# Grundlagen der Künstlichen Intelligenz - Informatik

Rapp, DHBW Lörrach

13.10.2023

# Inhaltsübersicht

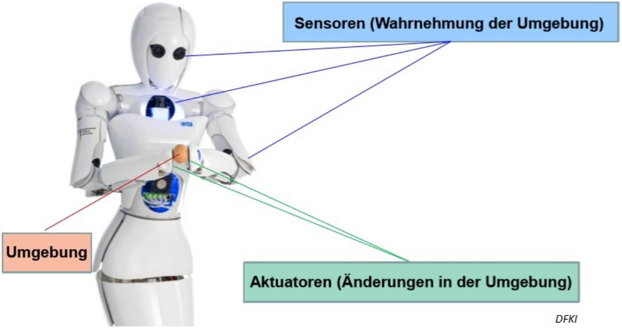
# Lernziele

## Meine 3 Lernziele für heute

1. Ich kenne den grundlegenden Aufbau eines wissensbasierten Agentenprogramms in der deklarativen Systementwicklung.
2. Mir sind wesentliche Konzepte von Syntax und Semantik der Aussagen- und Prädikatenlogik bekannt.
3. Ich bin mit dem Prozess zur Ableitung und Schlussfolgerung von explizitem aus implizitem Wissen in Ontologien vertraut.

# Agent - Typen

Mit den **Sensoren** kann der Agent Wahrnehmungen aus der Umgebung aufnehmen, mit den **Aktuatoren** Aktionen ausführen.




- Künstlicher Agent in physischer Welt = **Roboter**
- Künstlicher Agent in einer virtuellen Welt = **Softbot**

# Begriffsdefinitionen

**Wahrnehmung**

- Wahrgenommene Eingaben des Agenten zu einem beliebigen Zeitpunkt

**Wahrnehmungsfolge**

- Vollständiger Verlauf von allem, was der Agent je wahrgenommen hat

**Agentenfunktion**

- Die mathematische Agentenfunktion $f$ bildet jede beliebige Wahrnehmungsfolge auf eine Aktion ab
- Charakterisierung durch z.B. (ggf. unendlich große) Tabelle

**Agentenprogramm**

- Konkrete interne Implementierung der Agentenfunktion
- Läuft auf einem physischen System bzw. einer Architektur

**Methodische Herangehensweise**

Die Vorgehensweise der künstlichen Intelligenz besteht aus dem Entwurf eines Agentenprogramms, das die Agentenfunktion implementiert.

**Agent = Architektur + Agentenprogramm**

# Beispiel Staubsaugerwelt

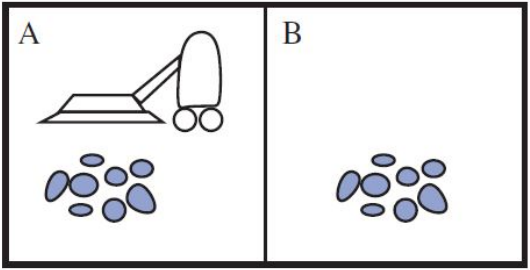



Abb.: Eine Staubsaugerwelt mit nur zwei Positionen.

- Der Staubsauger-Agent nimmt wahr, in welchem Quadrat er sich befindet und ob Schmutz in dem Quadrat liegt.
- Er kann entscheiden, sich nach links oder rechts zu bewegen, den Schmutz aufzusaugen oder nichts zu tun.

# Gutes Verhalten in der Staubsaugerwelt

Wenn das aktuelle Quadrat schmutzig ist, soll gesaugt werden, andernfalls soll eine Bewegung zum anderen Quadrat erfolgen:

| Wahrnehmungsfolge | Aktion |
|---|---|
| [*A*, *Sauber*] | *Rechts* |
| [*A*, *Schmutzig*] | *Saugen* |
| [*B*, *Sauber*] | *Links* |
| [*B*, *Schmutzig*] | *Saugen* |
| [*A*, *Sauber*], [*A*, *Sauber*] | *Rechts* |
| [*A*, *Sauber*], [*A*, *Schmutzig*] | *Saugen* |
| ... | ... |
| [*A*, *Sauber*], [*A*, *Sauber*], [*A*, *Sauber*] | *Rechts* |
| [*A*, *Sauber*], [*A*, *Sauber*], [*A*, *Schmutzig*] | *Saugen* |
| ... | ... |

*Ausschnitt aus einer tabellarischen Darstellung einer einfachen* **Agentenfunktion** *für die Zuordnung einer Wahrnehmungsfolge auf Aktionen in der Staubsaugerwelt.*

**„Gutes“ Verhalten:** Die **beste Agentenfunktion** ist diejenige, für die jeder Eintrag in der Tabelle **korrekt ausgefüllt** ist.

# Rationaler Agent in der PEAS Beschreibung

1. **(P)erformance Element**
   - Leistungsmaß für die Definition des Erfolgskriteriums
2. **(E)nvironment**
   - Vorwissen des Agenten über die Umgebung
3. **(A)ktuators**
   - Aktionen, die der Agent durchführen kann
4. **(S)ensors**
   - Bisherige Wahrnehmungsfolge des Agenten

**Rationaler Agent**

Ein **rationaler Agent** soll für jede mögliche Wahrnehmungsfolge eine **Aktion** auswählen, von der erwartet werden kann, dass sie seine **Leistungsmaß maximiert**, wenn man seine **Wahrnehmungsfolge** sowie vorhandenes Wissen, über das er verfügt, in Betracht zieht.

