

INF623

2024/1



Inteligência Artificial

A13: Representação do conhecimento III

Plano de aula

- ▶ Regras de inferência e prova de teoremas
 - ▶ Modus Ponens
 - ▶ Contraposição
 - ▶ Eliminação do E, Dupla negativa, implicação, bicondicional
 - ▶ De Morgan
 - ▶ Distributiva
- ▶ Forma normal conjuntiva
- ▶ Inferência por resolução

Regras de inferência

A complexidade do algoritmo de inferência por verificação de modelo é $O(2^n)$, portanto ele é prático apenas para um número pequeno de símbolos.

- ▶ Ao invés de enumerar e verificar todos modelos, podemos aplicar **regras de inferência** na base de conhecimento para concluir $BC \models \alpha$
 - ▶ Modus Ponens
 - ▶ Eliminação do E, Dupla negativa, implicação, bicondicional
 - ▶ De Morgan
 - ▶ Distributiva

Modus Ponens

Se sabemos que uma implicação e seu antecedente são verdadeiros, então o consequente também é verdadeiro

$\alpha \Rightarrow \beta$ "Se estiver chovendo, Lucas vai estudar"

α "Está chovendo"

Modus ponens

β "Lucas vai estudar"

Contraposição

Se sabemos que uma implicação é verdadeira, então a sua contrapositiva também é verdadeira.

α	β	$\alpha \Rightarrow \beta$
F	F	V
F	V	V
V	F	F
V	V	V

$\neg\beta$	$\neg\alpha$	$\neg\beta \Rightarrow \neg\alpha$
V	V	V
F	V	V
V	F	F
F	F	V

“Se estiver chovendo, Lucas vai estudar”

$$\alpha \Rightarrow \beta$$

Contraposição

$$\neg\beta \Rightarrow \neg\alpha$$

“Se não estiver chovendo, Lucas não vai estudar”

Eliminação do E

Se uma proposição E for verdadeira, então qualquer proposição atômica dentro dela também será verdadeira.

$\alpha \wedge \beta$ "Lucas é colega de Carlos e Marcos"

Eliminação do E

α "Lucas é colega de Carlos"

β "Lucas é colega de Marcos"

Eliminação de dupla negativa

Uma proposição que é negada duas vezes é verdadeira.

“Não é verdade que Lucas não passou no teste”

$$\neg(\neg\alpha)$$

Eliminação de dupla negativa

$$\alpha$$

“Lucas passou no teste”

Eliminação de implicação

Uma implicação é equivalente a uma relação OU entre o antecedente negado e o conseqüente

"Se estiver chovendo, então Lucas vai estudar"

α	β	$\alpha \Rightarrow \beta$
F	F	V
F	V	V
V	F	F
V	V	V

$$\frac{\alpha \Rightarrow \beta}{\neg \alpha \vee \beta} \quad \text{Eliminação de implicação}$$

"Não está chovendo" ou "Lucas vai estudar"

Eliminação de bicondicional

Uma proposição bicondicional é equivalente a uma implicação e seu inverso com um conectivo E.

“Está chovendo se e somente se Lucas vai estudar”

α	β	$\alpha \Leftrightarrow \beta$
F	F	V
F	V	F
V	F	F
V	V	V

$$\alpha \Leftrightarrow \beta$$

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)} \quad \text{Eliminação de bicondicional}$$

“Se estiver chovendo, então Lucas vai estudar”

“Se Lucas vai estudar, então está chovendo”

De Morgan

Transformar um conectivo \wedge (e) em um conectivo \vee (ou)

“Não é verdade que tanto Lucas quanto Carlos passaram no teste”

$$\frac{\neg(\alpha \wedge \beta)}{\neg\alpha \vee \neg\beta} \quad \text{De Morgan}$$

“Lucas não passou no teste ou Carlos não passou no teste”

De Morgan

Transformar um conectivo \vee (ou) em um conectivo \wedge (e)

“Não é verdade que Lucas ou Carlos passaram no teste”

$$\frac{\neg(\alpha \vee \beta)}{\neg\alpha \wedge \neg\beta} \quad \text{De Morgan}$$

“Lucas não passou no teste e Carlos não passou no teste”

Distributiva

Uma proposição com dois elementos agrupados com conectivos E ou OU pode ser distribuída ou dividida em unidades menores consistindo de E e OU.

