Grundlagen der Künstlichen Intelligenz

7. Prädikatenlogische Agenten

Syntax und Semantik, Normalformen, Unifikation
Generalisierter Modus Ponens

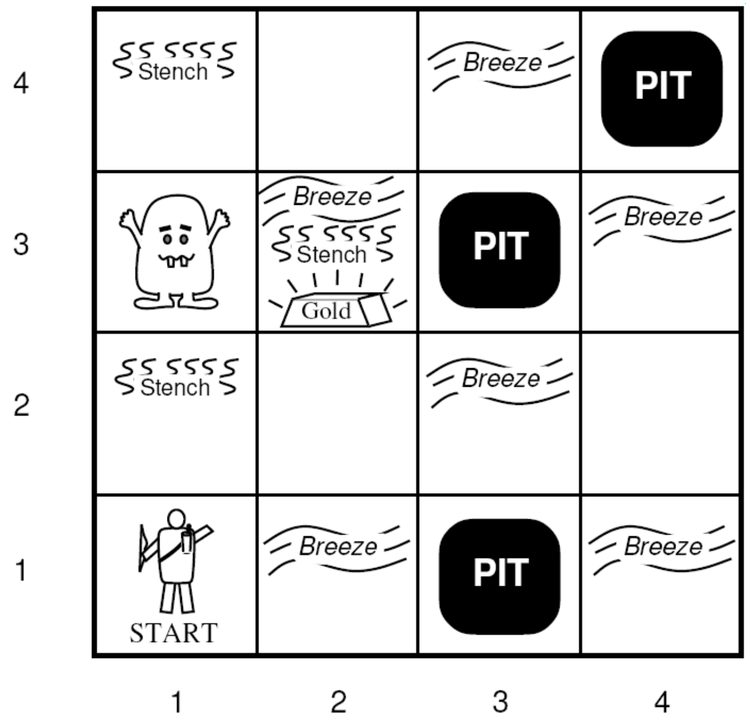
Volker Steinhage

Inhalt

- Motivation
- Syntax und Semantik der Prädikatenlogik 1. Stufe
- Normalformen der Prädikatenlogik 1. Stufe
- Unifikation als Voraussetzung prädikatenlogischer Inferenz
- Der Generalisierte Modus Ponens zur prädikatenlogischen Inferenz

Motivation (1)

In der Wumpus-Welt untersuchten wir die Aussagenlogik (AL) als ersten Aufschlag für eine Wissensrepräsentationssprache.



Vorteile:

1) Semantik basiert auf Wahrheitsrelation zw. Sätzen und mögl. Weltzuständen

2) Unvollständige Information ist darstellbar über Negation (z.B. keine Fallgrube in (1,2)); und Disjunktion (z.B. Fallgrube in (2,2) oder in (3,1)).

3) Kompositionalität: Bedeutung eines Satzes folgt aus den Bedeutungen seiner Komponenten (z.B. $S_{1,2} \wedge S_{2,3}$ ergibt sich aus $S_{1,2}$ und $S_{2,3}$).

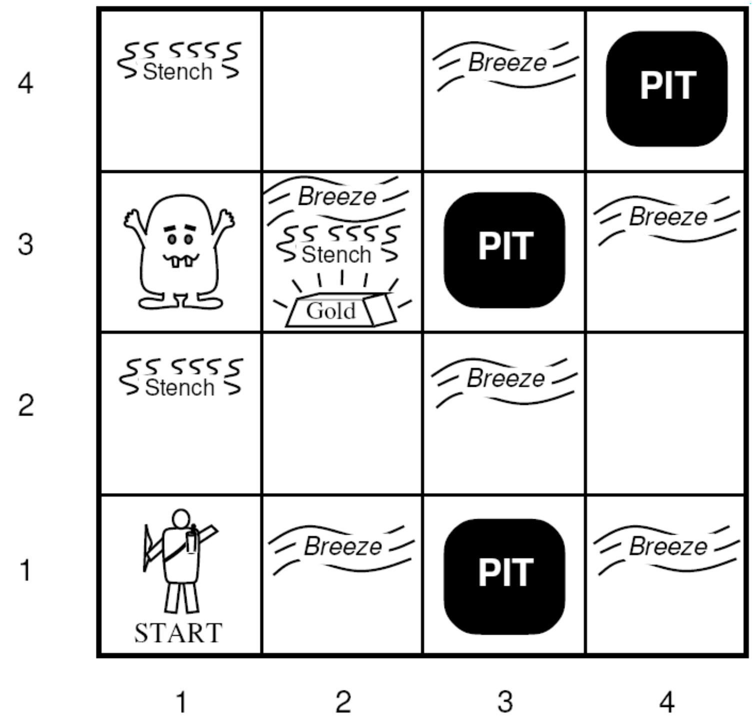
Folgerung: In der Wumpus-Welt waren damit Fakten (z.B. $S_{1,2}$) sowie deren Zusammensetzungen (z.B. $S_{1,2} \wedge S_{2,3}$) geeignet darstellbar.

Motivation (2)

Nachteil: keine *kompakte* Kodierung wegen unzureichender Sprachmächtigkeit der *AL*

1) Allg. Regeln, die auf mehrere oder nur bestimmte Weltzustände/Objekte anwendbar sind, müssen für jede mögliche *Instanz* neu formuliert werden

2) Funktionen und Relationen von bzw. zwischen Objekten müssen durch Aufzählungen der jeweiligen *Instanzen* ausgedrückt werden (z.B. Nachbarn((2,2)) = {(1,2),(2,1),(2,3),(3,2)}



Motivation (3)

Allg.:

- *Mit AL* sind Wahrheitswerte atomarer Aussagen und zusammengesetzter Formeln repräsentierbar sowie Schlussfolgerungen von diesen ableitbar.
- Allerdings ist es nicht möglich, auf die *Struktur* der atomaren Aussagen und damit der Komposite und Schlussfolgerungen einzugehen.

Beispiel:

„Alle Blöcke sind rot“ wird in AL zu atomarer Aussage *P*

„Es gibt einen Block A“ wird in AL zu atomarer Aussage *Q*

Daraus sollte folgen: „A ist rot“

- In *AL* nicht möglich! Es fehlen die sprachl. Mittel, um Eigenschaften (Attribute) und Beziehungen (Relationen) sowie Quantifizierungen über Objekten auszudrücken.

