Grundlagen der Künstlichen Intelligenz

6 Logische Agenten

Rationales Denken, Logik, Resolution

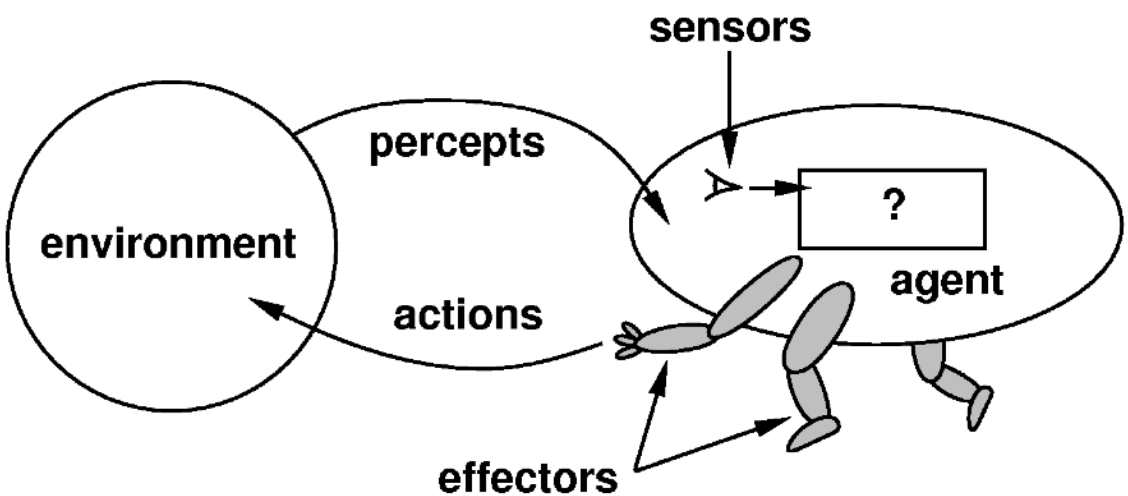
Volker Steinhage

Inhalt

- Rational denkende Agenten
- Die Wumpus-Welt
- Aussagenlogik: Syntax & Semantik
- Logische Folgerbarkeit
- Logische Ableitungen: Resolution

Rational denkende Agenten (1)

- Bisher lag das Schwergewicht auf *rational* *handelnden* Agenten.



- Für komplexe Szenarien setzt rationales Handeln auch *rationales* (logisches) *Denken* durch den Agenten voraus.

- Dazu müssen Teile der Welt symbolisch in einer *Wissensbasis* (*Knowledge base*, kurz: *KB*) repräsentiert sein

Rational denkende Agenten (2)

Eine *Wissensbasis* (*Knowledge base*, kurz: *KB*)

- enthält *Sätze* in einer Sprache, der sog. *Wissensrepräsentationssprache* (*Knowledge representation language*),
- mit einer *Wahrheitstheorie* (Logik), d.h. wir – als Außenstehende – können Sätze als *Aussagen* über die Welt *interpretieren*.

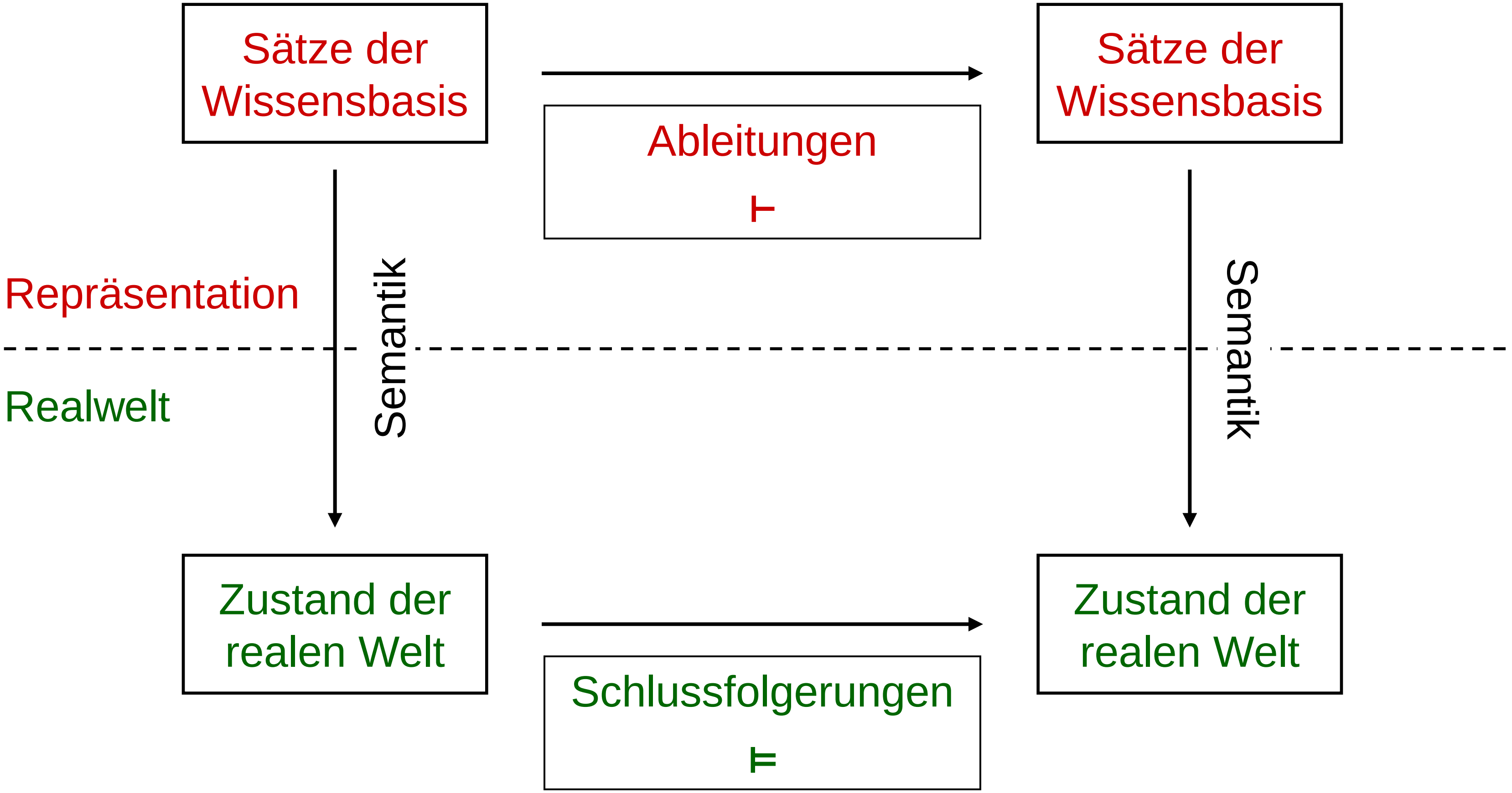
Semantik

- Die Sätze haben allein durch ihre *Struktur* einen *kausalen Einfluss* auf das Verhalten des Agenten in einer Weise, die in Korrelation zum *Inhalt* der Sätze steht.

Syntax

Rational denkende Agenten (3)

Zu Semantik und Syntax:



Rational denkende Agenten (4)

In dieser Vorlesung ist die Interaktion mit KB vereinfacht durch *ASK* und *TELL*:

- *TELL(KB,a):* Ergänze KB um *a*.
- *ASK(KB,a):* Leite *a* aus KB ab.

Prinzipiell auch *FORGET(KB,a):* entferne *a* aus KB.

Würde bei nicht-monotoner Logik auftreten – wird

hier aber nicht behandelt!

function KB-Agent (*percept*) **returns** an *action*

static: *KB*, a knowledge base

t, a time counter (initially 0)