	$(\alpha \wedge (\beta \vee \gamma))$	$(\alpha \vee (\beta \wedge \gamma))$	
Distributiva	<hr/>	<hr/>	Distributiva
	$(\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$	$(\alpha \vee \beta) \wedge (\alpha \vee \gamma)$	

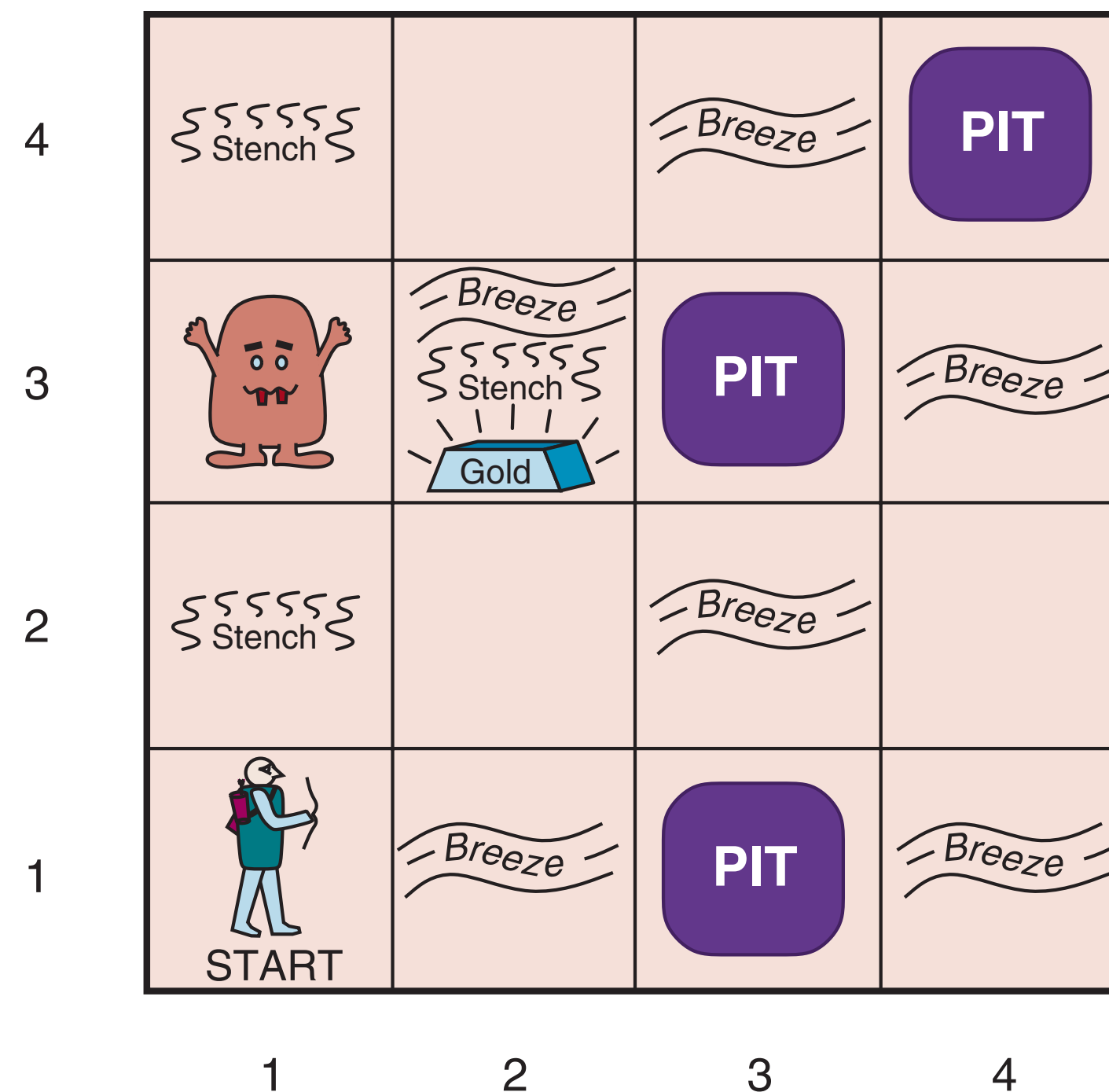
Prova de teorema como problema de busca

Prova de teorema é um outro método para inferência lógica. Ao invés de enumerar todos os modelos, a inferência é modelada por busca no espaço de estados:

- ▶ Estado inicial: BC inicial
- ▶ Ações: regras de inferência
- ▶ Modelo de transição: novas sentenças geradas com as aplicações das regras
- ▶ Estado final: estado com a sentença (consulta) que queremos provar
- ▶ Custo do caminho: número de passos da prova

Exemplo 3: o mundo de wumpus

O **mundo de wumpus** é um jogo onde o objetivo é fugir de uma caverna com uma pedra de ouro



Ambiente

- ▶ A posição de entrada e saída da caverna é sempre [1,1]
- ▶ A posição do wumpus e da pedra de ouro são escolhidas aleatoriamente
- ▶ Cada uma das outras posições podem ter um poço, com probabilidade 0,2
- ▶ As posições adjacentes de um poço possuem uma briza
- ▶ As posições adjacentes do wumpus possuem um mal cheiro

Agente

- ▶ Pode andar para frente, virar à esquerda ou à direita
- ▶ Pode deixar a caverna sem a pedra de ouro
- ▶ Possui uma única flecha que pode atirar na direção do wumpus para matá-lo
- ▶ Não conhece a posição do wumpus e dos poços

Exemplo 3: o mundo de wumpus

Símbolos Proposicionais

Para cada localização (x, y) :

- ▶ $P_{x,y}$ é verdadeiro se existe poço em (x, y)
- ▶ $W_{x,y}$ é verdadeiro se existe um wumpus em (x, y)
- ▶ $B_{x,y}$ é verdadeiro se existe uma briza em (x, y)
- ▶ $S_{x,y}$ é verdadeiro se existe um mal cheiro em (x, y)
- ▶ $L_{x,y}$ é verdadeiro se o agente está na posição (x, y)

Base de conhecimento (inicial)

- ▶ $L_{1,1}$ – A posição inicial do agente é $(1,1)$
- ▶ $\neg P_{1,1}$ – Não existe um poço em $(1,1)$
- ▶ $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
- ▶ $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
... declarar $B_{i,j}$ para cada posição (i, j)
- ▶ $S_{1,1} \Leftrightarrow (W_{1,2} \vee W_{2,1})$
- ▶ $S_{2,1} \Leftrightarrow (W_{1,1} \vee W_{2,2} \vee W_{3,1})$
... declarar $S_{i,j}$ para cada posição (i, j)

Após a visita de $(1,1)$:

- ▶ $\neg B_{1,1}$ – Não existe uma briza em $(1,1)$

Exemplo 3: o mundo de wumpus

Consulta

Existe poço em $P_{2,1}$?

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

Eliminação de bicondicional

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

Eliminação de E

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

Contrapositiva

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

Modus Ponens e $(\neg B_{1,1})$

$$\neg(P_{1,2} \vee P_{2,1})$$

De Morgan

$$\neg P_{1,2} \wedge \neg P_{2,1}$$

Prova por resolução

O método de prova de teorema pode não ser completo, mesmo se utilizarmos um algoritmo de busca completo.

- ▶ Pode não haver uma sequência de regras de inferência que produzam, a partir da base de conhecimento, a sentença que queremos provar
 - ▶ Por exemplo, sem a regra de eliminação de bicondicional, não conseguiríamos provar o teorema do Exemplo 3
- ▶ **Resolução** é uma **regra de inferência** que pode ser utilizada para criar um método de prova de teoremas **completo**!

Resolução

Se uma das duas proposições em uma proposição OU é falsa,
a outra tem que ser verdadeira

$$\alpha \vee \beta$$

"Lucas está no PVA ou Marcos está no PVB"

$$\neg \alpha$$

"Lucas não está no PVA"


$$\beta$$

Resolução


"Marcos está no PVB"

Resolução de cláusulas

A resolução pode ser generalizada para cláusulas com n símbolos.
Onde uma **cláusula** é uma disjunção (OUs) de **literais**


$$\alpha \vee \beta_1 \vee \beta_2 \vee \dots \vee \beta_n$$

$$\neg \alpha$$



Literais são símbolos proposicionais com ou sem negação (e.g, P , $\neg Q$)

