

Automatisches Beweisen

Seminar Künstliche Intelligenz

Dennis Ziska (WI4406)

Übersicht

- 1. allgemeine Problemstellung
- 2. Grundlagen
 - Logiken
 - Substitution, Unifikation
 - Formalisierung, Normalformen
 - Skolemisierung
 - Klauselsprache
- 3. Methoden
 - Tableaumethode
 - Resolutionskalkül
- 4. Konklusion

1. Allgemeine Problemstellung

Was bedeutet „automatisches Beweisen“?

=> Mathematische Beweise mittels
Computerprogramme führen

=> Mathematische Wahrheiten formal
modellieren und mit einer endlichen
Anzahl von Axiomen und Regeln formal
beweisen

1. Allgemeine Problemstellung

Genereller Ablauf:

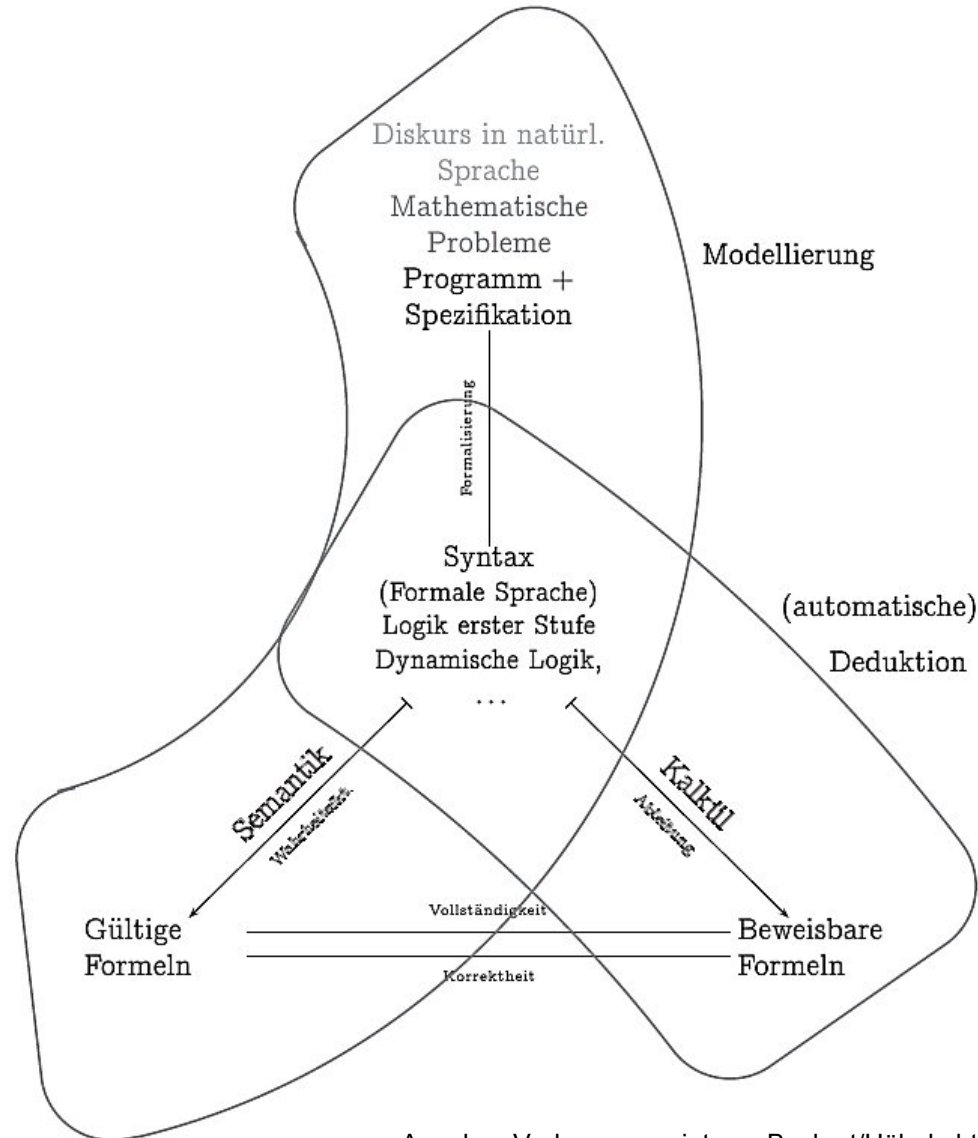
1. Modellierung:

- Mathematische Probleme formalisieren (logische Syntax)
- Jeder Aussage einen Wahrheitswert zuordnen (Semantik)
- Syntax und Semantik definieren Logik

2. Deduktion:

- Jede gültige Aussage ableitbar \Rightarrow Vollständigkeit
- Jede abgeleitete Aussage wahr \Rightarrow Korrektheit
- Menge aller Ableitungsregeln = Kalkül

1. Allgemeine Problemstellung



2. Grundlagen

2.1 Logiken

2.1.1 Aussagenlogik:

- Zeichen: $\neg, \wedge, \vee, \rightarrow, \Leftrightarrow, (,)$
- Aussagen sind unteilbar $(a, \neg a) \Rightarrow$ Atom, Literal
- Werte: true, false
- Signatur: $\Sigma = \{p_0, p_1, \dots\}$

2. Grundlagen

2.1 Logiken

2.1.2 Prädikatenlogik 1.Stufe

- neue Elemente: Quantoren \forall, \exists ; k-stellige Aussagen; Funktionen
- Signatur: $\Sigma = \langle F_\Sigma, P_\Sigma, \alpha_\Sigma \rangle$
- Formel: Verknüpfung von Variablen, Funktionen und Prädikaten mit Operatoren und Quantoren
- Variablen sind frei oder gebunden an Quantoren
- Geschlossene Formeln: keine freien Variablen
- Offene Formeln: keine gebundenen Variablen
- Beispiel: $\forall x(K(x,y) \wedge R(y,z) \vee \exists y K(x,y)) \wedge \neg R(x,y)$

2. Grundlagen

2.2 Substitution und Unifikation

- Substitution σ ist die Abbildung einer endlichen Menge von Variablensymbolen auf eine Menge von Termen
- Notation: $\{x_1/\sigma(x_1), \dots, x_n/\sigma(x_n)\}$ mit $\sigma(x_i) \neq x_i$
- Anwendung von σ auf Terme, Literale und Klauseln ist durch Konventionen definiert
- Beispiel: $\sigma = \{x/a, y/f(ax)\}$ für $t = g(xh(yz))$ gilt:
$$\sigma(t) = g(\sigma(x) \sigma(h(yz))) = g(ah(\sigma(y) \sigma(z))) = g(ah(f(ax)z))$$

2. Grundlagen

2.2 Substitution und Unifikation

- Unifikation ist eine spezielle Substitution
- Unifikator = Substitution, für die gilt: $\sigma f_1 = \sigma f_2$
- Beispiel: $D = \{f(x,a), f(z,y)\}$ ein Unifikator: $\sigma = \{x/c, y/a, z/c\}$
- Es gibt meistens eine Menge von Unifikatoren
- ein allgemeinsten Unifikator μ (mgu) besteht, wenn für alle Unifikatoren σ gilt: $\sigma = \rho \bullet \mu$; ρ .. Substitution
- Beispiel: $D = \{f(x,a), f(z,y)\}$, $\mu = \{x/z, y/a\}$
- Jede Problemstellung hat mgu oder ist nicht unifizierbar

2. Grundlagen

2.3 Normalformen

2.3.1 Negations- Normalform NNF:

- Enthält weder \rightarrow noch \leftrightarrow
- \neg nur vor Atomen
- Erreichbar durch Prädikatenlogik (Ersetzung und DeMorgan'sche Regeln)
- Grundform für weitere Schritte

2. Grundlagen

2.3 Normalformen

2.3.2 Pränexe Normalform PNF:

- Form: $\Delta_1 \times 1 (\Delta_2 \times 2 (\dots \Delta_n \times n B))$, $\Delta \in \{\forall, \exists\}$, B..Matrix = quantorenfreie Formel
- Berechenbar aus NNF: Quantoren über Umformung nach Außen ziehen
- Ausgangsform für Skolemisierung