Idee: Wir führen Individuenvariablen, Prädikate, Funktionen und Quantoren ein

→ *Prädikatenlogik 1. Stufe* (*PL1*)

Alphabet der Prädikatenlogik 1. Stufe

Symbole:

- *Operatoren*: $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
- (*Individuen-*) *Variablen*: $x_1, x_2, \ldots, x', x'', \ldots, y, \ldots, z, \ldots$
- *Klammersymbole*: (,), [,], { , }, …
- *Funktionssymbole* für Konstanten und funktional abhängige Objekte
- *Prädikatensymbole* für Eigenschaften und Relationen von Objekten
- *Quantoren*: $\forall$, $\exists$

Hinweise:

1. Prädikaten- und Funktionssymbole besitzen eine Stelligkeit (Zahl der Argumente)
   - 0-stellige Prädikate: aussagenlogische Atome
   - 0-stellige Funktionen: Konstanten
2. Wir nehmen abzählbar viele Prädikate und Funktionen jeder Stelligkeit an

Grammatik der Prädikatenlogik 1. Stufe (1)

Terme (stehen für Objekte):

1. Jede Variable ist ein Term.

2. Wenn $t_1, t_2, \ldots, t_n$ Terme sind und f ein n-stelliges Funktionssymbol, dann ist $f(t_1,t_2, \ldots ,t_n)$ auch ein Term.

   Terme ohne Variablen: *Grundterme*

Atomare Formeln (stehen für Aussagen über Objekten)

1. Wenn $t_1, t_2, \ldots, t_n$ Terme sind und P ein n-stelliges Prädikat ist, dann ist $P(t_1,t_2, \ldots,t_n)$ eine atomare Formel.

2. Wenn t_1 und t_2 Terme sind, dann ist $t_1 = t_2$ eine atomare Formel.

Atomare Formeln ohne Variablen: *Grundatome*.

Grammatik der Prädikatenlogik 1. Stufe (2)

Formeln:

1. Jede atomare Formel ist eine Formel.

2. Wenn φ und ψ Formeln sind und x eine Variable ist, dann sind auch Formeln:

   - $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \Rightarrow \psi$, $\varphi \Leftrightarrow \psi$ (Kompositionen)

   - $\exists\, x\, \varphi$, $\forall\, x\, \psi$ (Quantifizierte Formeln)

Zur Quantoren-Operatorenpräzedenz: $\exists$, $\forall$ binden so stark wie $\neg$

Folgerung: Die Aussagenlogik ist eine Teilsprache der PL1:
nur Formeln ***ohne*** (a) Individuenvariablen, (b) Funktionen, (c) Quantoren, (d) n-stell. Prädikate mit $n > 0$

Notation

Zur Notation in dieser Vorlesung:

- Klammern zur Strukturierung: (,)
- Verwendung anderer Klammern zwecks Leserlichkeit: z.B. {, }, [,], …
- Prädikate: z.B. *Person(x), Schön(x), Älter-als(x,y)*
- Funktionen: z.B. *vater-von(x), nachfolger(x), a, b*
- Variablen: z.B. *x, y, z,*
- geschachtelte Quantoren: $\forall x \, \forall y$... auch als $\forall x, y$...

Die Notation kann aber z.B. bzgl. der Groß- und Kleinschreibung von Funktions- und Prädikatsbezeichnern je nach Literaturvorlage variieren

---

Alternative Notationen: s. Anhang

Semantik von PL1-Formeln

Beispielformel: $\forall x\, [\, \texttt{Block}(x) \Rightarrow \texttt{Rot}(x)\,]$,

$\texttt{Block}(a)$

Quantoren

Attribute

Interpretation: Für alle Objekte x gilt: falls x ein Block ist, dann ist x rot.

a ist ein Block.

Generell:

- Terme werden als Objekte interpretiert
- universell quantifizierte Variablen stehen für alle Objekte des Universums
- existentiell quantifizierte Variablen stehen für mind. ein Objekt des Universums
- Prädikate stehen für Eigenschaften oder Relationen von Objekten des Universums

Ähnlich wie für Aussagenlogik definieren wir:

Interpretationen, Erfüllung, Modelle, Allgemeingültigkeit, Folgerung, . . .

Semantik von PL1: Interpretation

Interpretation: $I = \langle \mathbf{D}, \cdot^{I,\alpha} \rangle$ mit nicht-leerer Domänenmenge (bzw. Universum) $\mathbf{D}$

und Interpretationsfunktion $\cdot^{I,\alpha}$ mit

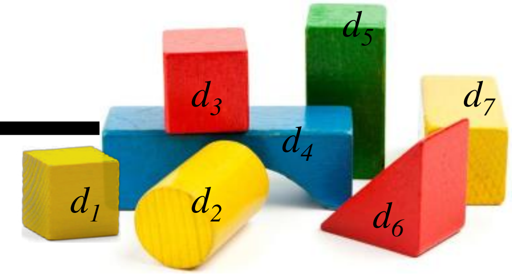
- $\cdot^{I,\alpha}$ bildet *n*-stell. Funktionssymbole auf Funktionen über $\mathbf{D}$ ab: $f^I \in [\mathbf{D}^n \to \mathbf{D}]$
- $\cdot^{I,\alpha}$ bildet Individuenkonstanten auf Elemente aus $\mathbf{D}$ ab: $a^I \in \mathbf{D}$
- $\cdot^{I,\alpha}$ bildet *n*-stell. Prädikatensymbole auf Relationen oder Attribute über $\mathbf{D}$ ab:

$P^I \subseteq \mathbf{D}^n$

Darauf aufbauend:

- *Interpretation* von Grundtermen:
  - $(f(t_1, \dots, t_n))^I = f^I(t_1^I, \dots, t_n^I) \quad (t_i^I \in \mathbf{D})$
- *Interpretation* von Grundatomen $P(t_1, \dots, t_n)$:
  - $I \models P(t_1, \dots, t_n)$ gdw. $\langle t_1^I, \dots, t_n^I \rangle \in P^I \quad (t_i^I \in \mathbf{D})$

