg A \vee \neg F \vee R) \wedge (\neg A \vee \neg H \vee R))$$

Übung: Umformungen

Zu zeigen: zu jeder Formel F gibt es eine äquivalente Formel, die nur die Operatoren $\neg$ und $\rightarrow$ enthält.

Zu zeigen: nicht zu jeder Formel F gibt es eine äquivalente Formel, die nur die Operatoren $\vee$, $\wedge$ und $\rightarrow$ enthält.

Normalformen

Ein *Literal* ist eine atomare Formel oder deren Negation.

F ist in *konjunktiver Normalform* (KNF), wenn sie eine “Konjunktion von Disjunktionen von Literalen” ist. D.h., sie hat die Form:

$$F = ((L_{1,1} \vee L_{1,2} \vee ...) \wedge (L_{2,1} \vee L_{2,2} \vee ...) \wedge ...)$$

oder auch

$$F = (\bigwedge_{i=1}^{n} (\bigvee_{j=1}^{m_i} L_{i,j}))$$

Mengenschreibweise (Mengen von *Klauseln*):

$$F = \{\{L_{1,1}, L_{1,2}, ...\}, \{L_{2,1}, L_{2,2}, ...\}, ...\}$$

F ist in *disjunktiver Normalform* (DNF), wenn sie eine “Disjunktion von Konjunktionen von Literalen” ist. D.h., sie hat die Form:

$$F = ((L_{1,1} \wedge L_{1,2} \wedge ...) \vee (L_{2,1} \wedge L_{2,2} \wedge ...) \vee ...)$$

oder auch

$$F = (\bigvee_{i=1}^{n} (\bigwedge_{j=1}^{m_i} L_{i,j}))$$

Induktionsbeweis, dass diese Formeln für jede Formel existieren, liefert Konstruktionsverfahren dafür. Aber: exponentielle Verlängerung...

Diätvorschriften

“Wenn ich kein Bier trinke, habe ich immer Fisch.”

“Wenn ich Bier und Fisch zur selben Mahlzeit habe, verzichte ich auf Eiscreme.”

“Wenn ich Eiscreme habe oder Bier meide, dann rühre ich den Fisch nicht an”

Formel:

$$G = ((\neg B \rightarrow F) \wedge ((B \wedge F) \rightarrow \neg E) \wedge ((E \vee \neg B) \rightarrow \neg F))$$

Warhheitstafel:

B	F	E	(¬B →F)	(B ∧ F)	..→¬E	(E ∨¬B)	..→¬F	G
0	0	0	0	0	1	1	1	0
0	0	1	1	0	1	1	0	0
0	1	0	0	0	1	1	1	0
0	1	1	1	0	1	1	0	0
1	0	0	1	0	1	0	1	1
1	0	1	1	1	1	0	1	1
1	1	0	1	0	1	1	0	0
1	1	1	1	1	0	1	1	0

Also: $((B \wedge \neg F \wedge \neg E) \vee (B \wedge \neg F \wedge E)) \equiv (B \wedge \neg F)$

Hornformeln

Hornformel: Formel in KNF, deren Disjunktionen höchstens ein positives Literal enthalten.

Beispiel:

$$G = (A \vee \neg B) \wedge \neg C \wedge D \wedge (\neg E \vee \neg F)$$

Umformung in Konjunktionen von Implikationen:

$$G \equiv (B \to A) \wedge (C \to 0) \wedge (1 \to D) \wedge (E \wedge F \to 0)$$

Test auf Erfüllbarkeit: Eingabe: Hornformel F

1. Markiere alle Vorkommen einer atomaren Formel A, falls Teilformel $(1 \to A)$ in F vorkommt.
2. Wenn es in F Teilformeln der Form $(A_1 \wedge ...A_n \to 0)$ gibt und $A_1, ...A_n$ markiert sind, gib “unerfüllbar” aus und stoppe.

   Wenn es in F Teilformeln der Form $(A_1 \wedge ...A_n \to B)$ gibt, $A_1, ...A_n$ markiert sind und B nicht, markiere alle Vorkommen von B.

   Wiederhole, bis keine der beiden Formelarten mehr vorkommt.
3. Gib “erfüllbar” aus. Markierte Atomformeln sind mit 1 belegt: $\mathcal{A}(A_i) = 1$ gdw. A_1 hat eine Markierung.

Resolutionskalkül

Aufgabe: Unerfüllbarkeit einer Formelmenge
(Anwendungen: F Tautologie? Folgt G aus $F_1 \wedge ... F_n$?)

Seien K_1, K_2, R Klauseln. R ist *Resolvent* von K_1 und K_2, falls es ein Literal L gibt mit $L \in K_1$ und $\overline{L} \in K_2$ und $R = (K_1 \backslash \{L\} \cup K_2 \backslash \{\overline{L}\})$.
($\overline{L}$ ist $\neg A_i$, wenn $L = A_i$ und A_i, wenn $L = \neg A_i$)
Leere Menge als Resolvent: $\square$, unerfüllbar.

Resolutionslemma:

Eine Klauselmenge F ist äquivalent zu $F \cup \{R\}$, wenn R Resolvent zweier Klauseln aus F ist.

Res(F) $::= F \cup \{R \mid R$ ist Resolvent zweier Klauseln in $F\}$
Res$^0(F) = F$, *Res*$^{n+1}(F) =$ *Res*$($*Res*(F), $n \geq 0$.

$$\textit{Res}^*(F) = \bigcup_{n \geq 0} \textit{Res}(F)$$

Resolutionssatz der Aussagenlogik:

Eine Klauselmenge F ist unerfüllbar genau dann, wenn $\square \in$ *Res*$^*(F)$.

Korrektheit: keine erfüllbare Formelmenge wird als unerfüllbar “erkannt”.
Vollständigkeit: alle unerfüllbaren Formelmengen werden erkannt.

Resolution, Beispiel

Zu zeigen: $(F \to G)$, also $(F \wedge \neg G)$?

$$F = ((\neg A \vee B) \wedge (\neg B \vee C) \wedge (A \vee \neg C) \wedge (A \vee B \vee C))$$
$$G = (A \wedge B \wedge C) \qquad \neg G = (\neg A \vee \neg B \vee \neg C) \text{ (deMorgan)}$$

$(F \wedge \neg G)$ als Klauselmenge:

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

$$\{B, C\} \qquad \{\neg A, \neg C\}$$

$$\{C\} \qquad \{\neg C\}$$

$$\square$$

Wozu Resolution?

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

enthält die gleichen positiven wie negativen Literale, ist aber weder gültig noch unerfüllbar. Beweis: Geben Sie $Res*(H)$ und $Res*(\neg H)$ an, $\square$ ist nicht Element davon.

Prädikatenlogik 1. Stufe

Erweiterung der Aussagenlogik um den Ausdruck von Eigenschaften von Objekten.

Beispiel:
$M = N$ genau dann, wenn alle x in M auch Element von N sind und alle y in N auch in M.

Beispiel:
“Alle Menschen sind sterblich”
“Sokrates ist ein Mensch”

Neu sind Variablen, Funktionen und Prädikate, sowie Quantoren.

Syntax der Prädikatenlogik

Variablen: $x_i, i = 1, 2, 3, ...$
Funktionssymbole: $f_i^k, i = 1, 2, 3, ..., k = 0, 1, 2, ...$
Prädikatssymbole: $P_i^k, i = 1, 2, 3, ..., k = 0, 1, 2, ...$
k ist die Stelligkeit, i Unterscheidungsindex

Terme:

1. Jede Variable ist ein Term.
2. Sei f ein Funktionssymbol mit Stelligkeit k und seien $t_1, ..t_k$ Terme, dann ist $f(t_1, ..t_k)$ ein Term.

"Nullstellige" Funktionen werden als Konstanten bezeichnet.

Formeln:

1. Seien $t_1, ..t_k$ Terme und P ein Prädikatssymbol mit der Stelligkeit k, dann ist $P(t_1, ..t_k)$ eine Formel.
2. Sei x eine Variable und F eine Formel, dann sind $\forall x(F)$ und $\exists x(F)$ Formeln.

   $\forall$ und $\exists$ heissen *Quantoren*, gelesen "für alle" und "es gibt ein ... so, dass"
3. Seien F, G Formeln, dann auch $\neg F$, $(F \vee G)$, $(F \wedge G)$, $(F \rightarrow G)$, $(F \leftrightarrow G)$.

Prädikatenlogik

Beispiel:

1. “Alle Menschen sind sterblich”: $\forall x(M(x) \rightarrow S(x))$
2. “Sokrates ist ein Mensch”: $M($sokrates$)$

gebundene / freie Variablen: innerhalb / ausserhalb des Wirkungsbereichs eines Quantors.

x ist in $\forall x(M(x) \rightarrow S(x))$ gebunden und frei in $(M(x) \rightarrow S(x))$.

Geschlossene Formel: keine freie Variable.

$\forall x(M(x) \rightarrow S(x))$ ist eine geschlossene Formel, $\forall x(M(x) \rightarrow S(y))$ eine offene.

Semantik der Prädikatenlogik

Eine *Struktur* $\mathcal{A} = (U_\mathcal{A}, I_\mathcal{A})$ besteht aus einer bel., nicht leeren *Grundmenge* $U_\mathcal{A}$, dem *Grundbereich, Universum*, und der Abbildung $I_\mathcal{A}$, *Interpretation*, die

- jedem k-stelligen Prädikatensymbol P (im Definitionsbereich von $I_\mathcal{A}$) ein k-stelliges Prädikat (= k-stellige Relation) $P^\mathcal{A}$ über $U_\mathcal{A}$,
- jedem k-stelligen Funktionssymbol f (im Definitionsbreich von $I_\mathcal{A}$) eine k-stellige Funktion $f^\mathcal{A}$ auf $U_\mathcal{A}$ und
- jeder Variable x (im Definitionsbreich von $I_\mathcal{A}$) ein Element $x^\mathcal{A}$ von $U_\mathcal{A}$ zuordnet.

$\mathcal{A}$ heisst zu einer Formel F passend, wenn $I_\mathcal{A}$ auf allen Variablen, Funktionssymbolen und Prädikatensymbolen in F definiert ist.

Semantik der PL

Sei F eine Formel, $\mathcal{A}$ eine zu F passende Struktur.
$\mathcal{A}\ (t)$ für Terme:

- falls t Variable, $t = x$: $\mathcal{A}\ (t) = x^{\mathcal{A}}$.
- falls $t = f(t_1, ..., t_k)$: $\mathcal{A}\ (t) = f^{\mathcal{A}}(\mathcal{A}(t_1), ..., \mathcal{A}(t_k))$

$\mathcal{A}\ (F)$ für Formeln:

- falls $F = P(t_1, ..., t_k)$:
  $\mathcal{A}\ (F) = \begin{cases} 1, & \text{falls } (\mathcal{A}(t_1), ...\mathcal{A}(t_k)) \in P^{\mathcal{A}} \\ 0, & \text{sonst} \end{cases}$
- falls $F = \forall x G$:
  $\mathcal{A}\ (F) = \begin{cases} 1, & \text{falls für alle } d \in U_{\mathcal{A}} \text{ gilt: } \mathcal{A}_{[x/d]}(G) = 1 \\ 0, & \text{sonst} \end{cases}$
- falls $F = \exists x G$:
  $\mathcal{A}\ (F) = \begin{cases} 1, & \text{falls für ein } d \in U_{\mathcal{A}} \text{ gilt: } \mathcal{A}_{[x/d]}(G) = 1 \\ 0, & \text{sonst} \end{cases}$

Dabei ist $\mathcal{A}_{[x/d]}$ diejenige Struktur die bis auf $\mathcal{A}\ (x)$ mit $\mathcal{A}$ identisch ist: $\mathcal{A}_{[x/d]}(x) = d$

$\wedge$, $\vee$ und $\neg$ wie in der Aussagenlogik...
alternative Schreibweise für $\mathcal{A}(F)$: $[\![F]\!]^{\mathcal{A}}$

Semantik der PL, Begriffe

Falls für $\mathcal{A}$ und F $\mathcal{A}$ (F) = 1, dann $\mathcal{A} \models F$.
"$\mathcal{A}$ erfüllt F, F gilt in $\mathcal{A}$, $\mathcal{A}$ ist Modell für F."

Falls für alle passenden $\mathcal{A}$ gilt: $\mathcal{A}$ (F) = 1, dann $\models F$.
"F ist gültig."

F heisst erfüllbar, falls F mindestens ein Modell hat, sonst unerfüllbar.

Übung: Formulierung in PL

1. “P. ist Anhänger der logischen Grammatik-Schule und spricht mit allen Kollegen, die keine Generativisten sind.”
2. “Q. vertritt die Lexikalisch-Funktionale Grammatik und diskutiert mit jedem, der über formale Grundlagen arbeitet.”
3. “R. ist Duden-Redakteur, formale Grundlagen können ihm gestohlen bleiben, aber er spricht gern mit jedem ohne Ansehen seiner grammatiktheoretischen Präferenzen.”
4. “S. ist Government & Binding-Theoretiker und spricht mit seinesgleichen.”
5. “T. hängt der Head-Driven Phrase Structure Grammar an, redet aber mit allen Generativisten.”
6. “Alle Anhänger der LFG, HPSG und GB sind Generativisten.”
7. “Logiker und Generativisten beschäftigen sich mit formaler Sprachbeschreibung.”
8. “Eine Unterhaltung findet statt, wenn zwei (verschiedene) Leute miteinander reden.”

Lösungen

(Bem.: wenn “spricht mit” heissen soll “spricht nur mit”, wird aus den Implikationen ein $\leftrightarrow$!)

1. Logiker(p) $\wedge$ $\forall$x ($\neg$ Generativist(x) $\rightarrow$SprichtMit(p,x))
2. LFG(q) $\wedge$ $\forall$x (Formalist(x) $\rightarrow$SprichtMit(q,x))
3. Duden(r) $\wedge$ $\neg$Formalist(q) $\wedge$ $\forall$x (SprichtMit(r,x))
4. GB(s) $\wedge$ $\forall$x (GB(x) $\rightarrow$SprichtMit(s, x))
5. HPSG(t) $\wedge$ $\forall$x (Generativist(x) $\rightarrow$SprichtMit(t,x))
6. $\forall$x ((LFG(x) $\vee$ HPSG(x) $\vee$ GB(x))$\rightarrow$Formalist(x))
7. $\forall$x ((Logiker(x) $\vee$ Generativist(x)) $\rightarrow$Formalist(x))
8. $\forall$x $\forall$y ((SprichtMit(x,y) $\wedge$ SprichtMit(y,x)
   $\wedge\neg$ Unterhaltung(x,x) $\wedge\neg$ Unterhaltung(y,y))
   $\rightarrow$Unterhaltung(x,y))
   $\wedge$ ($\forall$z ($\neg$ Unterhaltung(z,z)))

Äquivalenzen

Seien F und G Formeln.

- (a) $\neg\forall xF \equiv \exists x\neg F$ (b) $\neg\exists xF \equiv \forall x\neg F$
- Falls x nicht frei in G vorkommt:
  (a) $(\forall xF\wedge G) \equiv \forall x(F\wedge G)$ (b) $(\forall xF\vee G) \equiv \forall x(F\vee G)$
  (c) $(\exists xF\wedge G) \equiv \exists x(F\wedge G)$ (d) $(\exists xF\vee G) \equiv \exists x(F\vee G)$
- (a) $(\forall xF \wedge \forall xG) \equiv \forall x(F \wedge G)$
  (b) $(\exists xF \vee \exists xG) \equiv \exists x(F \vee G)$
- (a) $\forall x\forall yF \equiv \forall y\forall xF$ (b) $\exists x\exists yF \equiv \exists y\exists xF$

NICHT äquivalent sind:

$(\forall xF \vee \forall xG) \not\equiv \forall x(F \vee G)$

$(\exists xF \wedge \exists xG) \not\equiv \exists x(F \wedge G)$

Beweis durch Gegenbeispiele...

Substitution etc.

Sei F eine Formel, x eine Variable und t ein Term.

Dann ist $F[x/t]$ diejenige Formel, in der in F jedes freie Vorkommen von x durch t ersetzt wird.

$[x/t]$ ist eine *Substitution*.

Überführungslemma:

$$\mathcal{A}(F[x/t]) = \mathcal{A}_{[x/\mathcal{A}(t)]}(F)$$

Gebundene Umbenennung:

Sei $F = QxG$ eine Formel ($Q \in \{\exists, \forall\}$), y komme nicht in G vor. Dann:

$$F \equiv QyG[x/y]$$

Zu jeder Formel existiert äquivalente Formel in *bereinigter* Form, d.h. keine Variable kommt sowohl frei als auch gebunden vor, und hinter verschiedenen Quantoren stehen verschiedene Variablen.

Anwendung: Logikprogramme

Logikprogramm: Sequenz von Klauseln der Form:

```
P :- Q, R, S.
```

`P` ist *Kopf*, der Rest *Rumpf* der Klausel,
entspricht $((Q \wedge R \wedge S) \to P)$,
wenn Q, R und S erfüllt sind, dann auch P)

Klausel mit leerem Rumpf: *Fakt*:

```
P.
```

Klausel mit leerem Kopf: *Frage* :

```
?- P.
```

Anwendung: Eigenschaften von Relationen

- ein reflexives Prädikat:

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

- ein transitives:

$$\forall x \forall y \forall z((P(x,y) \land P(y,z)) \rightarrow P(x,z))$$

- ein symmetrisches:

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

- Die schlechte Nachricht: Antisymmetrie ist nicht in Prädikatenlogik erster Stufe darstellbar.

  Formulierungen wie:

$$\forall x \forall y((P(x,y) \land P(y,x)) \rightarrow Id(x,y))$$

  setzen Identität als Prädikat voraus; das ist aber ohne Einführung eines Quantors “es gibt genau ein...” nicht möglich.

  Die Formel:

$$F = \forall x \forall y((P(x,y) \land P(y,x)) \rightarrow P(x,y))$$

  drückt nur aus, dass $I_{\mathcal{A}}(P)$ eine Äquivalenzrelation auf der Grundmenge $U_{\mathcal{A}}$ sein muss, damit $\mathcal{A} \models F$ gilt. Identität und Äquivalenz können in der Prädikatenlogik nicht unterschieden werden!

Grammatiken als Logikprogramme

kontextfreie Grammatikregeln:

non-terminal $\rightarrow$ body, z.B.

s $\rightarrow$ np, vp.	det $\rightarrow$ [the].
np $\rightarrow$ det, noun.	noun $\rightarrow$ [cat].
vp $\rightarrow$ trans_verb, np.	noun $\rightarrow$ [dog].
vp $\rightarrow$ intrans_verb.	intrans_verb $\rightarrow$ [sleeps].
	trans_verb $\rightarrow$ [chases].

beschreiben eine *formale Sprache*.

($\rightarrow$ ist hier nicht mit der logischen Implikation zu verwechseln!)

Der Rumpf (die rechte Seite) einer Regel kann aus einem Terminalsymbol (entsprechend den Wörtern einer natürlichen Sprache) oder einer Folge von Nichtterminalsymbolen (entsprechend den Wortklassen/–arten) bestehen.

Eine *Grammatik* besteht aus den endlichen Mengen der Terminalen, der Nichtterminalen und der Regeln, sowie einem der Nichtterminalen, dass als Startsymbol ausgezeichnet wird, hier `s`.

Ein *Wort* einer von einer Grammatik erzeugten Sprache ist eine Folge von Terminalsymbolen, die durch Anwendung der Regeln aus dem Startsymbol abgeleitet werden können.

Grammatiken als Logikprogramm

kf. Regel in Klauselform
(die `S` sind Variablen für Wortpositionen):

```
sentence(S0,S) :- np(S0, S1), vp(S1, S).
np(S0,S) :- det(S0, S1), noun(S1, S).
...
```

Das Lexikon:

```
det(S0,S) :- connects(S0, the, S).
noun(S0,S) :- connects(S0, cat, S). ...
```

Der Satz: “the cat chases the dog” wird codiert als:

```
connects(1,the,2).
connects(2,cat,3).
connects(3,chases,4).
connects(4,the,5).
connects(5,dog,6).
```

Test, ob der Satz grammatikalisch ist:

```
?-sentence(1,6).
```

Definite Clause Grammars

Kontextfreie Grammatiken für natürliche Sprachen nicht adäquat (...)

Erweiterung der kf. Grammatikregeln:

- Nicht-Terminale sind komplexe Terme mit Variablen: `np(X, S)` oder `s(S)`

  Zusätzliche Argumente dienen zum Aufbau des Parsebaumes.
- Auf der rechten Regelseite sind Funktionsaufrufe erlaubt:

  noun(N) $\rightarrow$ [W], {rootform(W,N), is_noun(N)}.

Als Logikprogramm:

```
noun(N,S0,S):- connects(S0,W,S), rootform(W,N),
is_noun(N).
```

Diese Transformationen können von den meisten PROLOG-Interpretern automatisch durchgeführt werden.

Erweiterungen

Lexikon

Statt

noun(N) $\rightarrow$ [dog]

für jedes Wort der Kategorie `noun':

noun(N) $\rightarrow$ [N], {is_noun(N)}

und is_noun(...) für alle Wörter.

(Zusätzlichge Argumente und Prozeduren möglich)

Kontextabhängigkeit: **Kongruenz**

Einfügen eines zusätzlichen Arguments für jede Eigenschaft, die kongruieren soll:

is_determiner(every, singular)

is_determiner(all, plural)

– ebenso in allen Grammatikregeln, in denen Artikel vorkommen, Kongruenz wird durch Verwendung von Variablen erzwungen:

noun_phrase(Number, np(Det,Noun)) $\rightarrow$ determiner(Number, Det), noun(Number, Noun).

Termunifikation in der Prädikatenlogik

Um das Resolutionskalkül auf die Prädikatenlogik anzuwenden, müssen nicht nur die Formeln in KNF umgeformt, sondern auch geeignete Substitutionen für die Variablen ausgeführt werden, um die Literale möglichst zu vereinfachen. Dieser Vorgang heisst *Termunifikation*.

Def: *(allgemeinster) Unifikator*

Eine Substitution sub ist ein Unifikator einer endlichen Menge von Literalen **L** $= \{L_1, ..., L_k\}$, wenn $L_1 sub = L_2 sub = ... = L_k sub$.

sub ist ein allgemeinster Unifikator von **L**, wenn für jeden anderen Unifikator sub' gilt: es gibt s mit $sub' = sub\, s$ (d.h. für alle Formeln F gilt $Fsub' = Fsubs$)

Beispiel:

$\{P(x), P(f(y))\}$ wird durch $[x/f(y)]$ unifiziert.

Unifikationsalgorithmus:

Eingabe: nichtleere Literalmenge **L**

$sub := []$;

L$sub := \{L_1 sub, ..., L_k sub\}$;

solange $|$ **L**$sub\, | > 1$:

- Durchlaufe **L**sub, bis sich (von links nach rechts gelesen) zwei Literale darin in mindestens einem Zeichen unterscheiden.

  Wenn keines der beiden Zeichen eine Variable ist, stoppe mit "nicht unifizierbar".

  sonst:

  sei x die Variable, t der andere Term; wenn x in t vorkommt, stoppe mit "nicht unifizierbar".

  sonst:

  $sub := sub[x/t]$;

  L$sub := \{L_1 sub, ..., L_k sub\}$

Gib sub als allgemeinsten Unifikator aus.

Probleme der prädikatenlogischen **Resolution**: Explosion der Kombinationsmöglichkeiten, Auswahl der Substitutionen, etc.

Abhilfe: Strategien, Restriktionen: Bedingungen, unter denen zwei Klauseln resolviert werden können.

Merkmallogik

Grundmenge: Merkmalstrukturen (*feature structures*), Mengen aus Namen (*labels*) (für Merkmale, Attribute) und deren Werten.