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*

Ein wissensbasierter Agent

Ein wissensbasierter Agent benutzt seine Wissensbasis, um

(1) sein Hintergrundwissen zu repräsentieren,

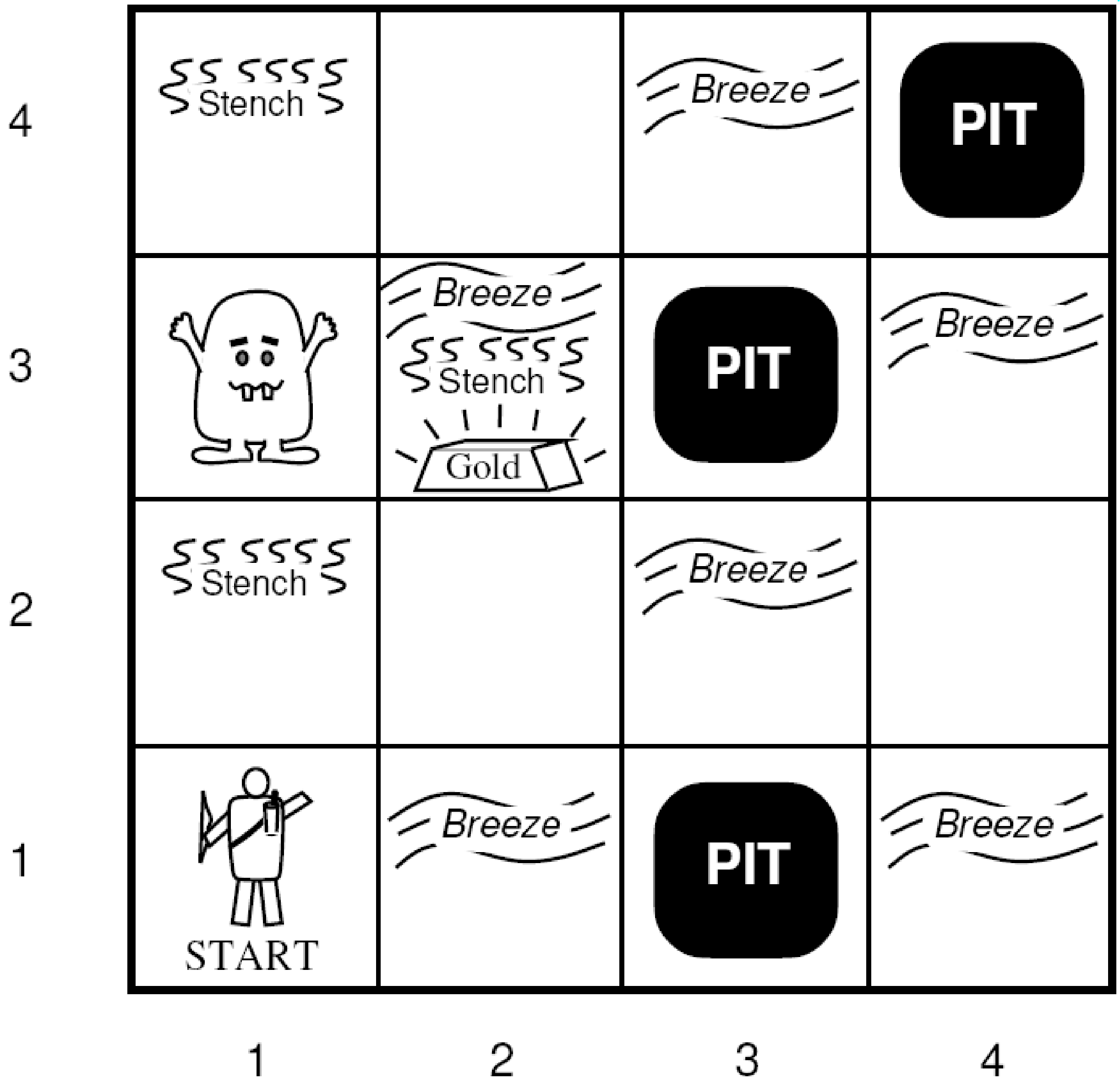
(2) seine Beobachtungen zu repräsentieren,

(3) daraus neue Aktionen abzuleiten,

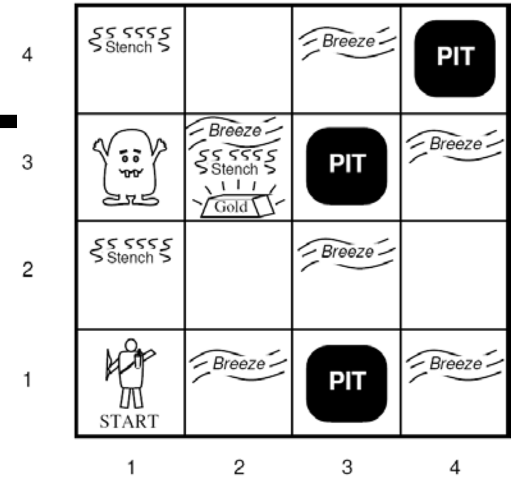
(4) die durchgeführten Aktionen zu repräsentieren.

> **function** KB-Agent (*percept*) **returns** an *action*
>
> static: *KB*, a knowledge base (1)
>
> t, a time counter (initially 0)
>
> TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept,t*)) (2)
>
> *action* ←ASK(KB,MAKE-ACTION-QUERY(*t*)) (3)
>
> TELL(*KB*, MAKE-ACTION-SENTENCE(*action,t*)) (4)
>
> $t = t + 1$
>
> **return** *action*

Die Wumpus-Welt (1): Eine Beispielkonfiguration



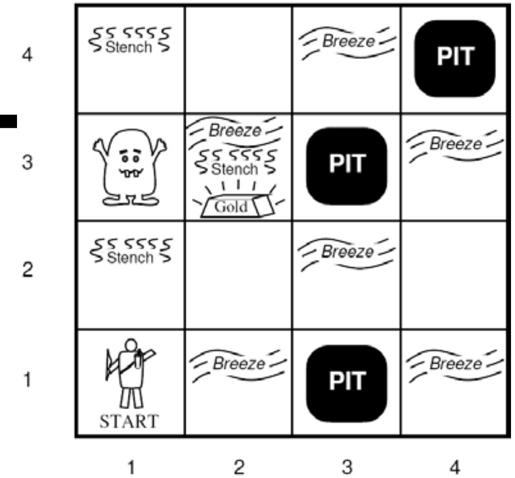
Die Wumpus-Welt (2)



Wumpus-Welt: eine 4×4-Umgebung.

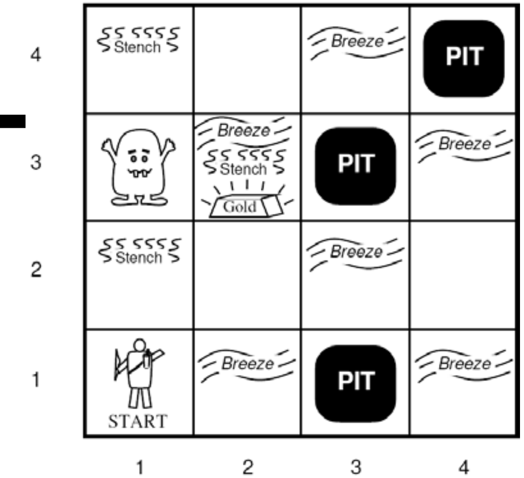
- Felder/Ereignisse und Perzepte
  - Im Feld des Wumpus' und in den direkt horizontal und vertikal benachbarten Feldern ist Geruch (Stench) wahrnehmbar.
  - Im Feld einer Fallgrube (Pit) und in den direkten horizontalen und vertikalen Nachbarfeldern ist Luftzug (Breeze) wahrnehmbar.
  - Im Feld mit dem Gold ist Glitzern (Glitter) wahrnehmbar.
  - Wenn der Agent in eine Wand läuft, bekommt er einen Stoß (Bump).
  - Wird der Wumpus getötet, ist überall sein Schrei (Scream) wahrnehmbar.
  - Wahrnehmungen werden als binäre 5-Tupel dargestellt: z.B. bedeutet [Stench, Breeze, Glitter, -Bump, -Scream], dass Geruch, Zug und Glitzern, aber kein Stoß und kein Schrei wahrnehmbar sind.

Die Wumpus-Welt (3)



- Aktionen und Zustände:
  - Aktionen:
    - greife ein Objekt im selben Feld,
    - gehe vorwärts, 90° nach rechts drehen, 90° nach links drehen,
    - schieße (es gibt nur einen Pfeil),
    - verlasse die Höhle (funktioniert nur im Feld [1,1]).
  - Der Agent stirbt, wenn er in eine Fallgrube fällt oder dem lebenden Wumpus begegnet.
  - Anfangszustand: Agent in [1,1] nach Osten schauend, irgendwo 1 Wumpus, 1 Haufen Gold und 3 Fallgruben.
  - *Ziel*: Hole das Gold und verlasse die Höhle.

Die Wumpus-Welt (4)

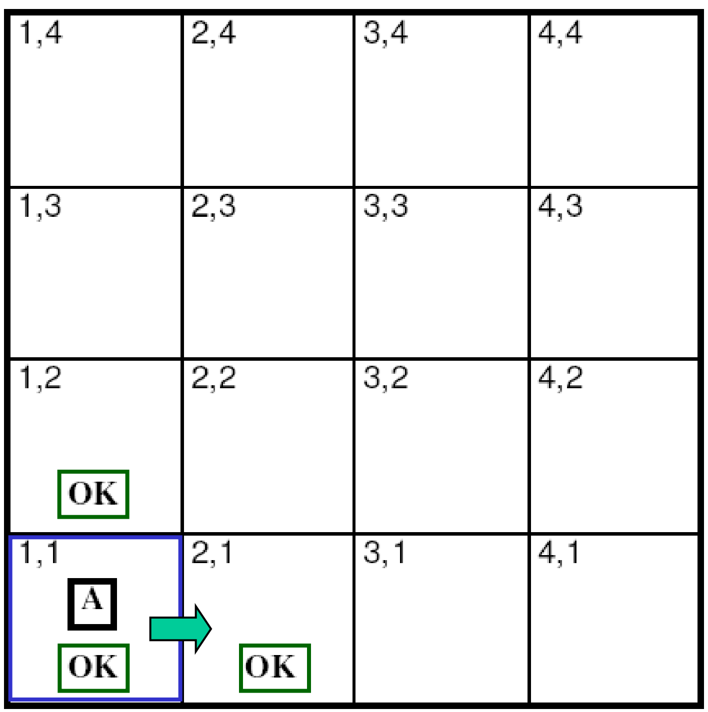


Wahrnehmungen, Inferenzen und Aktionen

(a) Agent in [1,1]: no Breeze, no Stench

→ [1,2] und [2,1] sicher (OK) → gehe z.B. zu [2,1]

