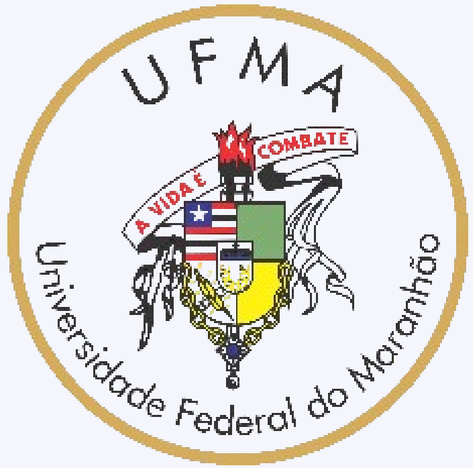

Lógica Formal

APLICAÇÃO DE LÓGICA USANDO PROLOG

COMPONENTES

EMANUELLE DA SILVA LAUNÉ

LUCYENE PINHEIRO NEVES

VINÍCIUS ANDRÉ ALMEIDA PEREIRA

SUMÁRIO

Introdução

Introdução à lógica de primeira ordem

- É a lógica em que o predicado de uma frase ou declaração pode ter apenas um sujeito como referido.

- É importante por ser um sistema de lógica formal capaz de formalizar de maneira computável as linguagens naturais.

- Com a lógica de primeira ordem é possível representar frases de modo formal,e a partir disso fazer conclusões, formular ideias e constatar teorias.

Introdução

Linguagem Natural

- É subtema da inteligência artificial que usa softwares para permitir que máquinas compreendam, processem e gerem linguagem humana de maneira adaptável e eficiente, facilitando a interação entre seres humanos e computadores.

- Possui objetos: pessoas, casas, números, cores, jogos, séculos...

- Relações: Unitárias, n-árias e funções.

- A lógica de primeira ordem é projetada em base de objetos e relações.

Introdução

Lógica de primeira ordem X lógica proposicional

- A lógica proposicional lida com proposições simples que são avaliadas como verdadeiras ou falsas, sem considerar a estrutura interna dessas proposições.

- Já a lógica de primeira ordem é mais estruturada, o que permite a inclusão de quantificadores (como "para todo" e "existe") e predicados, o que possibilita expressar relações mais complexas entre objetos e suas propriedades.

- Deste modo, a maior diferença entre as duas lógicas é chamada de compromisso ontológico.

Componentes Essenciais

- Objetos/termos
- Variáveis quantificadas
- Predicados
- Conectivos lógico

Objetos

- Objetos/termos: Entidades no domínio da lógica. Podem ser constantes (objetos específicos), variáveis (qualquer objeto do domínio) ou funções (geram objetos a partir de outros).

- Constantes: Representam objetos específicos no domínio de discurso. Exemplo: a, b, Maria, 3.

- Variáveis: Representam qualquer objeto ou elemento no domínio. Exemplo: x,y,z.

- Funções: São termos formados a partir de outros termos. Exemplo: irmaos(Maria, João), soma(x,y).

Predicado

- **Predicados:** Funções que retornam valores de verdade (verdadeiro ou falso), usadas para expressar propriedades ou relações entre objetos.

- Aplica-se a termos para descrever características ou relações.

- É uma maneira de guardar informações que podem ser classificadas como verdadeiras ou falsas, dependendo dos valores ou entidades aos quais se referem.

Predicado

- Exemplos em Prolog:

Fatos são as declarações mais simples em Prolog. Expressam relações ou propriedades que são consideradas verdadeiras e são formadas por um um predicado e argumentos entre parênteses.

- Ex: gosta(maria, doce). (Maria gosta de doce)

Regras são feitas para definir relações condicionais ou dependentes de outras condições.

- Ex: irmaos(X, Y) :- pai(X, Z), pai(Y, Z), X \= Y. (X e Y são irmãos se têm o mesmo pai Z e X é diferente de Y)

Variáveis

V**ariáveis quantificadas**: Variáveis sobre as quais fazemos afirmações gerais, quantificadas com os quantificadores existencial (∃) ou universal (∀), indicando a amplitude das afirmações.

Quantificador Universal (∀): Todas as variáveis em uma regra são universalmente quantificadas.

Ex: maior(X, Y) :- X > Y. (Para todo X e Y, X é maior que Y se X for maior que Y.)

Variáveis

Quantificador Existencial (∃): Indica que existe pelo menos um valor que satisfaz a condição.

Ex: tem_idade(joao, 25).

existe_alguem_com_idade(X) :- tem_idade(_, X).

(Existe alguém com a idade X se alguma pessoa tiver essa idade.)

Conectivos Lógicos

- **Conectivos lógicos:** Operadores que combinam ou manipulam expressões lógicas, como:
  - **Negação (¬)**: Indica a negação de uma proposição.
  - **Conjunção (∧)**: Ambas as proposições devem ser verdadeira.
  - **Disjunção (∨)**: Ao menos uma proposição deve ser verdadeira.
  - **Implicação (→)**: Se uma proposição AAA for verdadeira, então BBB também será.

Lógica Formal | Dedução vs Indução

DEDUÇÃO

- Raciocínio que parte de princípios gerais para chegar a conclusões específicas e logicamente necessárias.