# Beispiel Spamfilter

**Ziel des Spamfilters:** 1000 Emails in die richtige Klasse einteilen.

**Agent 1**

| | | korrekte Klasse | |
|---|---|---|---|
| | | erwünscht | SPAM |
| Spamfilter entscheidet | erwünscht | 189 | 1 |
| | SPAM | 11 | 799 |

**Agent 2**

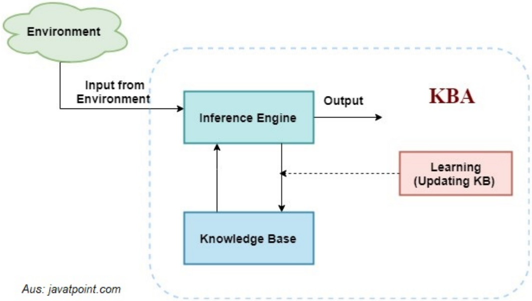
| | | korrekte Klasse | |
|---|---|---|---|
| | | erwünscht | SPAM |
| Spamfilter entscheidet | erwünscht | 200 | 38 |
| | SPAM | 0 | 762 |

**Fragestellung:** Welcher Agent ist besser?

# Wissensbasierter Agent

# Wissensbasierter Agent

Agent, der betrachtet werden kann, als würde er etwas über die **Welt wissen** und über mögliche **Aktionsfolgen schlussfolgern:**




**Vorteile gegenüber problemlösenden Agenten**

- Neue Aufgabe als explizit beschriebene Ziele
- Schneller Kompetenzerwerb durch Mitteilung oder Lernen
- Adaption an Umweltänderungen durch Wissensadaption

# Wissensbasiertes Agentenprogramm

## Grundstruktur wissensbasierter Agentenprogramme

```
function KB-AGENT(percept) returns eine action
  persistent: KB, eine Wissensbasis
              t, ein Zähler, anfangs 0, gibt die Zeit an
  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
  action ← ASK(KB, MAKE-ACTION-QUERY(t))
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))
  t ← t + 1
  return action
```

Der Agent fügt seiner Wissensbasis eine gegebene Wahrnehmung hinzu, fragt die Wissensbasis nach der besten Aktion ab und teilt der Wissensbasis mit, dass er die Aktion tatsächlich ausgeführt hat.

**Deklarative Systementwicklung**

Wissensbasis wird aufgebaut, indem dem System ein Satz nach dem anderen mitgeteilt wird bzw. es Sätze aufgrund von Wahrnehmungen lernt.

# Allgemeine Komponenten eines Agenten

- **Wissensbasis** *(Knowledge Base, KB)*
  Enthält Repräsentationen von Fakten über die Welt.
- **Satz** *(Sentence)*
  Eine individuelle Repräsentation.
- **Wissensrepräsentation** *(Knowledge Representation Language)*
  Sprache, in der die Sätze formuliert werden.
- **Tell und Ask**
  Hinzufügen neuer Sätze und Abfragen von Sätzen, die aus dem vorhandenen Wissen folgen.
- **Inferenzmechanismus**
  Neue Sätze werden aus den bekannten Sätzen hergeleitet.
- **Hintergrundwissen**
  Apriori vorhandenes Wissen in der Wissensbasis

# Google's Natural Language Understanding examples

**Youtube-Links**

- Watch Google's AI **LaMDA** program talk to itself (2021)
  **https://www.youtube.com/watch?v=aUSSfo5nCdM**
- Google **Duplex:** A.I. Assistant Calls Local Businesses To Make Appointments (2018)
  **https://www.youtube.com/watch?v=D5VN56jQMWM**

# Ontologie

# Definition in der Philosophie

**Ontologie**

Die Ontologie ist die Lehre des Seins.

*Aus: Aristoteles, Metaphysik, IV, 1*

Die Ontologie ist eine Disziplin der Philosophie, die sich mit der **Einteilung des Seienden** und den **Grundstrukturen der Wirklichkeit** befasst.

# Definition in der Informatik

**Ontologie**

Sprachlich gefasste und formal geordnete Darstellung einer Menge von Begriffen und der zwischen ihnen bestehenden Beziehungen in einem bestimmten Gegenstandsbereich.

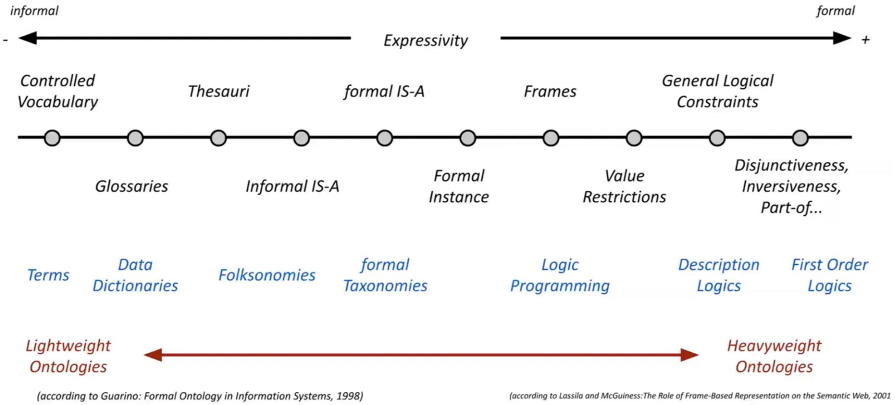
## Mögliches Ziel von Ontologien aus der Sicht der Informatik

- Wissensaustausch in digitalisierter und formaler Form zwischen Anwendungsprogrammen.

## KI-Kontext

- Teilbereich der Wissensrepräsentation

# Ontologie Typen und Kategorien




*Ontologie Typen und Kategorien entsprechend semantischer Ausdrucksfähigkeit*

# Bestandteile einer Ontologie am Beispiel Landkarte

**Klassen** *(auch: Konzepte, engl. Concepts)*
Beschreibung gemeinsamer Eigenschaften

- z.B. Stadt oder Land

**Instanzen** *(auch: Individuen, engl. Individuals)*
Repräsentieren Objekte in der Ontologie. Sie werden anhand vorher definierter Begriffe erzeugt.

