AGENTES LÓGICOS CAP 7

Parcialmente adaptado de

http://aima.eecs.berkeley.edu

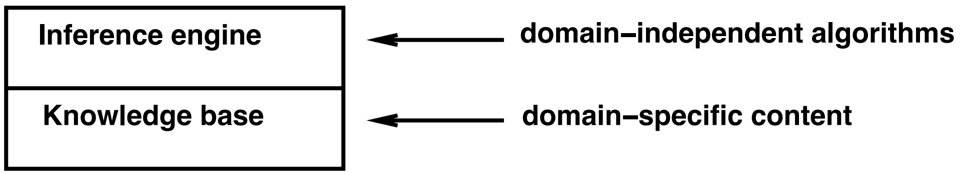
Resumo

- Agentes baseados em conhecimento
- O mundo Wumpus
- Lógica em geral – modelos e consequência
- Lógica Proposicional (Booleana)
- Equivalência, validade, satisfatibilidade
- Regras de Inferência e demonstração de teoremas
  - encadeamento para a frente (forward chaining)
  - encadeamento para trás (backward chaining)
  - resolução

Agentes Lógicos

- Agentes reactivos encontram o caminho de Arad para Bucareste por sorte.
- Programas jogadores de xadrez calculam as jogadas legais para o rei, mas não sabem que uma peça não pode estar em duas casas ao mesmo tempo.
- Agentes baseados em lógica combinam conhecimento geral com as percepções correntes para inferir aspectos escondidos do estado actual, antes de seleccionarem as acções.
  - Crucial em ambientes parcialmente observáveis.

Bases de Conhecimento



- Base de conhecimento = conjunto de frases numa linguagem formal
- Aproximação declarativa na construção de um agente (ou outro sistema):

  TELL ⇐ informar o sistema do que precisa de saber
- Seguidamente, o sistema pode perguntar a si próprio (ASK) o que deve fazer – respostas obtidas (implicitamente) a partir da KB
- Os agentes podem ser analisados quanto ao seu nível de conhecimento
  - i.e., aquilo que sabem, independentemente da sua implementação
- Ou quanto ao seu nível de implementação
  - i.e., estruturas de dados na KB e algoritmos que as manipulam

Um agente simples baseado em conhecimento

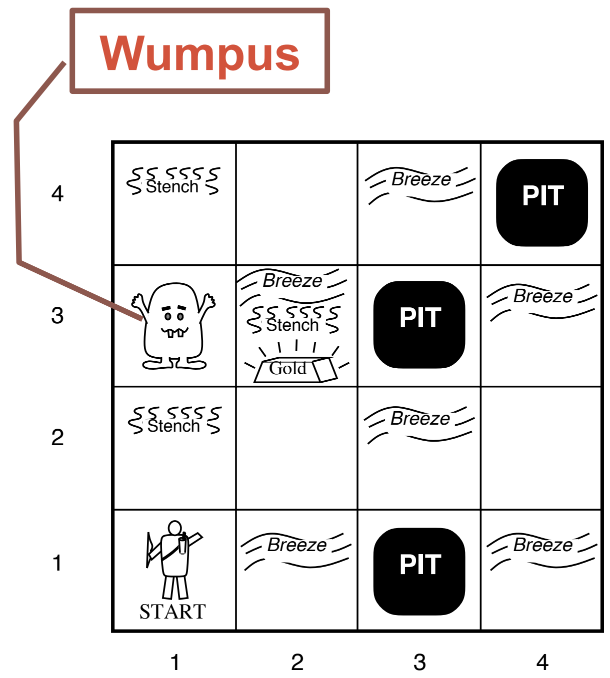
```
function KB-AGENT(percept) returns an action
    static: KB, a knowledge base
            t, a counter, initially 0, indicating time

    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
```

- O agente deve ser capaz de:
  - Representar estados, acções, etc.
  - Incorporar novas percepções
  - Actualizar representações internas do mundo
  - Deduzir propriedades escondidas do mundo
  - Deduzir acções apropriadas

Descrição do mundo do Wumpus (PEAS)

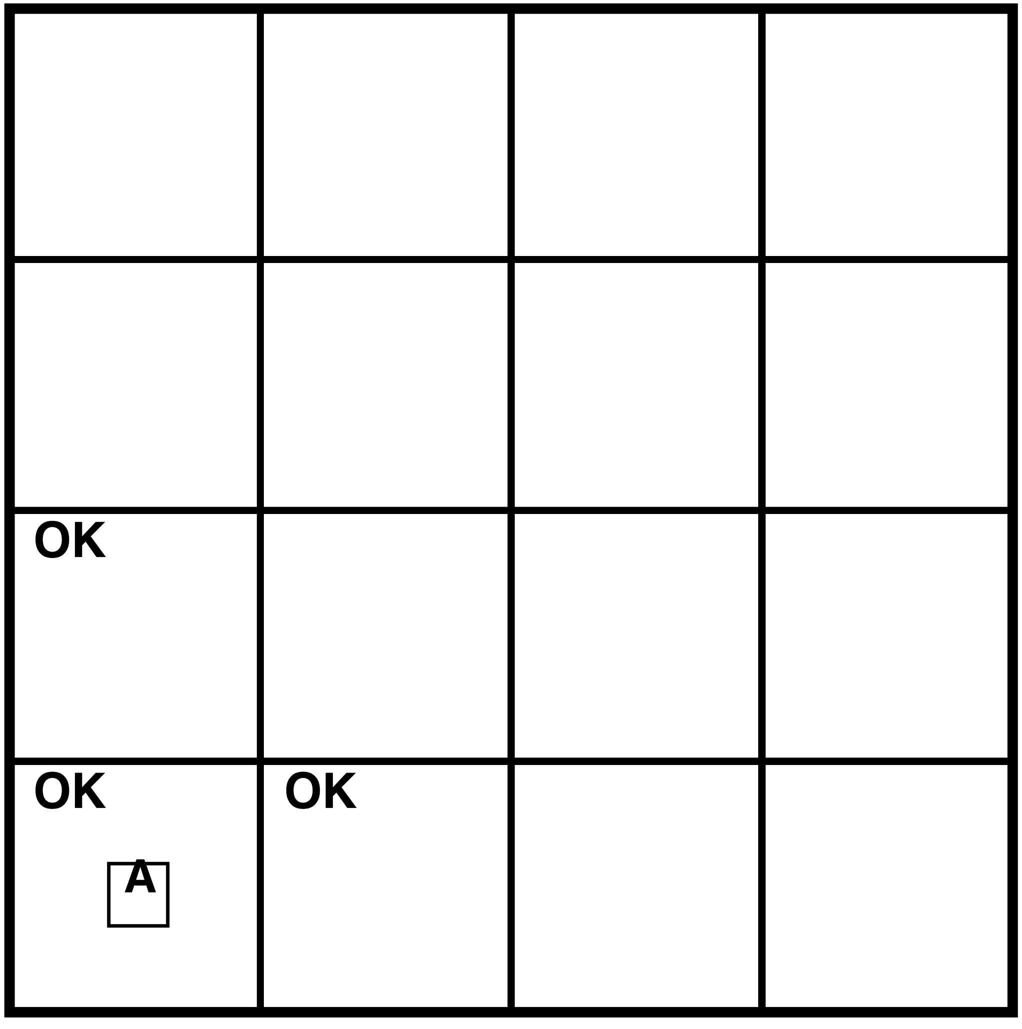
- Medida de desempenho
  - Sair com ouro +1000, morte -1000
  - -1 por passo, -10 por utilizar a seta
- Ambiente
  - Casas adjacentes ao wumpus são malcheirosas
  - Casas adjacenctes a um poço são ventosas
  - Brilho sse ouro está na mesma casa
  - Disparo mata wumpus se estiver defronte dele
  - Disparar gasta a única seta
  - Agarrar apanha o ouro da casa
  - Largar deixa o ouro na mesma casa
- Sensores
  - Brisa, Brilho, Cheiro, Grito, Batida
- Actuadores
  - Rodar Esquerda, Rodar Direita,
  - Avançar, Agarrar, Largar, Disparar, Sair