Formal: partielle Funktion von Merkmalen auf Werte.

Notation: [*name: wert*]

- Werte selbst können wieder Merkmalstrukturen sein:

$$\begin{bmatrix} \textit{cat:} & np \\ \textit{agreement:} & \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix} \end{bmatrix}$$

- dieselben Werte können von mehreren Attributen geteilt werden, dh., es kann auf einen einmal definierten Wert verwiesen werden (*reentrance, structure sharing*, Koreferenz, Koindizierung):

$$\begin{bmatrix} \textit{f:}\boxed{1} & \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix} \\ \textit{g:}\boxed{1} & \end{bmatrix}$$

Dabei sind zirkuläre Verweise nicht erlaubt!

Grundbegriffe

atomare Merkmalstrukturen:

einfache Symbole wie 3, S, np...

komplexe Ms.: [*name: wert*]

leere Ms., auch *Variable*: $[\,]$

Extraktion:

Sei D eine Ms., f eines ihrer Attribute. Dann bezeichnet $D(f)$ den Wert von f in D.

Bsp.: $D = \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix}$, dann $D(\textit{person}) = 3$.

Definitions- und Wertebereich wie bei Funktionen:

Definitionsbereich, Urbild: dom(D) = $\{$*person*, *number*$\}$

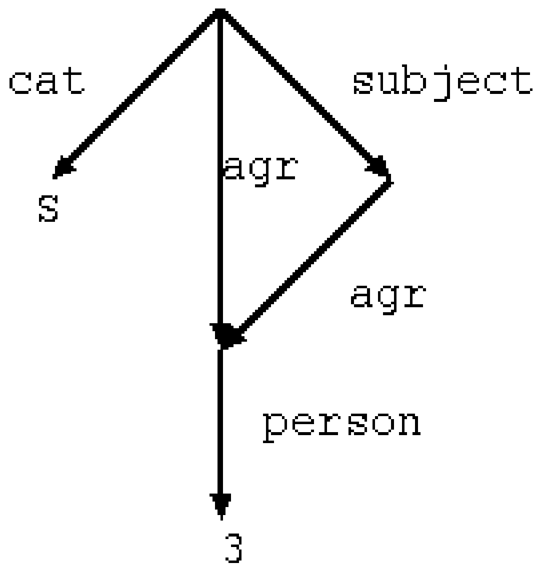
Wertebereich, Bild: range(D) = $\{3, S\}$

Ein *Pfad* innerhalb einer Ms. ist eine Folge von Attributen: $\langle$ *agreement person* $\rangle$

Die Extraktion kann auch auf Pfade angewendet werden...

Merkmalstrukturen als Graphen

$$\begin{bmatrix} \textit{cat:} & S \\ \textit{subject:} & \begin{bmatrix} \textit{agreement:}\boxed{1} \; [\textit{person:}3] \end{bmatrix} \\ \textit{agreement:}\boxed{1} & \end{bmatrix}$$



(Abb. geborgt von Martin Volk)

Merkmalstrukturen entsprechen den Knoten, Merkmalsnamen den Kanten.

Subsumption

Ordnungsrelation $\sqsubseteq$:

$D \sqsubseteq D'$, wenn D weniger (oder dieselbe) Information enthält als D'. D ist allgemeiner, D' spezieller.

Shieber:

$D \sqsubseteq D'$ gdw. für alle $l \in$ dom$(D) : D(l) \sqsubseteq D'(l)$
und für alle Pfade p, q mit $D(p) = D(q) : D'(p) = D'(q)$

D subsummiert D' (D' wird von D subsummiert), wenn für alle Attribute in D gilt, dass ihre Werte von denen in D' subsummiert werden.

Für atomare Strukturen gilt: $a \sqsubseteq b$ und $b \sqsubseteq a$ gdw. $a = b$

Die leere Struktur subsummiert alle anderen.

Beispiele:

$$[\,] \sqsubseteq \left[\textit{cat: } np\right] \sqsubseteq \begin{bmatrix} \textit{cat:} & np \\ \textit{agreement:} & [\textit{genus: } mas] \end{bmatrix}$$

Mit der Relation $\sqsubseteq$ bilden Ms. einen Verband!

Unifikation

Unifikation bildet aus zwei *verträglichen* Merkmalstrukturen eine neue, die von beiden subsummiert wird:

$D = D' \sqcup D''$ gdw.
D ist die allgemeinste Struktur mit:
$D' \sqsubseteq D$ und $D'' \sqsubseteq D$

Beispiel:

$$[\textit{number:}S] \sqcup [\textit{person:}3] = \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix}$$

Unifikation zwischen nicht verträglichen Strukturen scheitert, d.h. liefert den Wert *fail* oder $\bot$.

Beispiel:

$$[\textit{number:}S] \sqcup [\textit{number:}P] = \textit{fail}$$

Eigenschaften der Unifikation

neutrales Element:

$[\,] \sqcup D = D \sqcup [\,] = D$

Die Verbandseigenschaften:

1. Kommutativität:
   $D \sqcup E = E \sqcup D$
2. Assoziativität:
   $(D \sqcup E) \sqcup F = D \sqcup (E \sqcup F)$
3. Idempotenz:
   $D \sqcup D = D$

Unterschiede zur Termunifikation:

- Mengen vs. N-Tupel, dh. Reihenfolge in der Termunifikation entscheidend.
- Stelligkeit: eine Eigenschaft wie Kongruenz müßte in Prädikatenlogik mit fester Anzahl von Parametern definiert werden.
- Identität und Gleichheit lassen sich in Merkmalstrukturen ausdrücken...

Unifikation mit Identität und Gleichheit

bei Identität (Koreferenzierung)

$$\begin{bmatrix} \textit{agreement:}\ \boxed{1} & [\textit{number:}\ S] \\ \textit{subject:} & [\textit{agreement:}\ \boxed{1}] \end{bmatrix}$$

$$\sqcup\ [\textit{subject:}\ [\textit{agreement:}\ [\textit{person:}\ 3]\ ]\ ]$$

$$= \begin{bmatrix} \textit{agreement:}\ \boxed{1} & \begin{bmatrix} \textit{number:}\ S \\ \textit{person:}\ 3 \end{bmatrix} \\ \textit{subject:} & [\textit{agreement:}\ \boxed{1}] \end{bmatrix}$$

bei Gleichheit:

$$\begin{bmatrix} \textit{agreement:} & [\textit{number:}\ S] \\ \textit{subject:} & [\textit{agreement:} [\textit{number:}\ S]\ ] \end{bmatrix}$$

$$\sqcup\ [\textit{subject:}\ [\textit{agreement:}\ [\textit{person:}\ 3]\ ]\ ]$$

$$= \begin{bmatrix} \textit{agreement:} & [\ \textit{number:}\ S\ ] \\ \textit{subject:} & \begin{bmatrix} \textit{agreement:} & \begin{bmatrix} \textit{number:}\ S \\ \textit{person:}\ 3 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Unifikation, Erweiterungen

Disjunktion

aufwendig...

Mengen

Negation

zwei Arten: closed/open World

Typisierte Merkmalstrukturen

Wohlgeformtheit

Typenhierarchien...

Inferenz, Schlussfolgerung

Kalkül: Logik + syntaktische Ableitung:

$$F, G \vdash H$$

Prämisse: Aussage, die als wahr angenommen wird.

Schluss (Konklusion): Aussage, die aus der Wahrheit der Prämisse folgt.

Gültigkeit eines Arguments:

Es gibt keine Belegung/Struktur, unter der die Prämisse wahr, aber der Schluss falsch wäre.

Syntaktisch: (Prämisse $\rightarrow$ Schluss) ist gültig.

gültige Regel z.B.: $\dfrac{A, A \rightarrow B}{B}$ (Modus Ponens)

Beispiel: $\dfrac{\text{“Wenn es regnet, gehe ich ins Kino”} \quad \text{“Es regnet”}}{\text{“Ich gehe ins Kino”}}$

ungültige Regeln z.B.:

$$\frac{A \rightarrow B, B}{A} \quad \text{und} \quad \frac{\neg A, A \rightarrow B}{\neg B}$$

Weitere Regeln

$$\frac{\neg B, A \rightarrow B}{\neg A}$$ (Modus Tollens)

Beispiel:
$$\frac{\text{“Wenn es regnet, gehe ich ins Kino”} \quad \text{“ich gehe nicht ins Kino”}}{\text{“Es regnet nicht”}}$$

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

Beispiel:
$$\frac{\text{“Sie trinkt Tee oder Kaffee”} \quad \text{“Sie trinkt keinen Kaffee”}}{\text{“Sie trinkt Tee”}}$$

$$\frac{A}{A \vee B}$$

Beispiel:
$$\frac{\text{“Sie trinkt Tee”}}{\text{“Sie trinkt Tee, oder draussen vor der Tür faucht ein grüner Drachen”}}$$

Arten der Inferenz

Deduktion: von Klassen auf Exemplare schliessen:

$$\frac{\text{alle Menschen sind sterblich, Sokrates ist ein Mensch}}{\text{Sokrates ist sterblich}}$$

unsichere Inferenzen:

Induktion: Generalisierung, Analogiebildung; von Exemplaren auf die Gesamtheit:

$$\frac{\text{Sokrates ist sterblich, Sokrates ist ein Mensch}}{\text{alle Menschen sind sterblich}}$$

Abduktion: Klassenzuordnung anhand von beobachteten Eigenschaften:

$$\frac{\text{alle Menschen sind sterblich, Sokrates ist sterblich}}{\text{Sokrates ist ein Mensch.}}$$

Gegenbeispiele für diese Formen der Inferenz?

Modell, Denotation

schon bekannt aus der Prädikatenlogik:

Struktur $\mathcal{A} = (U, I)$ mit U als Grundmenge, Universum, und I als Interpretation aller verwendeten (nicht-logischen) Symbole als Funktionen, Relationen oder Konstanten in U, Formeln als 0 oder 1.

Oft davon getrennt betrachtet: Belegungsfunktion h für Variablen x: $h(x) \in U$.

Die Interpretation eines logischen Ausdrucks (Term oder Formel) heisst auch *Denotat* ($\rightarrow$“denotationale Semantik”).

Schreibweise:

$[\![F]\!]^{\mathcal{A},h}$, $[\![t]\!]^{\mathcal{A},h}$

Eine Struktur $\mathcal{A}$ ist ein Modell für eine Formelmenge Σ, gdw. jedes Element F von Σ in $\mathcal{A}$ wahr ist.

Eine Formel G folgt aus einer Formelmenge Σ, gdw. G in jedem Modell von Σ wahr ist.

Theorien

Eine *Theorie* **T** ist ein Menge von Formeln, die unter Folgerung abgeschlossen ist, d.h. jede Formel, die aus einer oder mehreren der Formeln in **T** folgt, ist selbst in **T**.

Modelltheoretische Definition:

$$\mathbf{T} ::= Th(\mathcal{A}) = \{F \mid \mathcal{A} \models F\}$$

axiomatische Methode:

Menge M von Formeln vorgegeben mit:

$$\mathbf{T} ::= \textit{Cons}(M) = \{\ F \mid \text{ es gibt } F_1, ...F_n \in M, \text{ so dass } G \text{ aus } M \text{ folgt}\}$$

$$\mathbf{T} ::= \textit{Cons}(\emptyset) = \{\ F \mid F \text{ ist gültig}\}$$

Eigenschaften formaler Systeme

Konsistenz:

Es können nicht gleichzeitig eine Aussage und deren Negation abgeleitet werden.

Inkonsistente Theorien haben kein Modell!

Wenn Inkonsistenz bewiesen werden soll: Widerspruch ableiten.

Wenn Konsistenz bewiesen werden soll: Modell angeben.

Unabhängigkeit:

eine Frage der Eleganz: ist die Menge der Axiome minimal, d.h. lässt sich keines der Axiome durch andere ausdrücken?

Korrektheit:

Es können keine “unerwünschten” Aussagen abgeleitet werden.

Wenn $F_1, F_2, ...F_n \vdash F$, dann $F_1, F_2, ...F_n \models F$

Eigenschaften formaler Systeme II

Vollständigkeit:

Es werden alle “erwünschten” Aussagen abgeleitet.

Wenn $F_1, F_2, ...F_n \models F$, dann $F_1, F_2, ...F_n \vdash F$

weitere Verwendungen des Begriffs:

- formal vollständig (jede Aussage kann bewiesen oder widerlegt werden)
- semantisch vollständig in Bezug auf ein Modell (alle im Modell wahren Aussagen können abgeleitet werden)

Bedauerlicherweise sind interessante Systeme oft unvollständig:

Diagonalisierungsargument...

Axiomatisierung von Stringkonkatenation

Alphabet: Menge von Symbolen

String, Zeichenkette: Folge von Symbolen

Konkatenation: Aneinanderreihung, zweistellige Operation "," auf einer Menge A von Zeichenketten.

ϵ: Zeichenkette der Länge 0.

- Axiomatisierung ohne leeren String (Halbgruppe)
  1. Abgeschlossenheit:
     $\forall x \forall y((x \in A \wedge y \in A) \rightarrow x, y \in A)$
  2. Assoziativität:
     $\forall x \forall y \forall z((x, y), z = x, (y, z))$

  Beispiel für ein Modell dafür: $A = \{z \mid z$ besteht aus einer geraden Anzahl von 'a's und / oder 'b's$\}$.
- Axiomatisierung mit leerem String (Monoid)
  1. Abgeschlossenheit:
     $\forall x \forall y((x \in A \wedge y \in A) \rightarrow x, y \in A)$
  2. Assoziativität:
     $\forall x \forall y \forall z((x, y), z = x, (y, z))$
  3. neutrales Element:
     $\exists e \forall x(x, e = e, x = x)$

Axiomatisierung der Mengentheorie

Mehrere Möglichkeiten, hier Zermelo-Fränkel.

Vorausgesetzte Relation: “ist Element von”

1. Extensionalität:
   Wenn x und y dieselben Elemente haben, gilt $x = y$.
2. Regularität / Begründung:
   Für jede nicht leere Menge x gibt es ein $y \in x$ mit: es gibt kein z mit $z \in x$ und $z \in y$
3. Leere Menge:
   Es gibt eine Menge ohne Elemente.
4. Ungeordnetes Paar:
   Wenn x und y Mengen sind, gibt es eine Menge z so, dass für alle w gilt: $w \in z$ gdw. $w = x$ oder $w = y$, d.h. $\{x, y\}$ existiert.
5. Vereinigung:
   Für alle x gibt es ein y so, dass gilt: $z \in y$ gdw. es gibt ein $w \in x$ mit $z \in w$.
6. Teilmenge/Potenzmenge:
   Für jedes x gibt es ein y so, dass für alle z gilt: $z \in y$ gdw. $z \subseteq x$ (für alle w: $w \in z \rightarrow w \in x$).

7. Ersetzung:

   Sei P eine Funktion und x eine Menge. Dann ist der Wertebereich von P beschränkt auf x eine Menge. D.h., es gibt y so, dass für alle z gilt: $z \in y$ gdw. es ein w gibt mit: $P(w) = z$

8. Unendlichkeit:

   Es gibt eine Menge x so, dass es ein z gibt mit $z \in x$ und, wenn $y \in x$, dann $(y \cup \{y\}) \in x$.

9. Auswahl:

   Jede Menge von nicht leeren Mengen hat eine Auswahlfunktion: eine Funktion, die aus jeder der Mengen ein Element bestimmt.

   (nicht allgemein als Axiom akzeptiert, ist aber konsistent mit den übrigen und ermöglicht einfachere Beweise)

Axiomatisierung der Prädikatenlogik

Axiome:

1. $(F \to (G \to F))$
2. $((F \to (G \to H)) \to ((F \to G) \to (F \to H)))$
3. $((\neg F \to \neg G) \to (G \to F))$
4. $(\forall x F(x) \to F(a))$
   $F(x)$ eine Formel, in der x frei vorkommt

Inferenzregeln:

1. Modus Ponens :
   $$\frac{F, F \to G}{G}$$
2. $\dfrac{F \to G(a)}{F \to \forall x G(x)}$, falls a nicht in F vorkommt.

Warum Axiome <u>und</u> Regeln?

Axiome sind gültig!

Regeln ermöglichen, aus bereits erfüllten Formeln weitere herzuleiten.

formale Semantik

Natürlichsprachliche Äusserungen haben nicht prinzipiell einen Wahrheitswert.

Ob “Es schneit” wahr oder falsch ist, hängt von den Umständen ab, unter denen die Äusserung gemacht wird.

Tarski: “Schnee ist weiss” ist genau dann wahr, wenn Schnee weiss ist.

Aufgabe der formalen Semantik:

- Angaben der Bedingungen, unter denen eine natürlichsprachliche Äusserung wahr oder falsch wird.
- Feststellung der Referenzen von Ausdrücken auf Objekte der Welt
- ...

Logiken für die Semantik natürlicher Sprache

Probleme der Merkmalslogik:

Darstellung semantischen Eigenschaften ist gut möglich, geeignete Schlussverfahren (ausser Typenverträglichkeit) stehen aber nicht zur Verfügung

Probleme der Prädikatenlogik:

- Ausdrucksmächtigkeit

  z.B. Gleichheit: kann nicht von Äquivalenzklassenbildung unterschieden werden.
- Wie konstruiert man die Repräsentation?
- Kompositionalität bei Zeit und Modalität: Intension vs. Extension

Intension vs. Extension

Modalausdrücke:

"Peter ist der Bürgermeister"
"Maria will Peter heiraten"

"Maria will den Bürgermeister heiraten" ?

Temporaladverbien:

"Gestern haben wir frischen Fisch gekauft"
"Heute essen wir, was wir gestern gekauft haben"

"Heute essen wir frischen Fisch" ?

Kripke-Semantik

Mögliche-Welten-Semantik:

sei U eine Grundmenge,

P eine Folge/Menge von Weltzuständen (Informations-Zustände, zeitlich geordnet)

Denotat von	im einfachen Modell	im Kripke-Modell
Term:	Element von U	$P \to U$
Formel	Element von $\{0, 1\}$	$P \to \{0, 1\}$

Nebenbedingung: eine einmal wahre Formel bleibt in allen folgenden Zuständen wahr.

Kripke-Semantik, formal

Weltzustände P + zeitliche Ordnung $\leq$:
partiell geordnete Menge, *Kripke-Frame* **P** $= \langle P, \leq \rangle$

Die Interpretation einer Aussage $s \in S$ entspricht derjenigen Teilmenge von P, in der s wahr ist. Die o.a. Nebenbedingung bewirkt, dass solche Teilmengen unter $\leq$ abgeschlossen sind (*Filter* von **P**).

Eine *Kripke-Valuierung* V ist eine Abbildung von der Menge der Aussagen auf die Menge der Filter von **P**,

$$V : S \rightarrow 2^P$$

Ein *Kripke-Modell* ist ein Paar **M** = $\langle \mathbf{P}, V \rangle$.
M $\models_p s$: s ist wahr im Zustand p im Modell **M**, d.h. $p \in V(s)$

$\vee$ und $\wedge$...
M $\models_p \neg s$ gdw. für alle p' mit $p \leq p'$ nicht **M** $\models_{p'} s$
(s kann nie mehr verifiziert werden....)
M $\models_p s \rightarrow s'$ gdw. für alle p' mit $p \leq p'$ gilt:
wenn **M** $\models_{p'} s$, dann **M** $\models_{p'} s'$
(wenn s wahr ist, muss auch weiterhin s' wahr sein...)

Modal- und Temporallogik

Kripke: Mögliche-Welten-Semantik, Abbildung einer Aussage nicht auf die zwei-elementige Menge $\{0, 1\}$, sondern auf eine Folge (oder einen Baum) ihrer Elemente.

Modallogik:

Zwei neue Operatoren:

$\Box$: notwendig , $\Diamond$: möglich

intuitive Bedeutung:

$\Box\, A$ gilt immer in allen Welten,

$\Diamond\, A$ gilt wenigstens einmal in einer Welt (in allen Welten).

Temporallogiken:

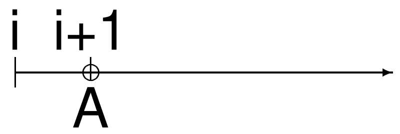
Interpretation der Modaloperatoren als zeitliche (auf die Zukunft bezogene) Operatoren, der Semantik auf diskrete Zeitzustände:

$\Box$: immer, $\Diamond$: irgendwann einmal.

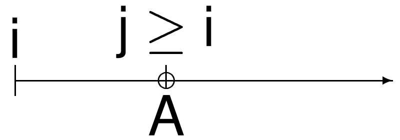
Erweiterung auf Vergangenheit...
weitere Operatoren...

Temporale Operatoren und ihre Semantik

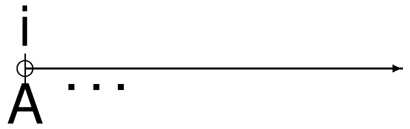
next $\circ A$



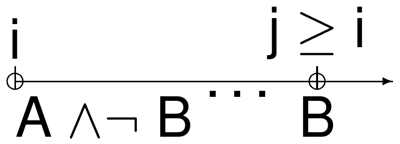
eventually $\Diamond A$



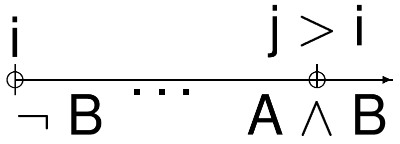
always $\Box A$



until $A \mathcal{U} B$



atnext $A \ \mathcal{N} \ B$



Beschreibungslogiken

Aussagen über einstellige Prädikate (= Konzepte, Menge von Objekten), individuelle Objekte (= Instanzen) und zweistellige Prädikate(= Rollen).

Operatoren zur Definition von Konzepten und Rollen:

Beschr.L.	**Präd.L.**	**Semantik**
C	$C(x)$	$\{d \in \mathcal{D} \mid d \in [\![C]\!]^{\mathcal{I}}\}$
$C \sqcap D$	$C(x) \wedge D(x)$	$[\![C]\!]^{\mathcal{I}} \cap [\![D]\!]^{\mathcal{I}}$
$C \sqcup D$	$C(x) \vee D(x)$	$[\![C]\!]^{\mathcal{I}} \cup [\![D]\!]^{\mathcal{I}}$
$\neg C$	$\neg C(x)$	$\mathcal{D} \setminus [\![C]\!]^{\mathcal{I}}$
$\forall R : C$	$\forall y : R(x,y) \Rightarrow C(y)$	$\{d \in \mathcal{D} \mid [\![R]\!]^{\mathcal{I}}(d) \subseteq [\![C]\!]^{\mathcal{I}}\}$
$\exists R : C$	$\exists y : R(x,y) \wedge C(y)$	$\{d \in \mathcal{D} \mid [\![R]\!]^{\mathcal{I}}(d) \cap [\![C]\!]^{\mathcal{I}} \neq \emptyset\}$
$\exists_{\geq n} R$	$\exists y_{1..n} R(x, y_{1..n})$	$\{d \in \mathcal{D} \mid \lvert [\![R]\!]^{\mathcal{I}}(d) \rvert \geq n\}$
$\exists_{\leq n} R$	$\exists i \exists y_{1..i} \forall y_{i+1} : i \leq n \wedge$ $R(x, y_{1..i}) \wedge \neg R(x, y_{1+i})$	$\{d \in \mathcal{D} \mid \lvert [\![R]\!]^{\mathcal{I}}(d) \rvert \leq n\}$
R	$R(x,y)$	$\{(d,e) \in \mathcal{D} \times \mathcal{D} \mid (d,e) \in [\![R]\!]^{\mathcal{I}}\}$
$R \sqcap S$	$R(x,y) \wedge S(x,y)$	$[\![R]\!]^{\mathcal{I}} \cap [\![S]\!]^{\mathcal{I}}$
R^{-1}	$R(x,y) \Leftrightarrow R^{-1}(y,x)$	$\{(d,d') \in \mathcal{D} \times \mathcal{D} \mid (d',d) \in [\![R]\!]^{\mathcal{I}}\}$

Beschreibungslogiken

Axiome: Aussagen über die Zusammenhänge zwischen Konzepten (T-Box, terminologische Aussagen) und über die Konzeptzugehörigkeit von Objekten (A-box, assertionale Aussagen).