- z.B. München als Instanz des Begriffs topologischer Ort vom Typ Stadt oder Deutschland als Instanz des Begriffs topologischer Ort vom Typ Land

**Relationen**
Werden verwendet, um zu beschreiben, welche Beziehungen zwischen den Instanzen bestehen.

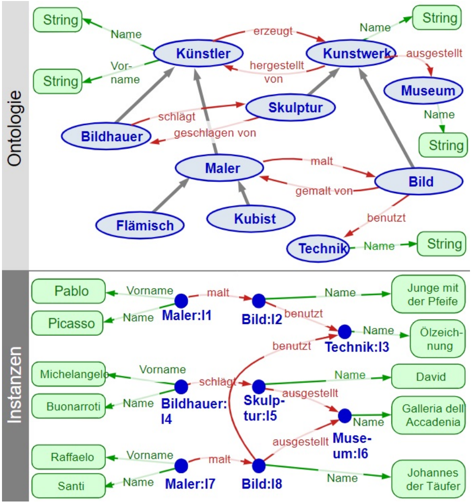
- z.B. Stadt München liegt im Land Deutschland

**Axiome**
Aussagen innerhalb der Ontologie, die immer wahr sind.

- z.B. Zwischen Amerika und Europa existiert keine Zugverbindung

# Beispiel Ontologie mit Instanzen

Ontologie als Netzwerk von Informationen mit logischen Relationen




**Ontologie**

- Ellipsen: Begriffe
- Dünne Pfeile: Relationen
- Rechtecke: Container für Informationen
- Dicke Pfeile: Vererbung

**Instanzen**

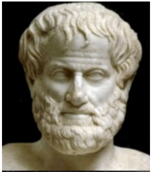
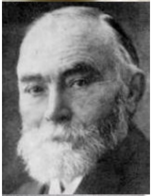
- Blaue Punkte: Instanzen der Ontologie
- Beispiel: Instanz des Malers Raffaello Santi verwendet bereits existierende Instanzen I3 vom Typ Ölzeichnung und I6 vom Typ Galleria dell'Accademia

# Aussagenlogik

# Geschichte der Logik

## Geschichte

- **Aristoteles** (384-322 v. Chr.)
  Kunst der Logik als Werkzeug der Wissenschaft
- **Gottlob Frege** (1848-1925) Begriffsschrift,
  *'Eine der arithmetischen nachgebildete Formelsprache des reinen Denkens'*
- **Charles S. Pierce** (1839-1914)
  Einführung der Standardnotation für die Prädikatenlogik erster Ordnung


# Aussagen mit Junktoren

*„Lörrach ist eine Stadt."* → **Atomare Aussage**

- Aussage, die nicht aus anderen Aussagen zusammengesetzt ist
- enthält keine aussagenverknüpfende logische Operatoren

*„Lörrach ist eine Stadt mit fünfzigtausend Einwohnern."*
→ **Zusammengesetzte Aussage**

- zusammengesetzt aus Junktoren (z.B. Konjunktion, Disjunktion)

Bei einer aussagenlogischen Analyse von Argumenten ist es zentral, die Formulierungen in **atomare Aussagen** zu untergliedern, da nur so die für die Argumentstruktur wichtigen **Junktoren** formalisiert werden können.

Hinweis: In der Aussagenlogik ist nur der *formale* und nicht der *inhaltliche* Wahrheitswert von Aussagen von Bedeutung.

# Junktoren in der Aussagenlogik

Ein Junktor ist eine logische Verknüpfung zwischen zwei Aussagen $P$ und $Q$ innerhalb der Aussagenlogik.

- **Negation:** $\neg P$, einstelliger Junktor
- **Konjunktion:** $P \wedge Q$, das logische Und: "Sowohl P als auch Q"
- **Disjunktion:** $P \vee Q$, das inklusive Oder: "P oder Q oder beides"
- **Implikation:** $P \rightarrow Q$, hinreichende Bedingung: "Wenn P, dann Q"
- **Äquivalenz:** $P \leftrightarrow Q$ *oder* $P \equiv Q$, hinreichende und notwendige Bedingung: "Q genau dann, wenn P"

# Wahrheitstafeln

**Wahrheitstafeln ein- und zweistelliger Junktoren in zweiwertiger Logik**

## Negation

| $A$ | $\bar{A}$ |
|---|---|
| w | f |
| f | w |

$\bar{A}$ wird auch $\neg A$ geschrieben

## Konjunktion

| $A$ | $B$ | $A \wedge B$ |
|---|---|---|
| w | w | w |
| w | f | f |
| f | w | f |
| f | f | f |

## Disjunktion

| $A$ | $B$ | $A \vee B$ |
|---|---|---|
| w | w | w |
| w | f | w |
| f | w | w |
| f | f | f |

## Implikation

| $A$ | $B$ | $A \Rightarrow B$ |
|---|---|---|
| w | w | w |
| w | f | f |
| f | w | w |
| f | f | w |

## Äquivalenz

| $A$ | $B$ | $A \Leftrightarrow B$ |
|---|---|---|
| w | w | w |
| w | f | f |
| f | w | f |
| f | f | w |

**Semantische Äquivalenz $F \equiv G$**

Zwei Formeln $F$ und $G$ heißen (semantisch) äquivalent, wenn sie für alle Belegungen die gleichen Wahrheitswerte annehmen.