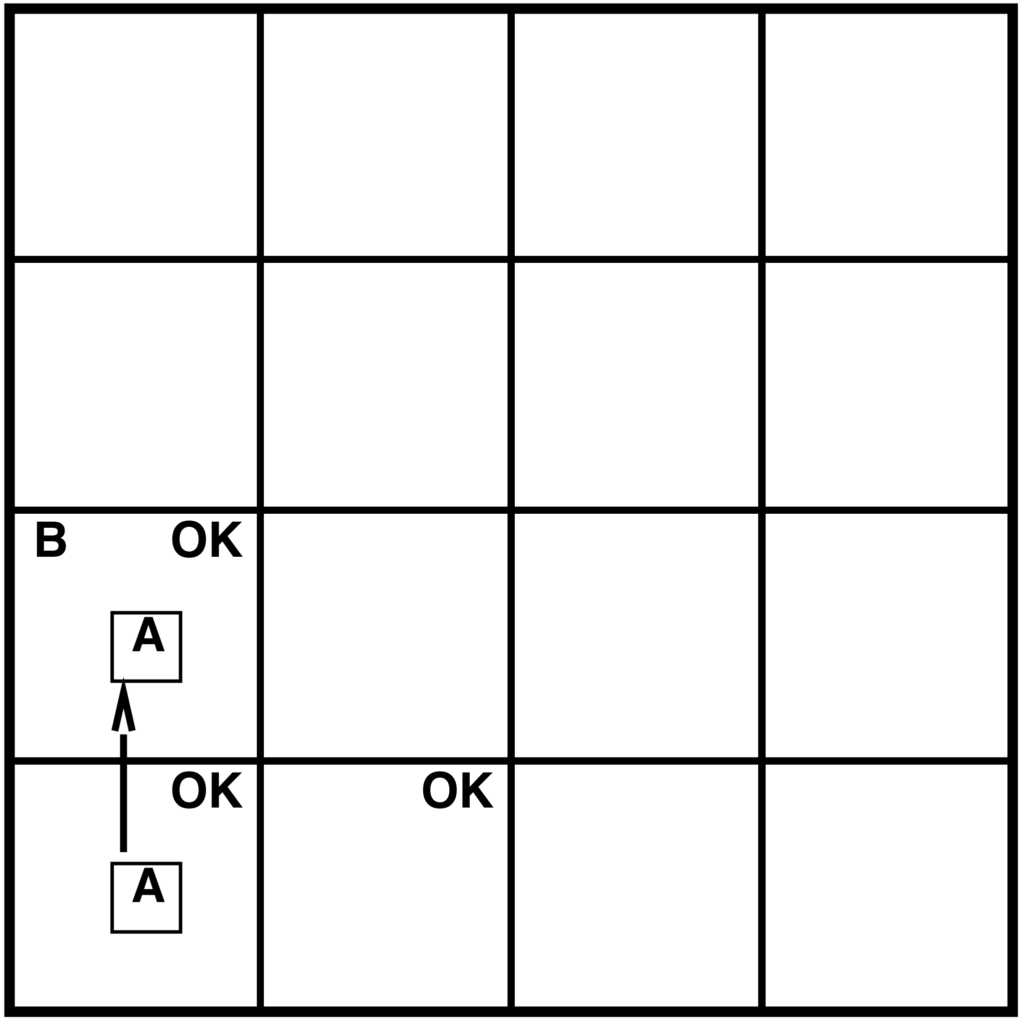
Caracterização do mundo do Wumpus

- Observável??
  - Não – apenas percepções locais
- Determinista??
  - Sim – os resultados estão especificados exactamente
- Episódico??
  - Não – sequencial ao nível das acções
- Estático??
  - Sim – Wumpus e poços não se movem
- Discreto??
  - Sim
- Agente único??
  - Sim – Wumpus é basicamente uma propriedade do ambiente