T-Box	
Axiom	Semantik
$D \doteq C$	$[\![D]\!]^{\mathcal{I}} = [\![C]\!]^{\mathcal{I}}$
$D \sqsubseteq C$	$[\![D]\!]^{\mathcal{I}} \subseteq [\![C]\!]^{\mathcal{I}}$
$Q \doteq R$	$[\![Q]\!]^{\mathcal{I}} = [\![R]\!]^{\mathcal{I}}$
$Q \sqsubseteq R$	$[\![Q]\!]^{\mathcal{I}} \subseteq [\![R]\!]^{\mathcal{I}}$
A-box	
Axiom	Semantik
$a : C$	$[\![a]\!]^{\mathcal{I}} \in [\![C]\!]^{\mathcal{I}}$
$a\ R\ b$	$([\![a]\!]^{\mathcal{I}}, [\![b]\!]^{\mathcal{I}}) \in [\![R]\!]^{\mathcal{I}}$

Prädikate, Mengen, Funktionen

Prädikate als Mengen:

$M_P^{\mathcal{A},g} = \{[\![x]\!]^{\mathcal{A},g} \mid [\![P(x)]\!]^{\mathcal{A},g} = 1\}$

Mengen als Funktionen:

charakteristische Funktion f_M

$$f_M : U \to \{0,1\}, f(x) = \begin{cases} 1, & \text{wenn } x \in M \\ 0, & \text{wenn } x \notin M \end{cases}$$

Typtheorie, Motivation

Informal: Unterscheidung zwischen verschiedenen Objekten

z.B.

- Terme und Formeln, Prädikate und Funktionen mit einer, zwei oder mehr Stellen
- 'einfache' Elemente, Mengen
- Relationen in bestimmten Mengen

Sprache: Valenzrahmen von Verben:

schenken, verschenken

glauben, vermuten

Typtheorie, formal

Die Menge der *Typen* ist definiert als:

1. e ist ein Typ (Objekte)
2. t ist ein Typ (Wahrheitswerte)
3. wenn a und b Typen sind, dann auch $\langle a, b\rangle$

Die Definition kann auf n-Tupel ausgedehnt werden, dann ist es aber sinnvoller, für Funktionen einen zusätzlichen zweistelligen Typoperator einzusetzen: $(a{\rightarrow}b)$. Damit können Funktionen als spezielle Paare unterschieden werden.

Eine typisierte (formale) Sprache ist eine Sprache, bei der jedem wohlgeformten Ausdruck ein Typ zugewiesen werden kann.

D_τ bezeichnet die Menge der Denotate der Ausdrücke vom Typ τ.

Sei U eine Grundmenge, dann:

$D_e = U, D_t = \{0, 1\}$ und $D_{\langle a,b\rangle} = \{f \mid f : D_a{\rightarrow}D_b\}$

Oder, bei n-Tupeln:

$D_e = U, D_t = \{0, 1\}$, $D_{\langle a,b\rangle} = \{p \mid p \in D_a \times D_b\}$ und $D_{a\rightarrow b} = \{f \mid f : D_a{\rightarrow}D_b\}$

Beispiele: prädikatenlogische Formalisierung

D_e = $\{$ *Peter, Maria, ein Buch* $\}$,
Prädikate *Schenken* und *Verschenken*.

Peter schnarcht

(0) *Schnarchen*(*Peter*)

Peter verschenkt ein Buch ...

(1) *Verschenken*(*Peter*, *ein Buch*)

Maria schenkt Peter ein Buch...

(2) *Schenken*(*Maria*, *Peter*, *ein Buch*)

Typ von *Schnarchen*: $\langle e, t\rangle$

Typ von *Verschenken*: $\langle e, \langle e, t\rangle\rangle$

von *Schenken*?

Maria vermutet, dass Peter ihr ein Buch schenkt...

Maria vermutet stark, dass Peter ihr ein Buch schenkt...

Maria vermutet sehr stark, dass Peter ihr ein Buch schenkt...

Beispiele

Mengentheorie:
D_e: Menge aller Elemente und Teilmengen einer Menge M

- x : e
- $x \subset y$: t
- $\{x \mid x \in y\}$: e
- $\{x \mid x \subset y\}$: e
- $x \times y$: $\langle e, e \rangle$
- $x \in y$: t
- $f : x \rightarrow y$: $\langle e, e \rangle$

Typregel

Funktionsapplikation:
Sei f vom Typ $(\alpha\rightarrow\beta)$ und a vom Typ α.
Dann ist $f(a)$ vom Typ β.

Beispiele: $D_e = \mathcal{N}$ (die Menge der natürlichen Zahlen)
Sei $f, f(x) = x^2$ ... Typ von f : $\langle e, e\rangle$
Sei $f, f(x) = (x^3 \leq 0)$ ... Typ von f : $\langle e, t\rangle$

Lambda-Operator

Church (1941): Notation für Definition und Anwendung von Funktionen.

Statt “sei $f : A \to B$ mit $f(x) = x^2 + 3$ ”:

(Abstraktion) $\lambda x.x^2 + 3$

Alternative Schreibweisen: $\lambda x(x^2 + 3), \lambda x\ x^2 + 3$

Anwendung: statt f(4) (und Def. wie oben) Auswertung, Ersetzung der Variable durch das Argument

(λ-Konversion) $(\lambda x.x^2 + 3)(4) \equiv 4^2 + 3 = 19$

Der λ-Operator bindet eine Variable wie ein Quantor!

Lambda-Abstraktion, formal

Syntax

Sei x eine Variable vom Typ α,

u ein Ausdruck vom Typ β, in dem x nicht gebunden vorkommt.

Dann ist $\lambda x\ (u)$ ein Ausdruck vom Typ $(\alpha{\rightarrow}\beta)$.

Semantik

Sei x eine Variable vom Typ α, u ein Ausdruck vom Typ β, in dem x nicht gebunden vorkommt,

dann ist $[\![\lambda x\ (u)]\!]^{\mathcal{M},h}$ eine Funktion von D_α nach D_β mit:

für alle k in D_α: $[\![\lambda x\ (u)]\!]^{\mathcal{M},h}\ (k) = [\![u]\!]^{\mathcal{M},h'}$

und h' = h bis auf $h'(x)$, $h'(x) = k$

Lambda-Abstraktion, Beispiel

Denotation von Verben:

Bruno raucht. ... *Rauchen(Bruno)*

$\forall k : [\![\lambda x(\textbf{\textit{Rauchen}}(x))]\!]^{\mathcal{M},h}(k) =$
$[\![\textbf{\textit{Rauchen}}(x)]\!]^{\mathcal{M},h'}, h'(x) = k$

Sei $\mathcal{M} = \langle U, I \rangle$ mit:
$U = \{$***Anna, Bruno, Clara***$\}$,
$I($***Rauchen***$) = \{$Anna$\}$

$[\![\lambda x\ (\textbf{\textit{Rauchen}}(x))]\!]^{\mathcal{M},h}\ ([\![Bruno]\!]^{\mathcal{M},h}) = 0$

Die Sprache TL, Syntax

Typisierte Logik: versammelt Prädikatenlogik, Typen, λ-Operator und Identität:

Symbole:

- Operatoren $\wedge, \vee, \neg$
- Quantoren $\forall, \exists$
- zu jedem Typ τ eine Menge von Variablen V_τ
- zu jedem Typ τ eine Menge von Konstanten K_τ
- der Lambda-Operator λ
- das Gleichheitssymbol $=$
- Man nehme Klammern nach Belieben: (,), [,], ...

Die Menge der <u>Typen</u>, T:

- $e \in T$ und $t \in T$
- wenn a und b in T sind, dann auch $\langle a, b\rangle$

Die Sprache TL, Syntax, Ausdrücke

Die Menge der Ausdrücke vom Typ τ, E_τ:
Für alle Typen a, b

- Alle Konstanten und Variablen vom Typ τ sind Elemente von E_τ.
- Wenn $x \in E_a$ und $u \in E_b$, dann ist $\lambda x\, u$ in $E_{a \longrightarrow b}$.
- Wenn $u \in E_{a \longrightarrow b}$ und $x \in E_a$, dann ist $u(x)$ in E_b.
- Wenn ϕ und ψ in E_t sind (also Formeln), dann auch: $\neg\phi, \phi \vee \psi, \phi \wedge \psi$.
- Wenn $\phi \in E_t$ und x in V_a ist, dann sind $\forall x \phi$ und $\exists x \phi$ in E_t.
- Wenn u und v in E_a sind, dann ist $u = v$ in E_t.

Viel aussagekräftiger als Prädikatenlogik: Prädikate und Funktionen höherer Ordnung, Quantifikation über beliebige Ausdrücke...

Die Sprache TL, Semantik

Typen

Sei U eine Grundmenge, die "Diskursdomäne". Die möglichen Denotate von Ausdrücken der Typen sind:

- $D_e = U$
- $D_t = \{0, 1\}$
- $D_{\langle a,b \rangle} = \{f \mid f : D_a {\rightarrow} D_b\}$

Ausdrücke

Tatsächliche Denotate von Ausdrücken in einem

Model $\mathcal{M} = \langle U, F \rangle$ mit einer Variablenbelegung h.

F ersetzt hier I und ist "nur" eine Belegung der Konstanten, keine Interpretation von Prädikaten- und Funktionssymbolen mehr (die werden mit Hilfe des Lambda-Operators "anonym" definiert)

TL, Semantik von Ausdrücken

Sei exp ein Ausdruck in TL, dann ist das Denotat von exp bez. $\mathcal{M}$ und h, $[\![exp]\!]^{\mathcal{M},h}$ wie folgt definiert:

- Sei c eine nicht-logische Konstante, dann:
  $[\![c]\!]^{\mathcal{M},h} = F(c)$.
- Sei x eine Variable, dann:
  $[\![x]\!]^{\mathcal{M},h} = g(x)$.
- (Abstraktion)
  Sei $x \in E_a$, $u \in E_b$, x nicht gebunden in u,
  dann ist $[\![\lambda x\ u]\!]^{\mathcal{M},h}$ eine Funktion von D_a nach D_b mit:
  für alle k in D_a: $[\![\lambda x\ u]\!]^{\mathcal{M},h}(k) = [\![u]\!]^{\mathcal{M},h'}$
  und h' = h bis auf $h'(x)$, $h'(x) = k$.
- (Konversion)
  Sei $u \in E_{a \rightarrow b}$, $x \in E_a$, dann:
  $[\![u(x)]\!]^{\mathcal{M},h} = [\![u]\!]^{\mathcal{M},h}([\![x]\!]^{\mathcal{M},h})$.

TL, Semantik von Ausdrücken II

- Seien ϕ und ψ in E_t, dann:
  $[\![\neg\phi]\!]^{\mathcal{M},h} = \neg[\![\phi]\!]^{\mathcal{M},h}$
  $[\![\phi \wedge \psi]\!]^{\mathcal{M},h} = [\![\phi]\!]^{\mathcal{M},h} \wedge [\![\psi]\!]^{\mathcal{M},h}$
  $[\![\phi \vee \psi]\!]^{\mathcal{M},h} = [\![\phi]\!]^{\mathcal{M},h} \vee [\![\psi]\!]^{\mathcal{M},h}$.
- Sei $\phi \in E_t$ und $x \in E_a$, dann:
  $[\![\forall x\phi]\!]^{\mathcal{M},h} = 1$ gdw. für alle Variablenbelegungen g, die ausser auf x mit h identisch sind, gilt:
  $[\![\phi]\!]^{\mathcal{M},g} = 1$.
  und:
  $[\![\exists x\phi]\!]^{\mathcal{M},h} = 1$ gdw. für mindestens eine Variablenbelegungen g, die ausser auf x mit h identisch sind, gilt:
  $[\![\phi]\!]^{\mathcal{M},g} = 1$.
- Seien u und v in E_a, dann:
  $[\![u = v]\!]^{\mathcal{M},g} = 1$ gdw. $[\![u]\!]^{\mathcal{M},g} = [\![v]\!]^{\mathcal{M},g}$.

Lambda-Kalkül, Λ

Weitere Anwendung des Lambda-Operators: das Lambda-Kalkül. Unterschied zu TL: keine logischen Operatoren, nicht typisiert; dafür Axiomatisierung.

Syntax von Λ:

Vokabular: Variablen x, y, λ, Klammern;

syntaktische Regeln:

- $x \in \Lambda$
- Wenn $M \in \Lambda$, dann auch $(\lambda x M)$.
- Wenn M, N in Λ, dann auch (MN).
- Wenn M, N in Λ, dann ist $M = N$ eine Formel.

Axiomatisierung

- $(\lambda x M)N = M'$, M' ist M, worin jedes Auftreten von x durch N ersetzt ist.
- $=$ ist reflexiv, symmetrisch und transitiv.
- Wenn $M = N$, dann $MZ = NZ$, $ZM = ZN$ und $(\lambda x M) = (\lambda x N)$

Anwendungen: Grundlage der Funktionstheorie, Charakterisierung von Berechenbarkeit, Syntax (Lisp) und Semantik von Programmiersprachen etc. Auch die Prädikatenlogik kann komplett im Lambda-Kalkül definiert werden...

Linguistische Anwendungen

Kompositionale Analyse von formalen Ausdrücken höherer Ordnung
$\sim$ kompositionale Semantik natürlicher Sprache.

Beispiele:

Denotation von Verben:

intransitiv, Abb. des Subjekts auf einen Wahrheitswert:

$$\lambda P[\lambda x[P(x)]]$$

transitiv, Abb. von Subjekt und Objekt auf Ww.:

$$\lambda y \lambda x[\lambda P[P(x)]](y), \text{ oder:}$$

$$\lambda y \lambda x[\lambda P[P(x, y)]]$$

Adverben - gesuchte Funktion muss ein Verb-Denotat auf ein Verb-Denotat abbilden:

Bsp:

$$\lambda P[\lambda x[\text{schnell}(P(x))]]$$

allgemein:

$$\lambda Q[\lambda P[\lambda x[Q(P(x))]]]$$

Weitere Beispiele: Lohnstein, [PMW]

Montague: Intensionale Logik

Kombination von allen bisher eingeführten Konzepten:

- Aussagenlogik: logische Operatoren, Wahrheitswerte
- Prädikatenlogik: Terme, Quantoren
- Modallogik: mögliche Welten, Informationszustände; Operatoren $\Box$ (notwendig), $\Diamond$ (möglich)
- Temporallogik: zeitlicher Verlauf, Zeitpunkte; Operatoren **P** (*Past*), **F** (*Future*)
- Lambda-Operator: Abstraktion über Variablen beliebigen Typs
- Unterscheidung zwischen Intension und Extension

(Formalisierung siehe Lohnstein oder [PMW])

Modelle dazu: Bäume von "einfachen Modellen", die mit Indizes $\langle w, t\rangle$ für die "mögliche Welt" w und den Zeitpunkt t versehen sind.

Intension und Extension

Frege: Sinn und Bedeutung

Der Abendstern, der Morgenstern vs. die Venus ...

Die Königin von England vs. Elisabeth, Victoria, ...

Intension als Funktion von Indizes
($\langle w, t \rangle$, = mögliche Welt und Zeitpunkt)
auf Menge von Extensionen (Individuen).

Neue Operatoren: Intensor $^\wedge$:

Sei α ein Ausdruck mit dem Denotat $[\![\alpha]\!]^{\mathcal{M},w,t,g}$.

Dann ist am Index $\langle w, t \rangle$ $[\![{}^\wedge\alpha]\!]^{\mathcal{M},w,t,g}$ die Intension von α

Extensor $^\vee$: Umkehrfunktion von $^\wedge$:

$[\![{}^{\vee\wedge}\alpha]\!]^{\mathcal{M},w,t,g} = [\![\alpha]\!]^{\mathcal{M},w,t,g}$

Nicht umgekehrt: Extension gibt es nur zu bestimmtem Index, Intension ermöglicht, an jedem Index die Extension zu bestimmen.

zusätzlicher Typ s:

Sei X vom Typ a, dann ist $^\wedge X$ vom Typ $\langle s, a \rangle$.

Sei X vom Typ $\langle s, a \rangle$, dann ist $^\vee X$ vom Typ a.

Aufgabe:

Versuchen Sie, Ausdrcke der Intensionalen Logik fr folgende zwei Stze zu finden. Geben Sie ein Modell dafr an.

Es besteht die entfernte Chance, dass sich nichts von allem, was folgt, je ereignet hat;

$\lambda x[\text{bestehen}(x)](\lambda y[\text{entfernt}(y) \wedge \text{Chance}(y)](C))$

mit $C = \lambda z[\forall z \neg \mathbf{P}(\text{ereignen}(z) \wedge \text{folgen}(z))]$

viel wahrscheinlicher ist jedoch, dass es sich, falls es sich je ereignet haben sollte, nie wieder ereignen wird.

(Peter Ustinov, Der Alte Mann und Mr. Smith)

$\lambda v[\text{wahrscheinlicher}(v, C)]$

$(\lambda u[\Diamond \mathbf{P}\text{ereignen}(u) \rightarrow \neg\, \mathbf{F}\text{ereignen}(u)])$

Probleme:

z und u können noch nicht bestimmt werden, müssen aber identisch sein (sollten vielleicht als eine Variable abstrahiert werden).

'viel' ist noch nicht integriert.

Die inhaltliche Interpretation würde eine Modellierung von 'entfernte Chance' und 'viel wahrscheinlicher' erfordern, so dass entschieden werden kann, was davon wirklich grösser ist.

Axiomatisierung von Stringkonkatenation

Alphabet: endliche, nicht-leere Menge von Symbolen.

String, Zeichenkette: Folge von Symbolen

Konkatenation: Aneinanderreihung, zweistellige Operation "," auf einer Menge A von Zeichenketten.

ϵ: Zeichenkette der Länge 0.

- Axiomatisierung ohne leeren String (Halbgruppe)
  1. Abgeschlossenheit:
     $\forall x \forall y((x \in A \wedge y \in A) \rightarrow x, y \in A)$
  2. Assoziativität:
     $\forall x \forall y \forall z((x, y), z = x, (y, z))$

  Beispiel für ein Modell dafür: $A = \{z \mid z$ besteht aus einer geraden Anzahl von 'a's und / oder 'b's$\}$.
- Axiomatisierung mit leerem String (Monoid)
  1. Abgeschlossenheit:
     $\forall x \forall y((x \in A \wedge y \in A) \rightarrow x, y \in A)$
  2. Assoziativität:
     $\forall x \forall y \forall z((x, y), z = x, (y, z))$
  3. neutrales Element:
     $\exists e \forall x(x, e = e, x = x)$

Schreibweise: statt x, y auch $x \frown y$ oder nur xy.

Wörter über Alphabeten

Sei Σ ein Alphabet, dann ist Σ^* die Menge aller endlichen *Wörter*, die sich durch Aneinanderreihung der Elemente von Σ bilden lassen, die Menge der Wörter über Σ.

Dazu gehört auch das *leere Wort*, ϵ.

$\Sigma^+ := \Sigma^* \backslash \{\epsilon\}$ ist die Menge der nicht-leeren Wörter über Σ.

Bsp: $\Sigma = \{a, b\}$. $\Sigma^* = \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, ...\}$.

Die *Länge* eines Wortes w, $\mid w \mid$, ist die Anzahl der Zeichen, aus denen es besteht.

Sei a ein Element eines Alphabets. a^n bezeichnet das Wort der Länge n aus lauter as.

Eine (formale) *Sprache* ist eine Menge von Wörtern.
Bsp: $L = \{a^n b^n \mid n \geq 1\}$

Grammatik

Definition:

Eine (*Phrasenstruktur-*)*Grammatik* G ist ein 4-Tupel,

$$(V, \Sigma, P, S)$$

mit folgenden Eigenschaften:

V ist eine endliche Menge, die *Variablen*, oder Nicht-Terminalsymbole.

Σ ist ein Alphabet, das *Terminalalphabet*, die Terminalsymbole. $V \cap \Sigma = \emptyset$.

P ist eine endliche Menge von *Regeln, Produktionen* der Form $u \to v$ mit $u \in ((\Sigma \cup V)^* \times V \times (\Sigma \cup V)^*)$ und $v \in (\Sigma \cup V)^*$, d.h. ein Wort, das mindestens aus einem Nicht-Terminalsymbol besteht, kann durch ein beliebiges Wort ersetzt werden.

$S \in V$ ist das *Startsymbol*, ein besonderes Element von V.

Eine Grammatik ist ein deduktives System: Axiome und Inferenzregeln. Das einzige Axiom einer Grammatik ist das Startsymbol, die Regeln sind die Produktionen.

formale Sprache

$\Rightarrow_G$ ist die Relation in $(V \cup \Sigma)^*$ mit:
$u \Rightarrow_G v$ gdw. $u = xyz, v = xy'z$ mit $x, z, y, y' \in (V \cup \Sigma)^*$ und $y \to y' \in P$.
"u geht unter G unmittelbar in v über "

$\Rightarrow_G^*$ ist die *reflexive und transitive Hülle* von $\Rightarrow_G$.
$x \Rightarrow_G^* y$, wenn gilt:
$x = y$, $x \Rightarrow_G y$ oder es gibt ein z mit $x \Rightarrow_G^* z$ und $z \Rightarrow_G y$.

Eine Grammatik G *erzeugt* ein Wort w, wenn gilt:

$$S \Rightarrow_G^* w$$

Die Sprache $L(G)$ ist die Menge der Wörter, die von der Grammatik G erzeugt werden.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow_G^* w\}$$

Eine Folge von Wörtern $(w_0, w_1, ..., w_n)$ mit $w_0 = S$, $w_n \in \Sigma^*$ und $w_0 \Rightarrow w_1 \Rightarrow ... \Rightarrow w_n$ heisst Ableitung von w_n.

Beispiel

$G = (V, \Sigma, P, S)$

$V = \{$ S, NP, VP, T-V, I-V, D, N $\}$

$\Sigma = \{$ [the], [cat], [dog], [sleeps], [chases] $\}$

$P = \{$
S $\rightarrow$ NP VP,
NP $\rightarrow$ D N,
VP $\rightarrow$ T-V NP,
VP $\rightarrow$ I-V,
N $\rightarrow$ [dog],
N $\rightarrow$ [cat],
I-V $\rightarrow$ [sleeps],
T-V $\rightarrow$ [chases],
D $\rightarrow$ [the],
$\}$

$S =$ S.

Erkennungsproblem

Problem

Ist ein gegebenes Objekt Element einer Menge?

Ein Problem ist *entscheidbar*, wenn in einer endlichen Anzahl von Berechnungsschritten festgestellt werden kann, ob eine Eingabe zur Menge gehört oder nicht.

Ein Problem ist *semi-entscheidbar*, wenn in einer endlichen Anzahl von Berechnungsschritten festgestellt werden kann, ob eine Eingabe zur Menge gehört, oder wenn in einer endlichen Anzahl von Berechnungsschritten festgestellt werden kann, ob eine Eingabe nicht zur Menge gehört!

Erkennungsproblem oder *Wortproblem*

Ist ein Wort w Element einer Sprache L?

Bäume

Ein Baum besteht aus *Knoten*, $n \in N$, und *Kanten*, $e \in E, E \subset N \times N$, wobei jeder Knoten maximal einen Vorgänger und beliebig, aber endlich viele Nachfolger hat:

Für alle $n \in N$: es gibt maximal ein $m \in N$ mit $\langle m, n \rangle \in E$.