Beispiel 1 zu Interpretation



Domänenmenge: $\mathbf{D} = \{d_1, \ldots, d_7\}$

Konstante: $a^I = d_1$ — Element aus **D**

Konstante: $b^I = d_2$ — Element aus **D**

Konstante: $c^I = \ldots$ — Element aus **D**

Prädikatensymbol: $\texttt{Keil}^I = \{ d_6 \}$ — Attribut über **D**

Prädikatensymbol: $\texttt{Gelb}^I = \{ d_1, d_2, d_7 \}$ — Attribut über **D**

Interpretation von Grundatom: $I \models \texttt{Gelb(b)}$

Interpretation von Grundatom: $I \not\models \texttt{Keil(b)}$

Beispiel 2 zu Interpretation

$$\mathbf{D} = \{1, 2, 3, \ldots\}$$

$$\mathtt{1}^I = 1$$

$$\mathtt{2}^I = 2$$

$$\ldots$$

Prädikatensymbol

$$\mathtt{Even}^I = \{2, 4, 6, \ldots\}$$

Attribut über **D**

Funktionssymbol

$$\mathtt{succ}^I = \{(1 \mapsto 2), (2 \mapsto 3), \ldots\}$$

Funktion über **D**

Interpretation von Grundatom

$$I \models \mathtt{Even(2)}$$

Interpretation von
Grundatom und Grundterm

$$I \not\models \mathtt{Even(succ(2))}$$

Semantik von PL1: Variablenbelegung

Die Funktion α belegt die Variablen der Variablenmenge V mit Elementen der Domänenmenge $\mathbf{D}$, also $\alpha : V \to \mathbf{D}$.

Notation: $\alpha\,[x/d]$ ist identisch mit α bis auf die Variable x.

Für x gilt die Belegung: $[x/d](x) = d$.

Interpretation von Termen unter I,α :

$x^{I,\alpha} = \alpha(x)$ für Variable x

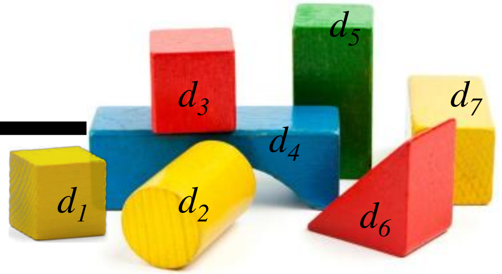
$a^{I,\alpha} = a^{I}$ für Konstante a

$$(f(t_1, \ldots, t_n))^{I,\alpha} = f^{I}(t_1^{I,\alpha}, \ldots, t_n^{I,\alpha})$$

Interpretation von atomaren Formeln unter I,α :

$I, \alpha \models P(t_1, \ldots, t_n)$ gdw. $\langle t_1^{I,\alpha}, \ldots, t_n^{I,\alpha} \rangle \in P^{I}$

Erweiterung von Beispiel 1



$$
\begin{aligned}
\mathbf{D} &= \{\, d_1, \ldots, d_7 \,\} \\
a^I &= d_1 \\
b^I &= d_2 \\
c^I &= \ldots \\
\texttt{Keil}^I &= \{\, d_6 \,\} \\
\texttt{Gelb}^I &= \{\, d_1, d_2, d_7 \,\} \\
I &\models \texttt{Gelb(b)} \\
I &\not\models \texttt{Keil (b)}
\end{aligned}
$$

Variablenbelegung

$$\alpha = \{(\, x \mapsto d_1 \,), (\, y \mapsto d_2 \,)\}$$

Interpretation atomarer Formel

$$I, \alpha \models \texttt{Gelb(x)}$$

$$I, \alpha[\, y/d_6 \,] \models \texttt{Keil(y)}$$

Interpretation atomarer Formel

Semantik von PL1: Erfüllbarkeit

Eine Formel φ wird von einer Interpretation I unter einer Variablenbelegung α erfüllt, d.h. $I, \alpha \models \varphi$:

$$I, \alpha \models \mathrm{T}$$

$$I, \alpha \not\models \mathrm{F}$$

$$I, \alpha \models \neg\varphi \text{ gdw. } I, \alpha \not\models \varphi$$

$$\ldots$$

und alle weiteren propositionalen Regeln sowie

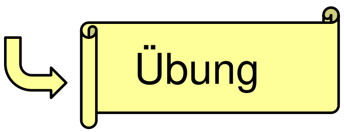
$$I, \alpha \models P(t_1, \ldots, t_n) \text{ gdw. } \langle t_1^{I,\alpha}, \ldots, t_n^{I,\alpha} \rangle \in P^I$$

$$I, \alpha \models \forall x\, \varphi \text{ gdw. für alle } d \in \mathbf{D} \text{ gilt } I\, \alpha\, [x/d] \models \varphi$$

$$I, \alpha \models \exists x\, \varphi \text{ gdw. es gibt ein } d \in \mathbf{D} \text{ mit } I\, \alpha\, [x/d] \models \varphi$$

Beispiel 3

Für die Übungen:



$$
\begin{aligned}
\mathbf{D} &= \{d_1, \ldots, d_n \mid n > 1\} \\
a^I &= d_1 \\
b^I &= d_2 \\
\texttt{Block}^I &= \{ d_1 \} \\
\texttt{Rot}^I &= \mathbf{D} \\
\alpha &= \{(x \mapsto d_1), (y \mapsto d_2)\}
\end{aligned}
$$

Fragen:

1. $I, \alpha \models \texttt{Block(b)} \lor \neg\texttt{Block(b)}$?
2. $I, \alpha \models \texttt{Block(x)} \Rightarrow (\texttt{Block(x)} \lor \texttt{Block(y)})$?
3. $I, \alpha \models \texttt{Block(a)} \land \texttt{Block}(\texttt{b})$?
4. $I, \alpha \models \forall\, \mathrm{x}\, (\texttt{Block(x)} \Rightarrow \texttt{Rot(x)})$?
5. $I, \alpha \models \Theta$?

mit Formelmenge Θ:

$$
\Theta = \left\{ \begin{array}{c} \mathrm{Block(a)}, \mathrm{Block(b)}, \\ \forall x\, (\mathrm{Block}(x) \Rightarrow \mathrm{Rot}(x)) \end{array} \right\}
$$

Freie und gebundene Variablen

In $\forall x(\varphi)$ bzw. $\exists x(\varphi)$ bezeichnet „(φ)“ den *Wirkungsbereich des Quantors* $\forall x$ bzw. $\exists x$. Jedes Vorkommen der Variablen x heißt dann in $\forall x(\varphi)$ bzw. $\exists x(\varphi)$ durch den Quantor *gebunden*. Eine Variable heißt *frei*, wenn sie durch keinen Quantor gebunden ist.