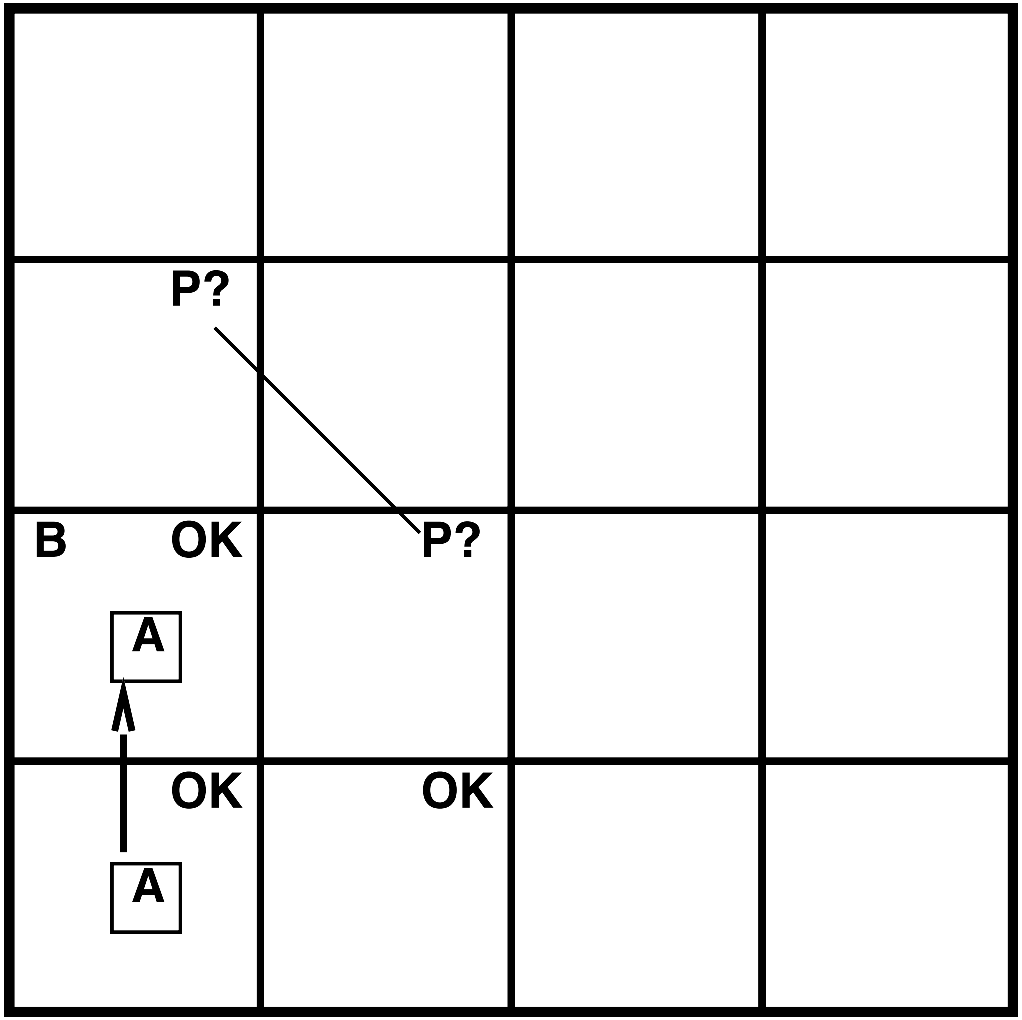
Explorando um mundo do Wumpus



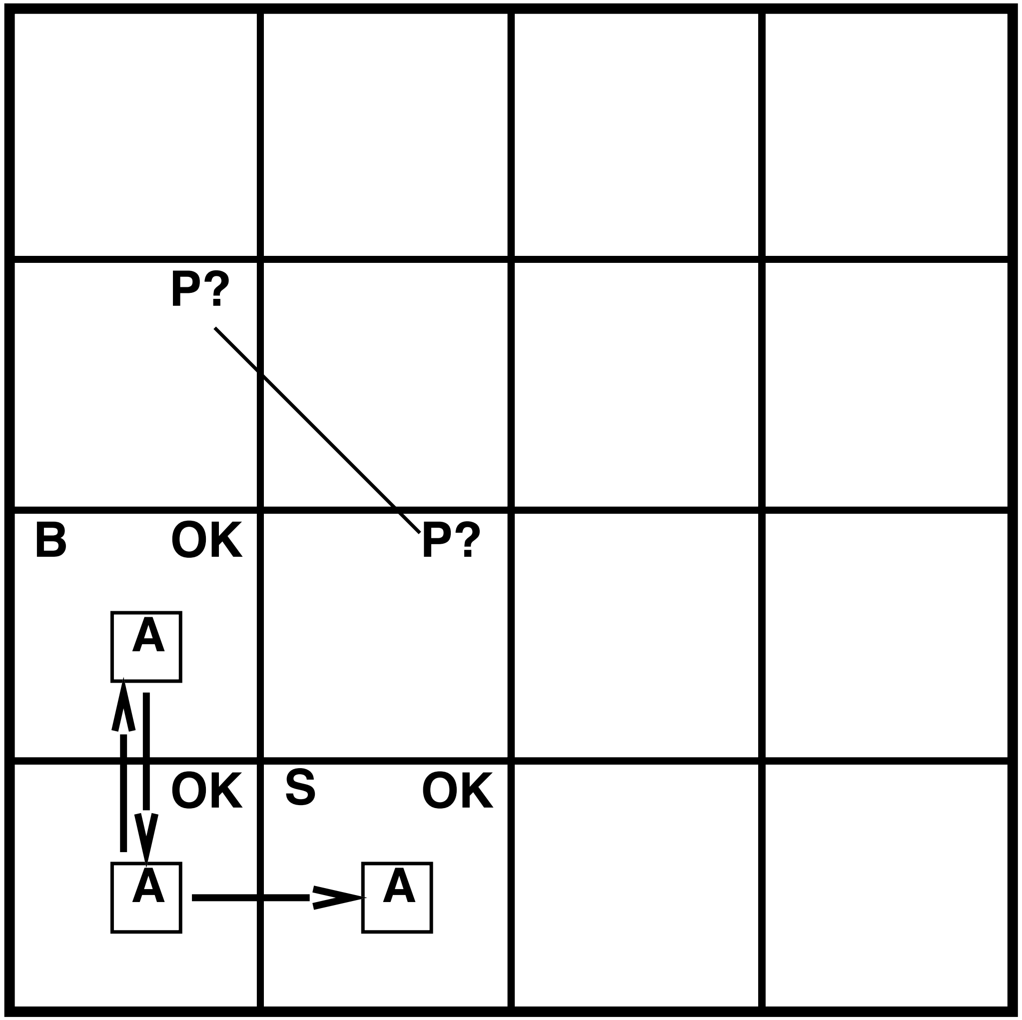
Explorando um mundo do Wumpus



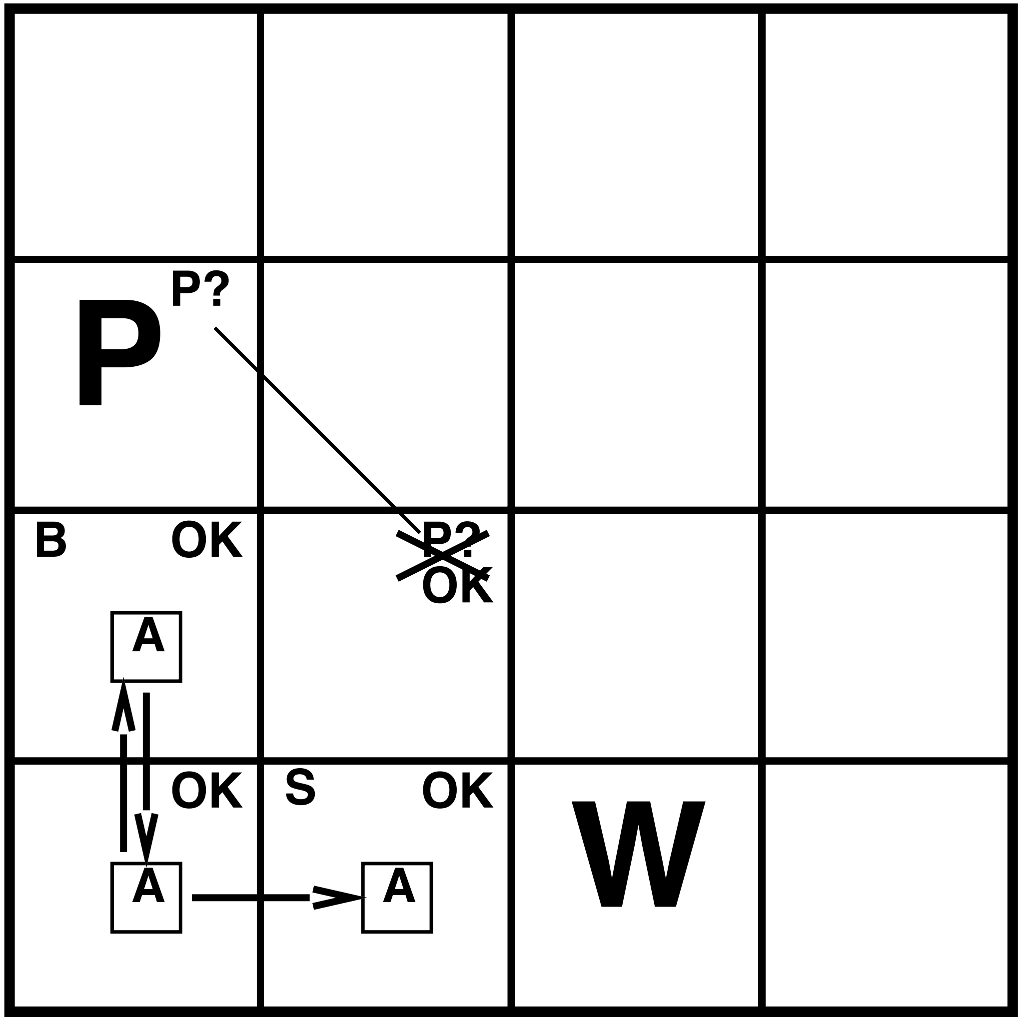
Explorando um mundo do Wumpus



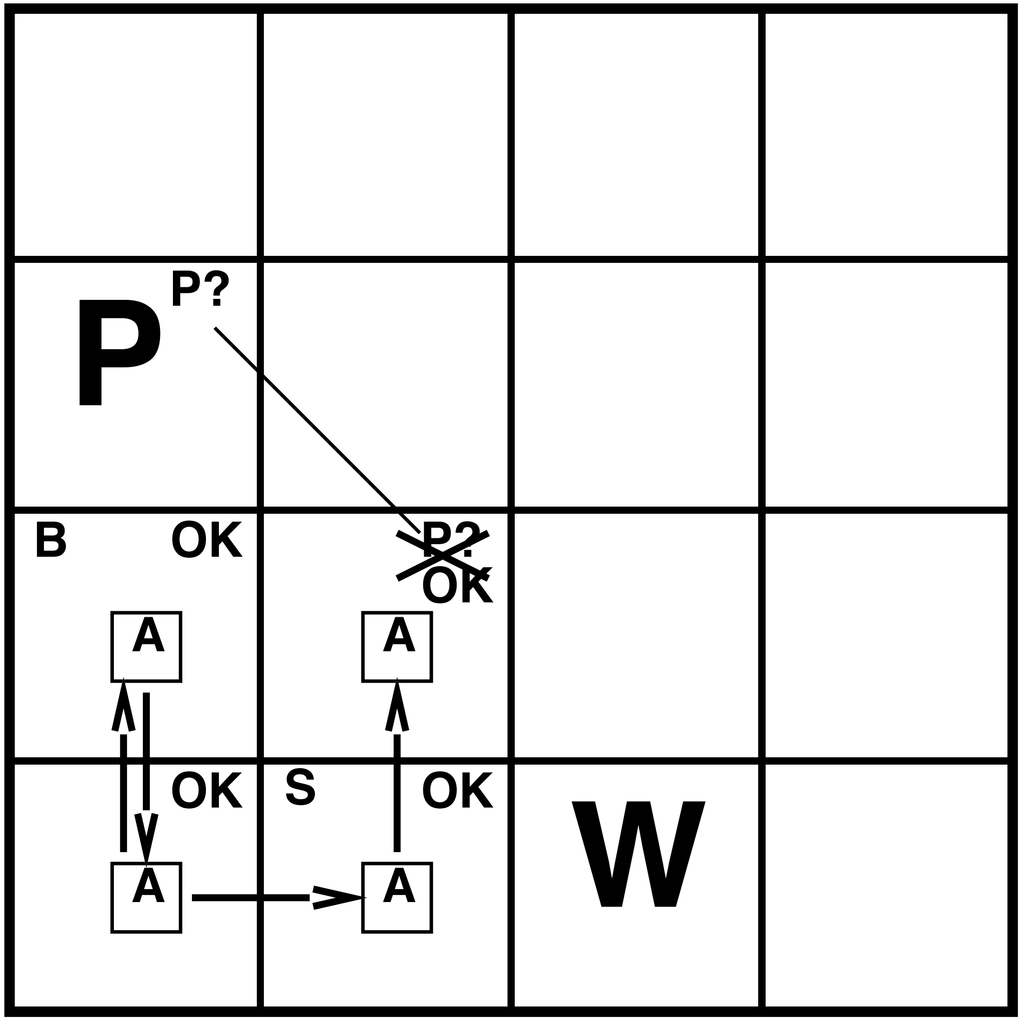
Explorando um mundo do Wumpus



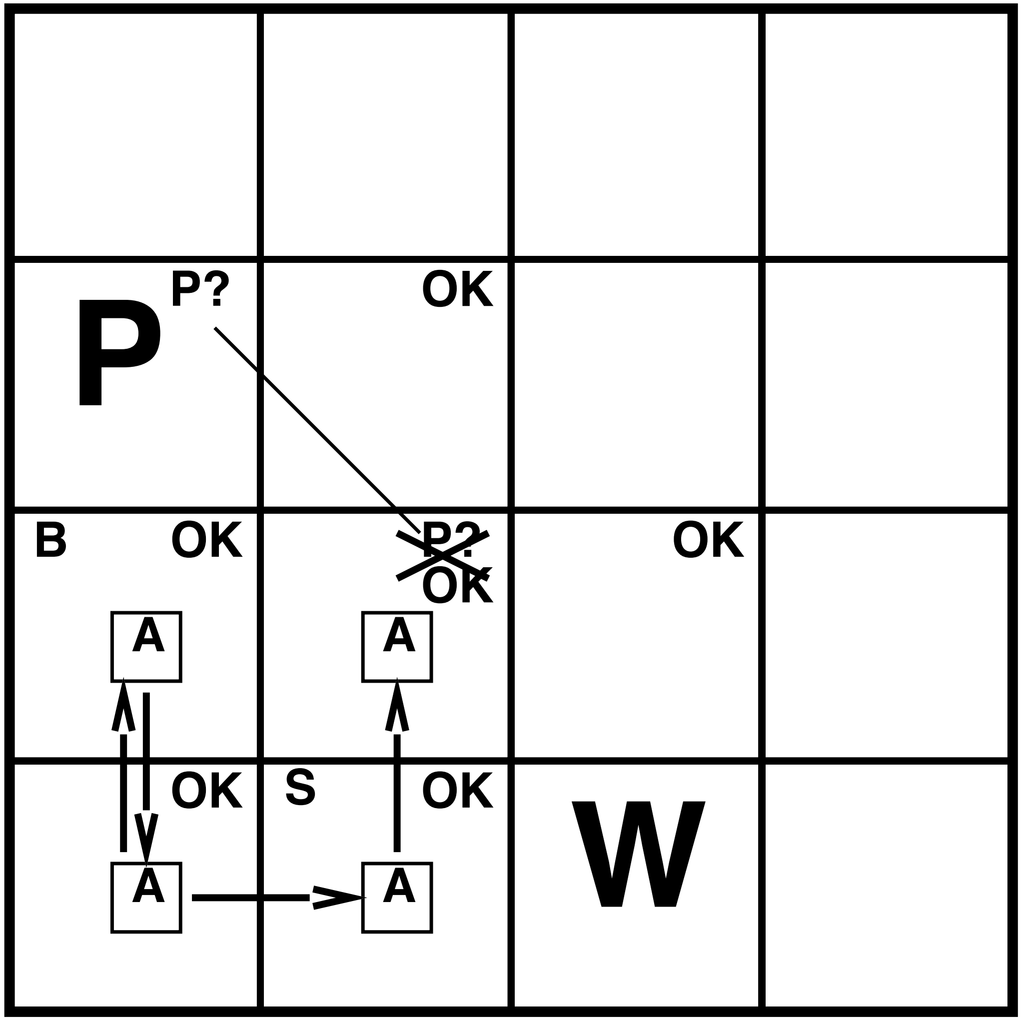
Explorando um mundo do Wumpus



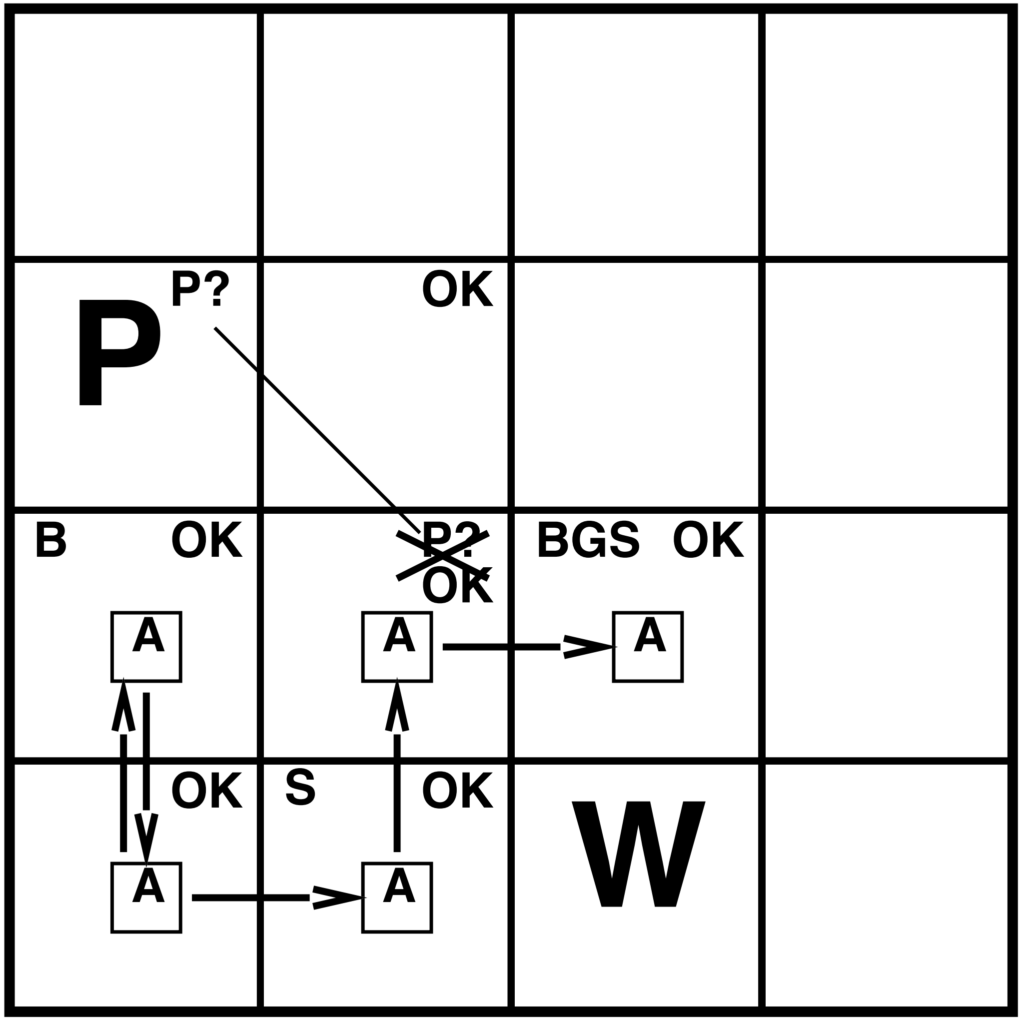
Explorando um mundo do Wumpus



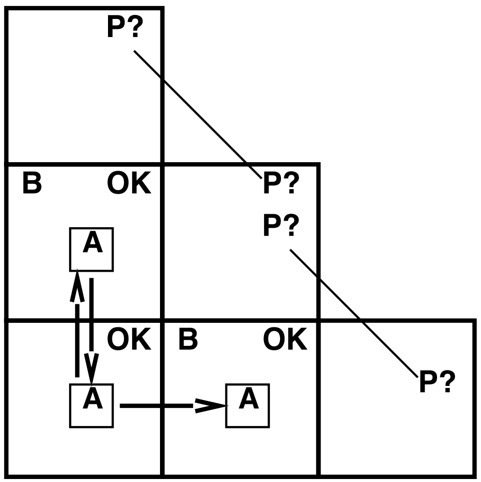
Explorando um mundo do Wumpus



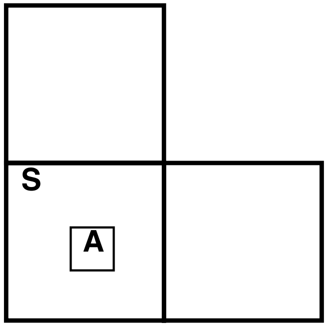
Explorando um mundo do Wumpus



Outras situações difíceis



- Vento em (1,2) e (2,1)
  - não existem acções seguras
- Assumindo poços uniformemente distribuídos, (2,2) tem poço c/ prob 0.86, vs. 0.31



- Cheiro em (1,1)
  - não se pode mover
- Pode recorrer a estratégia de coerção:
  - disparar em frente