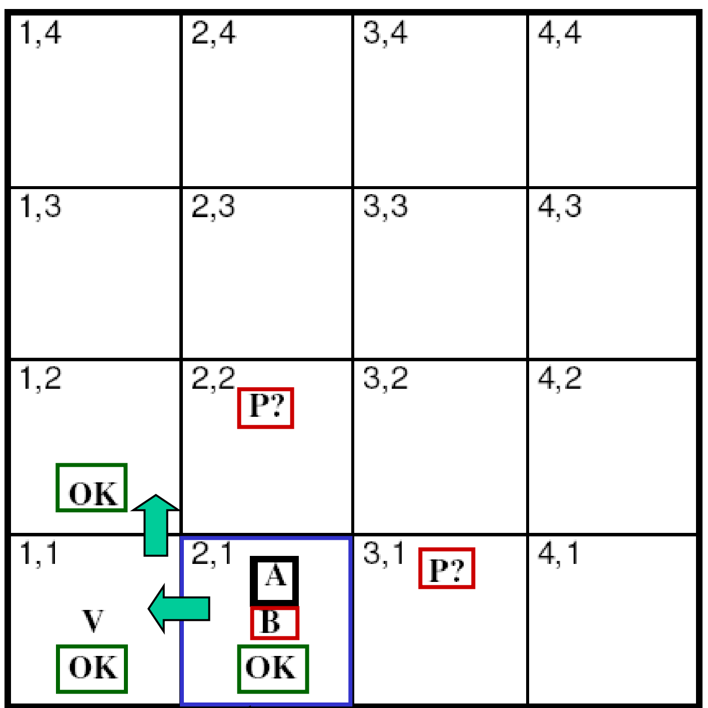
(b) Agent in [2,1]: Breeze → Pit in [2,2] oder [3,1] → gehe zu [1,2] über [1,1]



no Breeze and no Stench

(a)

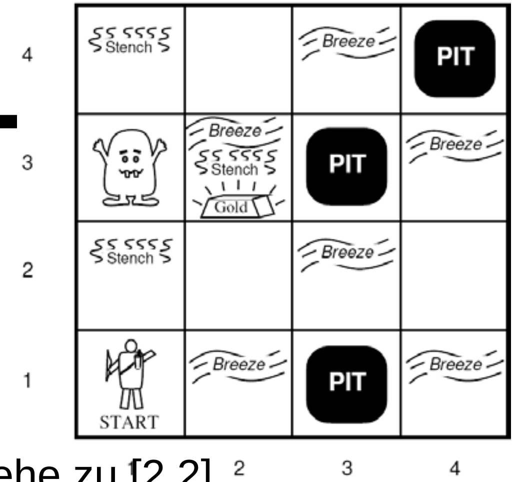
A = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*



no Stench but Breeze

(b)

Die Wumpus-Welt (5)



Wahrnehmungen, Inferenzen und Aktionen

(a) Agent in [1,2]: Stench in [1,2] → Wumpus in [1,3],

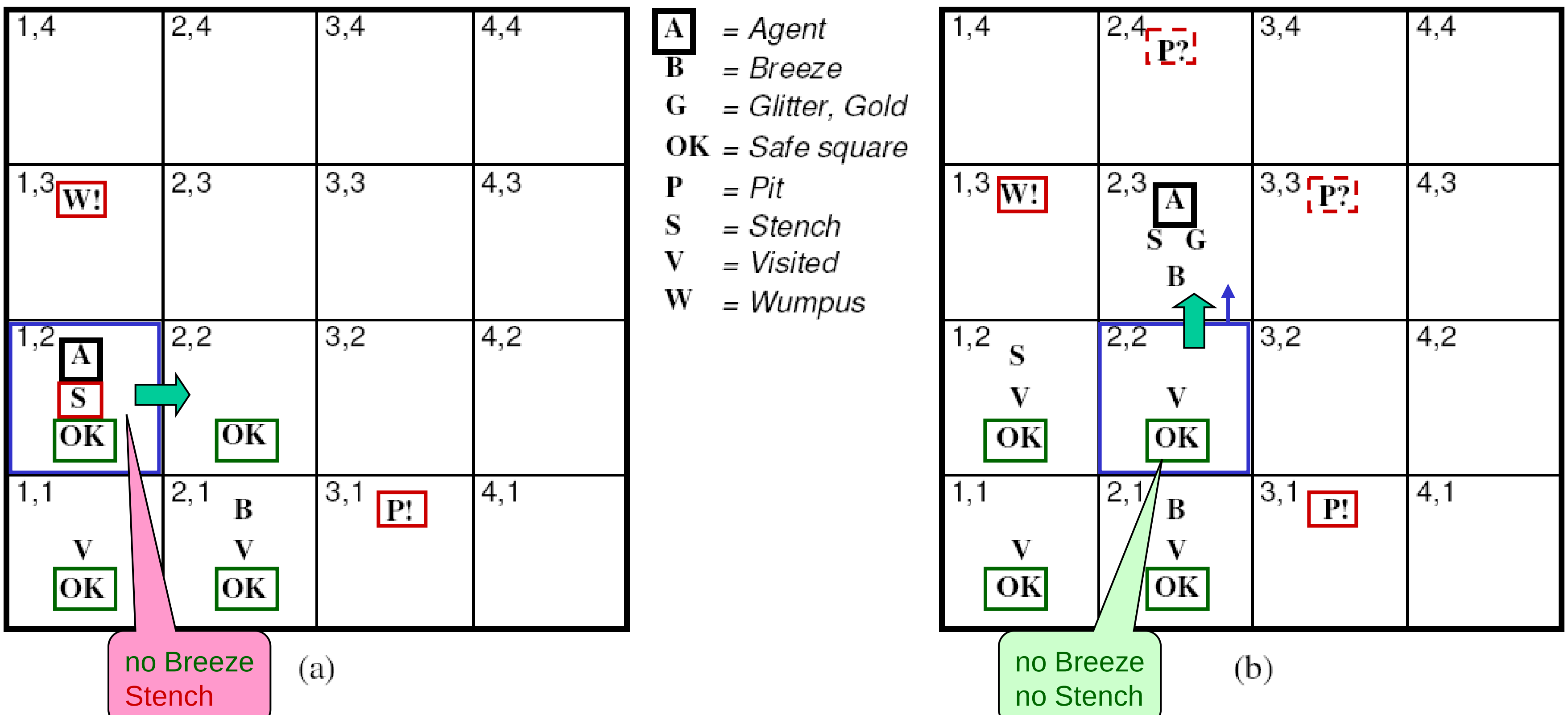
aber nicht in [2,2], da sonst Stench in [2,1] gewesen wäre

KB

KB

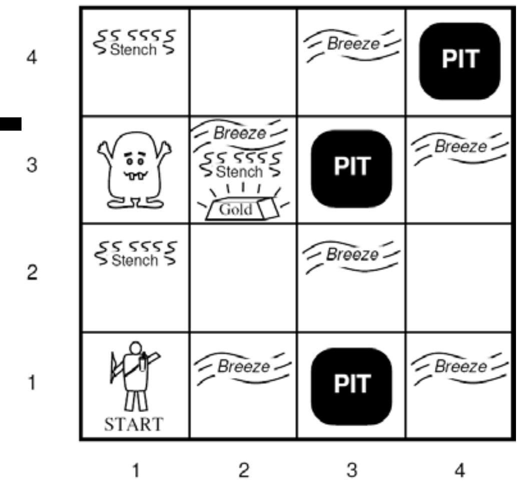
No Breeze in [1,2] → kein Pit in [2, 2], aber in [3,1] → [2,2] OK → gehe zu [2,2]

(b): No Breeze, no Stench, no Glitter in [2,2] → gehe z.B. zu [2,3] → 1. Erfolg: Gold ist gefunden



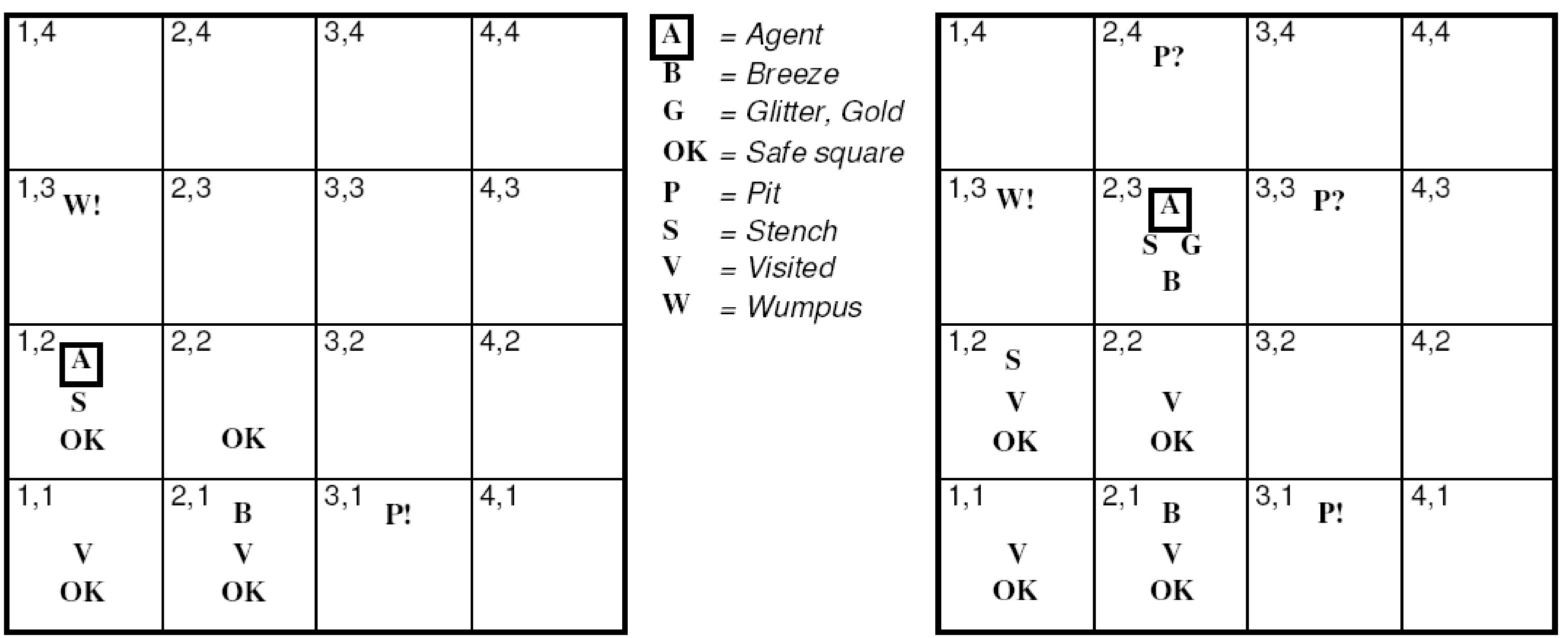
(a) (b)

Die Wumpus-Welt (6)



Wie soll der Agent

- sein Wissen, z.B. „no Stench in [1,1]“,
- seine Vermutungen, z.B. „Pit in [3,1] oder [2,2]“,
- seine Schlüsse, z.B. „no Stench in [1,1] ⇒ no Wumpus in [1,2] und [2,1]“)
- seine bisherige Aktions- und Zustandsfolge repräsentieren und auf diesen arbeiten?