Das Beispiel zeigt freie Variablen gerahmt und gebundene ungerahmt:

$$\forall x[R(\boxed{y}, \boxed{z}) \wedge \exists y\{\neg P(y, x) \vee R(y, \boxed{z})\}]$$

Formeln *ohne freie Variable* heißen *geschlossene* Formeln. Bei der Formulierung von Theorien benutzen wir nur geschlossene Formeln.

Grund: Die Begriffe *Äquivalenz, Erfüllbarkeit, Folgerbarkeit* usw. sind bei geschlossenen Formeln unabhängig von der Variablenbelegung.

Bei geschlossenen Formeln wird somit keine Variablenbelegung α auf der linken Seite des Modellbeziehungszeichens notiert: $I \vDash \varphi$.

Terminologie

Eine Interpretation I heißt *Modell* von φ unter der Belegung α, wenn

$$I, \alpha \models \varphi.$$

Eine Formel φ der PL1 kann - ebenso wie in der Aussagenlogik - *erfüllbar*, *unerfüllbar*, *falsifizierbar* oder *allgemeingültig* sein.

Analog sind zwei Formeln *logisch äquivalent* ($\varphi \equiv \psi$), wenn für alle I, α gilt: *

$$I, \alpha \models \varphi \text{ gdw. } I, \alpha \models \psi.$$

Analog *folgt* Formel ψ *logisch* aus Formel φ ($\varphi \models \psi$), wenn für alle I, α gilt:

$$\text{Wenn } I, \alpha \models \varphi \text{, dann } I, \alpha \models \psi.$$

Frage: Wie können wir einen Ableitungsbegriff definieren – z.B. die Resolution?

* Beachte: $P(x) \not\equiv P(y)$, aber $P(x) \equiv P(x) \vee P(x)$

Pränex-Normalform

Wegen der Quantoren können wir nicht direkt die KNF einer Formel bilden.

Erster Schritt: Bilden der *Pränex-Normalform*:

Quantorenpräfix + (quantorenfreie) Matrix φ.

$$\forall x_1 \, \forall x_2 \, \exists x_3 \, \ldots \, \forall x_n \; \varphi$$

Gültige Äquivalenzen für Umformungen

Seien φ und χ Formeln, welche die Variable x enthalten.

Sei ψ eine Formel, welche x nicht enthält.

Dann gelten die folg. Äquivalenzen:

$$
\begin{aligned}
(\forall x\, \varphi) \wedge \psi &\equiv \forall x(\varphi \wedge \psi) \\
(\forall x\, \varphi) \vee \psi &\equiv \forall x(\varphi \vee \psi) \\
(\exists x\, \varphi) \wedge \psi &\equiv \exists x(\varphi \wedge \psi) \\
(\exists x\, \varphi) \vee \psi &\equiv \exists x(\varphi \vee \psi) \\
\forall x\, \varphi \wedge \forall x\, \chi &\equiv \forall x(\varphi \wedge \chi) \\
\exists x\, \varphi \vee \exists x\, \chi &\equiv \exists x(\varphi \vee \chi) \\
\neg \forall x\, \varphi &\equiv \exists x \neg \varphi \\
\neg \exists x\, \varphi &\equiv \forall x \neg \varphi
\end{aligned}
$$

Weiterhin gelten alle aus der Aussagenlogik bekannten Äquivalenzen!

Erzeugen der Pränex-Normalform

(1) Eliminierung von $\Rightarrow$ und $\Leftrightarrow$

(2) $\neg$ nach innen

(3) Quantoren nach außen

Beispiel:

$$\neg\forall x[(\forall x\, P(x)) \Rightarrow Q(x)] \quad \leadsto \quad \neg\forall x\, [\neg(\forall x\, P(x)) \vee Q(x)] \quad \leadsto \quad \exists x\, [(\forall x\, P(x)) \wedge \neg Q(x)]$$

Problem: Zwei verschieden gebundene Variablenvorkommen x

Lösung durch *Variablenumbenennung von gebundenen Variablen*:
Die Umbenennung $\varphi[x/t]$ entsteht aus der Formel φ, indem alle gebundenen Vorkommen von x in φ durch den Term t ersetzt werden.

Lemma: Sei y eine Variable, die nicht in φ vorkommt. Dann gilt

$$\forall x\, \varphi \equiv \forall y\, \varphi\, [x/y] \text{ und } \exists x\, \varphi \equiv \exists y\, \varphi[x/y].$$

Satz: Die Pränex-Normalform ist für jede Formel φ effektiv berechenbar.

Ergebnis der Pränex-Normalform

Pränex-NF = Quantorenpräfix + (quantorenfreie) Matrix φ:

$$\forall x_1 \, \forall x_2 \, \exists x_3 \, \ldots \, \forall x_n \, \varphi$$

Alle Quantoren sind also im Quantorenpräfix zusammengefasst.

Nächstes Ziel: Elimination aller Quantoren!

$\rightarrow$ Skolemisierung:

- Elimination der Existenzquantoren durch Funktionen, die uns *ein „richtiges“* Element liefern.
- Elimination der Allquantoren, indem die verbliebenen Variablen als allquantiziert interpretiert werden!

Skolemisierung (1)

Idee: Eliminierung der Existenzquantoren durch Funktionen, die uns ein „richtiges“ Element liefern.