  - wumpus estava lá ⇒ morto ⇒ seguro
  - wumpus não estava lá ⇒ seguro

Noções de Lógica

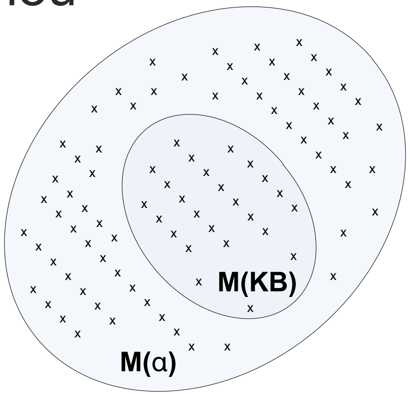
- Lógicas são linguagens formais para de representação de informação que permitem a extração de conclusões
- Sintaxe define as frases permitidas da linguagem
  - E.g., na linguagem da aritmética
    - $x + 2 \geq y$ é uma frase (proposição); $x2 + y >$ não é uma frase
- Semântica define o significado das frases;
  - i.e., define verdade de uma frase em cada um dos mundos possíveis
  - E.g., na linguagem da aritmética
    - $x + 2 \geq y$ é verdade sse o número $x + 2$ não for menor do que o número y
    - $x + 2 \geq y$ é verdade num mundo em que $x=7$, $y =1$
    - $x + 2 \geq y$ é falso num mundo em que $x=0$, $y =6$

Conclusão Lógica

- Conclusão (ou consequência) significa que algo segue de outrem:
- KB $\models$ α
