Universidade do Minho
Escola de Engenharia
Departamento de Informática

Programação em Lógica

LICENCIATURA EM ENGENHARIA INFORMÁTICA
MESTRADO integrado EM ENGENHARIA INFORMÁTICA
Inteligência Artificial
2025/26

ISLab
Synthetic Intelligence Lab

Da última aula

Limitações da resolução de problemas por procura

- Sistemas de procura são muito eficientes na solução de problemas que podem ser formalizados por:
  - um estado inicial, ações e estado final (ou estados finais).
- mas, não são capazes de resolver problemas que exigem **raciocínio baseado em conhecimento sobre o *mundo***:
  - Porque o seu *modelo do mundo* é pobre e o raciocínio é limitado
    - e.g., diagnóstico médico, sistemas especialistas em geral,…
  - mesmo em casos (aparentemente) resolúveis por procura (usando planeamento), pode ser necessário adicionar conhecimento explícito.

ISLab
Synthetic Intelligence Lab

Da última aula
A Lógica

- Axiomas: conjunto inicial de fórmulas lógicas
- Teoremas: fórmulas derivadas a partir dos axiomas e/ou teoremas (consequências semânticas)
- Regras de Inferência: conjunto de regras de derivação
  - *Modus ponens* { (A se B), B } ⊢ A (*sound* – válida)
  - *Modus tollens* { (A se B), ¬A } ⊢ ¬B (*sound* – válida)
  - *Modus mistakens* { (A se B), A } ⊢ B (*unsound* – não válida)
- Sistema de Inferência: união dos axiomas e das regras de derivação $\mathcal{R}$
- Prova: sequência $<s_1, s_2, \ldots>$ de s_i que são axiomas ou são derivações usando $\mathcal{R}$ e um subconjunto dos membros da sequência que precedem s_i;
  - A sequência é uma prova para s_n (derivação ou dedução)
- Teoria: união dos axiomas e de todos os teoremas derivados usando $\mathcal{R}$
  - Diz-se consistente sse não existe nenhuma fórmula s tal que, na teoria $\mathcal{T}$, exista s e $\neg s$
- **Nenhuma destas considerações toma em linha de conta o significado!**
  Apenas a estrutura sintática!!!

- Conhecimento e Raciocínio;
- Lógica;
- **Programação em Lógica;**
- **Sistemas Baseados em Conhecimento.**

ISLab
Synthetic Intelligence Lab

A Programação em Lógica

- A Programação em Lógica é um formalismo computacional que combina 2 princípios básicos:

1. Usa a Lógica para representar conhecimento
   (representação de pressupostos e de conclusões)

2. Usa a Inferência para manipular o conhecimento
   (estabelecer as relações lógicas entre os pressupostos e as conclusões)
   (mecanizar os procedimentos de prova; raciocinar)

ISLab
Synthetic Intelligence Lab

Caracterização da Programação em Lógica PROLOG

- Um programa em PROLOG é criado pela adição de fórmulas designadas por **cláusulas**
- As cláusulas podem ser de 3 tipos:
  - Factos: expressam algo que é sempre verdadeiro

    p. `filho( xico,quim ).`

  - Regras: expressam algo que é verdadeiro, dependente da veracidade das condições

    p se q. `pai( josé,joão ) se filho( joão,josé ).`

  - Questões: expressam algo que é verdadeiro, dependente da veracidade das condições

    ?q. `? pai( josé,joão ).`

    ¬q. `¬pai( josé,joão ).`

ISLab
Synthetic Intelligence Lab

Caracterização da Programação em Lógica PROLOG

- Um programa em PROLOG é criado pela adição de fórmulas designadas por **cláusulas**
- As cláusulas podem ser de 3 tipos:
  - Factos: expressam algo que é sempre verdadeiro

    p. filho(xico,quim).