Ausserdem ist E irreflexiv und intransitiv, d.h. es darf keine zirkuläre Folge von Paaren in E geben.

Ein Knoten ohne Vorgänger heisst *Wurzel*, ohne Nachfolger *Blatt* des Baumes.

Ableitungen in Grammatiken, bei deren Regeln immer nur ein einzelnes Nichtterminalsymbol auf der linken Seite steht, kann man einen *Syntaxbaum* zuordnen.

Die Wurzel ist dann mit dem Startsymbol, die inneren Knoten sind mit den verwendeten Nichtterminalsymbolen, die Blätter mit den Terminalsymbolen beschriftet.

Beispiel

(Links-)Ableitung (d.h., das jeweils am weitesten links stehende Nichtterminal wird ersetzt):

S ⇒ NP VP ⇒ D N VP ⇒ [the] N VP ⇒

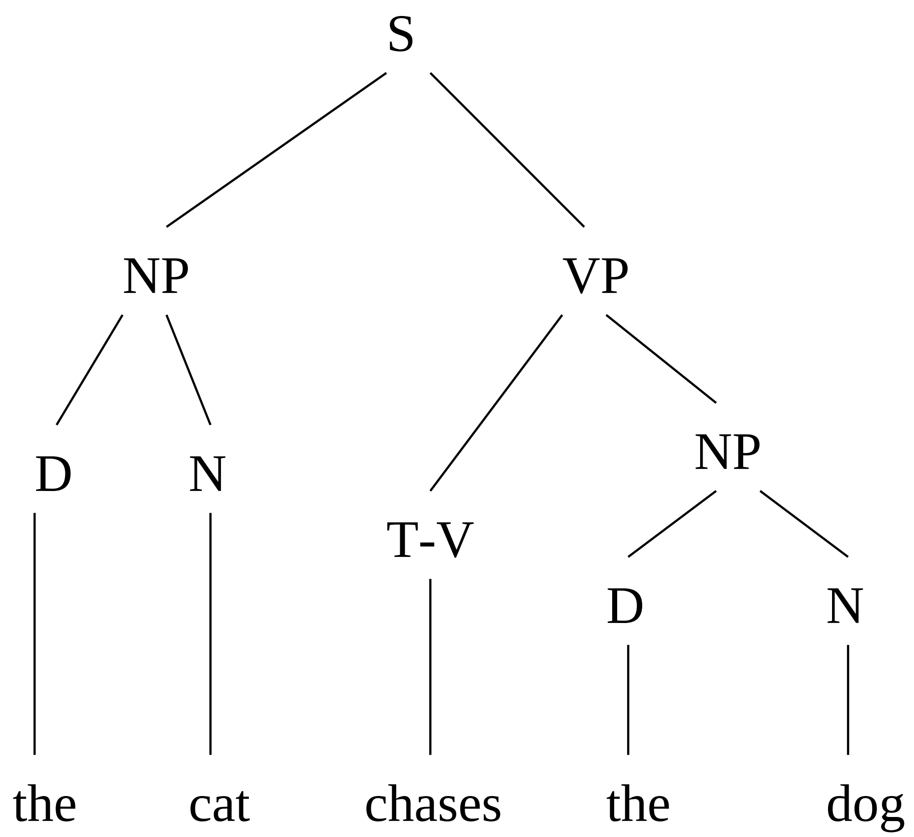
[the] [cat] VP ⇒ [the] [cat] T-V NP ⇒

[the] [cat] [chases] NP ⇒

[the] [cat] [chases] D N ⇒

[the] [cat] [chases] [the] N ⇒

[the] [cat] [chases] [the] [dog]



Syntaxbäume

Auch eine andere Ableitung (z.B. mit NP VP $\Rightarrow$ NP T-V NP) hätte denselben Syntxbaum ergeben...

Es gilt:

$x \in L(G)$,

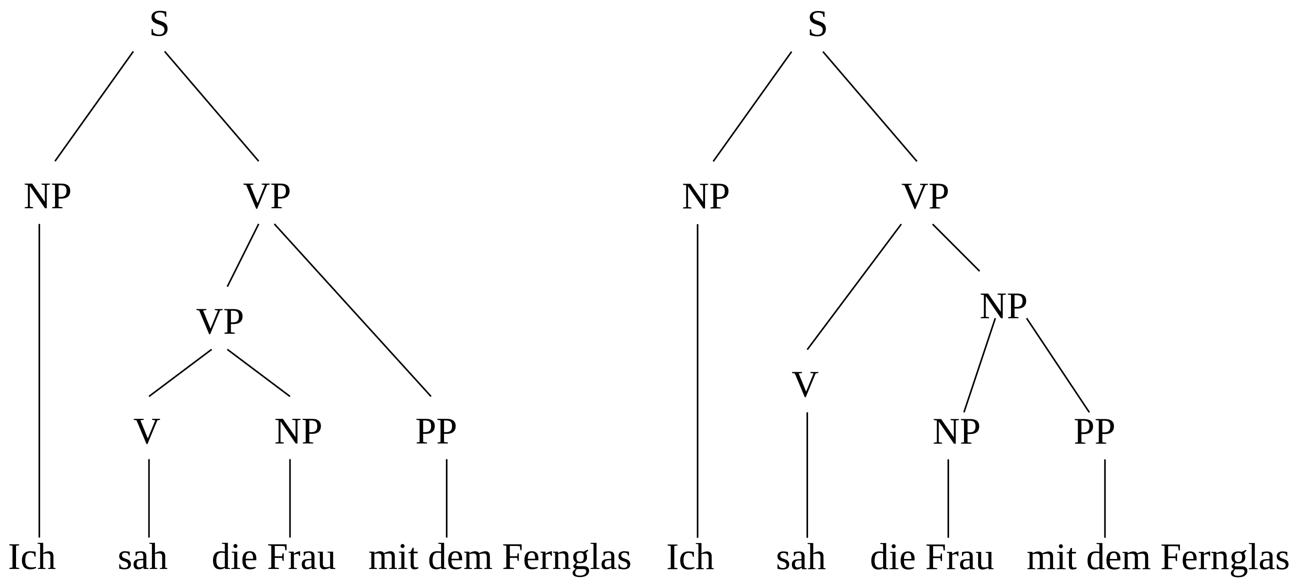
gdw. es gibt einen Syntaxbaum mit x an den Blättern,

gdw. es gibt eine Linksableitung für x

mehrdeutige Grammatik: für dasselbe Wort gibt es mehrere Syntaxbäume.

Beispiel:

G mit $P = \{$ S $\rightarrow$NP VP, NP $\rightarrow$NP PP, NP $\rightarrow$D N, VP $\rightarrow$VP PP, VP $\rightarrow$V NP $\}$.



Chomsky-Hierarchie

Je nach Form der Regeln lassen sich Grammatiken einem (maximalen) *Typ* zuordnen:

Typ 0:

Allgemeine Phrasenstrukturgrammatik, keinerlei Beschränkungen

Typ 1:

kontextsensitive Grammatik:

Für alle Regeln $w_1 \to w_2$ in P gilt: $\mid w_1 \mid \; \leq \; \mid w_2 \mid$.

Typ 2

kontextfreie Grammatik:

Für alle Regeln $w_1 \to w_2$ in P gilt, dass w_1 ein einzelnes Nichtterminal ist, d.h. $w_1 \in V$.

Typ 3

reguläre Grammatik:

Für alle Regeln $w_1 \to w_2$ in P gilt, dass w_2 ein einziges Terminal oder ein Terminal, gefolgt von einem Nichtterminal ist, d.h., $w_2 \in \Sigma \cup (\Sigma \times V)$

Eine Sprache ist vom Typ X, wenn sie von einer Grammatik vom Typ X erzeugt wird.

Es gilt: Typ 3 $\subset$ Typ 2 $\subset$ Typ 1 $\subset$ Typ 0.

Beispiele

Typ 1

$$L = \{a^n b^n c^n \mid n \leq 1\}$$

Typ 2

$$L = \{a^n b^n \mid n \leq 1\}$$

Typ 3

$$L = \{a^n \mid n > 1\}$$

Beispiel: arithmetische Ausdrücke:

$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, \times, (,)\}$ ...

Grammatik dazu:

P = $\{$ S $\rightarrow$T, T $\rightarrow$(T + T), T $\rightarrow$T $\times$ T,

T $\rightarrow$Z, Z $\rightarrow$$Z_1$ $\mid$ Z Z_0, Z_0 $\rightarrow$$Z_1$ $\mid$ 0,

Z_1 $\rightarrow$1 $\mid$ 2 $\mid$ 3 $\mid$ 4 $\mid$ 5 $\mid$ 6 $\mid$ 7 $\mid$ 8 $\mid$ 9$\}$.

Automaten

Ein Automat ist eine abstrakte Berechnungsmaschine, mit einer Menge von Eingaben und einem Verhalten dazu.

Eingaben: diskret (nicht kontinuierlich), Folgen von Zeichen eines Alphabets.

Innere Struktur: endliche Menge von *Zuständen*.

Ausgabe: Annahme oder Ablehnung einer Eingabe.

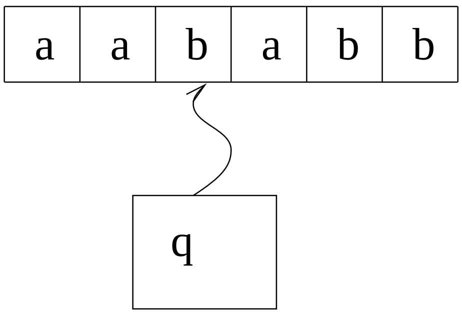
Verhältnis von Automaten und formalen Sprachen:

Automat kann als Akzeptant einer Sprache (und als ”Ablehner” ihres Komplements) betrachtet werden.

Endliche Automaten, DFAs

Ein *deterministischer endlicher Automat* (*deterministic finite Automaton*, DFA) liest eine endliche Folge von Zeichen nacheinander von links nach rechts ein, hält nach dem letzten Zeichen an und gibt "akzeptiert" oder "abgelehnt" aus.

Zu jedem Zeitpunkt befindet sich der Automat in einem von einer endlichen Anzahl von Zuständen. Es gibt einen Startzustand und eine Menge von Endzuständen.



Berechnungsschritte: wenn in einem bestimmten Zustand ein bestimmtes Zeichen gelesen wird, wird in einen bestimmten Zustand übergegangen.
Bsp: (q_1, a, q_2)

Endliche Automaten

Definition:

Ein *deterministischer endlicher Automat* ist ein 5-Tupel $(K, \Sigma, \delta, q_0, F)$ mit:

K eine endliche Menge von Zuständen,

Σ ein Alphabet,

δ eine Funktion von $K \times \Sigma$ nach K, die Zustandsübergangsfunktion,

$q_0 \in K$ der Startzustand,

$F \subseteq K$ die Endzustände.

Beispiel:

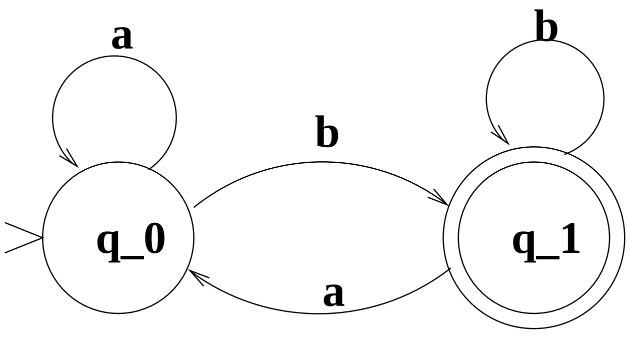
DFA mit Zuständen $\{q_0, q_1\}$,

Zustandsübergangsfunktion

$\delta(q_0, a) = q_0, \delta(q_0, b) = q_1, \delta(q_1, a) = q_0, \delta(q_1, b) = q_1,$

Startzustand q_0, Endzustand q_1.

Zustandsübergangsdiagramm:



Endliche Automaten

Situation eines Automaten: ”Schnappschuss”,

Position des Einlesekopfes auf der Eingabe und Zustand des Automatens.

Sei $M = (K, \Sigma, \delta, q_0, F)$. Eine Situation ist ein Tripel (x, q, y), wobei $q \in K$ und $x, y \in \Sigma^*$

So definierte Situationen müssen nicht erreichbar sein...

Relation ”erzeugt in einem Schritt” zwischen Situationen, $\vdash$:

$(x, q, y) \vdash (x', q', y')$ gdw. es gibt ein $\sigma \in \Sigma$ mit $y = \sigma y'$, $x' = x\sigma$ und $\delta(q, \sigma) = q'$.

$\vdash^*$ reflexive, transitive Hülle dazu: $s \vdash^* s'$, wenn s in keinem oder mehr Schritten s' erzeugt.

Nicht-Deterministische Automaten, NFAs:

statt Funktion δ Relation Δ.

Akzeptanz einer Sprache: wenn <u>ein</u> Pfad im Zustandübergangsdiagramm existiert...

Beide Arten endlicher Automaten akzeptieren dieselbe Klasse von Sprachen, nämlich die regulären Sprachen...

reguläre Sprachen

Def.: Seien A und B Mengen von Zeichenketten. Dann heisst AB Konkatenation von A und B, wenn gilt: $AB = \{x, y \mid x \in A$ und $y \in B\}$

Definition: *reguläre Sprache*

Sei Σ ein Alphabet. Dann gilt:

- $\emptyset$ ist eine reguläre Sprache.
- Für alle $x \in \Sigma^*$ ist $\{x\}$ eine reguläre Sprache.
- Wenn A und B reguläre Sprachen sind, dann auch A^*, $A \cup B$ und AB.

reguläre Ausdrücke

Reguläre Ausdrücke sind eine Meta-Notation für Wörter.

Definition: *reguläre Ausdrücke*

Sei Σ ein Alphabet. Dann gilt:

- $\emptyset$ ist ein regulärer Ausdruck.
- ϵ ist ein regulärer Ausdruck.
- Für alle $a \in \Sigma^*$ ist a ein regulärer Ausdruck.
- Wenn α und β reguläre Ausdrücke sind, dann auch $(\alpha)^*$, $(\alpha \mid \beta)$ und $\alpha\beta$.

$(\alpha \mid \beta)$ entspricht: " entweder ein Ausdruck α oder ein Ausdruck β".

Bsp: die Sprache, die der Beispiel-DFA akzeptiert: $L = (a \mid b)^* b$

Es gilt: Die Sprachen, die durch

- eine Typ-3-Grammatik beschrieben,
- einen endlichen Automaten (deterministisch oder nicht) akzeptiert
- und einen regulären Ausdruck beschrieben

werden, gehören derselben Klasse an.

Pumping-Lema für reguläre Sprachen

Sei L eine reguläre Sprache. Dann gibt es eine Zahl n, so dass gilt: alle Wörter $x \in L$ mit $\mid x \mid\ > 0$ lassen sich zerlegen in $x = uvw$ mit:

- $\mid v \mid\ \geq 1$
- $\mid uv \mid\ \leq n$
- für alle $i = 0, 1, 2, ...$ gilt: $uv^i w \in L$.

<u>Beweis</u>:

Es gibt einen Automaten M, der L akzeptiert.

Sei n die Anzahl der Zustände in M.

Bei Verarbeitung von x mit $\mid x \mid\ \geq n$ durchläuft M $\mid x \mid + 1$ Zustände.

Diese können nicht alle verschieden sein (es gibt nur n Zustände).

Also durchläuft M eine Schleife.

Wähle $x = uvw$ so, dass der Zustand nach u und uv derselbe ist.

Wenn also $uvw \in L$, dann auch $uw = uv^0 w$, $uv^1 w$,$uv^2 w$ etc.

Anwendung:

Das Pumping-Lemma ist zu gebrauchen, um zu zeigen, dass eine Sprache <u>nicht</u> regulär ist:

Bsp.: $L = \{a^m b^m \mid m \geq 1\}$.

Annahme: L sei regulär.

Es gibt n mit den Eigenschaften wie oben.

Man nehme das Wort $x = a^n b^n$ der Länge $2n$.

Die Zerlegung uvw muss erfüllen, dass v nicht leer ist, und nur aus as besteht.

Dann müsste sowohl v aus dem Wort x entfernt werden können, als auch beliebig oft eingefügt werden.

Es gilt aber: $m + \mid v \mid \neq m$.

Also war die Annahme falsch.

Eigenschaften regulärer Sprachen

Die Klasse ist *abgeschlossen* unter

- Vereinigung
- Komplement
- Schnitt
- Konkatenation
- Stern (Kleene)

Natürliche Sprachen $\subseteq$ Reguläre Sprachen?

Pumpinglemma...

kontextfreie Sprachen: Normalformen

Zur Erinnerung: Kontextfreie Sprachen sind diejenigen, die von Grammatiken erzeugt werden, die auf allen linken Regelseiten nur je ein Nichtterminalsymbol haben.

Aufgrund der Bedingung für Sprachen vom Typ 1, die linke Seite jeder Regel muss kürzer sein als die rechte, kann in diesen Sprachen ϵ nicht abgeleitet werden.

Es gilt aber: jede kontextfreie Grammatik G, die Regeln der Form $A \rightarrow \epsilon$ enthält, kann in eine äquivalente Grammatik G' ohne solche Regeln umgeformt werden, so dass gilt: $L(G) = L(G') \cup \{\epsilon\}$.

Dazu wird V zerlegt in $V_1 = \{A \mid A \Rightarrow^* \epsilon\}$ und $V_0 = V \backslash V_1$.

Dann werden alle Regeln der Form $A \rightarrow \epsilon$ aus P entfernt und für alle Regeln der Form $B \rightarrow xAy$ die Regel $B \rightarrow xy$ eingefügt.

kontextfreie Sprachen: Normalformen

<u>Definition</u>: Eine epsilon-freie, kontextfreie Grammatik G ist in *Chomsky-Normalform, CNF*, wenn alle Regeln entweder die Form $A \rightarrow BC$ oder $A \rightarrow a$ haben.

Zu jeder kf G gibt es eine äquivalente in CNF:

- Eliminierung von Regeln der Form $A \rightarrow B$:

  Falls es eine Menge von Nichtterminalen gibt mit $B_1 \rightarrow B_2, B_2 \rightarrow ..., B_k \rightarrow B1$, ersetze alle B_i durch B.

  Dann Nummerierung der Nichtterminalen so, dass gilt: aus $A_i \rightarrow A_j$ folgt $i < j$, also $V = \{A_1, A_2, ...A_n\}$. Jetzt wird diese Folge von Nichtterminalen von hinten abgearbeitet: für $k = n - 1, ...1$ werden alle Regeln der Form $A_k \rightarrow A_{k'}, k' > k$ eliminiert; für die Regeln $A_{k'} \rightarrow x_1 \mid ... \mid x_l$ werden die Regeln $A_k \rightarrow x_1 \mid ... \mid x_l$ hinzugefügt.

- Eliminierung von Regeln der Form $A \rightarrow xay$:

  Für jedes Terminal a wird ein neues Nichtterminal B sowie einer Regel $B \rightarrow a$ eingefügt. Dann wird jedes Vorkommen von a auf einer rechten Seite (länger als 1) durch B ersetzt.

- bleibt noch: Regeln der Form $A \rightarrow B_1 B_2 ... B_k, k > 2$:

  Für jede dieser Regeln werden neue Variablen $C_2, ..., C_{k-1}$ und Regeln $A \rightarrow B_1 C_2, C_2 \rightarrow B_2 C_3, ...,$ $C_{k-1} \rightarrow B_{k-1} B_k$ eingefügt.

kontextfreie Sprachen: Kellerautomaten, PDAs

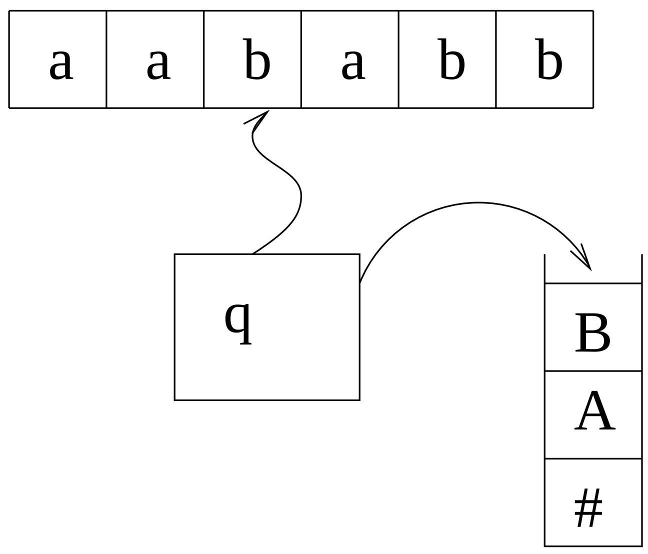
Sei $L = \{a_1, .., a_n\$a_n, .., a_1 \mid a_i \in \Sigma\}$.

Problem:

DFAs und NFAs haben kein “Gedächtnis”, beim Lesen von $ kann nur auf die Zustände zurückgegriffen werden, aber die sind endlich.

Lösung: Kellerautomaten:

Ein Kellerautomat liest ebenfalls ein Eingabeband von links nach rechts, aber auch in jedem Schritt das obere Symbol eines “Kellers” (*stack*). In Abhängigkeit vom Zustand, vom Eingabesymbol und vom Kellersymbol wird in den neuen Zustand übergegangen und ein neues Kellersymbol geschrieben.



Kellerautomaten

Definition:

Ein *nichtdeterministischer Kellerautomat* (*Pushdown Automata, PDA*) ist ein 6-Tupel $M = (K, \Sigma, \Gamma, \Delta, q_0, \#)$ mit:

K eine endliche Menge von Zuständen,

Σ ein Alphabet, das Eingabealphabet,

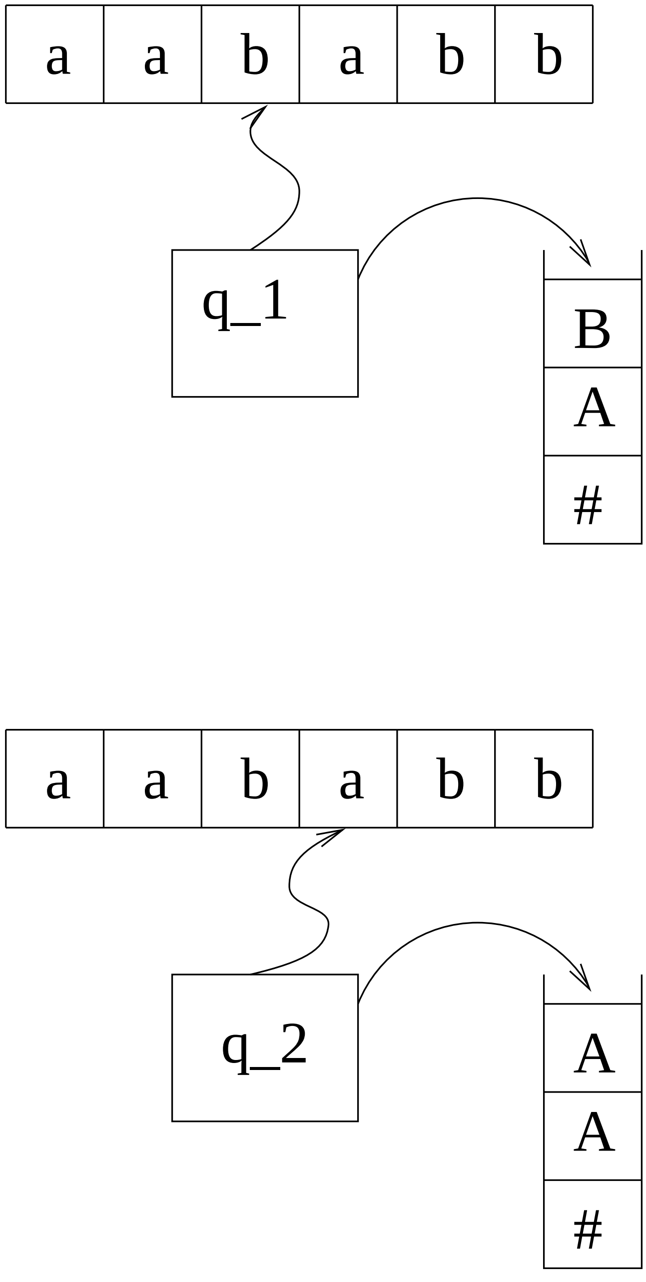
Γ ein Alphabet, das Kelleralphabet,

Δ eine Relation von $K \times (\Sigma \cup \{\epsilon\}) \times \Gamma$ nach $K \times \Gamma$, die Zustandsübergangsrelation,

$q_0 \in K$ der Startzustand,

$\#$ das unterste Kellerzeichen.