- Da base de conhecimento KB conclui-se a frase α:
  - KB $\models$ α se e só se α é verdade em todos os mundos em que KB é verdade.
- Da base de conhecimento KB contendo "a Académica ganhou" e "o Belenenses ganhou" conclui-se, por exemplo, "a Académica ganhou ou o Belenenses ganhou".
- E.g., de $x + y = 4$ conclui-se $4 = y + x$
- Conclusão Lógica é uma relação entre frases (i.e., sintaxe) que se encontra baseada na semântica.

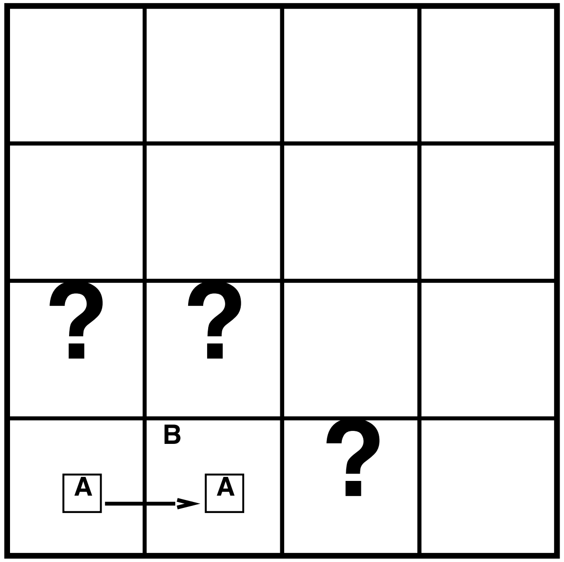
Modelos

- No contexto da lógica, normalmente pensa-se em termos de modelos, que são mundos formalmente estruturadas relativamente aos quais se pode avaliar a veracidade
- Diz-se que m é modelo de uma proposição α se α é verdade em m
- M(α) é o conjunto de todos os modelos de α
- Logo KB $\models$ α se e só se M(KB)$\subseteq$M(α)
  - E.g. KB = Académica ganhou e Belenenses ganhou
  - α = Académica ganhou