(a) (b)

Deklarative Sprachen

- Als *Wissensrepräsentationssprache* eignen sich insbes. deklarative Sprachen.

- Der Einsatz einer *deklarativen Sprache* bedeutet,

  - dass mit Hilfe dieser Sprache in der *Wissensbasis* nur die Fakten und Regeln der modellierten Welt repräsentiert werden,

  - dass der prozedurale Aspekt – also das Arbeiten mit dem Wissen (genauer das Ableiten von neuem Wissen) in einer anderen Komponente, nämlich der *Inferenzmaschine*, umgesetzt wird.

- Eine *erste einfache Möglichkeit* dafür: die *Aussagenlogik.*

Die Aussagenlogik kann „liefern“ (1)

Die Aussagenlogik (AL) kann Wissen repräsentieren:

- Die *Grundbausteine* der AL sind nicht weiter zerlegbare *atomare Aussagen* (atomare Propositionen), von denen sinnvoll zu sagen ist, dass sie *wahr* oder *falsch* sind.*

  Beispiele:

  - „Der Block ist rot“
  - „Der Wumpus ist in [1,3]“
  - „Alan Turing war ein engl. Mathematiker, der die Turing-Maschine erfand.“

- Die AL hat *logische Konnektoren* wie „*und*“, „*oder*“, „*nicht*“, mit denen aus einfachen atomaren Aussagen komplexere *Formeln* konstruierbar sind.

---

* Dichotomie (Zweiteilung) der Aussagenlogik in wahre und falsche Aussagen.

Die Aussagenlogik kann „liefern“ (2)

Wir können in der Aussagenlogik schlussfolgern!

- Die AL lässt die Beantwortung zu, ob ist eine Aussage *wahr* ist.

- Die AL lässt die Beantwortung zu, ob eine Aussage aus einer Wissensbasis KB *folgt* (i.Z.: KB $\models \varphi$).

  Semantik

- Für die AL gibt es einen syntaktischen *Ableitungsbegriff* (i.Z.: KB $\vdash \varphi$), der mit dem *Folgerungsbegriff* korrespondiert.

  Syntax

Die Aussagenlogik kann „liefern“ (3)

⇝ Wir können mit der Aussagenlogik die Bedeutung und Implementierung von ASK und TELL festschreiben und umsetzen!

function KB-Agent (*percept*) **returns** an *action*

static: *KB*, a knowledge base

t, a time counter (initially 0)

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*

Zur Erinnerung: die Syntax der Aussagenlogik (1)

Geg.: (1) abzählbares Alphabet Σ von *atomaren Aussagen* (*bzw. atomaren Sätzen*)

$True, False, P, Q, R, \ldots,$

(2) logische Verknüpfungen: Negation ($\neg$), Konjunktion ($\wedge$), Disjunktion ($\vee$), Implikation ($\Rightarrow$) und Äquivalenz ($\Leftrightarrow$).

Daraus ergeben sich alle *aussagenlogische Formeln* bzw. *Sätze:*

Satz	$\rightarrow$ atomarerSatz | komplexerSatz
atomarerSatz	$\rightarrow$ $True$ | $False$ | Symbol
Symbol	$\rightarrow$ P | Q | R | $\ldots$
komplexerSatz	$\rightarrow$ $\neg$ Satz | Satz $\wedge$ Satz | Satz $\vee$ Satz
	| Satz $\Rightarrow$ Satz | Satz $\Leftrightarrow$ Satz

Zur Erinnerung: die Syntax der Aussagenlogik (2)

Zur Vermeidung von zu vielen Klammerungen:

Operator-Präzedenz: $\neg > \wedge > \vee > \Rightarrow > \Leftrightarrow$

(dennoch muss ggf. geklammert werden).

Weiter zur Terminologie:

Atom: atomarer Satz, z.B. P, Q, $True$

Literal: positives oder negiertes Atom, z.B. *(+)*P, $\neg P$ *(oder* $\overline{P}$ *oder* -P*)*

Klausel: Disjunktion bzw. Konjunktion von Literalen,

z.B. $P \vee \neg Q \vee R$ bzw. $Q \wedge R \wedge \neg S$

Semantik (intuitiv)

- Atomare Aussagen sind

  entweder *wahr* (*True* oder **T**)

  oder *falsch* (*False* oder **F**).

- Der Wahrheitswert von Formeln ergibt sich aus den Wahrheitswerten der Atome (Wahrheitsbelegung oder Interpretation).

  Beispiel: $(P \vee Q) \wedge R$

  Wenn P und Q falsch sind und R wahr ist, ist die Formel falsch.

  Wenn dagegen P und R wahr sind (Q egal), ist die Formel wahr.

Semantik (formal)

Eine *Wahrheitsbelegung* der Atome in Σ oder *Interpretation* über Σ ist eine Funktion $\mathcal{I}$:

$$\mathcal{I} : \Sigma \rightarrow \{\mathit{True}, \mathit{False}\}.$$

$\mathcal{I}$ <u>*erfüllt*</u> φ ($\mathcal{I} \models \varphi$) oder φ ist wahr unter $\mathcal{I}$, falls $\mathcal{I}(\varphi) = \mathit{True}$.

Interpretation $\mathcal{I}(\varphi)$ bzw. $\varphi^{\mathcal{I}}$ einer Formel φ:

$\mathcal{I} \models \mathit{True}$ (*True* ist unter jeder Interpretation wahr)

$\mathcal{I} \not\models \mathit{False}$ (*False* ist unter jeder Interpretation falsch)

$\mathcal{I} \models P$ gdw. $P^{\mathcal{I}} = \mathit{True}$

$\mathcal{I} \models \neg\varphi$ gdw. $\mathcal{I} \not\models \varphi$

$\mathcal{I} \models \varphi \wedge \psi$ gdw. $\mathcal{I} \models \varphi$ und $\mathcal{I} \models \psi$ für Formeln φ, ψ

$\mathcal{I} \models \varphi \vee \psi$ gdw. $\mathcal{I} \models \varphi$ oder $\mathcal{I} \models \psi$ für Formeln φ, ψ

$\mathcal{I} \models \varphi \Rightarrow \psi$ gdw. wenn $\mathcal{I} \models \varphi$, dann $\mathcal{I} \models \psi$ für Formeln φ, ψ

$\mathcal{I} \models \varphi \Leftrightarrow \psi$ gdw. $\mathcal{I} \models \varphi$ genau dann, wenn $\mathcal{I} \models \psi$ für Formeln φ, ψ

Beispiel

Geg.: Interpretation I mit

$$I : \begin{cases} P \mapsto \mathrm{T} \\ Q \mapsto \mathrm{F} \\ R \mapsto \mathrm{F} \\ S \mapsto \mathrm{T} \\ \quad\vdots \end{cases}$$

Geg.: Formel φ mit

$$\varphi = ((P \vee Q) \;\Leftrightarrow (R \vee S)) \wedge (\neg(P \wedge Q) \vee (R \wedge \neg S)).$$

Frage: ist φ wahr unter I (I erfüllt φ)?

Frage: $I \vDash \varphi$?

Übung

Weitere Terminologie

Eine Interpretation $\mathcal{I}$ heißt *Modell* der Formel φ,

wenn $\mathcal{I}$ die Formel φ erfüllt (i.Z.: $\mathcal{I} \vDash \varphi$)

Eine Interpretation $\mathcal{I}$ heißt *Modell* einer *Menge von Formeln*,

wenn $\mathcal{I}$ Modell aller Formeln der Menge ist.

Eine Formel φ heißt

- *erfüllbar*, wenn es ein Modell $\mathcal{I}$ für φ gibt: $\exists \mathcal{I} : \mathcal{I} \vDash \varphi$.
- *unerfüllbar* (*inkonsistent*), wenn es kein Modell $\mathcal{I}$ für φ gibt: $\forall \mathcal{I} : \mathcal{I} \nvDash \varphi$.
- *falsifizierbar*, wenn es eine nicht erfüllende Interpretation $\mathcal{I}$ für φ gibt: $\exists \mathcal{I} : \mathcal{I} \nvDash \varphi$
- *allgemeingültig* (*tautologisch*), wenn jede Interpr. $\mathcal{I}$ Modell von φ ist: $\forall \mathcal{I} : \mathcal{I} \vDash \varphi$.