*Hinweis: Die leere Formel ist unter allen Belegungen wahr.*

# Junktor Rechenregeln

Die Operationen $\wedge$, $\vee$ sind kommutativ und assoziativ und es gilt:

| | |
|---|---|
| $\neg A \vee B \Leftrightarrow A \Rightarrow B$ | Implikation |
| $A \Rightarrow B \Leftrightarrow \neg B \Rightarrow \neg A$ | Kontraposition |
| $(A \Rightarrow B) \wedge (B \Rightarrow A) \Leftrightarrow A \Leftrightarrow B$ | Äquivalenz |
| $\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$ | De Morgan |
| $\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$ | De Morgan |
| $A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C)$ | Distributivgesetz |
| $A \wedge (B \vee C) \Leftrightarrow (A \wedge B) \vee (A \wedge C)$ | Distributivgesetz |
| $A \vee \neg A \Leftrightarrow w$ | Tautologie |
| $A \wedge \neg A \Leftrightarrow f$ | Widerspruch |
| $A \vee f \Leftrightarrow A$ | Disjunktion |
| $A \vee w \Leftrightarrow w$ | " |
| $A \wedge f \Leftrightarrow f$ | Konjunktion |
| $A \wedge w \Leftrightarrow A$ | " |

# Mathematisches Beweisverfahren für die Implikation

**Beweis** der ersten Äquivalenz

| A | B | $\neg A$ | $\neg A \vee B$ | $A \Rightarrow B$ | $(\neg A \vee B) \Leftrightarrow (A \Rightarrow B)$ |
|---|---|---|---|---|---|
| w | w | f | w | w | w |
| w | f | f | f | f | w |
| f | w | w | w | w | w |
| f | f | w | w | w | w |

$\square$

Die Beweise für die anderen Äquivalenzen laufen analog.

Operatorprioritäten in absteigender Reihenfolge: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$.

# Kombinatorische Explosion

## Zweiwertige Wahrheitstabelle mit 4 atomaren Aussagen

| Zeile | Binär | $A$ | $B$ | $C$ | $D$ |
|---|---|---|---|---|---|
| 15 | 1 1 1 1 | W | W | W | W |
| 14 | 1 1 1 0 | W | W | W | F |
| 13 | 1 1 0 1 | W | W | F | W |
| 12 | 1 1 0 0 | W | W | F | F |
| | | ... | | | |
| 3 | 0 0 1 1 | F | F | W | W |
| 2 | 0 0 1 0 | F | F | W | F |
| 1 | 0 0 0 1 | F | F | F | W |
| 0 | 0 0 0 0 | F | F | F | F |

**Frage:** Wie viele Zeilen sind bei n atomaren Aussagen notwendig?
**Antwort:** Es sind $2^n$ Zeilen notwendig, da für jede der n atomaren Aussagen die zwei Wahrheitswerte *wahr* und *falsch* als Belegung möglich sind. So sind bei 2 Teilaussagen 4, bei 3 Teilaussagen 8 und bei 4 Teilaussagen bereits 16 Zeilen notwendig.

# Konjunktive Normalform

# Definition

Um die automatischen Beweisverfahren möglichst einfach zu halten, arbeiten diese meist mit Formeln in konjunktiver Normalform:

## Konjunktive Normalform

Eine Formel ist in **konjunktiver Normalform (KNF)** genau dann, wenn sie aus einer **Konjunktion**

$$K_1 \wedge K_2 \wedge \ldots \wedge K_m$$

von Klauseln besteht. Eine **Klausel** $K_i$ besteht aus einer **Disjunktion**

$$L_{i1} \vee L_{i2} \vee \ldots \vee L_{in}, \quad i \in \{1, \ldots, m\}$$

von Literalen. Ein **Literal** schließlich ist eine atomare (positives Literal, z.B. $A$) oder eine negierte atomare Aussage (negatives Literal, z.B. $\neg B$).

**Beispiele**

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

Hinweis: Ein **Monom** besteht aus einer **Konjunktion** von Literalen.

# Konjunktive Normalform für automatische Beweisverfahren

Die **konjunktive Normalform** stellt keine Einschränkung der Formelmenge dar, denn es gilt der folgende

**Satz**

Jede aussagenlogische Formel lässt sich in eine äquivalente konjunktive Normalform transformieren.

**Beispiel:** Wir bringen $A \vee B \Rightarrow C \wedge D$ in konjunktive Normalform, indem wir die Äquivalenzen (s. Rechenregeln) anwenden.

- $A \vee B \Rightarrow C \wedge D$
- $\neg(A \vee B) \vee (C \wedge D)$ *Implikation*
- $(\neg A \wedge \neg B) \vee (C \wedge D)$ *de Morgan*
- $(\neg A \vee (C \wedge D)) \wedge (\neg B \vee (C \wedge D))$ *Distributivgesetz*
- $((\neg A \vee C) \wedge (\neg A \vee D)) \wedge ((\neg B \vee C) \wedge (\neg B \vee D))$ *Distributivg.*
- $(\neg A \vee C) \wedge (\neg A \vee D) \wedge (\neg B \vee C) \wedge (\neg B \vee D)$ *Assoziativgesetz*

# Varianten der Wahrheit

# Varianten der Wahrheit

Je nachdem, in wievielen Welten eine Formel wahr ist, teilt man die Formeln in folgende Klassen:

## Definition

Eine Formel heißt

- **erfüllbar**, falls sie bei mindestens einer Belegung wahr ist.
- **allgemeingültig** oder **wahr**, falls sie bei allen Belegungen wahr ist. Wahre Formeln heißen auch **Tautologien**.
- **unerfüllbar**, falls sie bei keiner Belegung wahr ist.

Jede erfüllende Belegung einer Formel heißt **Modell**.

## Schlussfolgerungen

- Offenbar ist die Negation jeder allgemeingültigen Formel unerfüllbar.
- Die Negation einer erfüllbaren aber nicht allgemeingültigen Formel $F$ ist erfüllbar.