2. Grundlagen

2.3 Normalformen

2.3.3 Konjunktive Normalform KNF:

- Form: $\bigwedge_{i=1..n}, \bigvee_{j=1..m}, L_{ij}$
- Formeln sind quantorenfrei
- Berechenbar aus NNF über Distributivgesetze

2. Grundlagen

2.3 Normalformen

2.3.4 Skolem Normalform SNF:

- Zwischenform
- Über Skolemisierung werden alle Existenzquantoren eliminiert
- Voraussetzung: Formel in PNF
- anschließende Überführung in KNF
- **Achtung:** Skolemisierung ist nicht äquivalenzerhaltend!

2. Grundlagen

2.4 Skolemisierung

- Voraussetzung: Formel in PNF
- Ziel: Existenzquantoren eliminieren
- Vorgehen: vom Quantor gebundene Variablen durch Skolemterme ersetzen
- Skolemterm (-funktion): $f(x_1 \dots x_k)$, neues k -stelliges Funktionssymbol
- Nachteile:
 - Bsp. $\forall x((\exists y p(y)) \vee q(x)) \Rightarrow \forall x(p(f(x)) \vee q(x))$ Einführung von x nicht nötig
 - Bsp. $(\exists x p(x)) \vee (\exists y p(y)) \Rightarrow p(c) \vee p(d)$ zwei verschiedene Skolemkonstanten sind unnötig

2. Grundlagen

2.4 Skolemisierung

- Verbesserung:
 - Skolemisierung ohne PNF
 - Feste Zuordnung Skolemsymbole zu Formeln
- Vorteile:
 - Vereinfachung durch Verwendung von gleichen Skolemsymbolen (Einschränken des Suchraumes)
 - Ergebnis hat polynomielle Größe

2. Grundlagen

2.5 Klauselsprachen

- Teilsprache der Prädikatenlogik
- Mengendarstellung von Ausdrücken in KNF
- Allquantifizierte Disjunktion repräsentiert durch Mengen von Atomen
- Konjunktionen repräsentiert durch Mengen dieser Mengen
- Klausel besteht aus endlicher Menge von Literalen => eine Klausel ist eine Menge
- Leere Klausel bedeutet Widerspruch

3. Methoden

3.1 Tableaumethode

- Widerspruchsbeweis durch Fallunterscheidung
- Tableau T für Formelmenge M ist ein Baum
- Baumstruktur: $|M|$ Knoten, endlichverzweigt
- Beispiel: Problem aus elementarer Mengenlehre: (1)-(4) \rightarrow (5)

$$\begin{array}{lclcl} (1) & S \cap Q & = & \emptyset & \\ (2) & P & \subseteq & Q \cup R & \\ (3) & P = \emptyset & \rightsquigarrow & Q \neq \emptyset & \\ (4) & Q \cup R & \subseteq & S & \\ \hline (5) & P \cap R & \neq & \emptyset & \end{array}$$

3. Methoden

3.1 Tableaumethode

- Ziel: nach Betrachten aller Regeln nur Widersprüche finden
- Widerspruch, wenn komplementäres Paar am Zweig => Zweig ist geschlossen
- Tableau heißt geschlossen, wenn alle Zweige geschlossen => Ziel erreicht

3. Methoden

3.1 Tableaumethode

- Grundtableaus: ersetzen gebundene Variablen durch Grundterme (vgl. Skolemisierung)
- Folge: nur geschlossene Formeln auf Grundtableau
- Nachteil: richtige Instanz zum Abschluss bei Regelanwendung nicht bekannt => extrem redundante Tableaus

- Beispiel: (1) $\neg \exists x (s(x) \wedge q(x))$
(2) $\forall x (p(x) \rightarrow (q(x) \vee r(x)))$
(3) $\neg \exists x (p(x)) \rightarrow \exists y (q(y))$
(4) $\forall x ((q(x) \vee r(x)) \rightarrow s(x))$
(Annahme) $\neg \exists x (p(x) \wedge r(x))$

3. Methoden

3.1 Tableaumethode

Betrachten der Formeln
in der Reihenfolge:

(3)