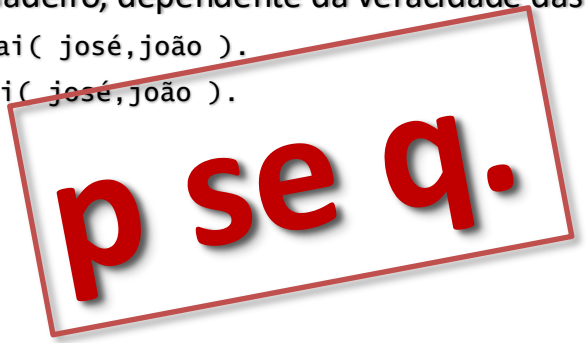
  - Regras: expressam algo que é verdadeiro, dependente da veracidade das condições

    p se q. pai(josé,joão) se filho(joão,josé).

  - Questões: expressam algo que é verdadeiro, dependente da veracidade das condições

    ?q. ? pai(josé,joão).

    ¬q. ¬pai(josé,joão).



ISLab
Synthetic Intelligence Lab

Cláusulas de Horn

- Clausulado de Horn (notação clausal da Lógica de primeira ordem)
  - É uma versão restrita do Cálculo Predicativo
  - É uma formula bem formada
  - Todas as fórmulas estão quantificadas universalmente
  - Todas as fórmulas são fechadas
  - As fórmulas lógicas admitem, apenas, 1 termo na disjunção positiva de literais

Cláusulas de Horn

  - $\neg q_i \vee p_j$
  - em que $i \geq 0$
    $0 \leq j \leq 1$

ISLab
Synthetic Intelligence Lab

Definições

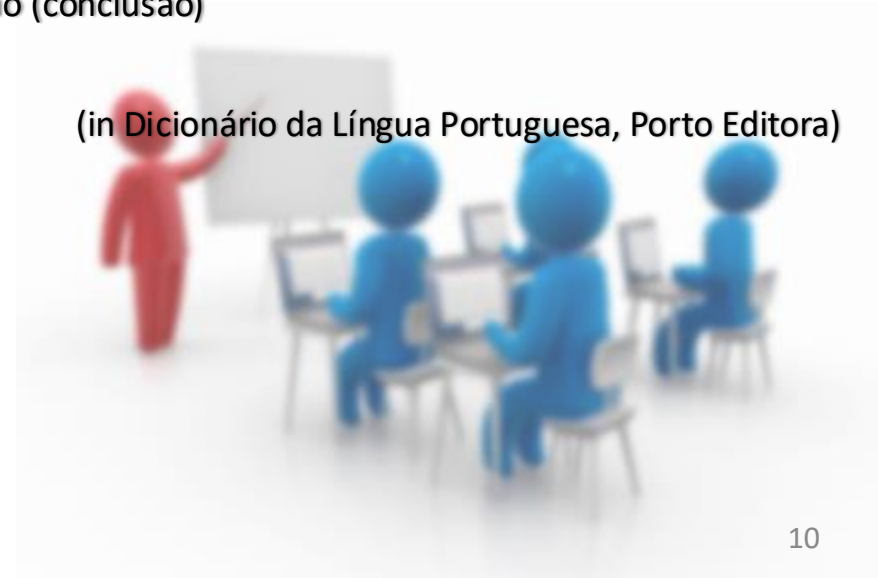
- Raciocínio:
  - Conjunto de pensamentos encadeados
  - Relacionamento de factos que levam a uma conclusão
  - Mistura dos 2 anteriores

ISLab
Synthetic Intelligence Lab

Definições

- Raciocínio:
  1. …
  2. Encadeamento lógico de pensamentos
  3. …
  4. [LÓGICA] Operação discursiva do pensamento mediante a qual concluímos que uma ou várias proposições (premissas) implicam a verdade (…) de uma outra proposição (conclusão)

(in Dicionário da Língua Portuguesa, Porto Editora)



ISLab
Synthetic Intelligence Lab

Raciocínio

- Três homens, António, Belmiro e Carlos, são cônjuges de Dulce, Eduarda e Filipa.
- Não se sabe quem é casado com quem.
- A formação deles é em Engenharia, Advocacia e Medicina.
- Não se sabe quem faz o quê.
- Com base nos dados abaixo, descubra o nome de cada esposa e a profissão de cada homem:
  - O médico é casado com Filipa;
  - Carlos é advogado;
  - Eduarda não é casada com Carlos;
  - Belmiro não é médico.