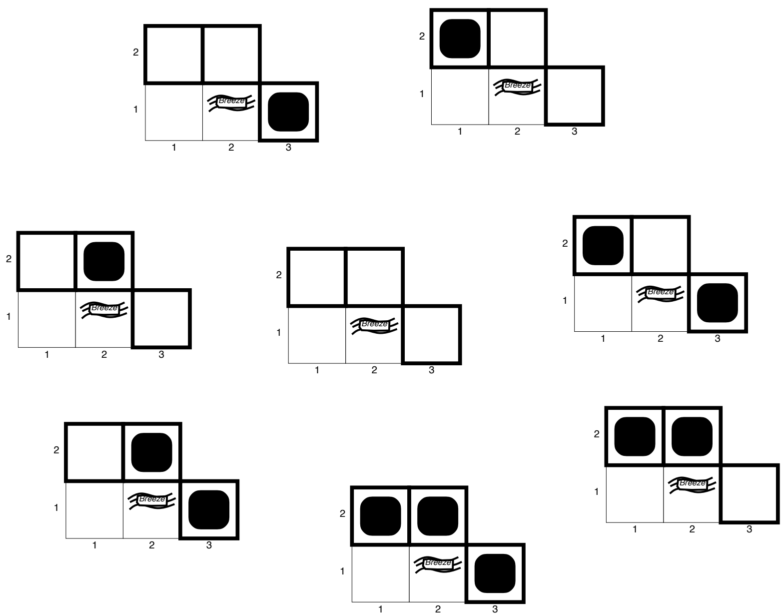


Conclusões no mundo do Wumpus

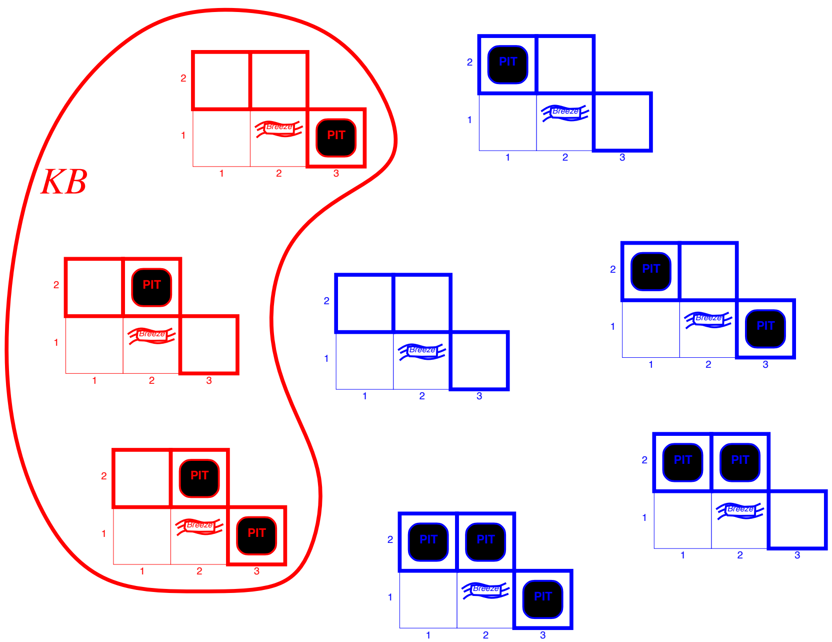
- Situação após
  - detectar nada em [1,1],
  - deslocação para a direita,
  - brisa em [2,1]
- Considerar modelos possíveis para "?" (assumindo apenas poços)
- 3 escolhas Booleanas ⇒ 8 mundos possíveis



Modelos Wumpus

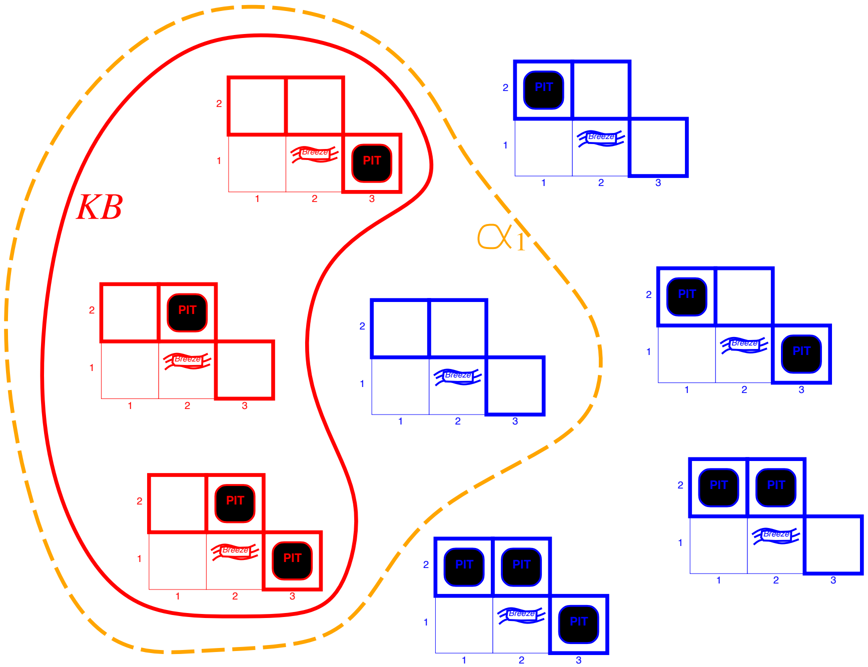


Modelos Wumpus



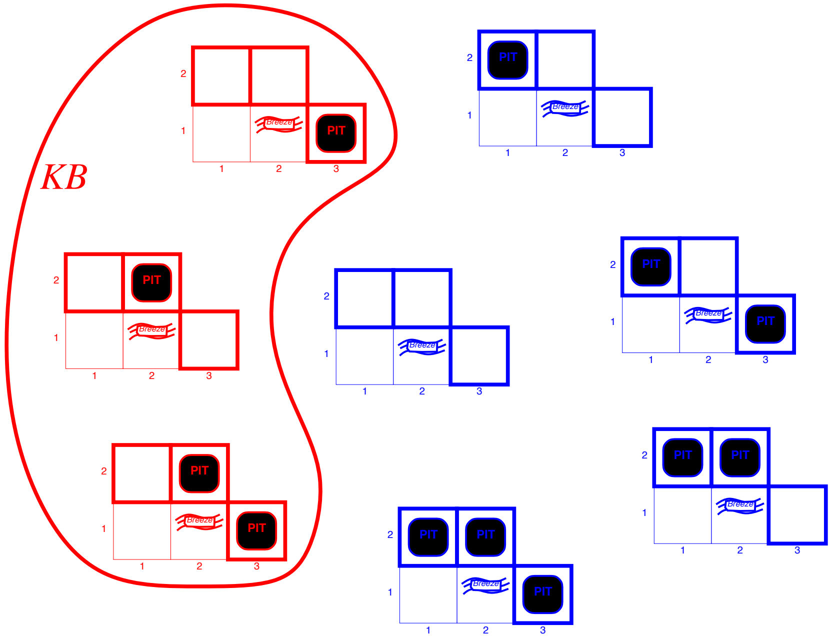
- KB = regras do mundo-wumpus + observações