Satz (*Skolem-Normalform*): Sei φ eine geschlossene Formel in Pränex-Normalform, so dass alle quantifizierten Variablen paarweise verschieden sind und die Funktionssymbole $g_1, g_2, \ldots$ nicht in φ auftreten. Genauer sei

$$\varphi = \forall x_1 \ldots \forall x_i \, \exists y \, \psi.$$

Dann ist φ erfüllbar gdw.

$$\varphi^* = \forall x_1 \ldots \forall x_i \, \psi[\, y/g_j(x_1, \ldots, x_i) \,]$$

erfüllbar ist.

Argumente der Skolem-Funktionen: alle allquantifizierten Variablen, in deren Gültigkeitsbereich der ersetzte Existenzquantor steht.

Skolemisierung (1)

Beispiel 1:

$\exists x\ \forall y\ \exists z\ [P(x,y) \vee Q(x,z)]$

$\rightarrow\ \forall y\ \exists z\ [P(f_0,y) \vee Q(f_0,z)]$ (0-stellige Skolem-Funktion (= Konstante) f_0 für $\exists x$)

$\rightarrow\ P(f_0,y) \vee Q(f_0,f_1(y))$ (1-stell. Skolem-Fktn. $f_1(y)$ für $\exists z$, Entfernen von $\forall y$)

Bemerkung: die Indizes der Skolem-Funktionen beziehen sich nicht auf deren Stelligkeit, sondern auf die Reihenfolge ihrer Erzeugung

Beispiel 2 mit Variablenumbenennung:

$\exists x\ [(\forall x\ P(x)) \wedge \neg Q(x)]$

$\rightarrow\ \exists y\ [(\forall x\ P(x)) \wedge \neg Q(y)]$ (Variablenumbenennung)

$\rightarrow\ \exists y\ [\forall x\ (P(x) \wedge \neg Q(y))]$ (Quantorverschiebung – s. Folie 22)

$\rightarrow\ \forall x\ (P(x) \wedge \neg Q(g_0))$ (0-stellige Skolem-Funktion g_0 für $\exists y$)

$\rightarrow\ P(x) \wedge \neg Q(g_0)$ (Entfernen des All-Quantors $\forall x$)

Skolem-Normalform

Skolem-Normalform:

Und die verbliebenen Allquantoren lassen wir weg in der Folgebearbeitung

Pränex-Normalform ohne Existenzquantoren.

Schreibweise: φ^* ist SNF von φ.

Satz: Die SNF φ^* ist für jede geschlossene Formel φ effektiv berechenbar.

Beachte: Die Skolemisierung ist *keine Äquivalenztransformation*,

sie erhält nur Erfüllbarkeit!

M.a.W.: *Erfüllbarkeitsäquivalent* heißt nicht *modellerhaltend*: eine Interpretation, welche die ursprüngliche Formel erfüllt, erfüllt nicht notwendigerweise auch die skolemisierte Formel

Die Idee der prädikatenlogischen Inferenz (1)

In der Aussagenlogik ist aussagenlogische Inferenz umsetzbar z.B. über

- AL-Modus Ponens: $\dfrac{\varphi, \varphi \Rightarrow \psi}{\psi}$

  Beim AL-Modus Ponens ist die Prämisse φ der Regel ($\varphi \Rightarrow \psi$) identisch dem Faktum φ.

- AL-Resolution: $\dfrac{C_1 \cup \{l\},\ C_2 \cup \{\bar{l}\}}{C_1 \cup C_2}$

  Bei der AL-Resolution sind die beiden Resolutionsliterale l und $\neg l$ identisch bis auf das Vorzeichen.

- Für die prädikatenlogische Inferenz werden wir ein Verfahren einsetzen, das Fakt und Regelprämisse beim Modus Ponens bzw. die Atome der Resolutionsliterale „gleich macht“ und damit die Anwendung von Modus-Ponens und Resolution für die prädikatenlogische Inferenz erlaubt.

Die Idee der prädikatenlogischen Inferenz (2)

Beispiel: gegeben sei die folg. prädikatenlogische Wissensbasis KB:

$$King(x) \wedge Greedy(x) \Rightarrow Evil(x)$$
$$King(john)$$
$$Greedy(y)$$
$$Brother(richard, john)$$

Vereinfachte Schreibweise der Klauselmenge: jede Zeile ist eine Klausel. Diese sind konjunktiv verknüpft zur Klauselmenge

1) Um die Regelklause anwenden zu können, wählen wir eine *Substitution* θ, so dass $King(x)$ und $Greedy(x)$ zu $King(john)$ bzw. $Greedy(y)$ passen. → Die Substitution θ ersetzt x und y jeweils durch $john$ und wir erhalten:

$$King(john) \wedge Greedy(john) \Rightarrow Evil(john)$$
$$King(john)$$
$$Greedy(john)$$

2) Damit können wir einen prädikatenlog. Modus Ponens anwenden: alle Klauseln sind konjunktiv verknüpft; also entsprechen die beiden konjunktiv verknüpften 1-Klauseln der Regelprämisse. Wir erhalten als Inferenzergebnis:

$$Evil(john)$$

Substitution und Unifikation (1)

Die *Substitution* θ, die $King(x)$ und $Greedy(x)$ zu $King(john)$ bzw. $Greedy(y)$ „passend macht", wird notiert als:

$$\theta = \{x/john,\ y/john\}.$$

Eine *Substitutionskomponente* ist eine Zuordnung von einem Term *t* zu einer Variablen *v*; i. Z.:

$$\{v/t\}.$$

Eine *Substitution* θ ist eine (ggf. leere) Menge von Substitutionskomponenten mit jeweils verschiedenen Variablen.