Zwei Formeln φ und ψ heißen

- *logisch äquivalent* ($\varphi \equiv \psi$), wenn für alle $\mathcal{I}$ gilt, dass $\mathcal{I} \vDash \varphi$ gdw. $\mathcal{I} \vDash \psi$.

Entscheidbarkeit?

Also wissen wir nun: Eine Formel φ heißt

- *erfüllbar*, wenn es ein Modell $\mathcal{I}$ für φ gibt: $\exists \mathcal{I} : \mathcal{I} \models \varphi$.
- *unerfüllbar* (*inkonsistent*), wenn es kein Modell $\mathcal{I}$ für φ gibt: $\forall \mathcal{I} : \mathcal{I} \not\models \varphi$.
- *falsifizierbar*, wenn es eine nicht erfüllende Interpretation $\mathcal{I}$ für φ gibt: $\exists \mathcal{I} : \mathcal{I} \not\models \varphi$.
- *allgemeingültig* (*tautologisch*), wenn jede Interpr. $\mathcal{I}$ Modell von φ ist: $\forall \mathcal{I} : \mathcal{I} \models \varphi$.

Wie entscheiden wir, ob eine Formel erfüllbar, allgemeingültig usw. ist?

Die Wahrheitstabellenmethode (1)

- Die einfachste Methode zur Überprüfung einer Formel auf Erfüllbarkeit, Allgemeingültigkeit usw. ist die Aufstellung der Wahrheitstabelle!

- Beispiel: Ist $\varphi = ((P \vee H) \wedge \neg H) \Rightarrow P$ allgemeingültig?

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
F	F	F	F	T
F	T	T	F	T
T	F	T	T	T
T	T	T	F	T

Die Wahrheitstabellenmethode (2)

- Beispiel: Ist $\varphi = ((P \vee H) \wedge \neg H) \Rightarrow P$ allgemeingültig?

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
F	F	F	F	T
F	T	T	F	T
T	F	T	T	T
T	T	T	F	T

$\leadsto$ φ ist unter allen Wahrheitsbelegungen wahr $\leadsto$ φ ist allgemeingültig.

- Entsprechend für Erfüllbarkeit, Falsifizierbarkeit, Unerfüllbarkeit.
- Hat die Wahrheitstabellen-Methode Nachteile?
  - Wenn ja: welche Alternativen gibt es?

Inferenz

- Nachteil ist: Die Wertetabellenmethode hat leider exponentielle Komplexität!
  - Wenn Formel φ insgesamt N verschiede Symbole enthält,

    dann enthält die Wertetabelle insgesamt 2^N Einträge (Zeilen).
  - Der Grund: Die Wertetabellenmethode ist eine *exhaustive Aufzählung*

    aller Interpretation der Formel φ.
- *Inferenz* stellt die effizientere Variante gegenüber der Wertetabellenmethode dar

➢ und: *Normalformen* steigern zusätzlich die Effizienz der Inferenz.

Normalformen

- Eine Formel φ ist in *konjunktiver Normalform* (*KNF*), wenn φ eine Konjunktion von disjunktiven Klauseln (Disjunktionen von Literalen $l_{i,j}$) ist:

$$\bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{m_i} l_{i,j} \right).$$

$$(\neg P \vee Q) \wedge (\neg Q \vee R)$$

- Eine Formel φ ist in *disjunktiver Normalform* (*DNF*), wenn φ eine Disjunktion von konjunktiven Klauseln (Konjunktionen von Literalen $l_{i,j}$) ist:

$$\bigvee_{i=1}^{n} \left(\bigwedge_{j=1}^{m_i} l_{i,j} \right).$$

$$(\neg P \wedge Q) \vee (\neg Q \wedge R)$$

- Zu *jeder* Formel φ existieren *äquivalente Formeln in KNF und DNF*.
- Eine Formel φ in *DNF ist erfüllbar* gdw. mind. ein Disjunkt von φ erfüllbar ist.
- Eine Formel φ in *KNF ist allgemeingültig* gdw. wenn jedes Konjunkt von φ allgemeingültig ist.

Normalformen $\rightarrow$ Klauselmengen (1)

- Formeln in KNF oder DNF lassen sich als Klauselmengen* darstellen.

- Für eine Formel φ in KNF:

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

Dabei verzichtet man auf das Schreiben von „$\vee$“ und „$\wedge$“.

  - Jede Disjunktion von Literalen wird als Literalmenge $\{l_{i,j}\}$ $(j \in \{1,\ldots,m_i\})$ notiert und als Klausel* C_i bezeichnet.

  - Die Konjunktion aller Klauseln C_i wird als Klauselmenge $\Delta = \{C_i\}$ $(i \in \{1,\ldots,n\})$ notiert.

- Die *leere Klausel* wird mit $\square$ notiert und ist unerfüllbar.

---

* Engl.: clauses – im Deutschen auch: Clausen, Klausen.

Normalformen → Klauselmengen (2)

- Formeln in KNF oder DNF lassen sich als Klauselmengen* darstellen.
- Für eine Formel φ in DNF

$$\bigvee_{i=1}^{n} \left(\bigwedge_{j=1}^{m_i} l_{i,j} \right).$$

  - Jede Konjunktion von Literalen wird als Literalmenge $\{\boldsymbol{l_{i,j}}\}$ $(j \in \{1,...,m_i\})$ notiert und als Klausel C_i bezeichnet.
  - Die Disjunktion aller Klauseln C_i wird als Klauselmenge $\Delta = \{\boldsymbol{C_i}\}$ $(i \in \{1,...,n\})$ notiert.

- Die KNF erlaubt eine einfache Schreib- und Leseweise von Fakten und Regeln einer Wissensbasis und wird daher im Weiteren gegenüber der DNF bevorzugt.

Erzeugen der KNF

Vier Schritte:

- Eliminiere „$\Rightarrow$“ und „$\Leftrightarrow$“ : $\alpha \Rightarrow \beta \;\rightarrow\; (\neg\, \alpha \vee \beta)$ usw.
- Schiebe „$\neg$“ nach innen: $\neg\, (\alpha \vee \beta) \;\rightarrow\; (\neg\, \alpha \wedge \neg\, \beta)$ usw.
- Verteile: „$\vee$“ über „$\wedge$“ : $((\alpha \wedge \beta) \vee \gamma) \;\rightarrow\; ((\alpha \vee \gamma) \wedge (\beta \vee \gamma))$ usw.
- Vereinfache: $(\alpha \vee \alpha) \;\rightarrow\; \alpha$ usw.

Übung

Ergebnis ist eine Konjunktion von Disjunktionen von Literalen!

Bemerkungen:

- Ein analoges Verfahren überführt eine beliebige Formel in eine äquivalente Formel in DNF.
- Formeln können bei der Transformation exponentiell aufgebläht werden.

Logische Folgerbarkeit (intuitiv)

- I.A. beschreibt eine Wissensbasis KB (= Menge von Formeln) die Welt nur unvollständig $\leadsto$ KB lässt die Wahrheitswerte einiger (oft vieler) Aussagen offen.

- *Beispiel*: $KB = \{P \lor Q, R \lor \neg P, S\}$ (in KNF)
  - *KB* ist definitiv bzgl. S, lässt aber P, Q, R offen (allerdings nicht beliebig).
  - Modelle von KB:

P	Q	R	S
F	T	F	T
F	T	T	T
T	F	T	T
T	T	T	T

In allen Modellen von KB sind auch $Q \lor R$ sowie $R \lor \neg P$ wahr,

d.h. $Q \lor R$ sowie $R \lor \neg P$ *folgen logisch* aus KB.

Logische Folgerbarkeit (formal)

Formel φ folgt aus KB (i.Z.: $KB \models \varphi$), wenn φ in allen Modellen von KB wahr ist:

$KB \models \varphi$ gdw. $\mathcal{I} \models \varphi$ für alle Modelle $\mathcal{I}$ von KB.

Einige beweisbare Eigenschaften der Folgerungsbeziehung:

- Deduktionssatz: $KB \cup \{\varphi\} \models \psi$ gdw. KB $\models \varphi \Rightarrow \psi$
- Kontrapositionssatz: $KB \cup \{\varphi\} \models \neg\psi$ gdw. KB $\cup \{\psi\} \models \neg\varphi$
- Widerspruchssatz: $KB \cup \{\varphi\}$ ist unerfüllbar gdw. $KB \models \neg\varphi$

Frage:

- Können wir $KB \models \varphi$ entscheiden, ohne alle Interpretationen betrachten zu müssen (die Wahrheitstabellenmethode)?

Inferenzen, Kalküle und Beweise (1)

Kernidee:

- Wir können aus den Formeln der KB solche neuen Formeln *syntaktisch ableiten*, die aus eben diesen Formeln der KB *semantisch folgen* müssen.

- Beispiel: Wenn $KB = \{ \ldots, (\varphi \Rightarrow \psi), \ldots, \varphi, \ldots \}$,

  dann gilt immer: $KB \models \psi$.

- Methodik: *Inferenzregeln* wie z.B. der *Modus Ponens*:

$$\frac{\varphi, \varphi \Rightarrow \psi}{\psi}.$$

Inferenzen, Kalküle und Beweise (2)

Terminologie*:*

- *Kalkül:* Menge von *Inferenzregeln* und logischen Axiomen.
- *Beweisschritt:* Anwendung einer *Inferenzregel* auf eine Menge von Formeln.
- *Beweis:* Sequenz von Beweisschritten, wobei
  - die mit jedem Schritt neu abgeleiteten Formeln zu *KB* hinzugefügt werden,
  - im letzten Schritt die *Zielformel* erzeugt wird.