(adaptado de: www.novaconcursos.com.br/portal/dicas/questao-raciocinio-logico-para-concursos)

Raciocínio e Inferência

- Para convencer de que a resposta não é “um tiro de sorte”, é necessário expor as razões que levam a atingir a conclusão!

Raciocínio e Inferência

- Para convencer de que a resposta não é “um tiro de sorte”, é necessário expor as razões que levam a atingir a conclusão!

- Para que a conclusão seja aceite, torna-se necessário explicar o mecanismo de raciocínio aplicado, ou seja, o **processo de inferência**.

ISLab
Synthetic Intelligence Lab

Definições

- Inferência:
  - Diz-se do processo aplicado, que permite passar das premissas à conclusão.
  - Inferência: dedução ou conclusão.

(in Dicionário Priberam da Língua Portuguesa)

ISLab
Synthetic Intelligence Lab

Definições

- Inferência:
  - Diz-se do processo aplicado, que permite passar das premissas à conclusão.
  - Inferência: dedução ou conclusão.

(in Dicionário Priberam da Língua Portuguesa)

- Regras de inferência:
  - *Modus ponens*:
    - de: $P , P \rightarrow Q$
    - infere-se: Q

ISLab
Synthetic Intelligence Lab

Definições

- Inferência:
  - Diz-se do processo aplicado, que permite passar das premissas à conclusão.
  - Inferência: dedução ou conclusão.

(in Dicionário Priberam da Língua Portuguesa)

- Regras de inferência:
  - *Modus ponens*: (*modus ponendo ponens Latim significa "a maneira que afirma afirmando")*
    - de: P , $P \rightarrow Q$
    - infere-se: Q

P	Q	P → Q
[?]	[?]	[?]
[?]	[?]	[?]
[?]	[?]	[?]
[?]	[?]	[?]

ISLab
Synthetic Intelligence Lab

Definições

- Inferência:
  - Diz-se do processo aplicado, que permite passar das premissas à conclusão.
  - Inferência: dedução ou conclusão.

(in Dicionário Priberam da Língua Portuguesa)

- Regras de inferência:
  - *Modus ponens*:
    - de: $P , P \rightarrow Q$
    - infere-se: Q
  - *Modus tollens*:
    - de: $P \rightarrow Q , \neg Q$
    - infere-se: $\neg P$

ISLab
Synthetic Intelligence Lab

Definições

- Inferência:
  - Diz-se do processo aplicado, que permite passar das premissas à conclusão.
  - Inferência: dedução ou conclusão.

(in Dicionário Priberam da Língua Portuguesa)

- Regras de inferência:
  - *Modus ponens*:
    - de: P , $P \rightarrow Q$
    - infere-se: Q
  - *Modus tollens* (Latim: modo que nega por negação):
    - de: $P \rightarrow Q$, $\neg Q$
    - infere-se: $\neg P$

P	Q	¬Q	P → Q
?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução

- A regra de inferência *modus ponens* permite derivar como verdadeira a conclusão de uma cláusula pela prova das respetivas condições:

$$\{ (A \text{ se } B), B \} \vdash A$$

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução

- A regra de inferência *modus ponens* permite derivar como verdadeira a conclusão de uma cláusula pela prova das respetivas condições:

$$\{ (A \text{ se } B), B \} \vdash A$$

- A aplicação da regra de inferência *modus tollens* permite dirigir a procura da prova para um ponto concreto do processo de raciocínio:

$$\{ (A \text{ se } B), \neg A \} \vdash \neg B$$

- o que permite desenvolver um **mecanismo de prova por contradição**.

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução

- Supor que temos um programa □, no qual queremos determinar a derivabilidade de uma questão A:
  i. Admita-se a negação de A:
     ¬A
  ii. Insira-se a negação de A no programa □:
     □ □ ¬A
- Se acontecer gerar-se uma contradição
  { A, ¬A } □ □
  tal significa que a questão inicial A é derivável de □.