Das Prinzip dieser *Gleichsetzung* bzw. *Vereinheitlichung* wird als *Unifikation* bezeichnet:

$$Unify(p,q) = \theta, \text{ wenn } p\theta = q\theta \text{ für zwei Ausdrücke } p \text{ und } q .$$

Substitution und Unifikation (2)

Beispiele für Substitutionen θ mit dem Ziel $\mathit{Unify}(p, q) = \theta$:

p	q	θ
Knows(john,x)	*Knows(john,jane)*	$\{x/jane\}$
Knows(john,x)	*Knows(y,bill)*	$\{x/bill,\ y/john\}$
Knows(john,x)	*Knows(y,mother(y))*	$\{y/john,\ x/mother(john)\}$
Knows(x,x)	*Knows(y,mother(y))*	*fail*
Knows(john,x)	*Knows(x,elizabeth)*	*fail*

$\{x/mother(\mathrm{x})\}$ führt zu zyklischer Substitution ohne Entsprechung im Universum

→ Occur-Check (F. 35)

Standardisierung

p	q	θ
Knows(john,x)	*Knows(x,Elizabeth)*	*fail*

I.A. ist es sinnvoll, wenn eine Variable nicht durch zwei verschiedene Terme belegt werden kann. Dies kann aber auch durch unglückliche, nämlich gleiche Benennung von verschiedenen Variablen in der Wissensbasis geschehen.

Stehen z.B. *p* und *q* des letzten Beispiels für zwei separate Sätze bzw. Klauseln, so bedeuten sie *„John kennt alle“* bzw. *„Alle kennen Elizabeth“.* Aus beiden Sätzen sollte der Schluss *„John kennt Elizabeth“* durchaus ableitbar sein. Der Konflikt rührt dann nur aus der gleichnamigen Benennung der allquantifizierten Variablen in beiden Sätzen.

Dies wird durch Standardisierung der Variablen vermieden, indem alle *Variablen verschiedener Klauseln* unterschieden bzw. geeignet umbenannt werden. Am einfachsten erfolgt die Standardisierung, indem die Klauseln durchnummeriert werden und deren Nummern als Indizes der Variablen gesetzt werden.

Beispiel 1 für Standardisierung

Klauselmenge:

$MotherOf(x,y) \wedge MotherOf(y,z) \Rightarrow GrandmotherOf(x,z)$

$MotherOf(queenMum,elizabeth)$

$MotherOf(elizabeth,anne)$

$Bird(x) \Rightarrow CanFly(x)$

$Bird(tweety)$

Vereinfachte Schreibweise der Klauselmenge: jede Zeile für eine Klausel. Diese konjunktiv verknüpft zur Klauselmenge

Nummerierte und standardisierte Klauselmenge:

K_1: $MotherOf(x_1,y_1) \wedge MotherOf(y_1,z_1) \Rightarrow GrandmotherOf(x_1,z_1)$

K_2: $MotherOf(queenMum,elizabeth)$

K_3: $MotherOf(elizabeth,anne)$

K_4: $Bird(x_4) \Rightarrow CanFly(x_4)$

K_5: $Bird(tweety)$

Frage:

$GrandmotherOf(queenMum,anne) \wedge CanFly(tweety)?$

Allgemeinste Unifikation

Die Unifikation von *Knows(john, x)* und *Knows(y, z)* kann erfolgen durch:

$\{y/john,\ x/z\}$ oder

$\{y/john,\ x/john,\ z/john\}$

- Die 2. Unifikation ist eine (unnötige) Spezialisierung der 1. Unifikation.
- Die 1. Unifikation ist die *allgemeinste Unifikation*.
- Der *allgemeinste Unifikator* (*Most general unifer, MGU)* ist eindeutig bis auf Umbenennung und wird durch den folg. Algorithmus UNIFY berechnet.

Unifikation: Algm. UNIFY

function UNIFY(x, y, θ) **returns** a substitution to make x and y identical
  inputs: x, a variable, constant, list, or compound
    y, a variable, constant, list, or compound
    θ, the substitution built up so far (optional, defaults to empty)

  if θ = *failure* **then return** *failure*
  else if $x = y$ **then return** θ
  else if VARIABLE?(x) **then return** UNIFY-VAR(x, y, θ)
  else if VARIABLE?(y) **then return** UNIFY-VAR(y, x, θ)
  else if COMPOUND?(x) **and** COMPOUND?(y) **then**
    return UNIFY(ARGS[x],ARGS[y],UNIFY(OP[x],OP[y],θ))
  else if LIST?(x) **and** LIST?(y) **then**
    return UNIFY(REST[x],REST[y],UNIFY(FIRST[x],FIRST[y],θ))
  else return *failure*

a

1. Argument = Variable

2 Funktionale/ Prädikate mit Argumenten

2 Argument-listen

[a] Compound expression $f(a, b)$: OP bearbeitet f und ARGS die Argumentliste (a, b).

Bsple.: UNIFY(P(a,x), P(y,f(y,z)), ∅), UNIFY(P(a,f(u,v)), P(y,f(y,z)), ∅), UNIFY(P(x,x), P(y,f(y)), ∅)

Unifikation: Algm. UNIFY-VAR

function UNIFY-VAR(var, x, θ) **returns** a substitution
  inputs: var, a variable
    x, any expression
    θ, the substitution built up so far
  if $\{var/val\} \in \theta$ **then return** UNIFY(val, x, θ)
  else if $\{x/val\} \in \theta$ **then return** UNIFY(var, val, θ)
  else if OCCUR-CHECK?(var, x) **then return** *failure*
  else return add$\{var, x\}$ to θ

Übung

Der OCCUR-CHECK? muss auch mögl. Substitutionen der Argumente des Ausdrucks x berücksichtigen. Bspl.: Variable $var = y$, Ausdruck $x = f(z)$ und Substitution $\{z/y\} \in \theta$.

Der OCCUR-CHECK verursacht die quadrat. Komplexität der Unifikation. Einige Systeme vernachlässigen diesen um den Preis von z.T. ungültigen Inferenzen. Komplexere Unifikationsalgorithmen können mit linearer Komplexität rechnen.

Generalized Modus Ponens (1)

Mit Hilfe der Unifikation können wir nun den Generalisierten Modus Ponens (GMP) für die Prädikatenlogik einführen.