# Tautologie Beispiel

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

| $A$ | $B$ | $C$ | $((A \Rightarrow B)$ | $\wedge$ | $(B \Rightarrow C))$ | $\Rightarrow$ | $(A \Rightarrow C)$ |
|---|---|---|---|---|---|---|---|
| W | W | W | W | W | W | **W** | W |
| W | W | F | W | F | F | **W** | F |
| W | F | W | F | F | W | **W** | W |
| W | F | F | F | F | W | **W** | F |
| F | W | W | W | W | W | **W** | W |
| F | W | F | W | F | F | **W** | W |
| F | F | W | W | W | W | **W** | W |
| F | F | F | W | W | W | **W** | W |

Wir sehen: Diese Aussage ist *immer* (d.h. für alle möglichen Belegungen) wahr (“Tautologie“).

# Herleitungsmechanismus in Prolog

# Herleitungsmechanismus in Prolog

Ein Prolog-Programm besteht aus dem eigentlichen Programm (Fakten und Regeln) und der Anfrage.

## Herleitungsmechanismus in Prolog

- Der Herleitungsmechanismus betrachtet die Anfrage als **negierte Aussage.**
- Er versucht zu beweisen, dass diese Aussage im **Widerspruch** zum Programm steht.
- Da das Programm aus Fakten und Regeln nicht selbst schon widersprüchlich sein kann, muss aus dem Programm dann das Gegenteil dieser Negation, das heißt die **vermutete Aussage** folgen.

## Beispielprogramm

```
es_regnet.
```

**Frage: regnet es?**

```
?- es_regnet.
```

**Beweis**

`es_regnet` $\wedge$ ¬`es_regnet` ist ein Widerspruch.

Also regnet es.

# Herleitungsverfahren = negativer Resolutionskalkül

Der Ablauf des Prolog-Programms beginnt mit einer Frage, z.B. `?- a, c.`

Weiter muss es ein Programm geben, das Aussagen zu a und b macht. Etwa `a:- b.` usw.

Die Prolog-Strategie geht von der Anfrage (engl. “goal“) aus.
Sie versucht zunächst das **linkeste Literal** durch Resolution weg zu bekommen (“goal order“).

Dazu wird die **erste passende Programmklausel** gesucht, die dieses Literal in positiver Form (als Regelkopf) enthält (“rule order“).
?- a, c.

a:- b.
\- - - - - - - - - -
**Restfrage:** ?- b, c.
*Das Ergebnis der Resolution = Resolvente fungiert in der Folge als neue Abfrage.*
?- b, c.
b.
\- - - - - - - - - -
**Restfrage:** ?- c.
?- c.
c.
\- - - - - - - - - -
[ ] = **Widerspruch**

# Beispiel einer Herleitung mit Backtracking

**Backtracking:** Wenn keine Resolution möglich ist, wird zu dem letzten Schritt zurückgegangen, bei dem eine andere Regel zur Auswahl stand.

**Beispielprogramm:**
R1: a.
R2: b:- c.
R3: b:- a.
Q: ?- a, b. *Frage* *(Komma in Prolog bedeutet logisches Und)*
**Beweis:**
-a, -b // Q (Komma in Resolutionsschreibweise bedeutet Disjunktion)
a // R1
\- - - - - - - - -
-b
b, -c // R2 Auswahlpunkt /choice point
\- - - - - - - - -
-c
\- - - fail - - - Backtracking: Rückkehr zum letzten Auswahlpunkt
-b
b, -a // R3
\- - - - - - - - -
-a
a // R1
\- - - - - - - - -
[ ] = **Widerspruch**

# Semantisches Modell

## Modell

Aus der Wissensbasis $WB$ folgt eine Formel $Q$, wenn jedes Modell von $WB$ auch ein Modell von $Q$ ist. Man schreibt dann $WB \models Q$.

- semantischer Begriff
- Sprechweisen für $WB \models Q$
  - “WB ist ein Modell für Q“
  - “Q folgt (semantisch) aus WB“

**Spezialfall: Tautologien**

- Tautologien wie zum Beispiel $A \vee \neg A$ schränken die Zahl der erfüllenden Belegungen nicht ein, denn ihre **Aussage ist leer**.
- Die leere Formel ist in allen Belegungen wahr.
- Für jede Tautologie $T$ gilt $\emptyset \models T$, kurz $\models T$.

# Beweiskalkül

# Die universelle Logikmaschine


In den 1970er Jahren schien die **Prädikatenlogik** als mächtige ausdrucksstarke Sprache zusammen mit einem **vollständigen Beweiskalkül** die universelle intelligente Maschine zur Repräsentation von Wissen und zur Lösung vieler schwieriger Probleme zu sein.

# Ableitung

## Ableitung

- Syntaktische Manipulation der Formeln $WB$ und $Q$ durch Anwendung von Inferenzregeln mit dem Ziel, diese so stark zu vereinfachen, dass man am Ende sofort erkennt, dass $WB \models Q$. Man schreibt dann $WB \vdash Q$.

## Kalkül

- Formales System von Regeln, mit denen sich aus der Wissensbasis weitere Aussagen syntaktisch beweisen *(auch: ableiten)* lassen.