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução

- Supor que temos um programa □, no qual queremos determinar a derivabilidade de uma questão A:
  i. Admita-se a negação de A:
     ¬A
  ii. Insira-se a negação de A no programa □:
     □ □ ¬A
- Se acontecer gerar-se uma contradição
  { A, ¬A } □ □
  tal significa que a questão inicial A é derivável de □.

- Regra de inferência modus tollens, com B = verdadeiro
  { (A se B), ¬A } □ ¬B
  { (A se □), ¬A } □ ¬□
  { A, ¬A } □ □
  { A, ¬A } □ □

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação

```
% filho: Filho,Pai □ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho □ {□,□}

pai( P,F ) :- filho( F,P ).
```

Considere-se o programa □ indicado ao lado

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (I)

- O João é filho do José?

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

Admita-se a colocação da questão escrita acima

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (I)

- O João é filho do José?

  ¬filho(joao,jose)

```
% filho: Filho,Pai □ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho □ {□,□}

pai( P,F ) :- filho( F,P ).
```

Em termos lógicos, a questão é descrita na formulação indicada

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (I)

- O João é filho do José?

  ¬filho(joao,jose)

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

De todas as cláusulas do programa ?, apenas as que oferecem conclusões sobre o predicado `filho/2` contribuirão para o desenvolvimento a prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (I)

- O João é filho do José?

  ¬filho(joao,jose)

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

De entre as três cláusulas assinaladas, a primeira será utilizada no primeiro passo do desenvolvimento da (árvore de) prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (I)

- O João é filho do José?

¬filho(joao,jose)

joao| joao, jose | jose

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

O termo joao unifica com joao; o termo jose unifica com jose;

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (I)

- O João é filho do José?

¬filho(joao,jose)

joao| joao, jose | jose

¬(verdade)

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

Nas condições estabelecidas pelas unificações, que utilizaram a primeira cláusula de □, a questão inicial é reduzida à atual, por se tratar de um facto

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (I)

- O João é filho do José?

¬filho(joao,jose)
 / _____ joao| joao, jose | jose
¬(verdade)
 / _____ --|--
□

```
% filho: Filho,Pai □ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho □ {□,□}

pai( P,F ) :- filho( F,P ).
```

De onde se retira uma contradição □ (independentemente de quaisquer unificações)

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

Pretende-se saber se o José é pai do João

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

  ¬pai(jose,joao)

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

Em termos lógicos, a questão é descrita na formulação indicada

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

  ¬pai(jose,joao)

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

De todas as cláusulas do programa ?, apenas as que oferecem conclusões sobre o predicado `pai/2` contribuirão para o desenvolvimento a prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

  ¬pai(jose,joao)

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

A única cláusula do predicado pai/2 assinalada será utilizada no primeiro passo do desenvolvimento da (árvore de) prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

¬pai(jose,joao)

jose| P, joao | F

```
% filho: Filho,Pai ▯ {▯,▯}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ▯ {▯,▯}

pai( P,F ) :- filho( F,P ).
```

O termo jose unifica com a variável P; o termo joao unifica com a variável F;

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

¬pai(jose,joao)

jose| P, joao | F

¬filho(joao,jose)

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

Para dar continuidade à (árvore de) prova, reduz-se a questão inicial à prova das condições da cláusula usada para proceder às unificações

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

¬pai(jose,joao)
jose| P, joao | F
¬filho(joao,jose)

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

De todas as cláusulas do programa □, apenas as que oferecem conclusões sobre o predicado `filho/2` contribuirão para o desenvolvimento a prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

¬pai(jose,joao)
jose| P, joao | F
¬filho(joao,jose)

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

De entre as três cláusulas assinaladas, a primeira será utilizada no primeiro passo do desenvolvimento da (árvore de) prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

¬pai(jose,joao)

jose| P, joao | F

¬filho(joao,jose)

joao| joao, jose | jose

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

O termo joao unifica com joao; o termo jose unifica com jose;

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

¬pai(jose,joao)
 jose| P, joao | F
¬filho(joao,jose)
 joao| joao, jose | jose
¬(verdade)

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

Nas condições estabelecidas pelas unificações, que utilizaram a primeira cláusula de ?, a questão anterior é reduzida à atual, por se tratar de um facto

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (II)

- O José é pai do João?