Modelos Wumpus



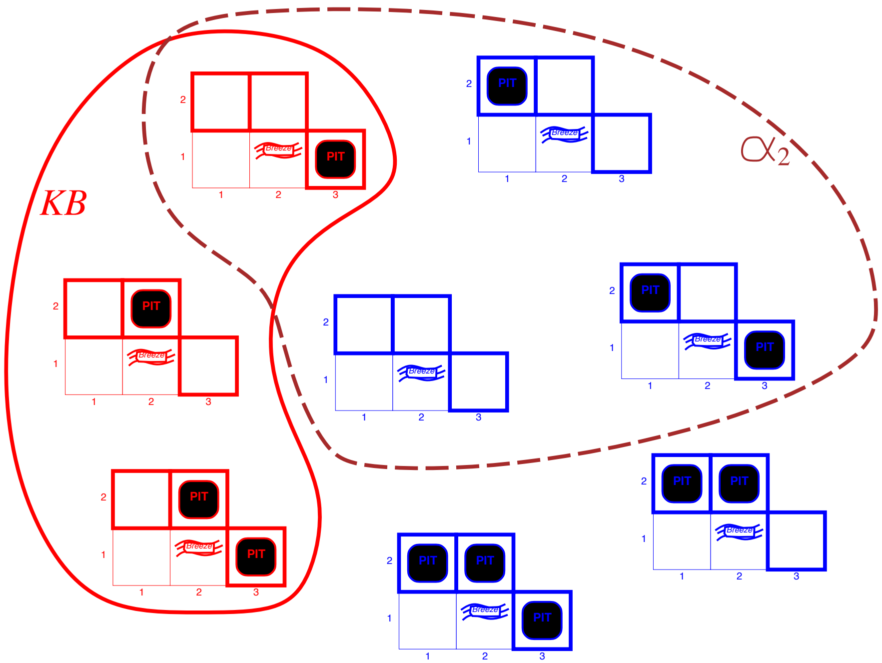
- KB = regras do mundo-wumpus + observações
- α_1 = “[1,2] é seguro”, KB ⊨ α_1, demonstrado por verificação de modelos

Modelos Wumpus



- KB = regras do mundo-wumpus + observações

Modelos Wumpus

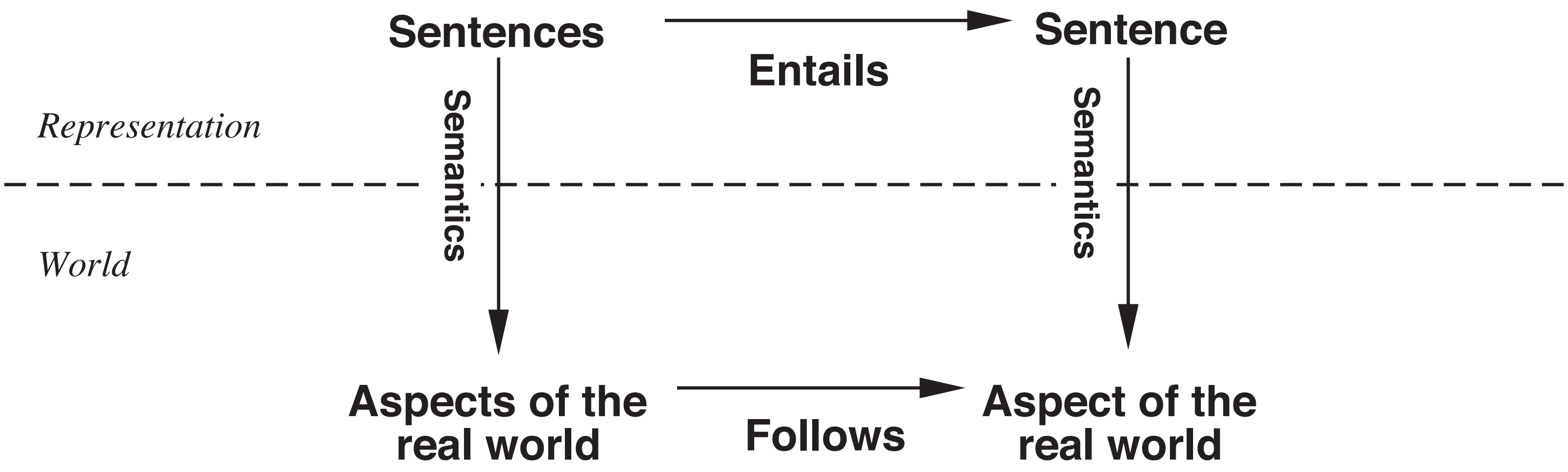


- KB = regras do mundo-wumpus + observações
- α_2 = “[2,2] é seguro”, KB $\nvDash$ α_2,

Inferência

- KB $\vdash_i$ α = proposição pode ser derivada de KB pelo procedimento de inferência i
- Consequências de KB são o palheiro; α é a agulha.
- Conclusão Lógica = agulha no palheiro; inferência = encontrá-la.
- Fidedigno: i é fidedigno (ou sólido) se
  - quando KB $\vdash_i$ α, então também é verdade que KB $\vDash$ α.
- Completo: i é completo se
  - quando KB $\vDash$ α, então também é verdade que KB $\vdash_i$ α.
- Procedimento de inferência fidedigno e completo:
  - responderá a qualquer questão que segue daquilo que é conhecido pela KB.

Perspectiva esquemática



- Se a KB é verdade no mundo real, então qualquer proposição α derivada de KB por um processo de inferência fidedigno (sólido) também é verdadeira no mundo real.

Lógica Proposicional: Sintaxe

- A lógica proposicional é a lógica mais simples
  - ilustra os conceitos básicos
- Os símbolos (ou variáveis) proposicionais P_1, P_2 etc são proposições (frases)
  - Se S é uma proposição, $\neg S$ é uma proposição (negação)
  - Se S_1 e S_2 são proposições, $(S_1 \land S_2)$ é uma proposição (conjunção)
  - Se S_1 e S_2 são proposições, $(S_1 \lor S_2)$ é uma proposição (disjunção)
  - Se S_1 e S_2 são proposições, $(S_1 \Rightarrow S_2)$ é uma proposição (implicação)
  - Se S_1 e S_2 são proposições, $(S_1 \Leftrightarrow S_2)$ é uma proposição (bicondicional)

Lógica Proposicional: Semântica

- Cada modelo atribui um valor de verdade verdadeiro/falso a cada símbolo proposicional.
  - E.g. $P_{1,2}$ verdadeiro $P_{2,2}$ verdadeiro $P_{3,1}$ falso
  - Com estes símbolos, 8 modelos possíveis podem ser enumerados automaticamente.
- Regras para avaliar, recursivamente, a veracidade de qualquer frase relativamente a um modelo m:
  - $\neg S$ é verdade sse S é falso
  - $S_1 \wedge S_2$ é verdade sse S_1 é verdade e S_2 é verdade
  - $S_1 \vee S_2$ é verdade sse S_1 é verdade ou S_2 é verdade
  - $S_1 \Rightarrow S_2$ é verdade sse S_1 é falso ou S_2 é verdade
    - i.e., é falso sse S1 é verdade e S_2 é falso
  - $S_1 \Leftrightarrow S_2$ é verdade sse $S_1 \Rightarrow S_2$ é verdade e $S_2 \Rightarrow S_1$ é verdade
- Um processo recursivo simples avalia uma proposição arbitrária, e.g.,

$$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = \neg verd \wedge (verd \vee falso) = falso \wedge verd = falso$$

Tabela de verdade para os conectivos

P	Q	¬P	P∧Q	P∨Q	P⇒Q	P⇔Q
falso	falso	verdade	falso	falso	verdade	verdade
falso	verdade	verdade	falso	verdade	verdade	falso
verdade	falso	falso	falso	verdade	falso	falso
verdade	verdade	falso	verdade	verdade	verdade	verdade

Proposições no mundo do Wumpus

- Seja $P_{i,j}$ verdade se existir um poço em $[i, j]$.
- Seja $B_{i,j}$ verdade se existir uma brisa em $[i, j]$.

$$\neg P_{1,1}$$

(não existe um poço em 1,1)

$$\neg B_{1,1}$$

(não existe brisa em 1,1)

$$B_{2,1}$$

(existe uma brisa em 2,1)

- "Poços causam brisa em casas adjacentes".