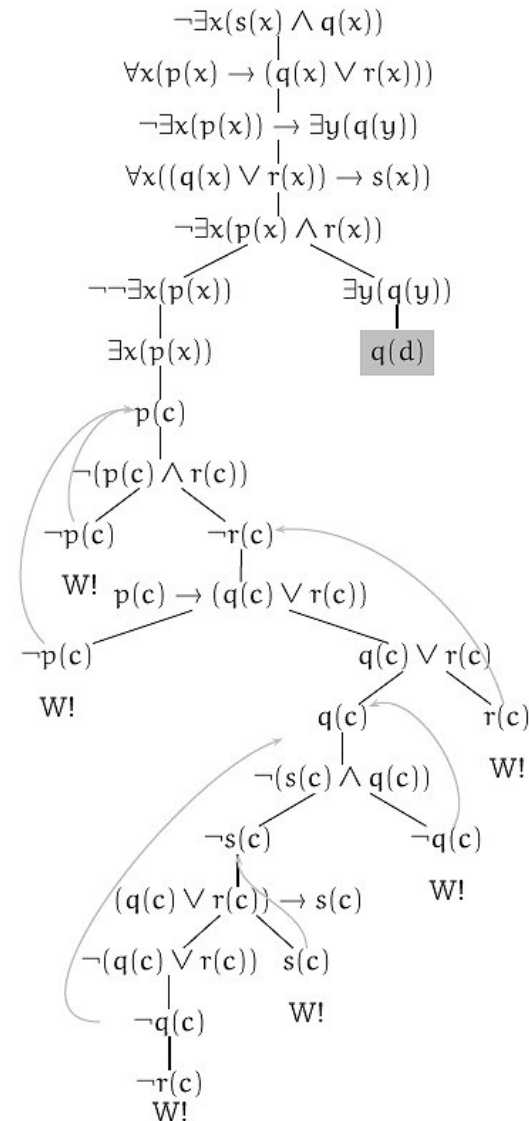
Annahme

(2)

(1)

(4)

Die Bögen zeigen die Widersprüche
bzw. die Abschlusspaare an.



3. Methoden

3.1 Tableaumethode

- Verbesserung der Größe durch freie-Variablen Tableaus (FV-Tableaus)
- Freie Variablen werden erzeugt, die bei Bedarf über Unifikation instantiiert werden
- FV-Tableaus sind vollständig, folgt aus Vollständigkeit von Grundtableaus
- Zu jedem geschlossenen Grundtableau existiert ein FV-Tableau
- Alle freien Variablen sind implizit allquantifiziert
- FV-Tableau ist erfüllbar, wenn es eine Interpretation D gibt, die für alle Belegungen auf mind. einem Zweig alle Formeln erfüllt

3. Methoden

3.1 Tableaumethode

Indeterminismus:

Problem: Vollständigkeit der Tableauregeln garantiert nur die Existenz eines geschlossenen Tableaus.

Es muss aber auch gefunden werden!

4 Typen von Indeterminismus:

1. Auswahl des zunächst zu betrachtenden Zweiges
2. Gewählten Zweig abschließen oder erweitern?
3. Bei Abschluss welche Formel? (mehrere mögliche Abschlusspaare)
4. Bei Erweitern welche Formel wählen?

3. Methoden

3.1 Tableaumethode

Bevorzugungsproblem (Fairness):

- Bevorzugung eines bestimmten Literals
gleiches Literal für Abschluss => Abschluss der übrigen Äste verhindert
- Bevorzugung einer bestimmten Formel
Gleiche Formel expandieren => kann zu Unvollständigkeit führen
- Bevorzugung eines Modus
z.B. Abschluss bevorzugen => kann zu Unvollständigkeit führen
- Lösung: faire Suchverfahren, z.B. Breitensuche, Backtracking, inkrementelle Tiefensuche...
- Vorteil von Fairness: Kalküle werden beweiskonfluent

3. Methoden

3.1 Tableaumethode

Optimierungen:

- Formeln nicht auswählen, die unabhängig vom bisherigen Beweis sind

Unabhängigkeit wenn:

- keine Unterformeln mit entgegengesetzter Polarität
 - Kein komplementäres Paar durch Substitution
- Wenn möglich Formeln generalisieren