¬pai(jose,joao)
  / ___ jose| P, joao | F
¬filho(joao,jose)
  / ___ joao| joao, jose | jose
¬(verdade)
  / ___ --|--
□

```
% filho: Filho,Pai □ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho □ {□,□}

pai( P,F ) :- filho( F,P ).
```

De onde se retira uma contradição □ (independentemente de quaisquer unificações)

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

Admita-se a colocação da questão escrita acima

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?

  ¬filho(jose,joao)

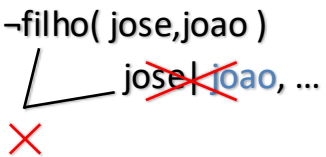
```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

Em termos lógicos, a questão é descrita na formulação indicada; de entre as três cláusulas do predicado `filho/2`, a primeira será utilizada para proceder ao desenvolvimento da (árvore de) prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?



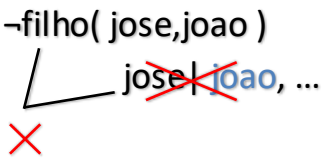

```
% filho: Filho,Pai ∈ {⍰,⍰}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {⍰,⍰}

pai( P,F ) :- filho( F,P ).
```

O termo jose não unifica com joao por se tratarem de duas constantes diferentes, pelo que o procedimento de prova não pode evoluir através deste ramo de prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?



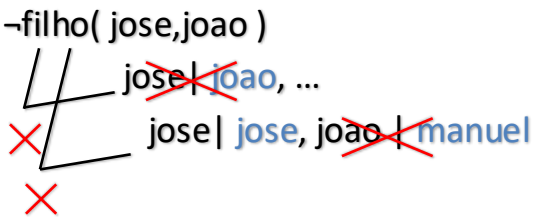

```
% filho: Filho,Pai ∈ {∅,∅}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {∅,∅}

pai( P,F ) :- filho( F,P ).
```

Abandonando a primeira cláusula, toma-se a segunda como alternativa para dar continuidade ao desenvolvimento da (árvore de) prova;

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?



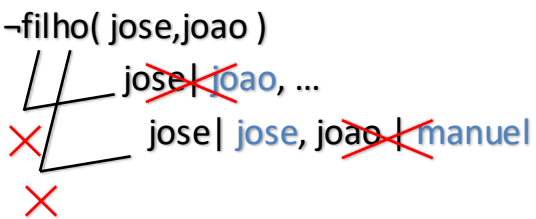

```
% filho: Filho,Pai ∈ {∅,∅}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {∅,∅}

pai( P,F ) :- filho( F,P ).
```

O termos jose unifica com jose; o termo joao não unifica com manuel por se tratarem de duas constantes diferentes, pelo que o procedimento de prova não pode evoluir através deste ramo de prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?



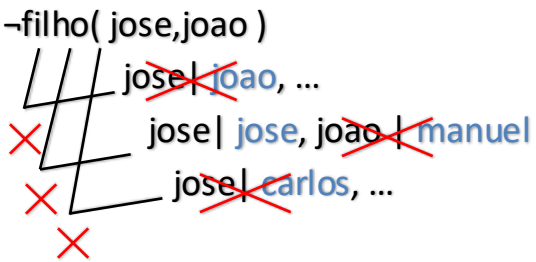

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

Abandonando, também, a segunda cláusula, toma-se a terceira (e última) como alternativa para dar continuidade ao desenvolvimento da (árvore de) prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?



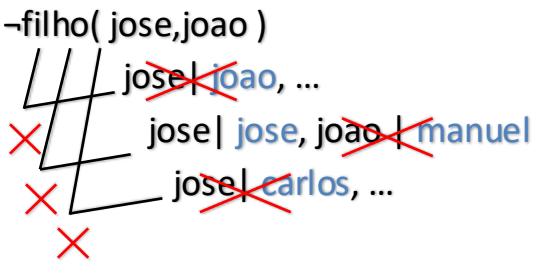