# Definitionen für korrekten und vollständigen Kalkül

## Korrektes Kalkül

Ein Kalkül heißt **korrekt,** wenn jede abgeleitete Aussage auch semantisch folgt, das heißt, wenn für Formeln $WB$ und $Q$ gilt:

$$\text{Wenn } WB \vdash Q \text{ dann } WB \models Q.$$

Alternative Formulierung: *"In einem korrekten Kalkül lässt sich keine Formel herleiten, die im gewählten Modell nicht wahr ist."*

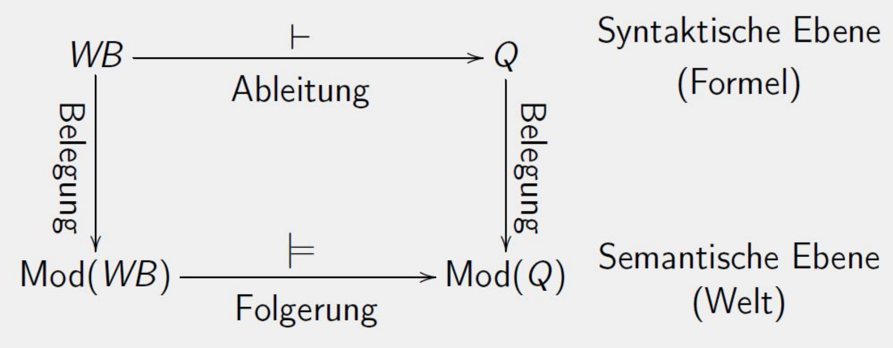
## Vollständiges Kalkül

Ein Kalkül heißt **vollständig,** wenn alle semantischen Folgerungen abgeleitet werden können, das heißt, wenn für Formeln $WB$ und $Q$ gilt:

$$\text{Wenn } WB \models Q \text{ dann } WB \vdash Q.$$

Die Vollständigkeit eines Systems besagt:
*"Was semantisch richtig ist, lässt sich auch ableiten."*

# Syntaktische Ableitung und semantische Folgerung




$Mod(X)$ steht für die Menge der Modelle einer Formel

# Beweiskalkül: Modus Ponens

Für das syntaktische Beweisen von aussagenlogischen Formeln benötigen wir einen Kalkül. Wir starten mit dem Modus Ponens, einer einfachen intuitiven Inferenzregel, die aus der Gültigkeit von A und A ⇒ B die Ableitung von B erlaubt:

**Modus Ponens**

$$\frac{A, A \rightarrow B}{B}$$

Bei gegebener erster Prämisse, "Wenn A, dann B", durch das *Setzen* (Annehmen) der zweiten Prämisse, A, wird der aus beiden folgende Satz B *gesetzt* (d.h. hergeleitet).

**Der Modus Ponens ist korrekt, aber nicht vollständig.**

# Prolog-Beispiel: Modus Ponens

Umsetzung des Modus Ponens an einem **Prolog-Beispiel**:
(1) $A \rightarrow B$: Wenn Jumbo ein Elefant ist, dann ist er ein Säugetier.

(2) $A$: Jumbo ist ein Elefant.
(3) $B$: Jumbo ist ein Säugetier.
**Wissensbasis:**

```
%Fakten
elefant(jumbo).
saeugetier(jumbo) :- elefant(jumbo)
```

**Anfrage an Prolog**

```
?- saeugetier(jumbo).
Yes
```

# Resolution als Verallgemeinerung des Modus Ponens

## Resolutionsregel (Spezialfall)

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

- Die abgeleitete Klausel $A \lor C$ wird **Resolvente** genannt.
- Mann nennt die Literale $B$ und $\neg B$ **komplementär**.

## Satz

Der Resolutionskalkül zum Beweis der Unerfüllbarkeit von Formeln in **konjunktiver Normalform** ist **korrekt und vollständig.**

**Voraussetzung: Die Wissensbasis $WB$ muss widerspruchsfrei sein!**
Andernfalls lässt sich aus der $WB$ alles herleiten.

# Hornklauseln

# Erfüllbarkeit

## Erfüllbarkeitsproblem der Aussagenlogik

Beschreibt die Frage, ob die Variablen einer aussagenlogischen Formel so **belegt** werden können, dass die Aussage **wahr** wird.

- gehört zur Klasse der NP-vollständigen Probleme und gilt damit im Allgemeinen als schwierig lösbar.
- gilt auch für Formeln, die in konjunktiver Normalform vorliegen.
- Eine Ausnahme bilden allerdings **Hornklauseln**, die einen Spezialfall der Formeln in Klauselnormalform darstellen und in **Polynomialzeit auf Erfüllbarkeit getestet** werden können.

# Definite und nicht definite Klausel

Eine Klausel in konjunktiver Normalform enthält positive und negative Literale und lässt sich daher darstellen in der Form

$$(\neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots \vee B_n)$$

Diese Klausel lässt sich umformen in

$$A_1 \wedge \ldots \wedge A_m \Rightarrow B_1 \vee \ldots \vee B_n.$$

**Beispiele**

„Wenn das Wetter schön ist und Schnee liegt, gehe ich Skifahren oder ich gehe arbeiten.“ $\rightarrow$ **nicht definite Klausel**

„Wenn das Wetter schön ist und Schnee liegt, gehe ich Skifahren.“
$\rightarrow$ **definite Klausel**

# Definition

## Hornklauseln

Hornklauseln sind Klauseln mit **höchstens einem positiven Literal** der Formen:

$$\neg A_1 \vee \ldots \vee \neg A_m \vee B \qquad \textbf{Definite Hornklausel}$$

*oder*

$$\neg A_1 \vee \ldots \vee \neg A_m \quad \textbf{Zielklausel}$$

*oder*

$$B \qquad \textbf{Faktenklausel}$$

beziehungsweise äquivalent:

$$A_1 \wedge \ldots \wedge A_m \Rightarrow B$$

*oder*

$$A_1 \wedge \ldots \wedge A_m \Rightarrow f$$

*oder*

$$B$$

# Modus Ponens mit Beispiel

## Satz

**Der Modus Ponens ist vollständig für aussagenlogische Hornklauseln.**

**Wissensbasis Beispiel**

$$(\text{Wetter\_schoen})_1$$
$$(\text{Schneefall})_2$$
$$(\text{Schneefall} \Rightarrow \text{Schnee})_3$$
$$(\text{Wetter\_schoen} \wedge \text{Schnee} \Rightarrow \text{Skifahren})_4$$