3. Methoden

3.1 Tableaumethode

Klauseltableaus:

- Vorbedingung: Formeln in SNF
- Vorteile:
 - Tableauprozedur und Vollständigkeitsbeweis werden einfacher und uniformer
 - Besserer Vergleich zu Resolution möglich
 - Gute Umsetzung in PROLOG möglich
- Nachteile:
 - Verständlichkeit der Tableaubeweise leidet
 - Nicht alle nichtklassischen Logiken in SNF darstellbar

3. Methoden

3.1 Tableaumethode

Klauseltableaus:

- Startklausel S nicht im initialen Tableau
- Klauseltableau für S : Baum mit genau einem Knoten S
- Zweig B mit n Nachfolgern von B erweitern \Rightarrow neues Klauseltableau für S
- Zweig ist eine Liste von Einerklauseln (nicht Literale)
- Modelle können unendliches Universum besitzen \Rightarrow unendliche Modelle
- Klauseltableaus stark verfeinerbar

3. Methoden

3.2 Resolution

- Dominiert den Bereich automatisches Beweisen
 - Unifikation zentraler Bestandteil der PL-Resolution
 - Einschränkung auf SNF => eine Inferenzregel für Beweisprozedur genügt
 - Beweiskonfluent und Fairness leicht zu erreichen
- Grundidee: ein Literal aus zwei Elternklauseln resolvieren
- Nutzt Klauselsprache
- Schlussregeln: Resolution und Faktorisierung
- Liste von Klauseln zur Herleitung in R

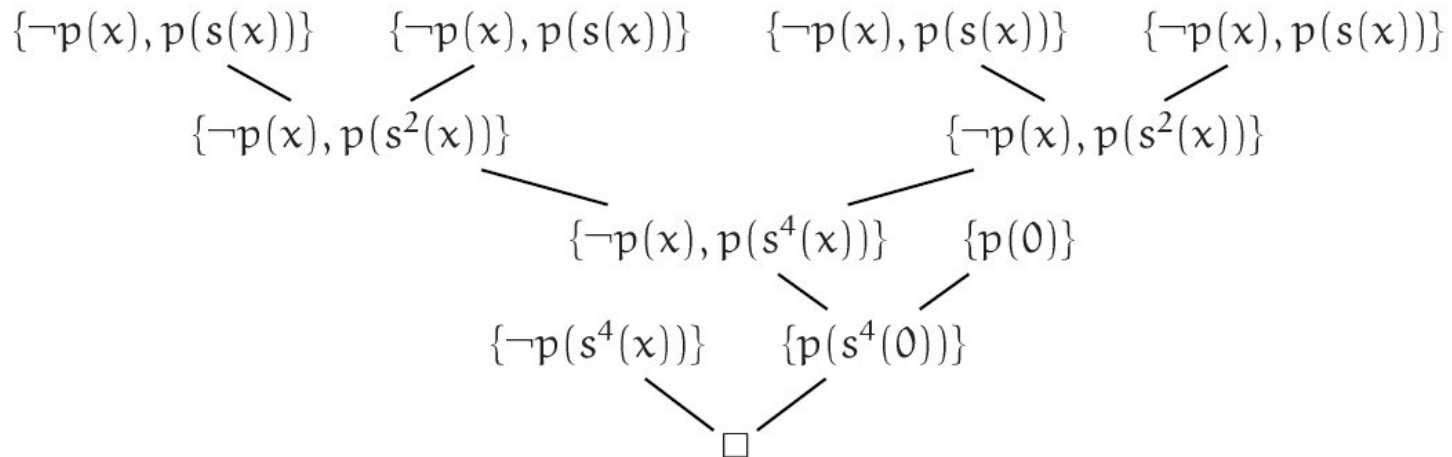
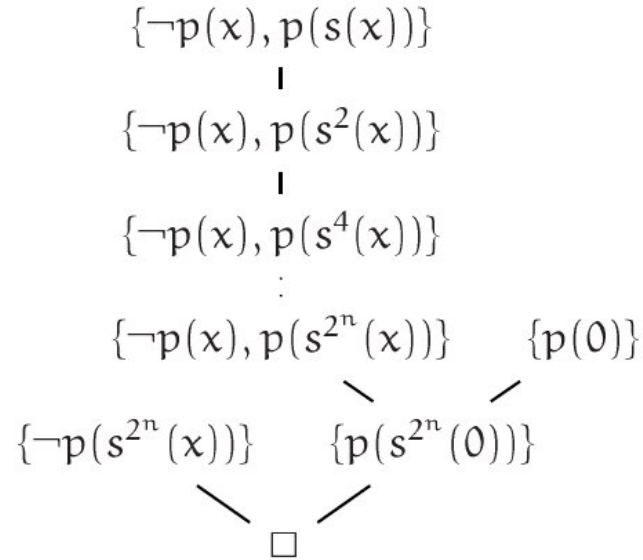
3. Methoden