Resolução

$$\beta_1 \vee \beta_2 \vee \dots \vee \beta_n$$

Resolução

Se uma das duas proposições em uma proposição OU é falsa,
a outra tem que ser verdadeira

$$\alpha \vee \beta$$

“Lucas está no PVA ou Marcos está no PVB”

$$\neg \alpha \vee \gamma$$

“Lucas não está no PVA ou Carlos está LBI”

Resolução

$$\beta \vee \gamma$$

“Marcos está na PVB ou Carlos está no LBI”

Resolução de cláusulas

A resolução pode ser generalizada para cláusulas com n símbolos.
Onde uma **cláusula** é uma sentença de **literais** conectados por OUs

$$\alpha \vee \beta_1 \vee \beta_2 \vee \dots \vee \beta_n$$

$$\neg \alpha \vee \gamma_1 \vee \gamma_2 \vee \dots \vee \gamma_m$$

Resolução

$$\beta_1 \vee \beta_2 \vee \dots \vee \beta_n \vee \gamma_1 \vee \gamma_2 \vee \dots \vee \gamma_m$$

Forma normal conjuntiva

Para estar na **forma normal conjuntiva (FNC)**, uma sentença lógica deve ser estruturada como uma conjunção (Es) de cláusulas, por exemplo:

$$(A \vee B \vee C) \wedge (D \vee \neg E) \wedge (F \vee G)$$

É possível converter qualquer sentença lógica para **FNC** usando regras de inferências:

- ▶ Eliminar bicondicionais – $\alpha \Leftrightarrow \beta$ em $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$
- ▶ Eliminar implicações – $\alpha \Rightarrow \beta$ em $\neg \alpha \vee \beta$
- ▶ Move \neg para dentro usando De Morgan – $\neg(\alpha \wedge \beta)$ em $\neg \alpha \vee \neg \beta$
- ▶ Usar distributiva para distribuir \vee sempre que possível

Exemplo 4: forma normal conjuntiva

$$(P \vee Q) \Rightarrow R$$

Eliminação de implicação

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

De Morgan

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

Distributiva

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

Inferência por resolução

Uma vez que temos sentenças na forma normal conjuntiva, podemos aplicar o algoritmo de inferência por resolução para provar $BC \models \alpha$:

► Para determinar se $BC \models \alpha$, verificamos se $(BC \wedge \neg\alpha)$ é uma contradição!

1. Converter $(BC \wedge \neg\alpha)$ para sua forma normal conjuntiva

2. Aplicar resolução para produzir novas cláusulas

3. Se nesse processo produzirmos uma cláusula vazia

- Então $(BC \wedge \neg\alpha)$ é uma contradição e portanto $BC \models \alpha$

4. Se não, α não é uma consequência lógica de BC

Inferência por resolução

$$(A \vee B) \wedge (\neg B \vee C) \wedge (\neg C) \models A?$$

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

$$(A \vee B) \quad \underline{(\neg B \vee C)} \quad \underline{(\neg C)} \quad (\neg A)$$

$$\underline{(A \vee B)} \quad (\neg B \vee C) \quad (\neg C) \quad (\neg A) \quad \underline{(\neg B)}$$

$$(A \vee B) \quad (\neg B \vee C) \quad (\neg C) \quad \underline{(\neg A)} \quad (\neg B) \quad \underline{(A)}$$

$$(A \vee B) \quad (\neg B \vee C) \quad (\neg C) \quad (\neg A) \quad (\neg B) \quad (A) \quad \underline{()}$$

Contradição!

Inferência por resolução

```
def retrocesso-recursivo(BC, alpha):  
1.     clauses = set(fnc(BC and not alpha))  
2.     new = {}  
3.     while True:  
4.         for (c_i, c_j) in pair-clauses(c clauses):  
5.             resolvents = resolution(c_i, c_j)  
6.             if [] in resolvents:  
7.                 return True  
8.             new = clauses | resolvents  
9.             if new == clauses:  
10.                 return False  
11.     clauses = clauses | new
```

Próxima aula

A14: Raciocínio Probabilístico I

Variáveis aleatórias, probabilidades, distribuições, probabilidades condicionais, probabilidades conjuntas, regras de probabilidade, teorema de Bayes