Kellerautomaten



$$\Delta((q_1, b, B), (q_2, A))$$

Kellerautomaten

Situation eines PDA:

Position des Lesekopfes, Zustand und Kellerinhalt: (x, q, y, z).

“erzeugt in einem Schritt”:

$(x, q, y, z) \vdash (x', q', y', z')$ gdw. es $a \in \Sigma \cup \{\epsilon\}$, $\gamma, \delta \in \Gamma$ und $w \in \Gamma^*$ gibt mit: $x' = xa, y = ay', z = \gamma w, z' = \delta w$ und $\Delta((q, a, \gamma), (q', \delta))$.

“erzeugt”, $\vdash^*$: die reflexive, transitive Hülle von $\vdash$

Akzeptanz eines Eingabestrings durch einen PDA:

1. Die Eingabe ist vollständig abgearbeitet,
2. der Keller ist leer.

(Alternativ kann man auch Endzustände definieren...)

Ein PDA M akzeptiert eine Sprache L, wenn gilt: für alle $x \in L$ gibt es ein q mit: $(\epsilon, q_0, x, \epsilon) \vdash^*_M (x, q, \epsilon, \epsilon)$

kontextfreie Sprachen und Kellerautomaten

Ein endlicher Automat ist ein Kellerautomat mit einem leeren Kelleralphabet.

Reguläre Sprachen können also auch durch Kellerautomaten erkannt werden.

Ansonsten gilt: eine Sprache ist (maximal) kontextfrei gdw. wenn sie von einem nichtdeterministischen Kellerautomaten erkannt wird.

Deterministischer Kellerautomat: statt Relation Δ Funktion δ, statt leerem Keller definierte Endzustände.

Die deterministischen kontextfreien Sprachen sind eine echte Teilmenge der kontextfreien...

Abschlusseigenschaften von kontextfreien Sprachen

Die Klasse ist *abgeschlossen* unter

- Vereinigung
- Konkatenation
- Stern (Kleene),

nicht aber unter

- Schnitt,
  Gegenbeispiel:
  $L_1 = \{a^i b^i c^j \mid i, j \geq 0\}$ und $L_2 = \{a^k b^l c^l \mid k, l \geq 0\}$,
  $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$: nicht kontextfrei.
- Komplement
  Beweis durch Widerspruch:
  seien zwei Sprachen L_1 und L_2 über Σ gegeben. Wenn die Komplemente $L_1' = \Sigma^* \backslash L_1$ und $L_2' = \Sigma^* \backslash L_2$ kontextfrei wären, dann auch ihre Vereinigung $L_1' \cup L_2'$. Deren Komplement ist laut deMorgan'schem Gesetz aber gerade $L_1 \cap L_2$ und nicht notwendig kontextfrei.

weiter: kontextfreie Sprachen

Aber: Der Schnitt einer regulären Sprache mit einer kontextfreien ist kontextfrei.

Anwendung:

Beweis, dass eine Sprache L nicht kontextfrei ist:

finde L_r regulär und L_n bewiesenermassen nicht kf mit $L = L_r \cap L_n$.

Es gibt *unentscheidbare* Probleme im Zusammenhang mit kontextfreien Grammatiken, z.B. ob zwei Grammatiken dieselbe Sprache erzeugen...

(das heisst nicht, dass das <u>nie</u> entschieden werden, sondern nur, dass es keine allgemeingültige Rechenvorschrift dafür geben kann!)

Anwendung: Natürliche Sprache kontextfrei?

Eher nicht...

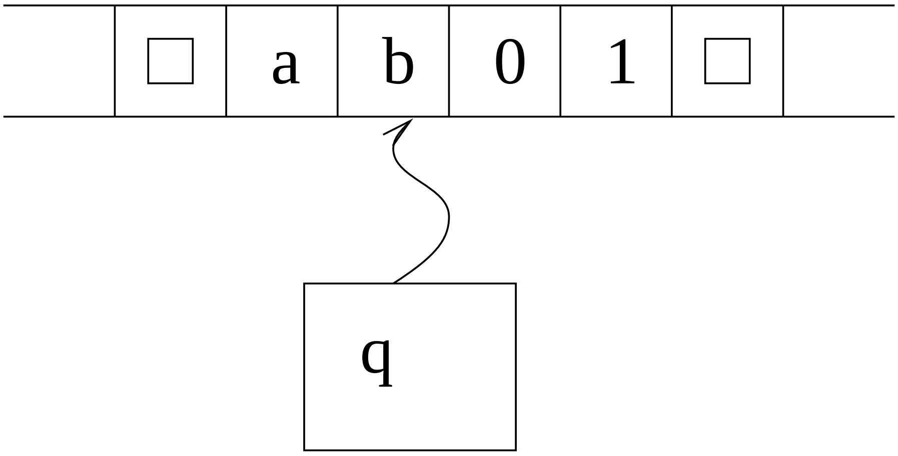
Methode: Nachweis von kreuzweisen Abhängigkeiten wie in $L = \{xx \mid x \in \{a, b\}^*\}$.

z.Bsp. im Schweizerdeutschen und Holländischen:

Wir wollen dem Kind dem Nachbarn den Garten umzugraben zu helfen erlauben.

...

Turingmaschine



Das Band ist unendlich und kann gelesen und beschrieben werden, der Schreiblesekopf kann in jedem Schritt um ein Feld nach rechts oder links versetzt werden oder stehen bleiben.
Das Eingabealphabet ist eine Teilmenge des Arbeitsalphabets.

Turingmaschine, formal

<u>Definition</u>

Eine *Turingmaschine* ist ein 7-Tupel $M = (K, \Sigma, \Gamma, \delta, q_0, \Box, F)$ mit:

K die endliche Menge der Zustände,

$\Sigma \subset \Gamma$ das Eingabealphabet,

Γ das Arbeitsalphabet,

$\delta : K \times \Gamma \rightarrow K \times \Gamma \times \{L, N, R\}$ die Zustandsübergangsfunktion, $\{L, N, R\}$ steht für links, neutral oder rechts (Bewegung des Kopfes),

$q_0 \in K$ der Startzustand,

$\Box \in \Gamma \backslash \Sigma$ das *Blank* - Zeichen für noch nicht besuchte Bandfelder,

F die Menge der Endzustände.

Die *Situation (Konfiguration)* einer Turingmaschine ist ein Wort $s \in \Gamma^* K \Gamma^*$

kontextsensitive Sprachen

Eine *nichtdeterministische linear beschränkte Turingmaschine* (*linear bounded Automaton, LBA*) ist eine Turingmaschine, die sich nicht über den Teil des Bandes, auf dem die Eingabe steht, hinausbewegt (dazu muss das letzte Zeichen der Eingabe besonders markiert werden):

Für alle $a_1a_2...a_{n-1}a_n \in \Sigma^*$ und alle Situationen $\alpha q\beta$ mit $q_0a_1a_2...a_{n-1}a_n' \vdash^* \alpha q\beta$ gilt: $|\,\alpha\beta\,| = n$.

(a_n' ist das markierte letzte Zeichen a_n)

Die von nichtdeterministischen linear beschränkten TMs akzeptierten Sprachen sind (maximal) kontextsensitiv.

Die Frage, ob LBAs und deterministische LBAs äquivalent sind, ist noch unentschieden!

Typ-1-Sprachen sind unter Schnitt, Vereinigung, Komplement, Konkatenation und Stern-Operation abgeschlossen.

rekursiv aufzählbare Sprachen

Allgemeine Turingmaschinen akzeptieren Typ-0-Sprachen, auch *rekursiv aufzählbar* genannt.

Nichtdeterministische Turingmaschinen können durch deterministische simuliert werden (durchsuche nach einem festen Schema den Berechnungsbaum...), die beiden Automatensorten sind äquivalent.

- **Entscheidbar (decidable):** Eine Menge heisst entscheidbar, wenn ihre charakteristische Funktion berechenbar ist.
- **Semi-entscheidbar (semi-decidable):** Eine Menge heisst semi-entscheidbar, wenn ihre charakteristische Funktion für ihre Elemente berechenbar ist. Für andere Objekte kann diese Funktion undefiniert sein.

Eine Sprache ist semi-entscheidbar, wenn sie rekursiv aufzählbar ist.

Das Wortproblem (ist ein gegebenes Wort Element der Sprache?) ist für diese Sprachen nicht entscheidbar (da die Turingmaschine evt. nicht anhält).

Eine Sprache, die selbst Turing-akzeptierbar ist und deren Komplement ebenfalls, ist *entscheidbar*.

Spezielles Halteproblem

Jede Turingmaschine lässt sich als Wort über $\{0, 1, \#\}$ kodieren:

L, R, N seien 0, 1, bzw. 2; Durchnumerierung der Elemente der Alphabete, des Startzustands und des Leerzeichens (beginnend bei 3), dann Binärdarstellung davon.

Regeln der Form $\delta(q_i, a_j) \rightarrow (q_{i'}, a_{j'}, y)$ als Wort aus den Binärdarstellungen der Symbole, getrennt von #, beginnend mit ##.

Diese Codierung lässt sich wiederum auf ein Wort über $\{0, 1\}$ abbilden (wie?).

Sei M' eine beliebige feste Turingmaschine.

$$M_w = \begin{cases} M, & \text{falls } w \text{ eine TM codiert} \\ M', & \text{sonst} \end{cases}$$

Dann heisst die Sprache K mit

$$K = \{w \in \{0, 1\}^* \mid M_w \text{ angesetzt auf } w \text{ hält.}\}$$

spezielles Halteproblem oder *Selbstanwendbarkeitsproblem*.

Dieses Problem ist nicht entscheidbar!

Die Turingmaschine U, die sich auf der Eingabe $w\#x$ so verhält wie die TM M_w auf x, heisst *Universelle Turingmaschine*.

Berechenbarkeitsbegriff

Eine (partielle) Funktion ist intuitiv berechenbar, wenn es eine Rechenvorschrift gibt, die auf den Argumenten, auf denen die Funktion definiert ist, nach endlich vielen Schritten den Funktionswert liefert.

So eine Rechenvorschrift heisst *Algorithmus*: endlich beschreibbar, mechanisch ausführbar, deterministisch, endet auf definierten Eingaben.

Formale Definitionen der Berechenbarkeit: Turingmaschine, μ-Rekursivität, WHILE-Programme, GOTO-Programme, ...

Church'sche These: Diese Formalisierungen erfassen genau den intuitiven Berechenbarkeitsbegriff.

Berechenbarkeit

Sind folgende Beispiele berechenbar?

$f, g, h : N \to \{0, 1\}$

$f(n) = 1$, falls n ein Anfangstück der Dezimalbruchentwicklung von π ist, 0 sonst.

(ja)

$g(n) = 1$, falls n irgendwo in der Dezimalbruchentwicklung von π vorkommt, 0 sonst.

(evt. nein, möglicherweise ist π aber soo zufällig, dass jede Ziffernfolge darin vorkommt. Dann ist $g(n) = 1$.)

$h(n) = 1$, falls n mal 7 in der Dezimalbruchentwicklung von π vorkommt, 0 sonst.

(ja. Entweder gibt es beliebig lange 7er-Folgen, oder es gibt dafür eine obere Grenze...)

O-Notation

Mit der O-Notation kann der Berechnungsaufwand für einen Algorithmus unabhängig von einer bestimmten Programmiersprache oder Rechnerarchitektur angegeben werden.

Angegeben wird eine obere Schranke, konstante Faktoren werden ignoriert.

Für eine Funktion $f : N \to N$ ist definiert:

$O(f) = \{g : N \to N \mid$ es gibt c und n_0, so dass für alle $n > n_0$ gilt: $g(n) \leq cf(n)\}$

Beispiel:

$$O(3\,n^4 + 34\,n \log n - 17) = O(n^4)$$

Komplexitätstheorie

In der Komplexitätstheorie wird versucht, den Berechnungsaufwand für die Lösung von Problemen (i.e. das Wortproblem für formale Sprachen) abzuschätzen.

Eine *obere* Grenze lässt sich durch die Angabe eines Algorithmus bestimmen, eine untere Grenze ist immer n, die Länge der Eingabe.

Probleme lassen sich in Komplexitätsklassen einordnen, neue Probleme können durch *Reduktion* auf bekannte ebenfalls klassifiziert werden.

Komplexitätsklassen

- **P (tractable):** Ein Problem kann mit polynomialem Zeitaufwand bzgl. der Länge der Eingabe gelöst werden.
- **NP-vollständig (NP-complete):** Ein Problem ist *NP-vollständig*, wenn es (noch?) nicht mit polynomialem Zeitaufwand gelöst werden kann.
- **NP-hart (NP-hard):** Ein Problem ist *NP-hart*, wenn es <u>mindestens</u> das Zeitverhalten wie ein NP-vollständiges Problem hat.
- **co-NP-hart (co-NP-hard):** Ein Problem ist *co-NP-hart*, falls das komplementäre Problem NP-hart ist.
- **PSPACE-vollständig (PSPACE-complete):** Ein Problem ist *PSPACE-vollständig*, falls es NP-vollständig ist und polynomialen Speicheraufwand hat.

P $\subseteq$ NP

P = NP ?

Die meisten NP-vollständigen Probleme hängen so zusammen, dass entweder jedes oder keines von ihnen doch polynomial gelöst werden kann.

Reduzierbarkeit

Das Problem **3SAT**:

Gegeben: eine aussagenlogische Formel F in konjunktiver Normalform mit höchstens 3 Literalen pro Klausel. Ist F erfüllbar?

Das Problem **VertexCover**:

Gegeben: ein endlicher Graph G und eine Zahl k. Gibt es eine Teilmenge der Knoten von G mit maximal k Elementen, so dass mindestens ein Endpunt jeder Kante von G darin enthalten ist?

3SAT und **VertexCover** sind als NP-vollständig bekannt.

Durch Reduktion auf **3SAT** oder **VertexCover** lassen sich auch andere Probleme als NP-vollständig charakterisieren...

Statistik: Überblick

- Beschreibende Statistik:
  Auswertung von Experimenten und Stichproben
- Wahrscheinlichkeitsrechnung:
  Schlüsse aus gegebenen Wahrscheinlichkeiten, Hilfsmittel: Kombinatorik
- Beurteilende Statistik:
  Schlüsse aus Experimenten, Beurteilung von exp. Ergebnissen (machen wir nicht)

Linguistische Anwendungen:

- Spracherkennung
- Textretrival
- probabilistische Grammatiken: z.B. Disambiguierung

Problem: woher Daten?

Systemevaluierung

Bsp:Text Retrieval: entspricht ein gefundenes Dokument der Anfrage?

Eventualitätstabelle zur Klassifikation von Beobachtungen (a + b + c + d = n, a echte positive, A positive Entscheidung; n echte negative, N negative Entscheidung):

	'Ja' ist richtig	'Nein' ist richtig
gefunden	a, $P(A \mid a)$, Treffer	b, $P(A \mid n)$, false pos.
nicht gef.	c, $P(N \mid a)$, false neg.	d,$P(N \mid n)$, true neg.

Weitere Masse

$$\text{Precision} = \frac{a}{a+b}$$

$$\text{Recall} = \frac{a}{a+c}$$

$$\text{Fallout} = \frac{b}{b+d}$$

Beschreibende Statistik

Statistische Erhebung: Bestimmung der *Ausprägung* eines *Merkmals* bei allen Individuen einer Grundgesamtheit.

qualitative vs. quantitative Merkmale,

diskrete vs. stetige qualitative Merkmale...

Beispiele:

- Geschlecht/ Gewicht aller Neugeborenen an einem Tag in einem Krankenhaus
- Anzahl der Wörter in jedem Artikel einer Ausgabe einer Tageszeitung
- Anzahl des Vorkommens von bestimmten Wörtern in einem Text-Korpus

Begriffe

- *absolute Häufigkeit*:
  Anzahl des Vorkommens einer Ausprägung.
- *relative Häufigkeit*:
  absolute H. / Anzahl der Individuen
- *Häufigkeitsverteilung*:
  Funktion von allen Ausprägungen eines Merkmals auf Häufigkeiten.
- *Zentralwert*:
  Bedingung: Ausprägungen geordnet. Der Zentralwert ist diejenige Ausprägung, für die gilt: es liegen nicht mehr als die Hälfte der Erhebungswerte darunter oder darüber.
- *arithmetisches Mittel* $\overline{x}$ von Erhebungswerten $x_1, .., x_n$:
  Bedingung: quantitatives Merkmal.
  $\overline{x} = \frac{1}{n}(x_1 + .. + x_n) = \frac{1}{n} \Sigma_{i=1}^{n} x_i$
- *Varianz, Streuung* $\overline{s}^2$ (mittlere quadratische Abweichung):
  $\overline{s}^2 = \frac{1}{n}((x_1 - \overline{x})^2 + .. + (x_n - \overline{x})^2) = \frac{1}{n} \Sigma_{i=1}^{n} (x_i - \overline{x})^2$
- *Standardabweichung*: Quadratwurzel aus Varianz

Zufallsexperimente

Zufallsvariable X : unsicherer Ausgang eines Zufallsexperiments mit endlicher Zahl möglicher Ausgänge $E_1, ..E_k$, Ausgangsmenge oder Ereignisraum $V(X)$.

Bsp: Werfen einer Münze, Ziehung der Lottozahlen, Alter des nächsten Passanten.

Jede Teilmenge von $V(X)$ heisst *Ereignis*, die einzelnen Elemente auch Elementarereignisse.

Das Komplement eines Ereignisses A heisst Gegenereignis $\overline{A}$.

relative Häufigkeit eines Ausgangs, $h(E)$:

$\#$ Eintreten von E / $\#$ Versuche.

Bemerkung zum Übergang zur Wahrscheinlichkeitsrechnung: es gelten gleiche Gesetzmässigkeiten, aber W'keitsrechnung lässt sich nicht statistisch begründen. Daher axiomatische Einführung mit gegebenen *Wahrscheinlichkeiten* der Elementarereignisse.

Axiome der Wahrscheinlichkeit (Kolmogoroff)

Wahrscheinlichkeit:

Sei $\{E_1, ..., E_k\}$ ein Ereignisraum mit den Elementarereignissen E_i.

Wahrscheinlichkeitsverteilung:

Funktion $P : \{E_1, ..., E_k\} \rightarrow [0, 1]$ (1)

mit $P(E_1) + P(E_2) + ... + P(E_k) = 1,$ (2)

$P(E_i)$ heisst *Wahrscheinlichkeit* von E_i.

Sei A ein Ereignis mit Ereignisraum wie oben.

Wahrscheinlichkeit von A:

$P(A) = P(E_1) + .. + P(E_i)$, falls $A = E_1 \cup .. \cup E_i$;

$P(A) = 0$, falls $A = \emptyset$ (3)

Folgerungen daraus: für alle Ereignisse A, B gilt:

$0 \leq P(A) \leq 1$

$P(\overline{A}) = 1 - P(A)$

$A \subset B \Rightarrow P(A) < P(B)$

$A \cap B = \emptyset \Rightarrow P(A \cup B) = P(A) + P(B)$

(A und B heissen *unvereinbar*)

Gleichverteilung

Gleichverteilung: W'keitsverteilung, bei der alle Elementarereignisse die gleiche W'keit haben.

Zufallsexperimente mit Gleichverteilung heissen *Laplace-Experimente*.

Für Laplace-Experimente gilt für Ereignis A :

$$P(A) = \frac{\text{Anzahl der günstigen Ausgänge}}{\text{Anzahl der möglichen Ausgänge}}$$

Beispiele:

- X_1: Augenzahl bei Wurf eines fairen (idealen) Würfels.
  $V(X_1) = \{1, 2, 3, 4, 5, 6\}, P(E_i) = 1/6$
- X_2: Augenzahl bei Wurf von zwei fairen Würfeln gleichzeitig.
  $V(X_2) = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$
  $P(2) = 2/36, P(3) = 4/36, ...$

Kombinatorik

Produktregel zur Bestimmung möglicher Kombinationen:

Sei folgendes Lexikon gegeben:

$\{$ die, keine, schönen, grünen, schnellen, Hunde, Katzen, Mäuse $\}$.

Wieviele NPs lassen sich unter Verwendung der Regel NP $\rightarrow$Det Adj N daraus bilden?

NP = # Det · # Adj · # N.

- Geordnete Stichprobe mit Zurücklegen, k-mal eines von n Elementen ziehen:
  n^k Möglichkeiten.
- Geordnete Stichprobe ohne Zurücklegen:
  $n \cdot (n-1) ... \cdot (n-k+1) = \frac{n!}{(n-k)!}$
- Geordnete Vollerhebung (Permutation): $n!$
- Ungeordnete Stichprobe ohne Zurücklegen (Bsp: Lottozahlen):
  Binomialkoeffizient, ”n über k”: $\begin{pmatrix} n \\ k \end{pmatrix} = \frac{n!}{k!(n-k)!}$

Bedingte Wahrscheinlichkeiten

$P(A \cup B) = P(A) + P(B)$ gilt nur wenn $A \cap B = \emptyset$.
Sonst: $P(A \cup B) = P(A) + P(B) - P(A \cap B)$
Wie kann man $P(A \cap B)$ bestimmen?

Bedingte relative Häufigkeit:

n Durchführungen, $n_1 > 0$ mal Ereignis A, davon k-mal auch B, dann ist k/n_1 die *relative Häufigkeit von* B ***bezüglich*** A, $h_A(B)$, auch $h(B \mid A)$.

$$h(A \cap B) = h(A) \cdot h_A(B)$$

Bedingte Wahrscheinlichkeit:

Gegeben: Ereignisse A und B, $P(A) \neq 0$, dann heisst

$$P(B \mid A) = P_A(B) = \frac{P(A \cap B)}{P(A)}$$

die *durch* A *bedingte Wahrscheinlichkeit von* B oder Wahrscheinlichkeit von B bezgl. A.

Allgemeiner Multiplikationssatz:

$P(A) \neq 0$, dann $P(A \cap B) = P(A) \cdot P_A(B)$

Beispiel

Wenn sich jemand noch genau erinnert, dass eines der beiden Kinder seiner Cousine ein Junge ist, wie gross ist dann die Wahrscheinlichkeit, dass beides Jungen sind? (P(Junge) = P(Mädchen) = 0.5).

Gesucht: $P_{1\ \text{Junge}}(2\ \text{Jungen})$

Lösung

Ohne Information:

P(2 Jungen) = 1/4

P(2 Mädchen)= 1/4

P(Junge/Mädchen) = 1/2

P(A) = P(1 Junge) = 1/4 + 1/2 = 3/4

P(B) = P(2 Jungen), $P(A \cap B) = P(B)$

Mit der Information, dass ein Kind ein Junge ist:

$P_{1\ \text{Junge}}(2\ \text{Jungen}) = P(2\ \text{Jungen}) / P(1\ \text{Junge})$

= 1/4 / 3/4 = 1/3

Weiter: bedingte W'keiten

Seien A und B Ereignisse mit $P(A) \neq 0, P(B) \neq 0$.
Dann gilt:

$$P_B(A) = P_A(B) \cdot P(A)/P(B).$$

verallgemeinert, Satz von Bayes:

Seien $A_1, ...A_n$ Ereignisse, die den Ereignisraum zerlegen, d.h. $P(A_1) + .. + P(A_n) = 1$, und $P(A_j) \geq 0$ für $1 \leq j \leq n$. Sei $P(B) > 0$.