3.2 Resolution

- Resolutionsregel:
 - Logik wie Schnittregel
 - Aber: Schnittformel ist bekannt
 - \Rightarrow Resolutionsableitung kann ein DAG (gerichteter antizyklischer Graph) sein
- Durch geeignete Substitution auf Elternklauseln ein komplementäres Paar erzeugen
- PL-Resolvente: $\text{Res}(C,L,D,K,\sigma) = \sigma(C-L) \cup \sigma(D-K)$
- Beispiel: $C1 = \{P(xb), P(by), \neg Q(xy)\}$ $C2 = \{\neg P(az), Q(zz)\}$
 $\Rightarrow R1 = \{P(by), \neg Q(ay), Q(bb)\} = \text{Res}(C1, P(xb), C2, \neg P(az), \{x/a, z/b\})$
oder kurz mit Index: $\text{Res}(C1, 1, C2, 1, \{x/a, z/b\})$

3. Methoden

3.2 Resolution



3. Methoden

3.2 Resolution

- Faktorisationsregel:
- Durch geeignete Substitution in einer Klausel ein äquivalentes Paar zu bilden
- Beispiel: $C1 = \{P(xb), P(by), \neg Q(xy)\}$ $C2 = \{\neg P(az), Q(zz)\}$
 $\Rightarrow F1 = \{P(bb), \neg Q(bb)\} = \text{Fak}(C1, \{x/b, y/b\})$
- PL-Resolution und Faktorisierung zusammenlegbar
 - Weniger Schritte zu Lösungsfindung
 - Weniger Übersichtlichkeit

3. Methoden

3.2 Resolution

- Festlegung: Klauseln sind variablenfremd
- Umsetzung durch Variablenumbenennung
- Sinn: Vermeiden von Unifikationskonflikten
- Beispiel einer R-Widerlegung nach Umbenennung:
 - (C1) $\{\neg P(x_1), P(f(x_1))\}$
 - (C2) $\{P(a), Q(b)\}$
 - (C3) $\{\neg Q(y_1), \neg Q(z_1), P(a)\}$
 - (C4) $\{\neg P(f(a))\}$
 - (C5) $\{\neg Q(y_2), P(a)\}$ Fak(C3, $\{z_1/y_1\}$)
 - (C6) $\{P(a)\}$ Res(C2, 2, C5, 1 $\{y_2/b\}$)
 - (C7) $\{P(f(a))\}$ Res(C1, 1, C6, 1 $\{x_1/a\}$)
 - (C8) \square Res(C4, 1, C7, 1, ε)

4. Konklusion

- Resolutionsmethoden praktisch häufig eingesetzt
- Teilweise Probleme bei Tableaumethoden bzgl. Speicherbedarf und Laufzeit
- Automatisches Beweisen ist ein weitläufiges Gebiet
- Denkbar viele Ansätze, z.B. Nichtklauselresolution, Inverse Methode, Dissolution...
- Noch Raum für Forschung

Literaturverzeichnis

- Handbuch der Künstlichen Intelligenz, Görz et al.
3.Auflage, Oldenbourg, 2000
- Vorlesungsscript von Beckert/Hähnle von
<http://i12www.ira.uka.de/~beckert/Lehre/Automatisches-Beweisen/>
- <http://wikipedia.org>
- <http://www.computerbase.de/lexikon/>