Zunächst stellen wir fest: unsere Wissensbasis KB enthält nur *definite Klauseln* mit *genau* einem positiven Literal:

$King(x) \wedge Greedy(x) \Rightarrow Evil(x)$
$King(john)$
$Greedy(y)$
$Brother(richard, john)$

$\Leftrightarrow$

$\{\ \{ -King(x), -Greedy(x), Evil(x) \},$
$\{ King(john) \},$
$\{ Greedy(y) \},$
$\{ Brother(richard, john) \}\ \}$

Daran müssen wir auch allgemein festhalten:

Satz: Auf prädikatenlogischen definiten Klauselmengen arbeitet der verallgemeinerte Modus Ponens korrekt, vollständig und effizient:

$$\frac{p_1', p_2', \ldots, p_n', (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta} \quad \text{mit } p'_i\,\theta = p_i\,\theta \text{ für alle } i.$$

Generalized Modus Ponens (2)

Beispiel: KB in KNF

$King(x) \wedge Greedy(x) \Rightarrow Evil(x)$ $King(john)$ $Greedy(y)$ $Brother(richard, john)$	$\Leftrightarrow$	$\{$ $\{$ $-King(x), -Greedy(x), Evil(x)$ $\}$, $\{$ $King(john)$ $\}$, $\{$ $Greedy(y)$ $\}$, $\{$ $Brother(richard, john)$ $\}$ $\}$

Anwendung des GMP: $\dfrac{p_1', p_2', \ldots, p_n', (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta}$ mit $p_i'\,\theta = p_i\,\theta\ \forall i.$

mit:					
p_1'	ist	$King(john)$	p_1	ist	$King(x)$
p_2'	ist	$Greedy(y)$	p_2	ist	$Greedy(x)$
θ	ist	$\{x/john, y/john\}$	q	ist	$Evil(x)$
$q\theta$	ist	$Evil(john)$			

Standardisierung und Klauselkopien in GMP-Beweisen

Klauselmenge:

$LessThan(x,y) \wedge LessThan(y,z) \Rightarrow LessThan(x,z)$

$LessThan(a,b)$

$LessThan(b,c)$

$LessThan(c,d)$

Nummerierte und standardisierte Klauselmenge:

K_1: $LessThan(x_1,y_1) \wedge LessThan(y_1,z_1) \Rightarrow LessThan(x_1,z_1)$

K_2: $LessThan(a,b)$

K_3: $LessThan(b,c)$

K_4: $LessThan(c,d)$

Frage:

$LessThan(a,d)$?

Standardisierung und Klauselkopien in GMP-Beweisen (Forts.)

Nummerierte und standardisierte Klauselmenge:

K_1: $LessThan(x_1,y_1) \wedge LessThan(y_1,z_1) \Rightarrow LessThan(x_1,z_1)$
K_2: $LessThan(a,b)$
K_3: $LessThan(b,c)$
K_4: $LessThan(c,d)$

Frage: $LessThan(a,d)?$

Für den Beweis brauchen das Transitivitätsgesetz (Klausel K_1) zweimal:

- Man kann y_1 nicht mit zwei verschiedenen Termen belegen: hier *b* und *c*!
- Allgemein: für Beweise können Klausenkopien nötig sein – diese müssen wieder standardisiert werden.
- Hier ist also eine Kopie von K_1 nötig:
  K_5: $LessThan(x_5,y_5) \wedge LessThan(y_5,z_5) \Rightarrow LessTan(x_5,z_5)$

Standardisierung und Klauselkopien in GMP-Beweisen (Forts.)

Erweiterte standardisierte Klauselmenge:

K_1: $LessThan(x_1,y_1) \wedge LessThan(y_1,z_1) \Rightarrow LessThan(x_1,z_1)$

K_2: $LessThan(a,b)$

K_3: $LessThan(b,c)$

K_4: $LessThan(c,d)$

K_5: $LessThan(x_5,y_5) \wedge LessThan(y_5,z_5) \Rightarrow LessThan(x_5,z_5)$

Frage: $LessThan(a,d)$?

Beweis:

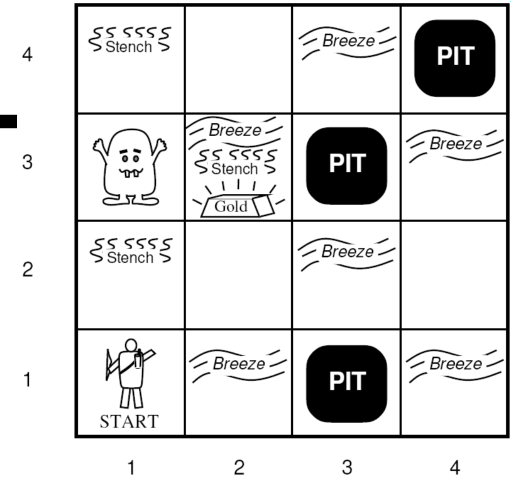
$$\frac{LessThan(a,b), LessThan(b,c), [LessThan(x_1,y_1) \wedge LessThan(y_1,z_1) \Rightarrow LessThan(x_1,z_1)]}{LessThan(a,c)}$$

$mit\ \theta = \{x_1/a,\ \ y_1/b,\ \ z_1/c\},$

$$\frac{LessThan(a,c), LessThan(c,d), [LessThan(x_5,y_5) \wedge LessThan(y_5,z_5) \Rightarrow LessThan(x_5,z_5)]}{\boxed{LessThan(a,d)}}$$

$mit\ \theta = \theta \cup \{x_5/a,\ \ y_5/c,\ \ z_5/d\}.$

Back to Wumpus World (1)



PL-Axiome der Wumpus Welt sind kompakter und drücken Sachverhalte in natürlicherer Weise ab als die AL-Axiome.