Korrektheit und Vollständigkeit

- Falls es bei Wissensbasis *KB* in einem Kalkül *C* einen Beweis für eine Formel φ gibt, schreiben wir (ggf. ohne Subskript *C*, wenn *C* eindeutig):

$$KB \vdash_C \varphi \, .$$

- Ein Kalkül C heißt *korrekt*, wenn alle aus einer *KB* ableitbaren Formeln auch tatsächlich logisch folgen:

$$KB \vdash_C \varphi \text{ impliziert } KB \models \varphi \, .$$

Folgt i.A. aus Korrektheit der Inferenzregeln und der logischen Axiome.

- Ein Kalkül heißt *vollständig*, falls jede Formel, die logisch aus KB folgt, auch aus *KB* ableitbar ist:

$$KB \models \varphi \text{ impliziert } KB \vdash_C \varphi \, .$$

Das Resolutionskalkül: Idee der Resolution

- *Idee:* Man versucht, *zielgerichtet* die Ableitungen so zu wählen, dass ein Zielzustand möglichst effizient erreicht wird. Dies geschieht über einen *Widerspruchsbeweis*.

- *Allerdings:* Es wird vorausgesetzt, dass alle Formeln in *KNF* vorliegen.

- *Aber:* In den meisten Fällen sind die Formeln nahe an KNF (und es existiert eine schnelle erfüllbarkeitserhaltende Transformation).

- *Dennoch:* Auch dieses Ableitungsverfahren benötigt im *schlechtesten Fall exponentiell viel Zeit*.

Das Resolutionskalkül: Repräsentation

Annahme: Alle Formeln der Formelmenge *KB* liegen in KNF vor.

(Äquivalente Annahme: *KB* ist eine Menge von Klauseln.)

Bezeichne Δ eine Menge von Klauseln. Bezeichnen C eine Klausel und $\square$ die leere Klausel. Bezeichnen l ein Literal und $\bar{l}$ bzw. $\neg l$ dessen Negation. Es gilt:

- Eine Interpretation erfüllt C gdw. es ein $l \in C$ gibt, so dass $\mathcal{I} \models l$.
- Keine Interpretation $\mathcal{I}$ erfüllt $\square$!
- $\mathcal{I}$ erfüllt Δ, falls für alle $C \in \Delta : \mathcal{I} \models C$
- $\mathcal{I} \not\models \square$, $\mathcal{I} \not\models \{\square\}$, $\mathcal{I} \models \{\,\}$ für alle $\mathcal{I}$.

Zur Erinnerung: Eine Formel in *KNF ist allgemeingültig* gdw. wenn jedes Konjunkt allgemeingültig ist.

Das Resolutionskalkül: die Resolutionsregel

Die Regel der (*Grund-)Resolution:*

$$\frac{C_1 \cup \{l\}, C_2 \cup \{\bar{l}\}}{C_1 \cup C_2}$$

Mengenschreibweise von Klauseln (s. Folie 29)

- $C_1 \cup C_2$ wird *(Grund-)Resolvente* der *Elternklauseln* $C_1 \cup \{ l \}$ und $C_2 \cup \{ \bar{l} \}$ genannt.
- l und $\bar{l}$ sind die *Resolutionsliterale*.

- Beispiel: $\{a, b, \neg c\}$ ist mit $\{a, d, c\}$ über c und $\neg c$ resolvierbar zu $\{a, b, d\}$.
- Beachte*:* die Resolvente ist *nicht* äquivalent mit den Elternklauseln, sie folgt aber aus diesen!

  Übung

- Bemerkg.: wir sprechen von Grundresolution, solange wir die Resolution in der Aussagenlogik anwenden. Später mehr dazu.

Das Resolutionskalkül: Resolutionsableitungen

- Die Anwendung der Grundresolution auf eine Klauselmenge Δ:

$$R(\Delta) = \Delta \cup \{C \mid C \text{ ist Resolvente zweier Klauseln aus } \Delta\}.$$

- Wir sagen, dass Klausel D aus Δ mit Hilfe von (Grund-)Resolution *abgeleitet* werden kann, i.Z.

$$\Delta \vdash D\,,$$

  wenn es $C_1, C_2, \ldots, C_n = D$ gibt, so dass

$$C_i \in R(\Delta \cup \{C_1\}, \ldots, \{C_{i-1}\}), \text{ für } 1 \leq i \leq n.$$

Korrektheit der (Grund)-Resolution

Lemma: (*Korrektheit der (Grund-)Resolution*):

Seien $C_1 = C_1' \vee l$

und $C_2 = C_2' \vee \neg l$,

dann ist die Resolvente $C_1' \vee C_2'$ logische Folgerung von C_1 *und* C_2 .

Beweisidee: Fallunterscheidung über l oder Wahrheitswerttabelle.

Satz: (*Korrektheit der Ableitung durch (Grund-)Resolution*):

Wenn $\Delta \vdash D$, dann $\Delta \models D$.

Beweisidee: Da alle $D \in R(\Delta)$ aus Δ logisch folgen, ergibt sich der Satz durch Induktion über die Länge der Ableitung.

Definition: Eine Ableitung von $\square$ aus Δ heißt *Widerlegung* von Δ .

Vollständigkeit der (Grund)-Resolution

Ist die (Grund-)Resolution auch vollständig: $\Delta \models \varphi$ *impliziert* $\Delta \vdash \varphi$?

- Höchstens für Klauseln!
- Aber: $$\left\{ \{a, b\}, \{\neg b, c\} \right\} \begin{array}{ll} \models & \{a, b, c\} \\ \not\vdash & \{a, b, c\} \end{array}$$

Jedoch ist die (Grund-)Resolution *widerlegungsvollständig*:

(Grund-)Resolutions-Theorem: Δ *ist inkonsistent* gdw. $\Delta \vdash \Box$.

D.h. wir können $KB \models \varphi$ zeigen mit Hilfe des Widerspruchssatzes (s. Folie 33):

$KB \cup \{\neg\varphi\}$ *ist inkonsistent* gdw. $KB \models \varphi$.

Das entspr. Theorem ist das *(Grund-)Resolutions-Theorem*:

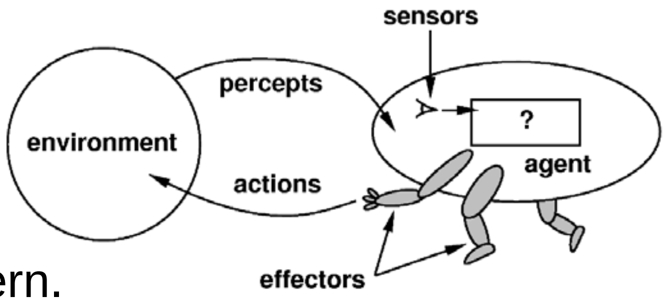
Δ ist inkonsistent gdw. $\Delta \vdash \Box$.*

* Anschaulicher Beweis über semantische Bäume im Anhang.

Vollständigkeit der (Grund)-Resolution

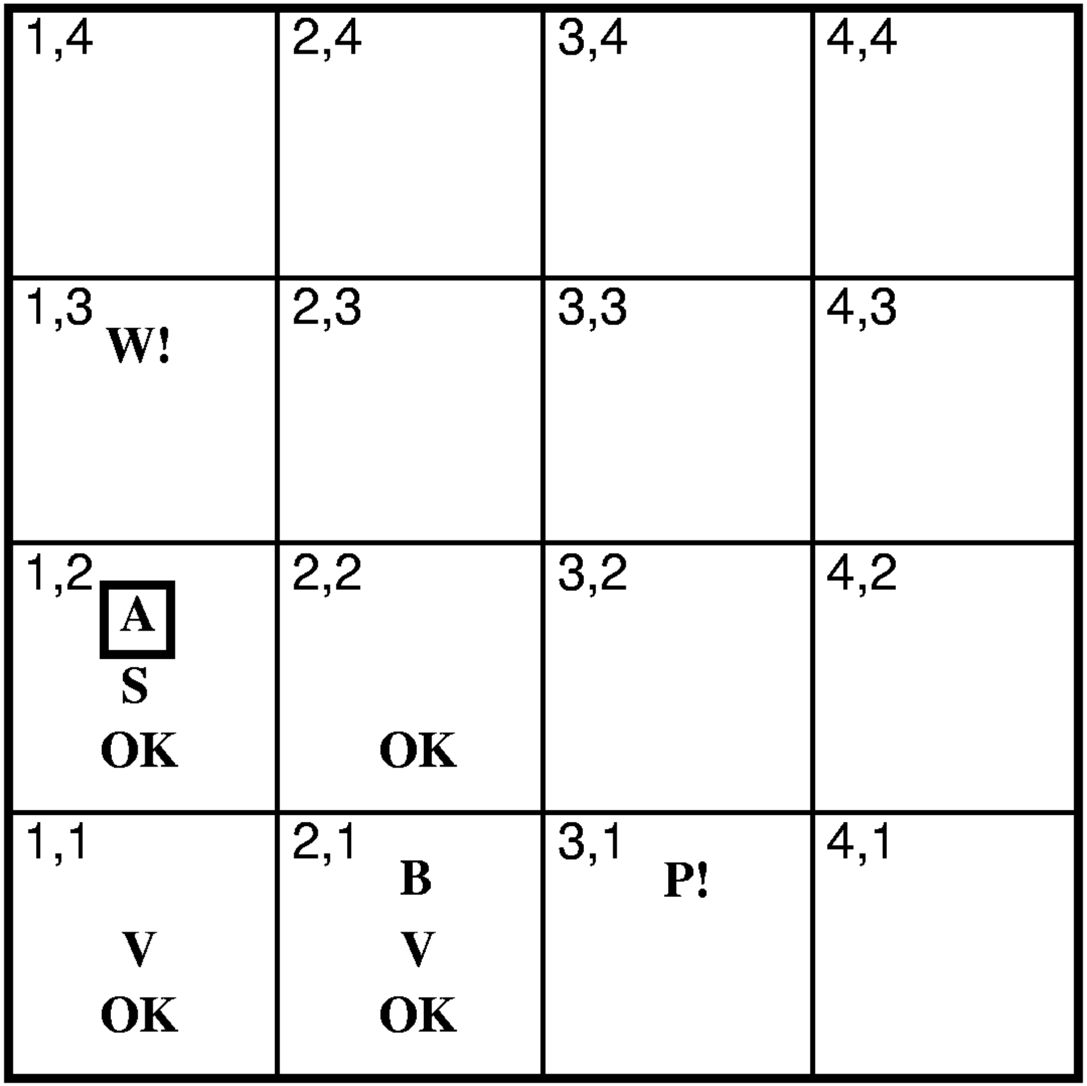
Fazit:

- Die Aussagenlogik mit der Grundresolution scheint geeignet zu sein für den Aufbau der Wissensbasis eines wissensbasierten Agenten, d.h. um
  - sein Hintergrundwissen zu repräsentieren,
  - seine Beobachtungen zu speichern,
  - daraus neue Aktionen abzuleiten,
  - die durchgeführten Aktionen zu speichern.



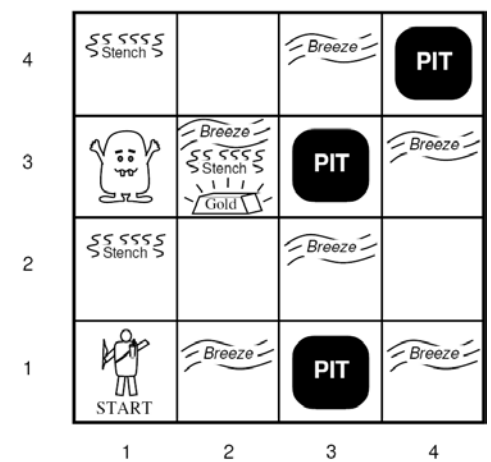
- Aussagenlogik und Grundresolution können
  - einfache und komplexe Aussagen repräsentieren,
  - korrekte und widerlegungsvollständige Ableitungen umsetzen.

Die Grundresolution für die Wumpus-Welt? – Die Situation



A = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*

Historie: Zuvor von [1,1] erst nach [2,1], dann über [1,1] nach [1,2].



Grundresolution für die Wumpus-Welt? – Situationswissen

- (Fakten-)Wissen über die Situation (u.a.) $\rightarrow$ Fakten-Klauseln $F_1, F_2, \ldots$

  $F_1: \neg S_{1,1}$ $\quad$ $F_2: \neg B_{1,1}$

  $F_3: \neg S_{2,1}$ $\quad$ $F_4: \ B_{2,1}$

  $F_5: \ S_{1,2}$ $\quad$ $F_6: \neg B_{1,2}$

  mit $B_{i,j}$ = Breeze in (i, j), $S_{i,j}$ = Stench in (i, j)

- (Regel-)Wissen über die Wumpus-Welt: $\rightarrow$ Regel-Klauseln $R_1, R_2, \ldots$

  $R_1: \quad \neg S_{1,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,1}$

  $R_2: \quad \neg S_{2,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{2,1} \wedge \neg W_{2,2} \wedge \neg W_{3,1}$

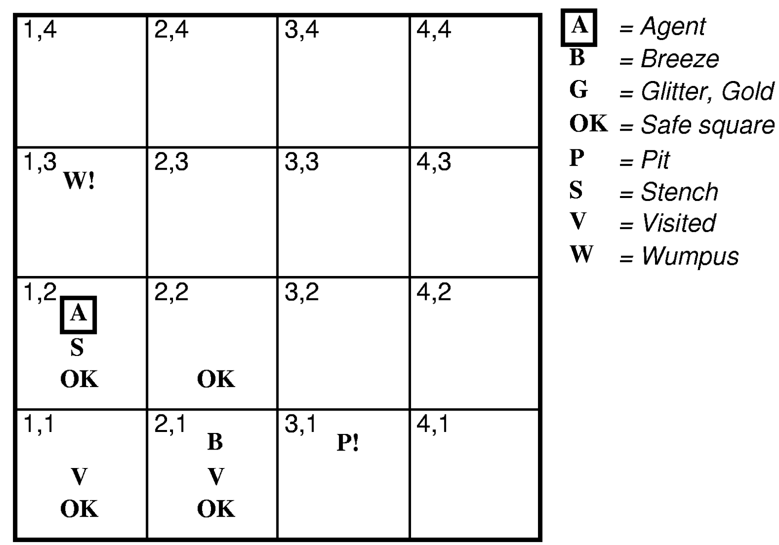
  $R_3: \quad \neg S_{1,2} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,2} \wedge \neg W_{1,3}$

  $R_4: \quad S_{1,2} \Rightarrow W_{1,3} \vee W_{1,2} \vee W_{2,2} \vee W_{1,1}$

  ...

- Ziel: zeige, dass der Wumpus in Feld (1,3) ist

  $KB \models W_{1,3}$



Grundresolution für die Wumpus-Welt? – Klauseldarstellung

- Situationswissen als Faktenklauseln der Länge 1 (sog. 1-Klauseln):

F_1: $\neg S_{1,1}$, F_2: $\neg B_{1,1}$, F_3: $\neg S_{2,1}$, F_4: $B_{2,1}$, F_5: $S_{1,2}$, F_6: $\neg B_{1,2}$, . . .

- Regelwissen als Regelklauseln der Länge n > 1:

$R_{1,1}$: $S_{1,1}, \neg W_{1,1}$, $R_{1,2}$: $S_{1,1}, \neg W_{1,2}$, $R_{1,3}$: $S_{1,1}, \neg W_{2,1}$, (drei 2-Klauseln)

$R_{2,1}$: $S_{2,1}, \neg W_{1,1}$, $R_{2,2}$: $S_{2,1}, \neg W_{2,1}$, $R_{2,3}$: $S_{2,1}, \neg W_{2,2}$, $R_{2,4}$: $S_{2,1}, \neg W_{3,1}$, (vier 2-Klauseln)

$R_{3,1}$: $S_{1,2}, \neg W_{1,1}$, $R_{3,2}$: $S_{1,2}, \neg W_{1,2}$, $R_{3,3}$: $S_{1,2}, \neg W_{2,2}$, $R_{3,4}$: $S_{1,2}, \neg W_{1,1}$, (vier 2-Klauseln)

R_4 : $\neg S_{1,2}, W_{1,3}, W_{1,2}, W_{2,2}, W_{1,1}$ Übung (eine 5-Klausel)

. . .

- Negierte Zielformel für widerspruchsvollständige Grundresolution:

Z: $\neg W_{1,3}$

Resolutionsbeweis für die Wumpus-Welt

Ein Beweis durch Resolution über acht abgeleitete Klausen A_i $(i \in \{1,..,8\})$:

$Z: \neg W_{1,3}$ — $R_4: \neg S_{1,2} \vee W_{1,3} \vee W_{1,2} \vee W_{2,2} \vee W_{1,1}$

A_1 aus $R(Z,R_4): \neg S_{1,2} \vee W_{1,2} \vee W_{2,2} \vee W_{1,1}$

$F_5: S_{1,2}$ — $A_1: \neg S_{1,2} \vee W_{1,2} \vee W_{2,2} \vee W_{1,1}$

A_2 aus $R(F_5,A_1): W_{1,2} \vee W_{2,2} \vee W_{1,1}$

$F_1: \neg S_{1,1}$ — $R_{1,1}: S_{1,1} \vee \neg W_{1,1}$

A_3 aus $R(F_1,R_{1,1}): \neg W_{1,1}$

$A_3: \neg W_{1,1}$ — $A_2: W_{1,2} \vee W_{2,2} \vee W_{1,1}$

A_4 aus $R(A_3,A_2): W_{1,2} \vee W_{2,2}$

$F_3: \neg S_{2,1}$ — $R_{2,3}: S_{2,1} \vee \neg W_{2,2}$

A_5 aus $R(F_3,R_{2,3}): \neg W_{2,2}$

$A_5: \neg W_{2,2}$ — $A_4: W_{1,2} \vee W_{2,2}$

A_6 aus $R(A_5,A_4): W_{1,2}$

$F_1: \neg S_{1,1}$ — $R_{1,2}: S_{1,1} \vee \neg W_{1,2}$

A_7 aus $R(F_1,R_{1,2}): \neg W_{1,2}$

$A_6: W_{1,2}$ — $A_7: \neg W_{1,2}$

$A_8 = \square$ aus $R(A_6,A_7)$

Übung

Von Wissen zu Aktionen

Wir können jetzt neue Fakten inferieren,

aber wie setzen wir das Wissen in Aktionen um?

1) Negative Selektion: Schließe alle beweisbar gefährlichen Aktionen aus

$$A_{1,1} \wedge East_A \wedge W_{2,1} \Rightarrow \neg Forward.$$

2) Positive Selektion: Schlage nur Aktionen vor, die beweisbar sicher sind

$$A_{1,1} \wedge East_A \wedge \neg W_{2,1} \Rightarrow Forward.$$

Aus den Vorschlägen muss der Agent sich noch eine Aktion „aussuchen"!