Dann gilt für A_i mit $1 \leq i \leq n$:

$$P_B(A_i) = \frac{P(A_i) \cdot P_{A_i}(B)}{P(A_1) \cdot P_{A_1}(B) + .. + P(A_n) \cdot P_{A_n}(B)}$$

Zwei Ereignisse heissen *unabhängig*, wenn gilt: $P_A(B) = P(B)$ (und $P_B(A) = P(A)$).

Spezieller Multiplikationssatz:

Sind A und B unabhängig, dann gilt:

$$P(A \cup B) = P(A) \cdot P(B)$$

bedingte W'keiten, linguistisch

Wortfolgen:

$$P(w_{1,n}) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1, w_2)...P(w_n \mid w_{1,n-1})$$

$$\text{der} \left\{ \begin{array}{c} \text{kleine} \\ \text{Schweine} \end{array} \right\} \text{Hund}$$

Sei P(kleine | der) = P(Schweine | der).

P_1 = P(der kleine Hund) =
P(der) P(kleine | der) P(Hund | der kleine)

P_2 = P(der Schweine Hund) =
P(der) P(Schweine | der) P(Hund | der Schweine)

$P_1 > P_2$,
falls P(Hund | der kleine) $>$ P(Hund | der Schweine)

Zufallsvariablen

Stochastische Variable oder *Zufallsvariable* X:

Abbildung eines Ereignisraumes S auf reelle Zahlen.

Das Ereignis $\{E_j \mid X(E_j) = x_j\}$ wird mit $X = x_j$ bezeichnet.

<u>*diskrete* Zufallsvariablen</u>

Sei P eine Wahrscheinlichkeitsverteilung über S, X eine Zufallsvariable über S, die die Werte $x_1, .., x_n$ annimmt.

Dann ist die Abbildung

$$f : S \rightarrow [0,1] \text{ mit } f(x_i) = P(X = x_i)$$

die *Wahrscheinlichkeitsverteilung* von X.

Erwartungswert einer diskreten Zufallsvariablen (entspricht arithmetischem Mittel):

$$\mu_X = E(X) = \sum_{x \in S} x \cdot f(x) = \sum_{i=1}^{n} x_i \cdot f(x_i)$$

Varianz σ^2:

$$\sigma_X^2 = \sum_{i=1}^{n} (x_i - \mu)^2 \cdot f(x_i)$$

Zufallsvariablen

stetige Zufallsvariablen

Die Wahrscheinlichkeit, dass der Wert einer stetigen Zufallsvariablen X genau eine reelle Zahl x annimmt, geht gegen 0.

Deshalb wird die summierte Wahrscheinlichkeit von $\underline{X \leq x}$ betrachtet:

$$F(x) = P(\{u \mid X(u) \leq x\}) = P(X \leq x) = \int_{-\infty}^{x} f(t)dt$$

$F : R \rightarrow [0, 1]$ heisst auch Verteilungsfunktion
(und f ist die Ableitung davon).

Erwartungswert:

$$\mu_X = E(X) = \int_{-\infty}^{\infty} x \cdot f(x)dx$$

Varianz σ^2:

$$\sigma_X^2 = \int_{-\infty}^{\infty} (x - \mu)^2 \cdot f(x)dx$$

Zufallsvariablen

mehrdimensionale Zufallsvariablen

Sei S ein Ereignisraum, P eine W'keitsverteilung darüber, X und Y Zufallsvariablen darüber mit den Werten $x_1, .., x_n$ bzw. $y_1, .., y_m$.
Dann ist die Abbildung

$$(x_i, y_j) \rightarrow P(X = x_i, Y = y_j)$$

die *gemeinsame Wahrscheinlichkeitsverteilung* oder -funktion von X und Y.

Gilt für alle (x_i, y_j):

$$P(X = x_i, Y = y_j) = P(X = x_i)P(Y = y_j),$$

dann sind X und Y *unabhängig*.

spezielle W'keitsverteilungen

Binomialverteilung

Bernoulli-Experimente:

Experimente mit nur zwei Ausgängen.

Bernoulli-Variable:

Zufallsvariable, bei der der eine Ausgang den Wert 0, der andere den Wert 1 erhält.

Bernoullische Formel:

Sei p die Wahrscheinlichkeit für den Ausgang 1 eines B.-Experiments. Dann ist die W'keit, dass bei n Ausführungen k-mal der Ausgang 1 eintritt:

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

Sei X eine Zufallsvariable, die die Werte 0, 1, 2, ..., n annehmen kann, mit $X(k) = \binom{n}{k} p^k (1-p)^{n-k} = B_{n,p}(k)$, heisst X *binomialverteilt* mit Parametern n und p.

Es gilt: $E(X) = np, v(X) = np(1-p)$.

Binomialverteilung ist abhängig von der Anzahl der Versuche! Betrachtung von $n \to \infty$: Verschiebung des Erwartungswerts auf 0, Streckung von k um $1/\sigma$ und von $B_{n,p}(k)$ um σ.

Gauss-Funktionen, Normalverteilung

die eulersche Zahl: $e = \sum_{n=0}^{\infty} \frac{1}{n!}$

<u>Gauss-Funktion:</u> $\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-1}{2}x^2}$

Graph von φ ist eine Glockenkurve...
Näherung für Binomialverteilung für $n > \frac{9}{p(1-p)}$

$$B_{n,p}(k) \approx \frac{1}{\sigma}\varphi\left(\frac{k-\mu}{\sigma}\right)$$

mit $\mu = np$ und $\sigma = \sqrt{np(1-p)}$

<u>Gauss'sche Summenfunktion:</u> $\Phi(x) = \int_{-\infty}^{x} \varphi(t)dt$

Sei X $B_{n,p}$-verteilt, dann gilt für genügend grosses n:

$$P(X \leq k) \approx \Phi(\frac{k-\mu}{\sigma}).$$

<u>Normalverteilung</u>

Zufallsvariable X mit $P(X \leq x) = \Phi(\frac{x-\mu}{\sigma})$,

für alle reellen Zahlen x heisst X *normalverteilt* mit Erwartungswert μ und Varianz σ^2 ($N(\mu, \sigma)$-verteilt).

Stochastische Prozesse

Ein *stochastischer* oder *Zufallsprozess* ist eine Folge von Zufallsvariablen $X_1, X_2, ...$ über demselben Ereignisraum.

Die möglichen Ausgänge heissen auch *Zustände* des Prozesses, der Prozess ist im Zustand x_t zum Zeitpunkt t.

Die X_i sind <u>nicht</u> zwingend unabhängig voneinander!

Zufallsprozesse können über diskrete oder stetige Zeitparameter und über diskrete oder stetige Zufallsvariablen betrachtet werden, hier aber nur diskrete Zeitschritte und endliche Ausgangsmengen.

Vollständige Charakterisierung eines Zufallsprozesses:

Wahrscheinlichkeit $P(X_1 = x_j)$ für alle Ausgänge x_j für den Anfangszustand.

für jeden folgenden Zustand $X_{t+1}, t = 1, 2, ...$ die bedingten W'keiten $P(X_{t+1} = x_{i_{t+1}} \mid X_1 = x_{i_1}, .., X_t = x_{i_t})$

N-Gramm-Modelle

Annahme: nur die letzten $n-1$ Wörter haben Einfluss auf die Wahrscheinlichkeit des nächsten. Gebräuchlich ist $n=3$: Trigramm-Modelle.

Wahrscheinlichkeit für ein Wort w_n nach der Wortfolge $w_{1,n-1}$:

$$P(w_n \mid w_{1,n-1}) = P(w_n \mid w_{n-2}, w_{n-1})$$

Die Wahrscheinlichkeit für eine Wortfolge $w_{1,n}$ beträgt dann:

$$\begin{aligned} P(w_{1,n}) &= P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1 w_2)..P(w_n \mid w_{1,n-1}) \\ &= P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1 w_2)..P(w_n \mid w_{n-2,n-1}) \\ &= P(w_1)P(w_2 \mid w_1) \prod_{i=3}^{n} P(w_i \mid w_{i-2,i-1}) \end{aligned}$$

$$P(w_{1,n}) = \prod_{i=3}^{n} P(w_i \mid w_{i-2,i-1})$$

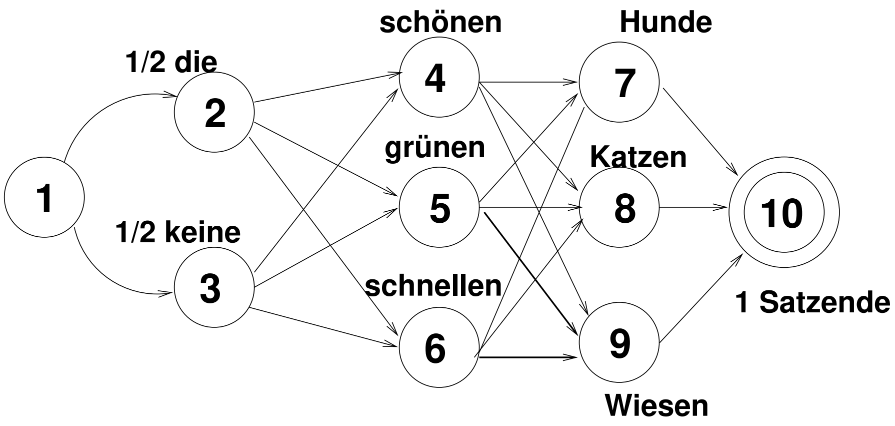
Markov-Kette

Eine *Markov-Kette* ist ein Zufallsprozess, bei dem die Wahrscheinlichkeit des nächsten Zustands nur vom aktuellen abhängt.

Die Markov-Eigenschaft ist also:

$$P(X_{t+1} = x_{i_{t+1}} \mid X_1 = x_{i_1}, .., X_t = x_{i_t})$$
$$= P(X_{t+1} = x_{i_{t+1}} \mid X_t = x_{i_t})$$

Beispiel:



Sei die Kante zwischen 5 und 9 mit 0.98 gewichtet und die zwischen 6 und 9 mit 0.02.

Stochastische Matrix

Sei eine (*endliche*) Markov-Kette mit n Zuständen gegeben.

Die *Übergangswahrscheinlichkeiten* von Zustand s_i in s_j, d.h. $P(X_{t+1} = s_j \mid X_t = s_i) = p_{ij}$ können in einer Übergangsmatrix dargestellt werden:

$$\mathbf{P} = \begin{bmatrix} p_{11} & \cdots & p_{1n} \\ \cdots & \cdots & \cdots \\ p_{n1} & \cdots & p_{nn} \end{bmatrix}, 0 \leq p_{ij} \leq 1, \sum_{j=1}^{n} p_{ij} = 1 \text{ für } i = 1, 2, .., n$$

Ein Vektor $\mathbf{v} = [v_1, .., v_n]$ mit $1 \geq v_i \geq 0$ und $\Sigma_{i=1}^{n} v_i = 1$ heisst Wahrscheinlichkeitsvektor, und kann z.B. für den ersten Zustand einer Markov-Kette gelten. Dann gilt: $v_i = P(X_i = s_i), i = 1, .., n$.

Der initiale Wahrscheinlichkeitsvektor zusammen mit der Übergangsmatrix bestimmen eine Markov-Kette vollständig, d.h. die Wahrscheinlichkeiten, dass sich der Prozess an einem best. Zeitpunkt t in einem best. Zustand s_i befindet, können daraus errechnet werden:

$$[p^{(t)}(s_1), .., p^{(t)}(s_n)] = \mathbf{v}\mathbf{P}^{t-1}$$

Beispielmatrix

Für das Beispiel zur Markov-Kette kann man folgende Zustandsübergangs-Wahrscheinlichkeits-Matrix auf-stellen:

$$\begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.34 & 0.33 & 0.33 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.34 & 0.33 & 0.33 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.33 & 0.33 & 0.34 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.01 & 0.01 & 0.98 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.49 & 0.49 & 0.02 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Matrix-Multiplikation

Wir brauchen nur quadratische, d.h. $(n \times n)$ Matrizen.

Seien $\mathbf{A}$ und $\mathbf{B}$ $(n \times n)$ Matrizen mit den Elementen a_{ij} und b_{ij}, $i, j = 1, .., n$, i die Zeile, j die Spalte.

Dann ist das Produkt $\mathbf{A} \cdot \mathbf{B}$ definiert als $(n \times n)$ Matrix $\mathbf{C}$ mit

$$c_{ij} = \sum_{k=1}^{n} (a_{ik} \cdot b_{kj}).$$

(Das ist das Produkt aus dem i-ten Zeilenvektor $z_i = [a_{i1}, .., a_{in}]$ und dem j-ten Spaltenvektor $s_j = [a_{j1}, .., a_{jn}]$)

Produkt aus $(n \times n)$ Matrix $\mathbf{A}$ und Vektor x (n-stellig):

$$\mathbf{A}x = \begin{bmatrix} z_1 x \\ \vdots \\ z_n x \end{bmatrix} = \begin{bmatrix} \Sigma_{k=1}^{n} a_{1k} x_k \\ \vdots \\ \Sigma_{k=1}^{n} a_{nk} x_k \end{bmatrix}$$

Markov-Modelle

Sei jeder Zustand einer Markov-Kette mit einer endlichen Menge von *Signalen* verbunden.

Nach jedem Zustandsübergang wird eines der zum aktuellen Zustand gehörenden Signale ausgegeben. Die Zufallsvariable η_t repräsentiert dieses Signal zum Zeitpunkt t.

Ein *Markov-Modell* (erster Ordnung) besteht aus:

- einer endliche Menge von Zuständen $\omega = \{s_1, .., s_n\}$
- einem Signal-Alphabet $\Sigma = \{\sigma_1, .., \sigma_m\}$
- einer $(n \times n)$-Zustandsübergangs-Matrix $\mathbf{P} = [p_{ij}]$ mit $p_{ij} = P(X_{t+1} = s_j \mid X_t = s_i)$
- einer $(n \times m)$-Signal-Matrix $\mathbf{A} = [a_{ij}]$ mit der Wahrscheinlichkeit $a_{ij} = p(\eta_t = \sigma_j \mid X_t = s_i)$ für jedes Zustands-Signal-Paar, dass σ_j im Zustand s_i ausgegeben wird.
- und einem initialer Vektor $\mathbf{v} = [v_1, .., v_n]$ mit $v_i = P(X_1 = s_i)$

Sei $p^{(t)}(\sigma_j)$ die Wahrscheinlichkeit, dass zur Zeit t das Signal σ_j ausgegeben wird. Der Vektor

$$[p^{(t)}(\sigma_1), .., p^{(t)}(\sigma_m)] = \mathbf{v}\mathbf{P}^{t-1}\mathbf{A}$$

enthält diese Wahrscheinlichkeiten für alle $\sigma \in \Sigma$.

Hidden Markov Models, HMMs

Wenn keine Beobachtung der Zustände möglich ist, sondern nur die Signale beobachtet werden können, liegt ein *Hidden Markov Model* (HMM) vor.

Sei $\mathbf{O} \in \Sigma^*$ eine Folge von beobachteten Signalen und $\mathbf{S} \in S^*$ die unbekannte Folge von Zuständen.

Die beste Schätzung für **S** ist die Folge mit dem grössten Wert für $P(\mathbf{S} \mid \mathbf{O})$

Laut Bayes'schem Satz gilt:

$$P(\mathbf{S} \mid \mathbf{O}) = \frac{P(\mathbf{O} \mid \mathbf{S}) \cdot P(\mathbf{S})}{P(\mathbf{O})}$$

und da $P(\mathbf{O})$ nicht von $\mathbf{S}$ abhängt, können wir auch $P(\mathbf{O} \mid \mathbf{S}) \cdot P(\mathbf{S})$ maximieren.

$P(\mathbf{O} \mid \mathbf{S})$ heisst Signalmodell, $P(\mathbf{S})$ Sprachmodell.

Anwendungen für HMMs

1. Schätzung der Wahrscheinlichkeit einer Signalfolge (Identifikation einer Sprache), $P(\mathbf{O})$
2. Bestimmung der wahrscheinlichsten Zustandsfolge, die zu einer Signalfolge geführt hat:

   <u>Tagging</u>

   Signale: Wörter eines Eingabetextes

   Zustände: Mengen von Wortarten

   Aufgabe: finde die wahrscheinlichste Folge von Wortartmengen, die den Wörtern zugeordnet werden können.

   <u>Spracherkennung</u>

   Signale: (Repräsentation der) akustischen Signale

   Zustände: mögliche Wörter

   Aufgabe: finde die wahrscheinlichste Folge von Wörtern, die die akustischen Signale hervorgerufen haben

3. Bestimmung der Parameter $\mathbf{P}, \mathbf{A}, \mathbf{v}$

1. $P(\mathbf{O})$

Sei $\mathbf{O} = (\sigma_{k_1}, .., \sigma_{k_T}), \mathbf{S} = (s_{i_1}, .., s_{i_T})$.
Dann:

$$\begin{aligned}
P(\mathbf{O} \mid \mathbf{S}) &= \Pi_{t=1}^{T} P(\eta_t = \sigma_{k_t} \mid X_t = s_{i_t}) = \Pi_{t=1}^{T} a_{i_t k_t} \\
\\
P(\mathbf{S}) &= v_{i_1} \cdot \Pi_{t=1}^{T} p_{i_{t-1} i_t} \\
\\
P(\mathbf{O} \cap \mathbf{S}) &= P(\mathbf{O} \mid \mathbf{S}) \cdot P(\mathbf{S}) \\
&= (\Pi_{t=1}^{T} a_{i_t k_t}) \cdot (\Pi_{t=1}^{T} p_{i_{t-1} i_t}) \\
&= a_{i_1 k_1} \cdot v_{i_1} \cdot \Pi_{t=2}^{T} p_{i_{t-1} i_t} a_{i_t k_t}
\end{aligned}$$

und:

$$P(\mathbf{O}) = \sum_{\mathbf{S}} P(\mathbf{O} \cap \mathbf{S})$$

und das ist viel zu aufwendig!

wie aufwendig?

$O(2Tn^T)$

Der Vorwärts-Algorithmus

Vorwärts-Variablen: Wahrscheinlichkeit, zum Zeitpunkt t im Zustand s_i zu sein und die Signalfolge $\mathbf{O}_{\leq t}$ bis dahin beobachtet zu haben.

$$\alpha_t(i) = P(\mathbf{O}_{\leq t}; X_t = s_i)$$
$$= P(\eta_1 = \sigma_{k_1}, .., \eta_t = \sigma_{k_t}; X_t = s_i).$$

$$P(\mathbf{O}) = \sum_{i=1}^{n} P(\eta_1 = \sigma_{k_1}, .., \eta_T = \sigma_{k_T}; X_T = s_i) = \sum_{i=1}^{n} \alpha_T(i)$$

$$\alpha_1(i) = a_{ik_1} \cdot v_i$$
und $\alpha_{t+1}(j) = (\Sigma_{i=1}^{n} \alpha_t(i) \cdot p_{ij}) \cdot a_{jk_{t+1}}$

Begründung (Markov-Annahme im zweiten Schritt):

$$P(\mathbf{O}_{\leq t+1}; X_{t+1} = s_j) =$$
$$= \Sigma_{i=1}^{n} P(\mathbf{O}_{\leq t}; X_t = s_i) \cdot P(\eta_{t+1} = \sigma_{k_{t+1}}; X_{t+1} = s_j \mid \mathbf{O}_{\leq t}; X_t = s_i)$$
$$= \Sigma_{i=1}^{n} P(\mathbf{O}_{\leq t}; X_t = s_i) \cdot P(\eta_{t+1} = \sigma_{k_{t+1}} \mid X_{t+1} = s_j) \cdot P(X_{t+1} = s_j \mid X_t = s_i)$$

Aufwand: $O(n^2 T)$

Der Rückwärts-Algorithmus

Rückwärts-Variablen: Wahrscheinlichkeit, zum Zeitpunkt t im Zustand s_i gewesen zu sein und die Signalfolge $\mathbf{O}_{>t}$ von da an beobachtet zu haben.

$$\beta_t(i) = P(\mathbf{O}_{>t}; X_t = s_i)$$
$$= P(\eta_{t+1} = \sigma_{k_{t+1}}, .., \eta_T = \sigma_{k_T}; X_t = s_i).$$

$$P(\mathbf{O}) = \sum_{i=1}^{n} P(\eta_1 = \sigma_{k_1}, X_1 = s_i) \cdot P(\eta_2, .., \eta_T = \sigma_{k_T}; X_1 = s_i)$$
$$= \sum_{i=1}^{n} a_{ik_1} \cdot v_i \cdot \beta_1(i)$$

Definiere $\beta_T(i) = 1$ für $i = 1, .., n$.

$$\beta_t(i) = \sum_{j=1}^{n} p_{ij} \cdot a_{jk_{t+1}} \cdot \beta_{t+1}(j)$$

weil:

$$P(\mathbf{O}_{>t} \mid X_{t+1} = s_i) =$$

$$= \Sigma_{j=1}^{n} P(\mathbf{O}_{>t}; X_{t+1} = s_j \mid X_t = s_i)$$
$$= \Sigma_{j=1}^{n} P(\mathbf{O}_{>t} \mid X_t = s_i; X_{t+1} = s_j)$$
$$\cdot P(X_{t+1} = s_j \mid X_t = s_i)$$

$$= \Sigma_{j=1}^{n} P(\eta_{t+1} = \sigma_{k_{t+1}} \mid X_{t+1} = s_j) \cdot P(\mathbf{O}_{>t} \mid X_{t+1} = s_j)$$
$$\cdot P(X_{t+1} = s_j \mid X_t = s_i)$$

Der Vorwärts-Rückwärts-Algorithmus

$$
\begin{aligned}
P(\mathbf{O}) &= \Sigma_{i=1}^{n} P(\mathbf{O}; X_t = s_i) \\
&= \Sigma_{i=1}^{n} P(\mathbf{O}_{\leq t}; X_t = s_i) \cdot P(\mathbf{O}_{>t} \mid \mathbf{O}_{\leq t}; X_t = s_i) \\
&= \Sigma_{i=1}^{n} P(\mathbf{O}_{\leq t}; X_t = s_i) \cdot P(\mathbf{O}_{>t} \mid X_t = s_i) \\
&= \Sigma_{i=1}^{n} \alpha_t(i)\beta_t(i)
\end{aligned}
$$

Wahrscheinlichkeit, zum Zeitpunkt t im Zustand s_i zu sein, wenn **O** die gesamte beobachtete Sequenz von Zeit 1 bis T ist:

Vorwärts-Rückwärts-Variablen:

$$
\gamma_t(i) = P(X_t = s_i \mid \mathbf{O}) = \frac{P(\mathbf{O}; X_t = s_i)}{P(\mathbf{O})} = \frac{\alpha_t(i)\beta_t(i)}{\Sigma_{i=1}^{n} \alpha_t(i)\beta_t(i)}
$$

Datenstruktur Trellis

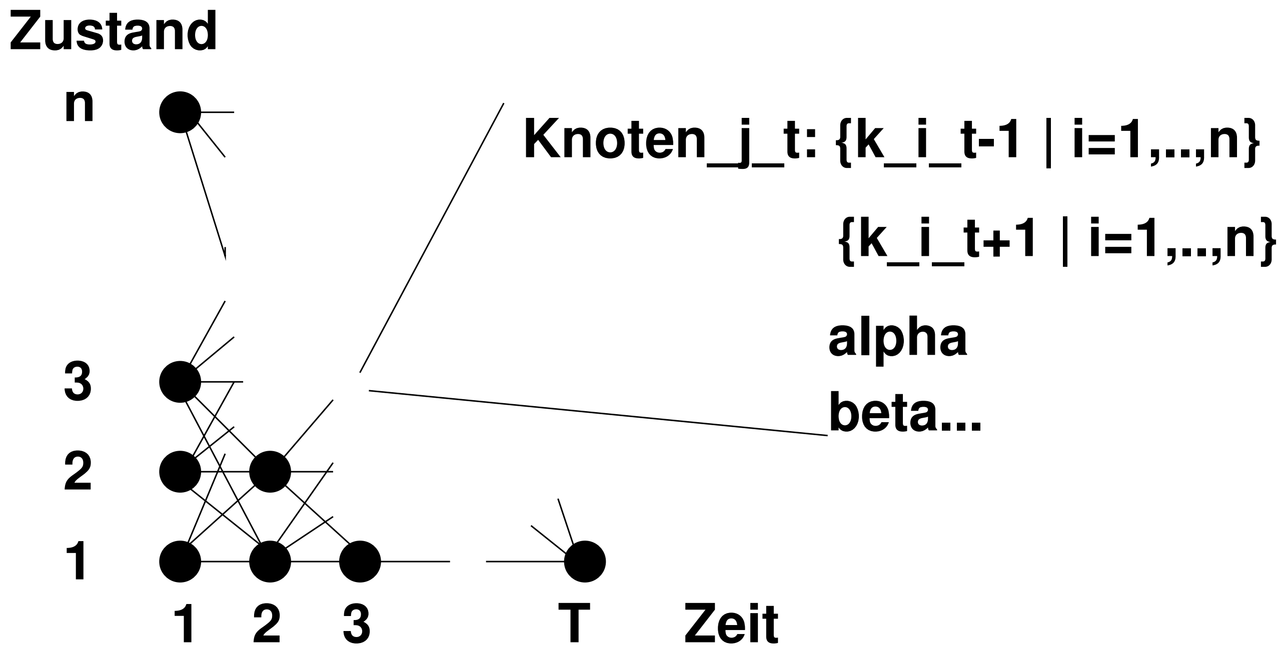
Ein Trellis ist ein Graph mit je einem Knoten für jeden Zustand an jedem Zeitpunkt.