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

…

- "Uma casa é ventosa sse existir um poço adjacente"

Inferência através de tabelas de verdade

- Enumerar todos os modelos e verificar se α é verdade em cada modelo em que KB é verdade (ex. α_1="$\neg P_{1,2}$"):

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
falso	falso	falso	falso	falso	falso	falso	falso	verdade
falso	falso	falso	falso	falso	falso	verdade	falso	verdade
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
falso	verdade	falso	falso	falso	falso	falso	falso	verdade
falso	verdade	falso	falso	falso	falso	verdade	verdade	verdade
falso	verdade	falso	falso	falso	verdade	falso	verdade	verdade
falso	verdade	falso	falso	falso	verdade	verdade	verdade	verdade
falso	verdade	falso	falso	verdade	falso	falso	falso	verdade
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
verdade	verdade	verdade	verdade	verdade	verdade	verdade	falso	falso

Inferência por enumeração

- Enumeração em profundidade primeiro dos modelos todos é solido e completo

function TT-ENTAILS?(KB, α) **returns** *true* or *false*

  symbols ← a list of the proposition symbols in KB and α
  return TT-CHECK-ALL(KB, α, *symbols*, [ ])

---

function TT-CHECK-ALL(KB, α, *symbols*, *model*) **returns** *true* or *false*
  if EMPTY?(*symbols*) **then**
    if PL-TRUE?(KB, *model*) **then return** PL-TRUE?(α, *model*)
    else return *true*
  else do
    P ← FIRST(*symbols*); *rest* ← REST(*symbols*)
    return TT-CHECK-ALL(KB, α, *rest*, EXTEND(P, *true*, *model*) **and**
        TT-CHECK-ALL(KB, α, *rest*, EXTEND(P, *false*, *model*)

- Para n símbolos, complexidade temporal de O(2^n) e espacial O(n).

Equivalência Lógica

- Duas proposições são logicamente equivalentes sse forem verdadeiras nos
- mesmos modelos: $\alpha \equiv \beta$ *sse* $\alpha \vDash \beta$ *e* $\beta \vDash \alpha$

$$
\begin{aligned}
(\alpha \land \beta) &\equiv (\beta \land \alpha) \quad \text{comutatividade de } \land \\
(\alpha \lor \beta) &\equiv (\beta \lor \alpha) \quad \text{comutatividade de } \lor \\
((\alpha \land \beta) \land \gamma) &\equiv (\alpha \land (\beta \land \gamma)) \quad \text{associatividade de } \land \\
((\alpha \lor \beta) \lor \gamma) &\equiv (\alpha \lor (\beta \lor \gamma)) \quad \text{associatividade de } \lor \\
\neg(\neg\alpha) &\equiv \alpha \quad \text{eliminação da dupla negação} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposição} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \lor \beta) \quad \text{eliminação da implicação} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)) \quad \text{eliminação do bicondicional} \\
\neg(\alpha \land \beta) &\equiv (\neg\alpha \lor \neg\beta) \quad \text{de Morgan} \\
\neg(\alpha \lor \beta) &\equiv (\neg\alpha \land \neg\beta) \quad \text{de Morgan} \\
(\alpha \land (\beta \lor \gamma)) &\equiv ((\alpha \land \beta) \lor (\alpha \land \gamma)) \quad \text{distributividade de } \land \text{ sobre } \lor \\
(\alpha \lor (\beta \land \gamma)) &\equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma)) \quad \text{distributividade de } \lor \text{ sobre } \land
\end{aligned}
$$

Validade e satisfatibilidade

- Uma proposição é válida se for verdadeira em todos os modelos,
  - e.g., True, $A \lor \neg A$, $A \Rightarrow A$, $(A \land (A \Rightarrow B)) \Rightarrow B$
- Validade está relacionado com consequência através do Teorema da Dedução:

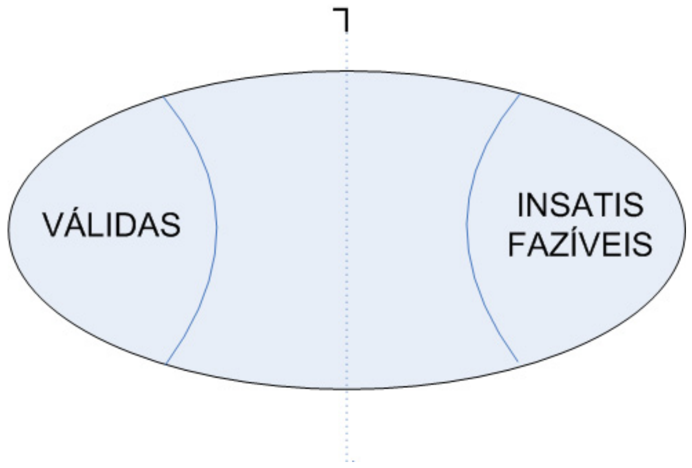
$$KB \vDash \alpha \text{ se e só se } (KB \Rightarrow \alpha) \text{ é válida}$$

- Uma proposição é satisfazível se é verdadeira em algum modelo
  - e.g., $A \lor B$, C
- Uma proposição é insatisfazível se for verdadeira em nenhum modelo
  - e.g., $A \land \neg A$
- Insatisfatibilidade relaciona-se com consequência através de:

$$KB \vDash \alpha \text{ se e só se } (KB \land \neg \alpha) \text{ é insatisfazível}$$

- i.e., demonstrar α por *reductio ad absurdum (i.e. por refutação ou contradição)*

Geografia das expressões booleanas



Eixo de simetria = negação

Regras de inferência em Lógica Proposicional

- Modus Ponens

$$\frac{\alpha, \qquad \alpha \Rightarrow \beta}{\beta}$$

- Eliminação de $\wedge$

$$\frac{\alpha \wedge \beta}{\alpha}$$

- Todas as equivalências do slide “Equivalência Lógica” podem ser usadas como regras de inferência:

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}$$

Exemplo

- Assumindo R_1 a R_5:

$$\neg P_{1,1}, \quad B_{1,1} \Leftrightarrow \left(P_{1,2} \vee P_{2,1}\right), \quad B_{2,1} \Leftrightarrow \left(P_{1,1} \vee P_{2,2} \vee P_{3,1}\right), \quad \neg B_{1,1}, \quad B_{2,1}$$

- Como provar $\neg P_{1,2}$?

$$R_6 : \left(B_{1,1} \Rightarrow \left(P_{1,2} \vee P_{2,1}\right)\right) \wedge \left(\left(P_{1,2} \vee P_{2,1}\right) \Rightarrow B_{1,1}\right)$$

$$R_7 : \left(P_{1,2} \vee P_{2,1}\right) \Rightarrow B_{1,1}$$

$$R_8 : \neg B_{1,1} \Rightarrow \neg\left(P_{1,2} \vee P_{2,1}\right)$$

$$R_9 : \neg\left(P_{1,2} \vee P_{2,1}\right)$$

$$R_{10} : \neg P_{1,2} \wedge \neg P_{2,1}$$

- Eliminação de bicondicional
- Eliminação de $\wedge$
- Contraposição
- Modens ponens
- De Morgan

Pesquisa de provas

- Encontrar provas é o mesmo que encontrar soluções para problemas de pesquisa
- A pesquisa pode ser feita para a frente (forward chaining) para derivar um golo (objectivo) ou para trás (backward chaining) a partir do golo
- Pesquisar provas não é mais eficiente do que enumerar os modelos, mas em muitos casos práticos é mais eficiente porque podemos ignorar propriedades irrelevantes.
- Monotonicidade: o conjunto de frases que se concluem só pode crescer à medida que mais informação é acrescentada à base de conhecimento.

para todo o α e β, se $KB \vDash \alpha$ então $KB \land \beta \vDash \alpha$.