```
% filho: Filho,Pai ▯ {▯,▯}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ▯ {▯,▯}

pai( P,F ) :- filho( F,P ).
```

O termo jose não unifica com carlos por se tratarem de duas constantes diferentes, pelo que o procedimento de prova não pode evoluir através deste ramo de prova

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?



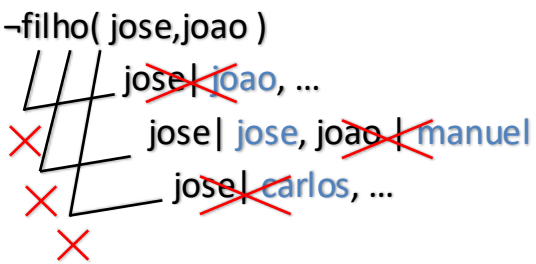

```
% filho: Filho,Pai ? {?,?}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ? {?,?}

pai( P,F ) :- filho( F,P ).
```

Não havendo mais cláusulas no predicado filho/2 que possam ser consideradas mais alternativas na (árvore de) prova, o procedimento de aplicação do algoritmo de resolução termina sem alcançar qualquer contradição

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (III)

- O José é filho do João?




```
% filho: Filho,Pai □ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho □ {□,□}

pai( P,F ) :- filho( F,P ).
```

Tal significa que a questão inicial não gera nenhuma contradição em □, logo, é falsa.

ISLab
Synthetic Intelligence Lab

Algoritmo de Resolução
Exemplo de aplicação (IV)

¬ filho(X,jose)

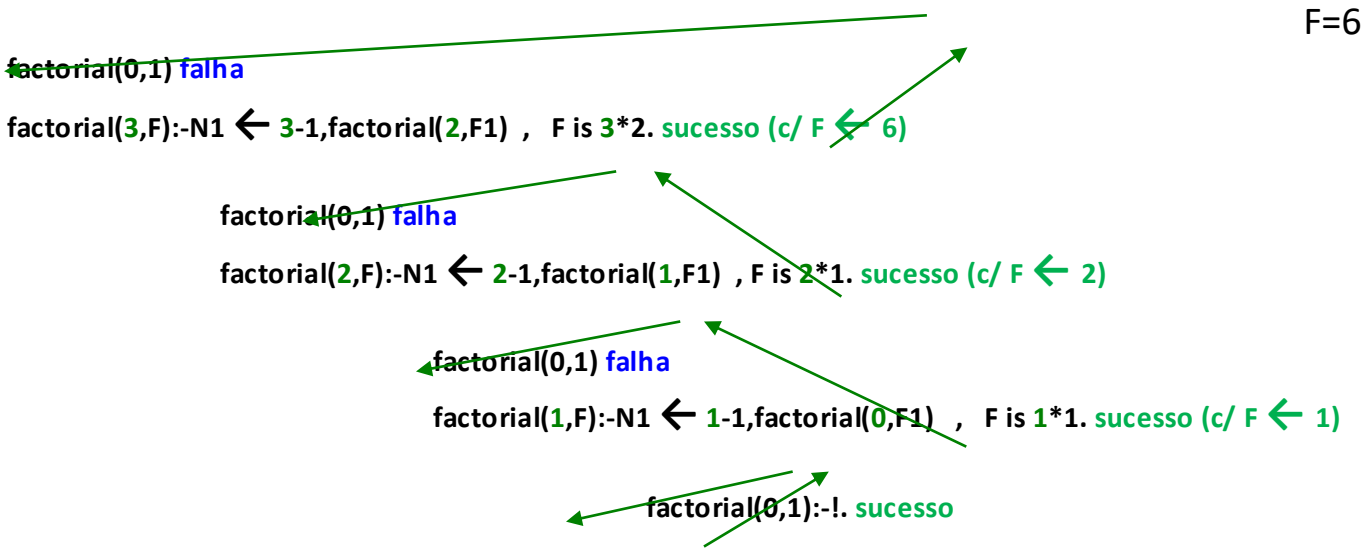
Qual é o significado desta questão

Desenvolver a árvore de prova para esta questão

```
% filho: Filho,Pai ∈ {□,□}

filho( joao,jose ).
filho( jose,manuel ).
filho( carlos,jose ).