Gilt Skifahren? Inferenzregel (verallgemeinerter Modus Ponens):

$$\frac{A_1 \wedge \ldots \wedge A_m, \quad A_1 \wedge \ldots \wedge A_m \Rightarrow B}{B}.$$

Beweis für Skifahren:

$$MP(2,3): \; (\text{Schnee})_5$$
$$MP(1,5,4): \; (\text{Skifahren})$$


Klausel
Hornklausel
Agent
DeklarativeSystementwicklung
Korrekt
Tautologie
Beweiskalkül
PEAS
WissensbasierterAgent
KonjunktiveNormalform
Agentenprogramm
Literal
DefiniteKlausel
Implikation
Vollständig
ModusPonens
Suchraumexplosion
Agentenfunktion
Prolog
Modell

# Prädidatenlogik erster Stufe

# Motivation

**Bisher: Aussagenlogik**

- Beispiel natürlicher Sprache:

  **Roboter 7 befindet sich an xy-Position (35,79)**

- Überführung in aussagenlogische Variable:

  **Roboter_7_befindet_sich_an_xy-Position_(35,79)**

**Beispiel für Limitierung**

100 Roboter auf Gitter mit 100x100 Punkten

$\Rightarrow 100 \cdot 100 \cdot 100 = 1.000.000 = 10^6$ Variablen **!!!**

**Besser: Prädikatenlogik**

**Roboter** $A$ **steht weiter rechts als Roboter** $B$.

# Von der Aussagenlogik zur Prädikatenlogik

## Bisher: Aussagenlogik

- Aussagen in der Aussagenlogik haben keine innere Struktur
- Verknüpfung nur über Wahrheitswert möglich und nicht über Inhalt von Aussagen

## Neu: Prädikatenlogik

- **Objekte/Individuen:** z.B. Menschen, Tiere, Ereignisse,...
- **Relationen:** z.B. vater_von(.,.), weiblich(.),...

## Vorteil von Prädikatenlogik gegenüber Aussagenlogik

- Prädikatenlogik ist insbesondere wegen der Möglichkeit zur Quantifizierung ausdrucksstärker als Aussagenlogik.

## Beweisverfahren für die Prädikatenlogik erster Stufe

- Der Resolutionskalkül der Aussagenlogik lässt sich so erweitern, dass man mit ihm ebenfalls Widerspruchsbeweise über prädikatenlogische Formeln führen kann.

# Prädikatenlogik erster Stufe

## Ziel der Prädikatenlogik

Beschreibung von **Objekten**, deren **Eigenschaften** und **Beziehungen** zueinander.

Analog zur Aussagenlogik unterscheidet die Prädikatenlogik zwischen *Syntax* und *Semantik*.

## Standardsystem der Prädikatenlogik

Die **Prädikatenlogik erster Stufe** stellt das Standardsystem der Prädikatenlogik dar.

# Beispiele

## Beispiele für Prädikatenlogik erster Stufe

- „Alle Kinder lieben Eiscreme.“
  $\forall x : Kind(x) \Rightarrow liebt\_Eiscreme(x)$
- „Es gibt einen Baum, der Nadeln hat.“
  $\exists x : Baum(x) \wedge hat\_Nadeln(x)$
- „Die Mutter einer Person ist dessen weibliches Elternteil.“
  $\forall x \forall y : Mutter(x, y) \Leftrightarrow (weiblich(x) \wedge Elternteil(x, y))$
- Die Relation *Geschwister* ist symmetrisch:
  $\forall x \forall y : Geschwister(x, y) \Rightarrow Geschwister(y, x)$

# Beispiele aus der Computerlinguistik

Engl. Natural Language Processing, *NLP*

| Natural language | Logical formula |
|---|---|
| John walks | $W(j)$ |
| John sees Mary | $S(j, m)$ |
| John gives Mary the book | $G(j, m, b)$ |
| He walks | $W(x)$ |
| John sees her | $S(j, x)$ |
| Someone walks | $\exists x W(x)$ |
| Some boy walks | $\exists x(B(x) \wedge W(x))$ |
| Everyone walks | $\forall x W(x)$ |
| Every girl sees Mary | $\forall x(G(x) \rightarrow S(x, m))$ |
| Someone creates everythings | $\exists x \forall y C(x, y)$ |

# Eigenschaften der Quantoren

**Kommutativität des All- und Existenzquantors**

- $\forall x \forall y : F$ ist semantisch äquivalent zu $\forall y \forall x : F$
- $\exists x \exists y : F$ ist semantisch äquivalent zu $\exists y \exists x : F$

**Aber**

$\exists x \forall y : F$ ist **nicht** semantisch äquvialent zu $\forall y \exists x : F$

Beispiel:

- $\exists x \forall y : liebt(x, y)$
  Es gibt jemanden, der alle in der Welt liebt.
- $\forall y \exists x : liebt(x, y)$
  Jeder in der Welt wird von mindestens einer Person geliebt.

# Umformung in Klauselform

1. **Negationsnormalform:** Äquivalenzumformungen einer prädikatenlogischen Formel, so dass alle Negationen ganz innen stehen
2. **Pränexform:** Alle Quantoren stehen am Anfang der Formel.
3. **Skolemform:** Entfernung aller Existenzquantoren. Die Skolemform ist genau dann erfüllbar, wenn die Ausgangsformel erfüllbar ist.
4. **Konjunktive Normalform:** Die Formel ist eine Konjunktion von Disjunktionen
5. **Klauselform:** Die konjunktive Normalform wird in die Schreibweise der Klauselform gebracht

$$(a \wedge (b \vee \neg c) \wedge (a \vee d)) \;\rightarrow\; \{a, \{b, \neg c\}, \{a, d\}\}$$

Darauf können wir dann analog zur Aussagenlogik die **Resolutionsmethode** anwenden.