---

* $East_A$ bedeutet, dass der Agent nach Osten orientiert ist (im folg. Algm. Auch als „*right*" bezeichnet).

Der aussagenlogische Wumpus–Agent (1)

function PL-Wumpus-Agent (percept) **returns** an action

PL hier für propositional logic

inputs: percept, a list of [*stench*, *breeze*, *glitter*, *bump*, *scream*]

static: *KB*, a knowledge base, initially containing the "physics" of the wumpus world,

x, *y*, *orientation*, the agent's position (initially [1,1]) and orientation (initially *right*),

visited, an array indicating which squares have been visited, initially *false* for all squares,

action, the agent's most recent action, initially *null*,

plan, an action sequence, initially *empty*

update *x*, *y*, *orientation*, *visited* based on *action*

Aktualisierungen durch Sensorik und Aktionsausführung

if *stench* **then** TELL(KB, $S_{x,y}$) **else** TELL(KB,$\neg S_{x,y}$)

if *breeze* **then** TELL(KB, $B_{x,y}$) **else** TELL(KB,$\neg B_{x,y}$)

if *glitter* **then** *action* $\leftarrow$ *grab*

fringe square = unbesuchtes Feld benachbart zu besuchtem Feld

else if *plan* is *nonempty* **then** *action* $\leftarrow$ POP(*plan*)

else if for some fringe square [i,j], ASK(KB, $\neg P_{i,j} \wedge \neg W_{i,j}$) is entailed by KB **or** // provably save

for some fringe square [i,j], ASK(KB, $P_{i,j} \vee W_{i,j}$) is not entailed by KB **then do** // poss. save

plan $\leftarrow$ A*-GRAPH-SEARCH(ROUTE-PROBLEM([*x*,*y*], *orientation*, [*i*,*j*], *visited*)) *

action $\leftarrow$ POP(*plan*)

else *action* $\leftarrow$ a randomly chosen move

return *action*

* ROUTE-PROBLEM erzeugt ein Suchproblem, dessen Lösung eine Aktionsfolge ist, die von [x,y] zu [i,j] führt und nur über besuchte Felder verläuft.

Der aussagenlogische Wumpus–Agent (2)

Das aussagenlogische Wumpus–Agentenprogramm PL-Wumpus-Agent

- aktualisiert seine Wissensbasis über TELL, ✓
  - ob im aktuell besuchten Feld Geruch oder Luftzug wahrnehmbar sind ✓
  - welche Felder schon besucht wurden ✓
- wählt die nächste Aktion aus: ✓
  - greift das Gold, wenn dieses im aktuellen Feld ist, ✓
  - führt sonst den nächsten Schritt des Planes durch, ✓
  - sucht sonst ein nächstes unbesuchtes Feld (*fringe square*) und generiert einen Plan, um vom aktuellen Feld zu diesem unbesuchten Feld zu kommen ✓
  - führt sonst eine zufällige Bewegungsaktion aus. ✓

Probleme mit der Aussagenlogik (1)

Obwohl Aussagenlogik für die Darstellung der WUMPUS-Welt ausreichend scheint, ist sie doch ziemlich umständlich.

1. *Allgemeine Regeln* müssen für *jedes einzelne Feld* aufgestellt werden ↯

$$R_1 : \neg S_{1,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,1}$$

$$R_2 : \neg S_{2,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{2,1} \wedge \neg W_{2,2} \wedge \neg W_{3,1}$$

$$R_3 : \neg S_{1,2} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,2} \wedge \neg W_{1,3}$$

$$\ldots$$

2. Tatsächlich müssten wir i. A. alle atomaren Aussagen *mit einem Zeitindex* versehen, der die zeitliche Gültigkeit der Aussage in der Zeit beschreibt ⇝ weitere Aufblähung der Regelmenge. ↯

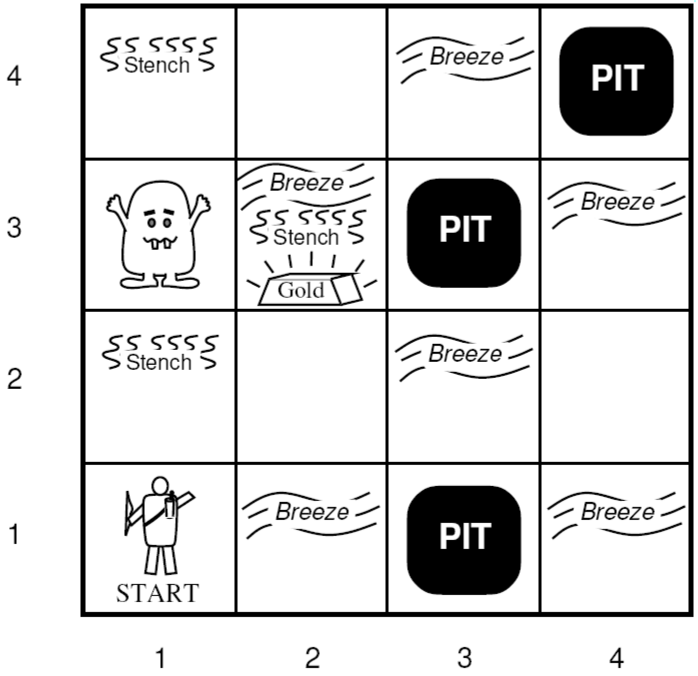
Probleme mit der Aussagenlogik (2)

- Letztlich gibt uns die Aussagenlogik keine sprachlichen Mittel an die Hand, mit denen die Zusammenhänge der repräsentierten Welt differenzierter ausdrücken können. ↯

↝ Eine *mächtigere Logik wäre wünschenswert*, in der wir *Objekte* und *Objektvariable* sowie deren *Eigenschaften* und *Relationen* beschreiben können!

↝ Ein Ausweg könnte die nächst höhere Logik sein, die *Prädikatenlogik 1. Stufe* (*first-order predicate logic*).

↝ Nächste Vorlesung!



Zusammenfassung

- Rationale Agenten benötigen *Wissen* über ihre Welt, um rationale Entscheidungen zu treffen.
- Dieses Wissen wird in einer *deklarativen* (Wissensrepräsentations-) Sprache dargestellt und in einer *Wissensbasis* gespeichert.
- Wir benutzen dafür (zunächst) die *Aussagenlogik*.
- Aussagenlogische Formeln können *allgemeingültig, erfüllbar* oder *unerfüllbar* sein.
- Wichtig ist der Begriff der *logischen Folgerbarkeit*.
- Inhaltliches Schlussfolgern kann durch einen *Kalkül* mechanisiert werden $\rightarrow$ Resolution.
- Aussagenlogik wird selbst für kleine Weltausschnitte sehr schnell unhandlich.

Anhang: (Grund-) Resolutions-Theorem (1)

(Grund-)Resolutions-Theorem: Δ ist inkonsistent gdw. $\Delta \vdash \square$.

Beweisidee:

- $\Delta \vdash \square$ impliziert Δ *ist inkonsistent*: wegen Korrektheit der Grundresolvente!
- Δ *ist inkonsistent* impliziert $\Delta \vdash \square$:
  - Sei $V = v_1, \ldots, v_m$ die (Atom-)Menge aller Variablen der Klauselnmenge Δ.
  - Ein vollständiger semantischer Baum B für Δ ist ein Binärbaum der Tiefe m, dessen Kanten der Ebene k vom Vorgängerknoten ausgehend mit den Literalen v_k bzw. $\neg v_k$ markiert sind.

  $\rightarrow$ Ein Pfad von der Wurzel bis zur Ebene k entspricht somit einer i.A. partiellen Interpretation $\mathcal{I}$ der Variablen $v_1, \ldots, v_k$ mit $v_i = T$, wenn v_i im Pfad bzw. $v_i = F$, wenn $\neg v_i$ im Pfad.

  - Die Knoten von B sind unmarkiert oder stehen für Klauseln aus Δ .
  - $\mathcal{I}(n)$ bezeichne die Interpretation, die dem Pfad von der Wurzel bis zum Knoten n entspricht.
  - Ein Knoten n heißt *Fehlerknoten*, wenn n der erste Knoten im Pfad ist, so dass $\mathcal{I}(n)$ eine Klausel C aus falsifiziert. n ist dann mit C markiert.

Anhang: (Grund-) Resolutions-Theorem (2)

- Ein semantischer Baum heißt *abgeschlossen*, wenn jeder Pfad mit einem Fehlerknoten endet.
- Ein Knoten n heißt *Inferenzknoten*, wenn *beide* Nachfolgeknoten Fehlerknoten sind.
- Beh.: Δ ist inkonsistent gdw. zu B ein abgeschlossener Teilbaum B' existiert.
- Bew. durch Induktion über Knotenzahl z des semantischen Baums B:
  - Induktionsanfang: z = 1: B hat nur Wurzelknoten, also $\square \in \Delta$.
  - Induktionsschritt: z > 1:
    - Es existiert mind. ein Inferenzknoten, sonst hätte jeder Knoten mind. einen Nichtfehlerknoten als Nachfolger und damit gäbe es mind. eine erfüllende Interpretation I von Δ
    - Bilde Resolvente aus den durch die beiden Fehlerknoten falsifizierten Clausen.
    - Entferne Nachfolger des Inferenzknotens aus B und mache Inferenzknoten zu neuem Fehlerknoten, da dieser nun die neue Resolvente falsifiziert.
    - Führe diesen Prozess weiter bis zur Reduktion von B zum Wurzelknoten!

Anhang: Beispiel für semantischen Baum

Δ umfasse 4 Klauseln:

C1:	+P	
C2:	+Q	+R
C3:	-P	-Q
C4:	-P	-R