% pai: Pai,Filho ∈ {□,□}

pai( P,F ) :- filho( F,P ).
```

```
factorial(0,1):-!.
factorial(N,F):-N1 is N-1,factorial(N1,F1),F is N*F1.
```

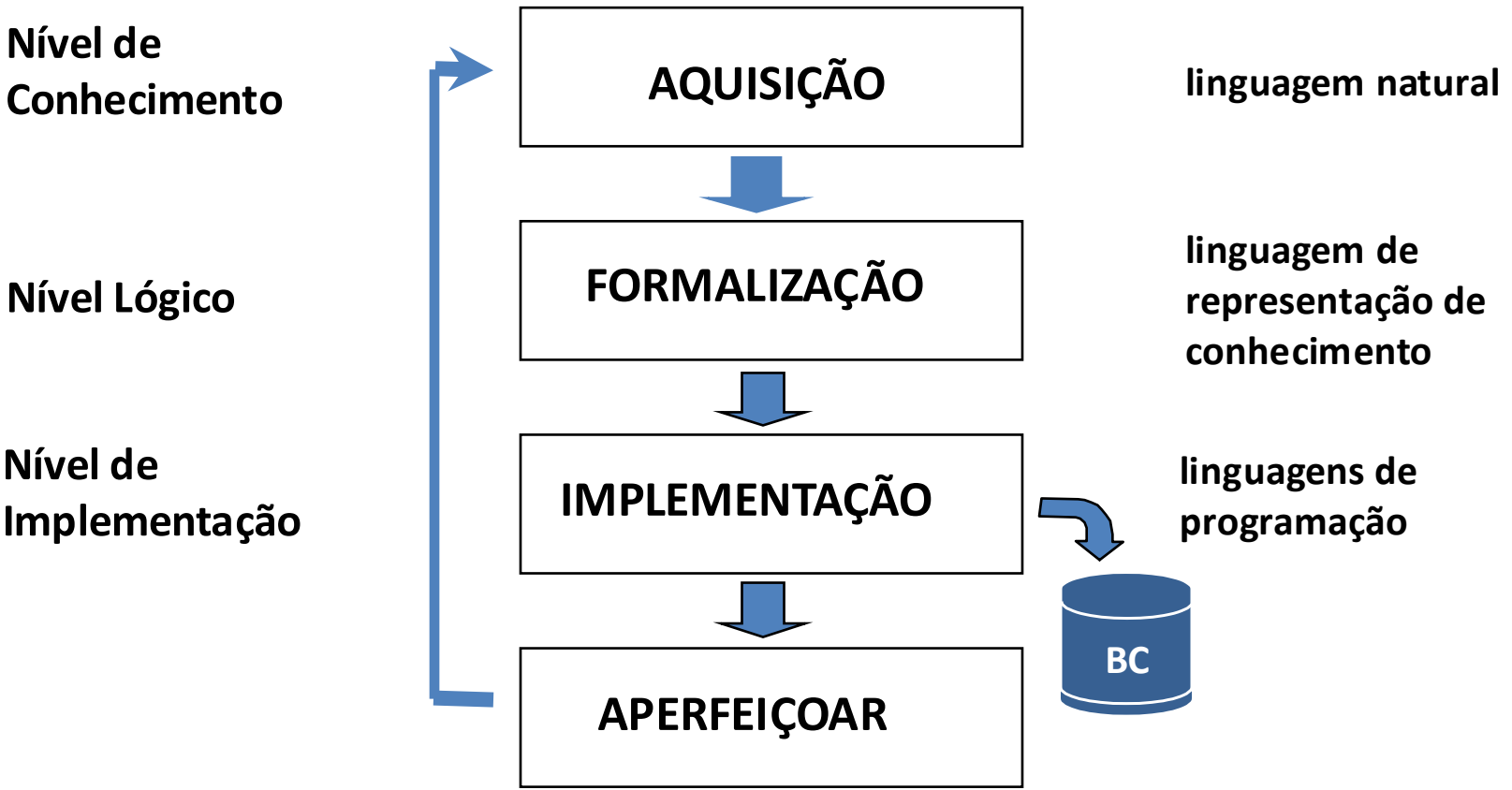
Vejamos o que acontece quando se efetua a chamada ?- factorial(3,F).



- São sistemas que
  - Utilizam conhecimento representado explicitamente para resolver problemas complexos;
  - Manipulam conhecimento e informação;
  - Têm "incrustado" a capacidade de raciocinar:
    - A habilidade de definir um conjunto de passos para a resolução eficiente de um problema;
    - O próprio mecanismo de inferência é conhecimento.

ISLab
Synthetic Intelligence Lab

Sistemas Baseados em Conhecimento



ISLab
Synthetic Intelligence Lab

Tipos de Raciocínio

- **Dedução**
  - factos + regras de inferência => novos factos
  - causa → efeito
    - Se há fogo (causa), há fumo (efeito). Aqui tem fogo, logo, aqui tem fumo (novo facto)
  - É o único tipo de inferência que mantém a verdade
    - *truth-preserving*
- **Abdução**
  - inverso da dedução: do efeito para a causa
    - Se há fumo, há fogo. Eu vi fumo (efeito), logo aqui tem fogo (causa)
  - Este tipo de inferência mantém a falsidade
  - Geramos uma possível explicação que terá de ser comprovada

Tipos de Raciocínio

- **Indução**
  - parte dos factos para gerar (novas) regras
    - facto1 + facto2 + facto3 → regra!
    - ex. Sr. Joaquim, assim como a D. Isabel, têm gripe e dor de cabeça, então todo a gente que tem gripe, tem dor de cabeça
  - Transforma factos (conhecimento em extensão) em conhecimento na forma de hipótese!
- **Analogia**
  - factos + similaridades + regras de adaptação