# Unifikation von Termen

- Gegeben: Literale $L_1, L_2$
- Gesucht: Variablensubstitution $\sigma$ derart, dass nach Anwendung auf $L_1$ und $L_2$ gilt: $L_1\sigma = L_2\sigma$

$\sigma$ heißt **Unifikator** von $L_1$ und $L_2$. **Beispiele:**

| $L_1$ | $L_2$ | $\sigma$ |
|---|---|---|
| p(X,X) | p(a,a) | [X/a] |
| p(X,X) | p(a,b) | n.a. |
| p(X,Y) | p(a,b) | [X/a, Y/b] |
| p(X,Y) | p(a,a) | [X/a, Y/a] |
| p(f(X),b) | p(f(c),Z) | [X/c, Z/b] |
| p(X,f(X)) | p(Y,Z) | [X/Y, Z/f(Y)] |
| p(X,f(X)) | p(Y,Y) | n.a. |

# Beispiel für Resolution in der Prädikatenlogik

$(p(X,f(Y)) \vee q(f(X),Y))$

$(\neg p(a,Z) \vee r(Z))$

*Resolution mit [X/a, Z/f(Y)] ergibt*

$(q(f(a),Y) \vee r(f(Y)))$.

→ In der Prädikatenlogik werden bei der Resolution durch **Substitutionen** zusätzlich auch **Variablenbindungen** berücksichtigt.

# Beispiel

# Beispielaufgabe

Transformieren Sie die prädikatenlogische Formel in Klauselnormalform und geben Sie dabei alle Zwischenschritte (bereinigte Form, Pränexform, Skolemform) an: $(\exists x(p(x,y))) \to (\exists x(q(x,x)))$

$$(\exists x(p(x,y))) \to (\exists x(q(x,x)))$$

$$=(\neg\exists x(p(x,y))) \lor (\exists x(q(x,x)))$$

$$=\forall x(\neg p(x,y)) \lor \exists x(q(x,x))$$

Bereinigte Form:

$$=\forall x(\neg p(x,y)) \lor \exists z(q(z,z))$$

Pränexform:

$$=\forall x \exists z((\neg p(x,y) \lor q(z,z))$$

Die freie Variable $y$ existenzquantifizieren:

$$\exists y \forall x \exists z(\neg p(x,y) \lor q(z,z))$$

Skolemform:

$$\forall x(\neg p(x,a) \lor q(f(x),f(x)))$$

Klauselnormalform:

$$\{\{\neg p(x,a), q(f(x),f(x))\}\}$$

# Beispielaufgabe

**Logikrätsel mit dem Titel Der Hochsprung**:
Drei junge Mädchen üben Hochsprung für die Sportprüfung im Abitur. Die Stange wurde bei 1.20 m befestigt. 'Ich wette', sagt das erste zum zweiten Mädchen, 'dass mir mein Sprung gelingen wird, wenn, und nur dann, wenn du versagst.' Wenn das zweite junge Mädchen das gleiche zu dem dritten sagt, welches wiederum das gleiche zu dem ersten sagt, wäre es dann möglich, dass keins von den dreien seine Wette verliert?

# Beispielaufgabe - Lösung 1/2

**Formalisierung:**

Der Sprung des ersten Mädchens gelingt: $A$,
der Sprung des zweiten Mädchens gelingt: $B$,
der Sprung des dritten Mädchens gelingt: $C$.
Wette des ersten Mädchens: $(A \Leftrightarrow \neg B)$,

Wette des zweiten Mädchens: $(B \Leftrightarrow \neg C)$,
Wette des dritten Mädchens: $(C \Leftrightarrow \neg A)$.
Behauptung: Es können nicht alle drei die Wette gewinnen:

$$Q \equiv \neg((A \Leftrightarrow \neg B) \wedge (B \Leftrightarrow \neg C) \wedge (C \Leftrightarrow \neg A))$$

Per Resolution ist nun zu zeigen, dass $\neg Q$ unerfüllbar ist.

# Beispielaufgabe - Lösung 2/2

**Resolutionsbeweis:** **Transformation in CNF** (konj. Normalform)

Wette des ersten Mädchens:

$$(A \Leftrightarrow \neg B) \equiv (A \Rightarrow \neg B) \wedge (\neg B \Rightarrow A) \equiv (\neg A \vee \neg B) \wedge (A \vee B)$$

Die Wetten der beiden anderen Mädchen werden analog transformiert und man erhält als negierte Behauptung

$$\begin{aligned} \neg Q \equiv & (\neg A \vee \neg B)_1 \wedge (A \vee B)_2 \wedge (\neg B \vee \neg C)_3 \wedge \\ & (B \vee C)_4 \wedge (\neg C \vee \neg A)_5 \wedge (C \vee A)_6 \end{aligned}$$

Daraus wird nun mit Resolution die leere Klausel abgeleitet:

$$\begin{aligned} Res(1,6):&\quad (C \vee \neg B)_7 \\ Res(4,7):&\quad (C)_8 \\ Res(2,5):&\quad (B \vee \neg C)_9 \\ Res(3,9):&\quad (\neg C)_{10} \\ Res(8,10):&\quad () \end{aligned}$$

# Lernkontrolle

**Meine 3 Lernziele für heute waren**

1. Ich kenne den grundlegenden Aufbau eines wissensbasierten Agentenprogramms in der deklarativen Systementwicklung.
2. Mir sind wesentliche Konzepte von Syntax und Semantik der Aussagen- und Prädikatenlogik bekannt.
3. Ich bin mit dem Prozess zur Ableitung und Schlussfolgerung von explizitem aus implizitem Wissen in Ontologien vertraut.