Jeder Knoten zum Zeitpunkt t ist mit den Knoten zu den Zeitpunkten $t-1$ und $t+1$ verbunden.

Die Knoten werden mit Variablen attributiert, in denen Werte akkumuliert werden können (z.B. Vorwärtsvariablen).

Wenn eine passende Gleichung vorliegt, können aus den Werten für t die für $t=1$ berechnet werden.

Beispiel:



2. Bestimmung von $\mathrm{S}, s^*_{i_t}$

Def.: max$_x f(x)$ ist der maximale Wert von $f(x)$

Def.: arg max$_x f(x)$ ist derjenige Wert von x, mit dem $f(x)$ maximal wird.

<u>Viterbi-Algorithmus</u>

Gesucht: Variablen δ mit:

$\delta_t(i) = \max_{\mathbf{S}_{\leq t-1}} P(\mathbf{S}_{\leq t-1}, X_t = s_i; \mathbf{O}_{\leq t})$

$= \max_{s_{i1},..s_{i\,t-1}} P(X_1 = s_{i_1}, .., X_{t-1} = s_{i_{t-1}}, X_t = S_{i_t}; \mathbf{O}_{\leq t}),$

der Wahrscheinlichkeit der wahrscheinlichsten Zustandsfolge von Zeit 1 bis t mit s_i am Zeitpunkt t und beobachteter Signalfolge $\mathbf{O}_{\leq t}$.

Der Vektor $\psi_t(i)$ bezeichnet den Vorgänger vom Zustand s_i im Pfad zu $\delta_t(i)$.

Unterschied zu Vorwärts-Algorithmus:

Maxima statt Summen.

Gleiche Komplexität $O(n^2T)$

Viterbi-Algorithmus

Es gilt:

$$\delta_1(i) = v_i \cdot a_{ik_1}, \qquad i = 1, .., n$$

$$\delta_t(j) = [\text{max}_i \delta_{t-1}(i) \cdot p_{ij}] \cdot a_{jk_t}, \qquad t = 2, .., T, j = 1, .., n$$

$$\psi_t(j) = \text{arg max}_{1 \leq i \leq n}(\delta_{t-1}(i) \cdot p_{ij}), \quad t = 2, .., T, j = 1, .., n$$

$$P^* = \text{max}_i \delta_T(i)$$

$$s^*_{k_t} = \text{arg max}_{1 \leq i \leq n} \delta_T(i)$$

$$s^*_{k_t} = \psi_{t+1}(s^*_{k_{t+1}}), \qquad t = 1, .., T-1$$

Viterbi-Algorithmus

Die Begründung für $\delta_t(j)$:

$$
\begin{aligned}
\delta_t(j) &= \text{max}_{\mathbf{S}_{\leq t-1}} P(\mathbf{S}_{\leq t-1}, X_t = s_i; \mathbf{O}_{\leq t}) \\
&= \text{max}_i \text{max}_{\mathbf{S}_{\leq t-2}} P(\mathbf{S}_{\leq t-2}, X_{t-1} = s_i, X_t = s_j; \mathbf{O}_{\leq t-1}, \eta_t = \sigma_{k_t}) \\
&= \text{max}_i \text{max}_{\mathbf{S}_{\leq t-2}} P(X_t = s_j; \eta_t = \sigma_{k_t} \mid \mathbf{S}_{\leq t-2}, X_{t-1} = s_i; \mathbf{O}_{\leq t-1}) \\
&\qquad \cdot P(\mathbf{S}_{\leq t-2}, X_{t-1} = s_i; \mathbf{O}_{\leq t-1}) \\
&= \text{max}_i \text{max}_{\mathbf{S}_{\leq t-2}} P(X_t = s_j \mid X_{t-1} = s_i) \cdot P(\eta_t = \sigma_{k_t} \mid X_t = s_j) \\
&\qquad \cdot P(\mathbf{S}_{\leq t-2}, X_{t-1} = s_i; \mathbf{O}_{\leq t-1}) \\
&= [\text{max}_i P(X_t = s_j \mid X_{t-1} = s_i) \cdot \text{max}_{\mathbf{S}_{\leq t-2}} P(\mathbf{S}_{\leq t-2}, X_{t-1} = s_i; \mathbf{O}_{\leq t-1})] \\
&\qquad \cdot P(\eta_t = \sigma_{k_t} \mid X_t = s_j) \\
&= [\text{max}_i p_{ij} \cdot \delta_{t-1}(i) \cdot a_{ik_t}
\end{aligned}
$$

Stochastisches Tagging

1. Einem Wort wird das wahrscheinlichste Tag, d.h. dasjenige, mit dem es im Trainingsset am häufigsten assoziert war, zugeordnet.

   Zu 90 % korrekt (Englisch, nach Allen 1995)

   Nachteil: kann unzulässige Folgen von Tags ergeben

2. Wahrscheinlichkeit einer Folge von Tags, Tag-N-Gramme, Viterbi-Algorithmus

3. HMMs, Kombination von Worthäufigkeit und Tag-Folgen-Wahrscheinlichkeit, Viterbi-Algorithmus; jedes Wort ist unabhängig von den anderen Wörtern, aber hängt von den n vorigen Tags ab.

Unbekannte Wörter:

- morphologische Information, Prefixe, Suffixe
- Menge von Default-Tags (offene Wortklassen), dann davon nach W'keit im N-Gramm davor auswählen
- W'keit aller Tags im Tagset am Ende des entsprechenden N-Gramms (ungeeignet für grosse Tagsets)

Tagging, Beispiel (deRose 1988)

2. Variante: (n-best) Tag-N-Gramme

"The man still saw her "

the	Art		
man	N	V	
still	Adv	N	V...
saw	N	V Pst	
her	PsPr	Pr Dat	

Häufigkeiten bei 4017 Bigrammen:

	N	Pr Dat	PsPr	Adv	V	V Pst	.
Art	186	0	0	8	1	8	9
N	40	1	3	40	9	66	186
Pr Dat	7	3	16	164	109	16	313
PsPr	176	0	0	5	1	1	2
Adv	5	3	16	164	109	16	313
V	22	694	146	98	9	1	59
V pst	11	584	143	160	2	1	91

1. Schritt, (Art) (eindeutig).

2. Schritt, möglich: Art - N und Art - V, beide expandieren (obwohl W'keit f. Art - V sehr gering).

3. Schritt, usw. besten Pfad zu jedem Tag aufheben, hier (Art - N - N), (Art - N - V) und (Art - N - Adv).

3. Parameterbestimmung für HMMs

Aufgabe: Bestimmung von Anfangsw'keiten $\mathbf{v}$, Übergangsw'keiten $\mathbf{P}$ und Signalw'keiten $\mathbf{A}$ für ein HMM.

1. Fall:

Annotierte Trainingsdaten liegen vor, d.h. Signal- und Zustandsfolge sind bekannt.

Dann können die relativen Häufigkeiten als Schätzwerte verwendet werden.

Problem: "sparse Data", spärliche Daten.

2. Fall:

Es gibt nur rohe Trainingsdaten, d.h. nur die Signalfolge ist bekannt.

Es werden Iterationsgleichungen definiert, die in jedem Schritt bessere Schätzwerte liefern.

Spärliche Daten

Problem: manche Phänomene sind so selten, dass sie im Trainingsset gar nicht auftreten.

Lösungen:

- *Smoothing* für Trigramme

  Bigramme und Unigramme hinzuziehen und gewichten:

  $$\begin{aligned} P(w_n \mid T_{n-2}, T_{n-1}) \approx & \quad \lambda_1 P_e(T_n) \\ & + \lambda_2 P_e(T_n \mid T_{n-1}) \\ & + \lambda_3 P_e(T_n \mid T_{n-2}, T_{n-1}) \end{aligned}$$

  mit $\lambda_1 + \lambda_2 + \lambda_3 = 1, \lambda_i > 0$, λ_3 “relativ“ hoch.

- ...

Parameterschätzungen durch Iteration

Baum-Welch-Verfahren oder Vorwärts-Rückwärts-Wiedereinschätzungsalgorithmus:

Definiere W'keiten:

$$\begin{aligned}\epsilon_t(i,j) &= P(X_t = s_i, X_{t+1} = s_j \mid \mathbf{O}) \\ &= \frac{P(\mathbf{O}; X_t = s_i, X_{t+1} = s_j)}{P(\mathbf{O})},\end{aligned}$$

d.h. gemeinsame W'keit, dass zum Zeitpunkt t Zustand s_i und zum Zeitpunk $t+1$ Zustand s_j vorlag unter der Bedingung, dass die Signalfolge $\mathbf{O}$ beobachtet wurde (von Zeit 1 bis T).

Es gilt:

$$\begin{aligned}\epsilon_t(i,j) &= \frac{\alpha_t(i) \cdot p_{ij} \cdot a_{jk_{t+1}} \cdot \beta_{t+1}(j)}{P(\mathbf{O})} \\ &= \frac{\alpha_t(i) \cdot p_{ij} \cdot a_{jk_{t+1}} \cdot \beta_{t+1}(j)}{\Sigma_{i,j} \alpha_t(i) \cdot p_{ij} \cdot a_{jk_{t+1}}}\end{aligned}$$

weiter: Baum-Welch

Weil:

$$
\begin{aligned}
& P(\mathbf{O}; X_t = s_i, X_{t+1} = s_j) \\
& \quad = P(\mathbf{O}_{\leq t+1}, \mathbf{O}_{>t+1}; X_t = s_i, X_{t+1} = s_j) \\
& \quad = P(\mathbf{O}_{\leq t+1}; X_t = s_i, X_{t+1} = s_j) \\
& \qquad \cdot P(\mathbf{O}_{>t+1} \mid \mathbf{O}_{\leq t+1}; X_t = s_i, X_{t+1} = s_j) \\
& \quad = P(\mathbf{O}_{\leq t+1}; X_t = s_i) \\
& \qquad \cdot P(\eta_{t+1} = \sigma_{k_{t+1}}; X_{t+1} = s_j \mid \mathbf{O}_{\leq t+1}; X_t = s_i) \\
& \qquad \cdot P(\mathbf{O}_{>t+1} \mid X_{t+1} = s_j) \\
& \quad = P(\mathbf{O}_{\leq t+1}; X_t = s_i) \cdot P(X_{t+1} = s_j \mid X_t = s_i) \\
& \qquad \cdot P(\eta_{t+1} = \sigma_{k_{t+1}} \mid X_{t+1} = s_j) \\
& \qquad \cdot P(\mathbf{O}_{>t+1} \mid X_{t+1} = s_j) \\
& \quad = \alpha_t(i) \cdot p_{ij} \cdot a_{j k_{t+1}} \cdot \beta_{t+1}(j)
\end{aligned}
$$

Diese ϵ heissen ebenfalls Vorwärts-Rückwärts-variablen, und hängen mit den γ wie folgt zusammen:

$$
\begin{aligned}
\gamma_t(i) & = P(X_t = S_i \mid \mathbf{O}) \\
& = \Sigma_{j=1}^{n} P(X_t = s_i, X_t + 1 = s_j \mid \mathbf{O}) \\
& = \Sigma_{j=1}^{n} \epsilon_t(i, j)
\end{aligned}
$$

weiter: Baum-Welch

Wir haben also:

- $\gamma_1(i)$ - W'keit, im Zustand s_i zu starten
- $\Sigma_{t=1}^{T-1}\epsilon_t(i,j)$ - erwartete Anzahl von Übergängen von s_i nach s_j
- $\Sigma_{t=1}^{T-1}\gamma_t(i)$ - erwartete Anzahl von Übergängen aus s_i
- $\Sigma_{t=1:\sigma_{k_t}=\sigma_j}^{T}\gamma_t(i)$ - erwartete Anzahl von Ausgaben des Signals σ_j im Zustand s_i

Die gesuchten Iterationsgleichungen:

$$\begin{aligned}\overline{v}_i &= \gamma_1(i)\\ \overline{p}_{ij} &= \frac{\Sigma_{t=1}^{T-1}\epsilon_t(i,j)}{\Sigma_{t=1}^{T-1}\gamma_t(i)}\\ \overline{a}_{ij} &= \frac{\Sigma_{t=1:\sigma_{k_t}=\sigma_j}^{T}\gamma_t(i)}{\Sigma_{t=1}^{T}\gamma_t(i)},\end{aligned}$$

wobei die Summen über i von 1 bis n der v_i, p_{ij} und a_{ij} immer 1 ergeben müssen.

Es kann gezeigt werden, dass $P(\mathbf{O} \mid (\mathbf{v}, \mathbf{P}, \mathbf{A}))$ mit jedem Schritt steigt oder die Parameter gleich bleiben.

stochastische kontextfreie Grammatik

Eine kontextfreie Grammatik G ist ein 4-Tupel,

$$(V, \Sigma, R, S)$$

mit Nicht-Terminalsymbolen V,
Terminalsymbolen Σ,
Regeln R der Form $X \rightarrow \beta$ (mit $X \in V, \beta \in (V \cup \Sigma)^*$)
und dem Startsymbol $S \in V$.

Warum Wahrscheinlichkeiten dazu?

- Wertung von Hypothesen (z.B. bei Spracherkennung),
- Auswahl des wahrscheinlichsten Ergebnisses (Parsing),
- frühzeitiges Ausschalten unwahrscheinlicher Analyseversuche (Parsing, Effizienzsteigerung)

stochastische kontextfreie Grammatik

Eine *stochastische* oder *probabilistische* kontextfreie Grammatik G ist ein 5-Tupel, (V, Σ, R, S, P),

mit V, Σ, R, S wie vorher und

P eine Funktion von R nach [0,1] mit

$$\forall X \in V : \sum_{\beta \in (V \cup \Sigma)^*} P(X \rightarrow \beta) = 1$$

Die Summe der Wahrscheinlichkeiten für die Regeln mit einem bestimmten Nichtterminal auf der linken Seite muss 1 sein.

Beispiel:

$$\begin{array}{lcll} S & \rightarrow & NP\,VP & (1.0) \\ VP & \rightarrow & V & (0.5) \\ VP & \rightarrow & V\ Adv & (0.5) \\ NP & \rightarrow & \textit{Hans} & (1.0) \\ V & \rightarrow & \textit{schläft} & (1.0) \\ Adv & \rightarrow & \textit{ständig} & (1.0) \end{array}$$

Wahrscheinlichkeit einer Zeichenfolge: Summe der W'keiten der Parsebäume dazu.

W'keit eines Parsebaums: W'keit der erzeugenden Linksableitung

Wahrscheinlichkeit einer Ableitung

Wahrscheinlichkeit einer Linksableitung:

Seien die Regeln durchnummeriert und die Ableitung repräsentiert durch die Folge der verwendeten Regeln. Sei X_m eine Zufallsvariable, die von der Regel bestimmt wird, die in Schritt m verwendet wurde.

Dann ist eine Linksableitung ein stochastischer Prozess $X_1, .., X_M$ mit der Regelmenge der Grammatik als Zustandsmenge.

Die W'keit einer Linksableitung ist:

$$
\begin{aligned}
&P(X_1 = r_{i_1}, .., X_M = r_{i_M}) \\
&\quad = P(X_M = r_{i_M} \mid X_1 = r_{i_1}, .., X_{M-1} = r_{i_{M-1}}) \\
&\qquad \cdot P(X_1 = r_{i_1}, .., X_{M-1} = r_{i_{M-1}}) \\
&\quad = \Pi_{m=1}^{M} P(X_m = r_{i_m} \mid X_1 = r_{i_1}, .., X_{m-1} = r_{i_{m-1}})
\end{aligned}
$$

nötig: alle Faktoren, d.h. alle bedingten W'keiten für jede Regel.

Annahme, Regeln unabhängig von vorangegangenen Ableitungsschritten:

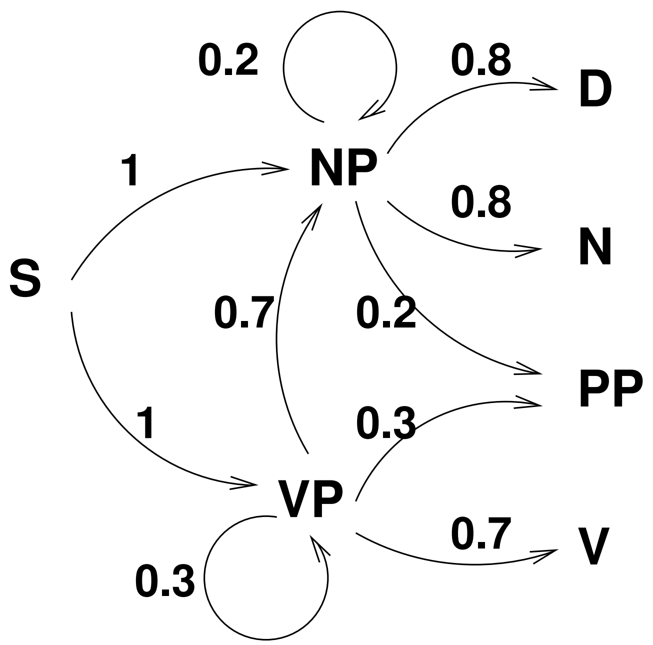
$$P(X_m = r_{i_m} \mid X_1 = r_{i_1}, .., X_{m-1} = r_{i_{m-1}}) = P(r_{i_m})$$

also $P(X_1 = r_{i_1}, .., X_M = r_{i_M}) = \prod_{m=1}^{M} P(r_{i_m})$

Parsingbeispiel

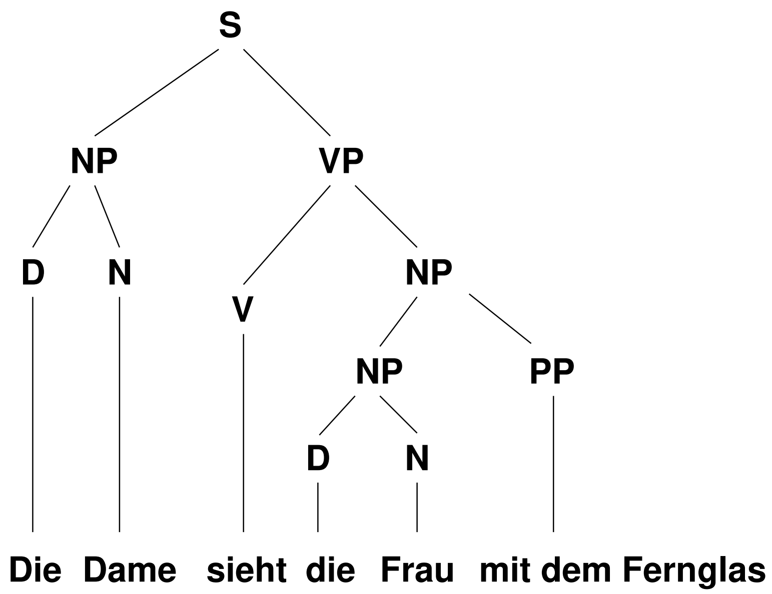
Grammatik:

1	S	$\rightarrow$	$NP\,VP$	(1.0)
2	VP	$\rightarrow$	$V\ NP$	(0.7)
3	VP	$\rightarrow$	$VP\ PP$	(0.3)
4	NP	$\rightarrow$	$D\ N$	(0.8)
5	NP	$\rightarrow$	$NP\ PP$	(0.2)
6	PP	$\rightarrow$	'mit dem Fernglas'	(1.0)
7	V	$\rightarrow$	'sieht'	(1.0)
8	N	$\rightarrow$	'Dame'	(0.4)
9	N	$\rightarrow$	'Frau'	(0.6)
10	D	$\rightarrow$	'die'	(1.0)

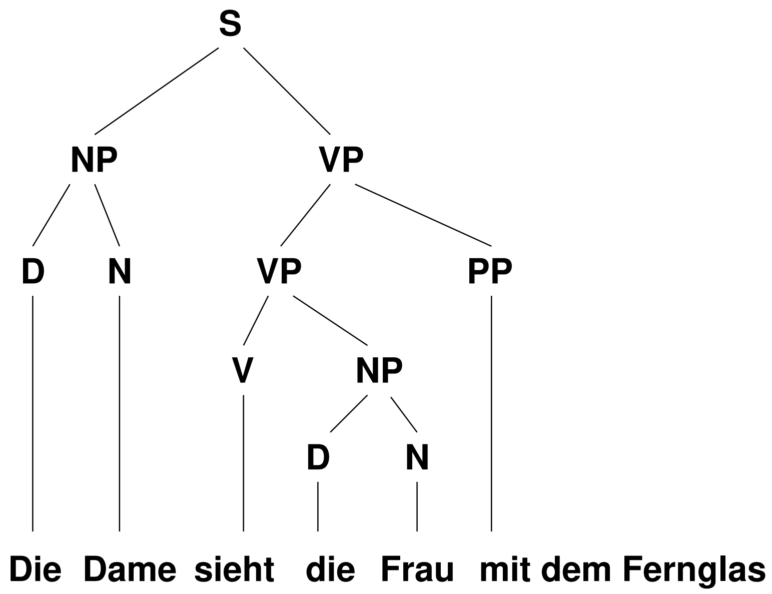


Lesarten von 'Die Dame sieht die Frau mit dem Fernglas'

I:



II:



Die Ableitungen

$S \Rightarrow^* w$ über (ohne lexikalische Ersetzungen):
I:
(S, NP VP, D N VP,
D N V NP,
D N V NP PP, D N V D N PP)
= (1, 4, 2, 5, 4)

II:
(S, NP VP, D N VP,
D N VP PP,
D N V NP PP, D N V D N PP)
= (1, 4, 3, 2, 4)

Welche davon ist wahrscheinlicher?

$$P(I) = (1 \cdot 0.8 \cdot 0.7 \cdot 0.2 \cdot 0.8) = 0.09$$
$$P(II) = (1 \cdot 0.8 \cdot 0.3 \cdot 0.7 \cdot 0.8) = 0.13$$

Notationen zu Parsebäumen

$\tau_1 \circ \tau_2$:

der am weitesten links stehende Nichtterminal-Knoten von τ_1 wird durch τ_2 ersetzt, linker Knoten und Wurzel müssen dabei das gleiche Nichtterminal als Label tragen.