Beispiele:

- Die Perzept-5-Tupel sowie die (diskrete) Zeitpunkte ihrer Wahrnehmung:

  Percept ([stench, breeze, glitter, none, none], 5)

- Die Perzepte führen zu Beschreibungen aktueller Zustände:

  $\forall$ t, s, g, m, c Percept ([s, breeze, g,m, c], t) $\Rightarrow$ Breeze(t)
  $\forall$ t, s, b, m, c Percept ([s, b, glitter,m, c], t) $\Rightarrow$ Glitter (t)

- Aktionen sind als Terme formulierbar:

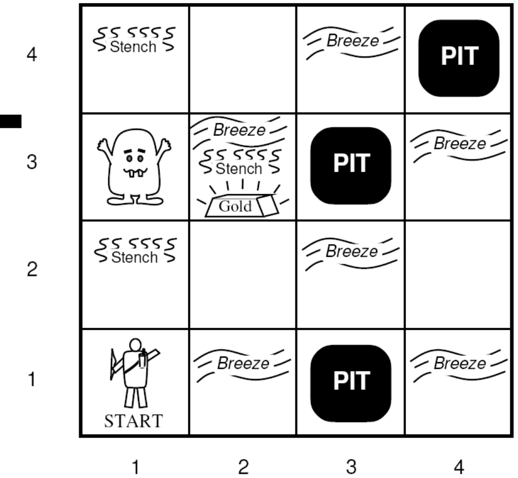
  turn(right), turn(left), forward, shoot, grab, climb

- Die beste Aktion ist auswählbar nach:

  ASKVARS($\exists$a BestAction(a, 5))
  und liefert eine Substitution {a/grab}

ASKVARS fragt nach Varibalenbelegung, die Aussage wahr macht

Back to Wumpus World (2)



- Einfaches Reflexverhalten:

$$\forall\, t\ \mathrm{Glitter}(t) \Rightarrow \mathrm{BestAction}(grab, t)$$

- Repräsentation des Spielfeldes:

Nachteil der separaten Feldbenennung, z.B. Feld (1,2): es gäbe es wieder „Extrafakten“ zu den Nachbarschaften für jedes Feldpaar. Besser: Komplexe Terme mit Integer-Paaren wie [1,2]. Nachbarschaften sind dann definierbar durch:

$$\forall\, x, y, u, v\ \mathrm{Adjacent}\,([u, v], [x, y]) \Leftrightarrow$$
$$(u = x \land (v = y - 1 \lor v = y + 1)) \lor (v = y \land (u = x - 1 \lor u = x + 1))$$

- Persistente Feldeigenschaften:

Position von Fallgrube sind fix. Also ist der Luftzug in einem Nachbarfeld zeitunabhängig:

$$\forall\, s, t\ \mathrm{At}(agent, s, t) \land \mathrm{Breeze}(t) \Rightarrow \mathrm{Breezy}(s)$$

- Inferenzen:

$$\forall\, s\ \mathrm{Breezy}(s) \Leftrightarrow \exists r\ \mathrm{Adjacent}\,(r, s) \land \mathrm{Pit}(r)$$

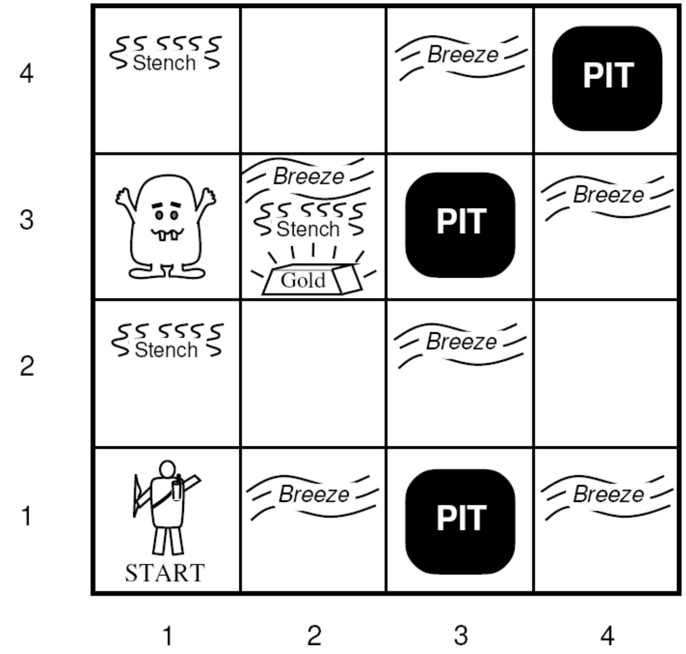
Back to Wumpus World (3)

Fazit:

Die PL-Formulierungen sind deutlich näher an den umgangssprachlichen Formulierungen, mit denen die Wumpus-Welt definiert wurde.

Gegenüber der AL erlaubt die PL prägnantere und damit kompaktere Formulierungen – insbesondere bei Quantifizierungen über Raum, Zeit und Objektdomänen.

Logische PL-Inferenz ist möglich über Skolemisierung, Unifikation und GMP.



Offen Fragen:

- In welcher Reihenfolge sind Inferenzschritte auszuwählen?

→ Ableitungsstrategien

- Wie sind Weltmodelle vollständig zu beschreiben?

→ Situationskalkül

Zusammenfassung

- PL1 erlaubt es, Aussagen zu strukturieren und gibt uns damit eine erheblich größere Ausdruckskraft als Aussagenlogik.
- Formeln bestehen aus *Termen* und *atomaren Formeln*, die mit Hilfe von *Konnektoren* und *Quantoren* zu Formeln zusammengesetzt werden.
- Interpretationen in PL1 bestehen aus einer *Domänenmenge* bzw. einem *Universum* und der *Interpretationsfunktion*.
- Durch die Unifikation können wir Methoden der AL-Inferenz für die Inferenz in der PL-1 einsetzen.
- Der Generalisierte Modus Ponens (GMP) ist die Umsetzung des Modus Ponens auf PL-1 mit Hilfe der Unifikation. Der GMP ist für definite Klauselmengen korrekt und vollständig.

Anhang: Alternative Notation

hier	sonst		
$\neg\varphi$	$\sim\varphi$	$\overline{\varphi}$	
$\varphi \wedge \psi$	$\varphi \& \psi$	$\varphi \cdot \psi$	φ, ψ
$\varphi \vee \psi$	$\varphi \vert \psi$	$\varphi; \psi$	$\varphi + \psi$
$\varphi \Rightarrow \psi$	$\varphi \rightarrow \psi$	$\varphi \supset \psi$	
$\varphi \Leftrightarrow \psi$	$\varphi \leftrightarrow \psi$	$\varphi \equiv \psi$	
$\forall x \varphi$	$(\forall x)\varphi$	$\bigwedge x \varphi$	
$\exists x \varphi$	$(\exists x)\varphi$	$\bigvee x \varphi$	