  - a partir de factos, da similaridade entre eles, resolve o problema sem gerar regras
    - e.g.: No caso anterior de gripe, eu tomei uma aspirina e não resolveu, logo não vou tomar aspirina neste semelhante caso

ISLab
Synthetic Intelligence Lab

Raciocínio na prática

- Dedução e Abdução
  - São usadas nos agentes baseados em conhecimento
- Indução e Analogia
  - São usadas na aprendizagem automática

- Dedução: existe dois grandes grupos
  - Lógica e afins
  - Tratamento de incerteza (e.g, Probabilístico ou *fuzzy*)

Implementações de Prolog

SWI-Prolog - A Free Software Prolog environment, licensed under the Lesser GNU public license. This popular interpreter was developed by Jan Wielemaker. This is the interpreter we used while developing this book.
http://www.swi-prolog.org/

SICStus Prolog - Industrial strength Prolog environment from the Swedish Institute of Computer Science.
http://www.sics.se/sicstus/

GNU Prolog - Another more widely used free Prolog compiler developed by Daniel Diaz.
http://www.gprolog.org

YAP Prolog - A Prolog compiler developed at the Universidade do Porto and Universidade Federa do Rio de Janeiro. Free for use in academic environments.
http://www.ncc.up.pt/~vsc/Yap/

ISLab
Synthetic Intelligence Lab

Bibliografia Recomendada

- Ivan Bratko, PROLOG: Programming for Artificial Intelligence, 4th Edition, Addison-Wesley Longman Publishing Co., Inc., ISBN-13: 978-0321417466, 2011.

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Programação em Lógica

LICENCIATURA EM ENGENHARIA INFORMÁTICA
MESTRADO integrado EM ENGENHARIA INFORMÁTICA
Inteligência Artificial
2025/26