$l(n)$: Label des Knotens n.

$R(\tau)$: Wurzel von τ.

$Y(\tau)$:

“yield“ des Baumes τ, Ergebnis der Regelanwendung, generierter String.

$L(\tau)$:

Label des am weitesten links stehenden Knoten in τ.

Korrespondenz zu partiellen Parsebäumen

Regel $X \rightarrow X_1, ..X_k$ entspricht Teilbaum mit Wurzel X und Knoten $X_1, ..X_k$.

Sei τ_m der Teilbaum zu r_m, $\tau_1, .., \tau_M = \mathbf{t}_M$ die Folge der Teilbäume der Ableitung $r_{i_1}, .., r_{i_M}$. Dabei ist τ_M der vollständige Parsebaum.

Dann ist die Wahrscheinlichkeit eines Parsebaums:

$$P(\tau) = P(\mathbf{t}_M) = \prod_{m=1}^{M} P(\tau_m \mid \mathbf{t}_{m-1})$$

$$P(\tau_m \mid \tau_1 \circ .. \circ \tau_{m-1}) = P(r_m \mid r_1, .., r_{m-1})$$

Gesucht: Extraktor-Funktion g, die die relevanten Eigenschaften von $\mathbf{t}_m$ zur W'keitbestimmung extrahiert, mit:

$$P(\tau_{k+1} \mid \mathbf{t}_k) \approx P(\tau_{k+1} \mid g(\mathbf{t}_k))$$

und das ist der Label des am weitesten links stehenden Knotens im Ergebnis von τ_k, $g(\tau) = L(Y(\tau))$.

Ein Parser für SKFGs

Adaption des Viterbi-Algorithmus, Variante des CYK-Parsers, Zeitkomplexität $O(N^3T^3)$, Speicherkomplexität $O(NT^2)$.

$V_N = X_1, ..X_N, S = X_1, \omega = w_1, ..w_T$, G in CNF (Analyse-W'keiten bleiben erhalten).

Akkumulator $\delta_n(X_i)$, Knoten n, Nichtterminale X_i.

Jeder Knoten n bestimmt einen Teilstring mit den Stringpositionen (s, t), sei $\mathbf{w}_{s,t} = w_{s+1}, .., w_t$ dieser Teilstring.

Die dazu inverse Funktion (nur partiell definiert!) bestimmt aus einem Teilstring denjenigen Parsebaumknoten, der am nächsten zur Wurzel liegt.

$$\delta_{s,t}(X_i) = \text{max}_{\tau:Y(\tau)=\mathbf{w}_{s,t}} P(\tau \mid l(R(\tau)) = X_i),$$
$$1 \leq i \leq N, 0 \leq s < t \leq T$$

W'keit des wahrscheinlichsten Baums mit $\omega = \mathbf{w}_{0T}$:

$$\delta_{0,t}(S) = \text{max}_{\tau:Y(\tau)=\omega} P(\tau \mid l(R(\tau)) = S)$$

Der Baum selbst ist:

$$\text{arg max}_{\tau:Y(\tau)=\omega} P(\tau \mid l(R(\tau)) = S)$$

weiter: Parser

Sei $p_{i \to jk} = P(X_i \to X_j X_k \mid X_i), p_{i \to w} = P(X_i \to w \mid X_i)$.
Konstruktion des Parsebaums, bottom-up:

1. Initialisierung
   $\forall i, t : 1 \le i \le N, 1 \le t \le T$
   $$\delta_{t-1,t}(X_i) = p_{i \to w_t}$$
2. Rekursion
   $\forall i, r, t : 1 \le i \le N, 0 \le r < t \le T$
   $$\delta_{r,t}(X_i) = \max_{\substack{1 \le j \le N \\ 1 \le k \le N \\ r < s < t}} p_{i \to jk} \delta_{r,s}(X_j) \delta_{s,t}(X_k)$$
   $$\begin{bmatrix} \iota_{r,t}(X_i) \\ \kappa_{r,t}(X_i) \\ \sigma_{r,t}(X_i) \end{bmatrix} = \arg\max_{\substack{1 \le j \le N \\ 1 \le k \le N \\ r < s < t}} p_{i \to jk} \delta_{r,s}(X_j) \delta_{s,t}(X_k)$$
3. Rekonstruktion, $n = (s, t)$
   $$\begin{aligned} \text{left}(n) &= \begin{cases} nil & \text{wenn } t - s \le 2 \\ (s, \sigma_{s,t}(l(n))) & \text{sonst} \end{cases} \\ \text{right}(n) &= \begin{cases} nil & \text{wenn } t - s \le 2 \\ (\sigma_{s,t}(l(n)), t) & \text{sonst} \end{cases} \\ l(\text{left}(n)) &= X_{\iota_{s,t}(l(n))} \\ l(\text{right}(n)) &= X_{\kappa_{s,t}(l(n))} \end{aligned}$$

weiter Parser

Begründungen

Initialisierung:

Die Nichtterminale, die die Terminale erzeugen, erhalten die W'keiten der lexikalischen Regeln.

Rekursion:

ι, κ sammeln die Indizes der rechten Seiten der verwendeten Regeln auf, σ speichert die String-Position zwischen X_ι und X_κ.

$\delta_{r,t}(X_i)$ wird aus den W'keiten für X_j und X_k berechnet:

$$
\begin{aligned}
&\text{max}_{\tau:Y(\tau)=\mathbf{w}_{r,t}} P(\tau \mid l(R(\tau)) = X_i) \\
&\quad = \text{max}_{j,k,s}[P(X_i \rightarrow X_j\ X_k \mid X_i) \\
&\qquad \cdot \text{max}_{\tau':Y(\tau')=\mathbf{w}_{r,s}} P(\tau' \mid l(R(\tau')) = X_j) \\
&\qquad \cdot \text{max}_{\tau'':Y(\tau'')=\mathbf{w}_{s,t}} P(\tau'' \mid l(R(\tau')) = X_k)]
\end{aligned}
$$

Parse, Initialisierung

Indizes des Eingabestrings:

0die 1 Dame 2 sieht 3 die 4 Frau 5 mit dem Fernglas6

Initialisierung: $\delta_{t-1,t}(X_i)$

t \ i	1 S	2 NP	3 VP	4 PP	5 V	6 N	7 D
die 1							1
Dame 2						0.4	
sieht 3					1		
die 4							1
Frau 5						0.6	
mit ...6				1			

$\delta_{r,t}$ für NP und VP

$$\delta_{r,t}(X) = \max_{\substack{1 \leq j \leq N \\ 1 \leq k \leq N \\ r < s < t}} p_{i \rightarrow jk} \delta_{r,s}(X_j) \delta_{s,t}(X_k)$$

NP, r \ t	1	2	3	4	5	6
0		$NP \rightarrow D\ N$ $0.8 \cdot 1 \cdot 0.4 = 0.32$				
1						
2						
3					$NP \rightarrow D\ N$ $0.8 \cdot 1 \cdot 0.6 = 0.48$	$NP \rightarrow NP\ PP$ $0.2 \cdot 0.48 \cdot 1 = 0.096$
4						
5						
6						

VP, r \ t	1	2	3	4	5	6
0						
1						
2					$VP \rightarrow V\ NP$ $0.7 \cdot 1 \cdot 0.48 = 0.336$	$VP \rightarrow VP\ PP, VP \rightarrow V\ NP$ $\max(0.3 \cdot 0.336, 0.7 \cdot 0.096)$ $= \max(0.1, 0.067) = 0.1$
3						
4						
5						
6						

Parameterschätzungen für prob. Grammatiken

- wenn ein annotierter Korpus vorliegt:

  relative Häufigkeiten verwenden, Achtung bei spärlichen Daten

- wenn nicht:
  - Zuerst alle gültigen Parsebäume erzeugen, dann eine Gleichverteilung für alle Bäume pro Satz annehmen.

    Häufigkeitszählung der verwendeten Regeln (nach linkem Symbol), gewichtet mit der W'keit des Baumes: ergibt neue Verteilung für die Regeln, daraus neue Verteilung für die Bäume; das kann beliebig wiederholt werden.

    Problem: Komplexität, Anzahl der Parsebäume exponentiell zur Stringlänge.
  - *Inside-Outside*-Algorithmus

Parameterschätzungen für prob. Grammatiken

Inside-Outside-Algorithmus

Idee: Verwende die aktuellen W'keiten der Regeln, um davon abhängige andere Masse einzuschätzen.

Ziel: Finde die Menge von Regel-W'keiten, mit denen es am ehesten möglich ist, den Trainings-Korpus zu generieren.

Die Variablen:

Die *Inside*-W'keit $I_i^\omega(s,t)$ schätzt die W'keit

$$P(X_i \Rightarrow^* \mathbf{w}_{s,t} \mid X_i),$$

dass $\mathbf{w}_{s,t}$ abgeleitet wird, wenn X_i vorliegt.

Die *Outside*-W'keit $O_i^\omega(s,t)$ schätzt die W'keit

$$P(S \Rightarrow^* \mathbf{w}_{0,s} X_i \mathbf{w}_{t,T} \mid S),$$

dass die o.a. Zeichenkette von S abgeleitet wird, wenn S vorliegt.

- Inside-Variablen-Initialisierung:

$\forall i, t : 1 \leq i \leq N, 1 \leq t \leq T$

$$I_i^\omega(t-1, t) = p_{i \longrightarrow w_t}$$

- Inside-Variablen-Rekursion

$\forall i, r, t : 1 \leq i \leq N, 0 \leq r < t \leq T$

$$I_i^\omega(r, t) = \sum_{j,k=1}^{N} \sum_{r<s<t} p_{i \longrightarrow jk} \cdot I_j^\omega(r, s) \cdot I_k^\omega(s, t)$$

- Outside-Variablen-Rekursion

$\forall i, t : 1 \leq i \leq N, 1 \leq t \leq T$

$$O_i^\omega(r, t) = \Sigma_{j,k=1}^{N} \quad (\Sigma_{s=0}^{r-1} p_{j \longrightarrow ki} \cdot O_j^\omega(s, t) \cdot I_k^\omega(s, r) + \Sigma_{s=t+1}^{T} p_{j \longrightarrow ik} \cdot O_j^\omega(r, s) \cdot I_k^\omega(t, s))$$

Herleitung

Inside:

$$P(X_i \Rightarrow^* \mathbf{w}_{r,t} \mid X_i) =$$
$$\Sigma_{j,k=1}^{N} P(X_i \Rightarrow X_j X_k \Rightarrow^* \mathbf{w}_{r,t} \mid X_i) =$$
$$\Sigma_{j,k=1}^{N} \Sigma_{r<s<t} P(X_i \Rightarrow X_j X_k \mid X_i) \cdot P(X_j \Rightarrow^* \mathbf{w}_{r,s} \mid X_j) \cdot P(X_k \Rightarrow^* \mathbf{w}_{s,t} \mid X_k)$$

Outside:

$$P(S \Rightarrow^* \mathbf{w}_{0,s} X_i \mathbf{w}_{t,T} \mid S) =$$
$$\Sigma_{j,k=1}^{N} \Sigma_{s=0}^{r-1} P(S \Rightarrow^* \mathbf{w}_{0,s} X_j \mathbf{w}_{t,T} \Rightarrow^* \mathbf{w}_{0,s} X_k X_i \mathbf{w}_{t,T} \Rightarrow^* \mathbf{w}_{0,s} \mathbf{w}_{s,r} X_i \mathbf{w}_{t,T}) +$$
$$\Sigma_{s=t+1}^{T} P(S \Rightarrow^* \mathbf{w}_{0,r} X_j \mathbf{w}_{s,T} \Rightarrow^* \mathbf{w}_{0,r} X_i X_k \mathbf{w}_{s,T} \Rightarrow^* \mathbf{w}_{0,r} X_i \mathbf{w}_{t,s} \mathbf{w}_{s,T}) =$$
$$\Sigma_{j,k=1}^{N} \Sigma_{s=0}^{r-1} P(S \Rightarrow^* \mathbf{w}_{0,s} X_j \mathbf{w}_{t,T}) P(X_j \Rightarrow^* X_k X_i \mid X_j) P(X_k \Rightarrow^* \mathbf{w}_{s,r} \mid X_k) +$$
$$\Sigma_{s=t+1}^{T} P(S \Rightarrow^* \mathbf{w}_{0,r} X_j \mathbf{w}_{s,T} P(X_j \Rightarrow^* X_i X_k \mid X_j) P(X_k \Rightarrow^* \mathbf{w}_{t,s} \mid X_k) =$$

Komplexität für beide Variablen $O(N^3 T^3)$.

Wiedereinschätzungsgleichung

Idee:

$$p_{i \to w} = P(X_i \to w \mid X_i) = \frac{P(X_i \to w)}{P(X_i)}$$

$$p_{i \to jk} = P(X_i \to X_j X_k \mid X_i) = \frac{P(X_i \to X_j X_k)}{P(X_i)}$$

mit:

$$P(X_i) = \frac{1}{\mid W \mid} \sum_{\omega \in W} \frac{P_i^\omega}{P^\omega},$$

P^ω die Wahrscheinlichkeit des Strings ω, P_i^ω die W'keit des Strings ω und das Vorkommen des Nicht-terminals X_i ind dessen Ableitung.

$$P(X_i \to w) = \frac{1}{\mid W \mid} \sum_{\omega \in W} \frac{\sum\limits_{1 \leq t \leq T, w_t = w} O_i^\omega(t-1, t) \cdot p_{i \to w}}{P^\omega}$$

$$P(X_i \to X_j X_k) =$$

$$\frac{1}{\mid W \mid} \sum_{\omega \in W} \frac{\sum\limits_{0 \leq r < s < t \leq T} O_i^\omega(r, t) \cdot p_{i \to jk} \cdot I_j^\omega(r, s) \cdot I_k^\omega(s, t)}{P^\omega}$$

Stochastische Verfahren für Semantik

Beispiel: R.Bonnema, R.Bod, R. Scha: A DOP Model for Semantic Interpretation, ACL '97

http://earth.let.uva.nl/ bonnema/acl-DOP-sem/acl-DOP-sem.html

Idee:*Data-oriented Processing* auf Semantik erweitern

1. Was ist DOP?

   Korpus mit syntaktisch annotierten, d.h. gelabelten Bäumen, als Operation darauf Komposition $\circ$. Neue Eingabe wird analysiert, indem vorhandene Teilbäume komponiert werden.

   Model: Einschätzung der W'keit, dass ein Teibaum t an einer bestimmten Stelle eingestzt wird: Anzahl der Vorkommen von t im Korpus / Anzahl aller Teilbäume mit dem gleichen Wurzellabel:

   $$P(t) = \mid t \mid / \sum_{t':R(t')=R(t)} \mid t' \mid$$

   W'keit einer Ableitung:

   $$P(t_1 \circ ...t_n) = \prod_i (P(t_i)$$

   W'keit eines Baumes, t_{id} der i-te Teilbaum in der Ableitung d von Baum T:

   $$P(T) = \sum_d \prod_i P(t_{id})$$

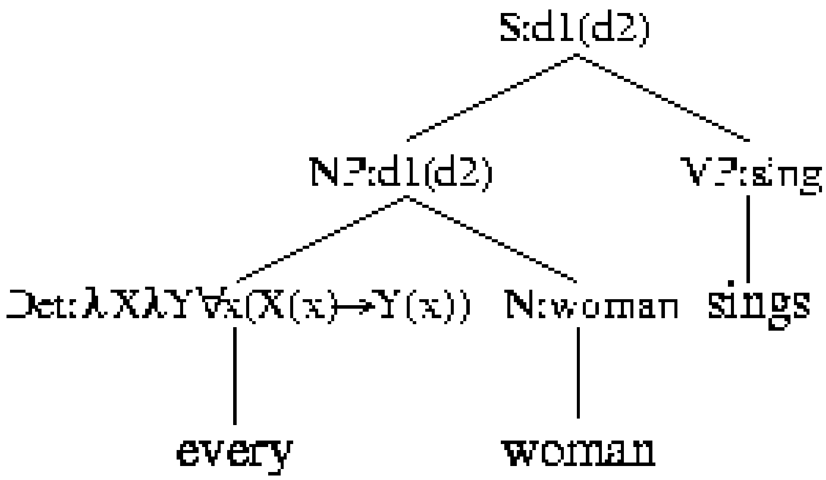
2. Erweiterung um semantische Annotationen Gewählter Semantik-Formalismus: Extensionale Typentheorie (Prädikatenlogik höherer Ordnung, Typen, λ-Abstraktion)

   Jeder syntaktisch annotierte Knoten wird jetzt ebenfalls semantisch annotiert:

   (pre-)lexikalische mit typen-logischer Formel, die die Bedeutung repräsentiert.

   nicht-lexikalische mit Formel-Schema, mit dem man die Formeln der Tochterknoten zusammensetzen kann.

   Problem: kompositionale Semantik angenommen



   Modifikation der Komposition bz. Extraktion:

   Wenn ein Teilbaum extrahiert wird, wird das entstehende Blatt im Oberbaum mit einer Unifikationsvariablen gelabelt. Bei Komposition wird sie mit der Formal am eingefügten Wurzelknoten unifiziert.

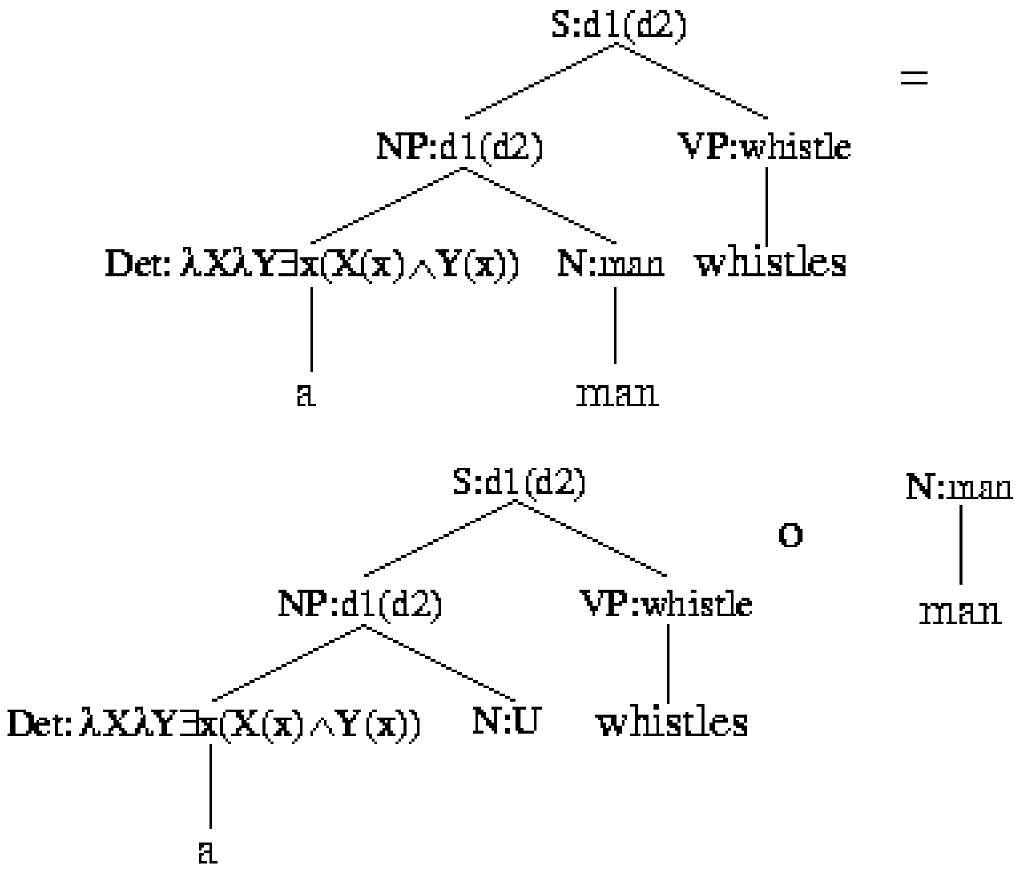


Figure: Decomposing a tree into subtrees with unification variables.

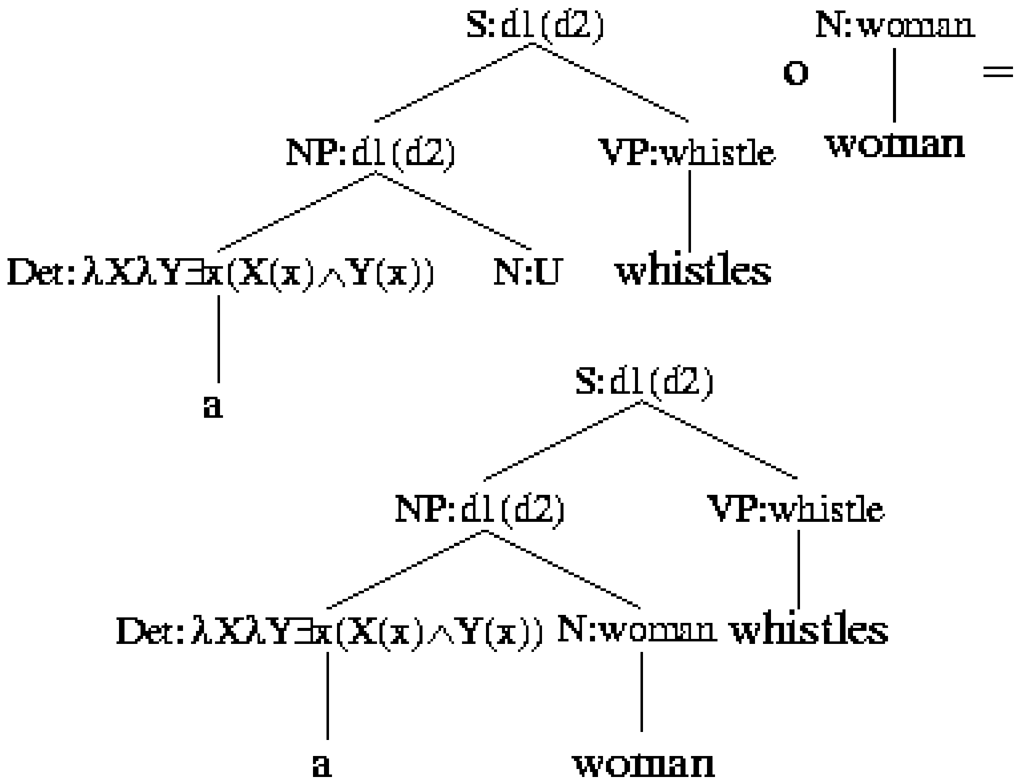


Figure: Generating an analysis for ''*A woman whistles*''.

3. Statistisches Modell

gleich dem syntaktischen:

$$P(t) = \mid t \mid / \sum_{t':R(t')=R(t)} \mid t' \mid$$

W'keit einer Ableitung:

$$P(t_1 \circ ...t_n) = \prod_i (P(t_i)$$

$$P(T) = \sum_d \prod_i P(t_{id})$$

Interpretation eines Strings: Semantische Annotation der Wurzel des Parsebaums.

W'keit einer Interpretation, p diejenigen Parses, die eine äquivalente Interpretation liefern:

$$P(I) = \sum_p \sum_d \prod_i P(t_{idp})$$