Características:

  - Conclusão garantida.
  - Implícita nas premissas.
  - Baseada em regras formais.
  - Não amplia o conhecimento.

Lógica Formal | Dedução vs Indução

DEDUÇÃO

Exemplo:

Todos os mamíferos têm pulmões.

Todos os cães são mamíferos.

Logo, todos os cães têm pulmões.

deducao.pro

```
% Definindo as premissas
mamifero_tem_pulmoes.
cao_e_mamifero.

% Regra dedutiva
cao_tem_pulmoes :-
    mamifero_tem_pulmoes,
    cao_e_mamifero.

% Consultando a conclusão
:- (cao_tem_pulmoes ->
    writeln('Conclusão: Todos os cães têm pulmões.');
    writeln('Conclusão: A dedução falhou.')).
```

```
7 ?- consult('deducao.pro').
Conclusão: Todos os cães têm pulmões.
true.
```

Lógica Formal | Dedução vs Indução

INDUÇÃO

- Raciocínio que parte de casos específicos para formular generalizações universais, com conclusões prováveis.

Características:

- A conclusão não é necessariamente verdadeira.
- Amplia o conhecimento, gerando novas hipóteses.
- Baseada em observações e dados.

Lógica Formal | Dedução vs Indução

INDUÇÃO

Exemplo:
O cobre é condutor de eletricidade.

O ferro é condutor de eletricidade.

O ouro é condutor de eletricidade.

Concluímos que todos os metais são condutores de eletricidade.

inducao.pro

```
% Fatos: Metais conhecidos como condutores de eletricidade
condutor(cobre).
condutor(ferro).
condutor(ouro).

% Regra: Verifica se todos os metais fornecidos são condutores
todos_metais_sao_condutores([]). % Base: Lista vazia é sempre verdadeira.
todos_metais_sao_condutores([Metal | Resto]) :-
    condutor(Metal),                % Verifica se o metal atual é condutor.
    todos_metais_sao_condutores(Resto). % Recursivamente verifica o restante.

% Consulta para testar a indução
:- (todos_metais_sao_condutores([cobre, ferro, ouro, aluminio, zinco]) ->
    writeln('Conclusão: Todos os metais fornecidos são condutores de eletricidade.');
    writeln('Conclusão: Nem todos os metais fornecidos são condutores de eletricidade.')).
```

```
8 ?- consult('inducao.pro').
Conclusão: Nem todos os metais fornecidos são condutores de
true.
```

Representação Lógica do Conhecimento

- Representação lógica do conhecimento envolve estruturar informações sobre o mundo em formatos manipuláveis por algoritmos.

- **Principais Objetivos:**
  - Capturar fatos e regras de um domínio.
  - Facilitar inferências e tomadas de decisão.

Representação Lógica do Conhecimento

Sistemas Baseados em Regras

- Representação do conhecimento envolve estruturar informações sobre o mundo em formatos manipuláveis por algoritmos.

Principais Objetivos:

- Capturar fatos e regras de um domínio.

- Facilitar inferências e tomadas de decisão.

Representação Lógica do Conhecimento

Sistemas Baseados em Regras

- **Base de Fatos:** Contém o que já sabemos sobre o paciente.
- **Regras:** Lógicas que deduzem novas informações a partir dos fatos.
- **Inferência:** O sistema verifica as regras e imprime a conclusão baseada nas premissas.

medico_sis_baseado_regras.pro

```
% Base de Fatos
fato(paciente_tem_febre).
fato(paciente_tem_dor_de_garganta).

% Regras de Inferência
possivel_infeccao :- fato(paciente_tem_febre).
possivel_faringite :- fato(paciente_tem_febre), fato(paciente_tem_dor_de_garganta).

% Verificação de Inferência
verificar_inferencia :-
    (possivel_faringite -> writeln('Premissas 1 e 2 satisfeitas: O paciente pode ter faringite.');
     possivel_infeccao -> writeln('Apenas premissa 1 satisfeita: O paciente pode ter infecção.');
     writeln('Nenhuma condição satisfeita.')).
```

```
11 ?- verificar_inferencia.
Premissas 1 e 2 satisfeitas: O paciente pode ter faringite.
true.
```

Representação Lógica do Conhecimento

Base de Conhecimento

- Estruturas que armazenam proposições, predicados e regras para raciocínio lógico.

Características:

- Capacidade de expansão e atualização.

- Estruturada para consultas rápidas e inferência lógica.

Representação Lógica do Conhecimento

Base de Conhecimento

- **Fato:** Cachorro é um animal.
- **Regra:** Se é um animal, então respira.
- **Inferência:** Cachorro respira.

base.pro

```
% Base de Fatos
animal(cachorro).  % Fato: Cachorro é um animal

% Regra: Se algo é um animal, então respira
respira(X) :-
    animal(X).  % Se X é um animal, então X respira

% Consultar se o cachorro respira
consulta :-
    respira(cachorro),
    write('Cachorro respira.'), nl.
```

```
3 ?- consulta.
Cachorro respira.
true.
```

Introdução à Prolog

- Programas em Prolog seguem o estilo da Lógica de Primeira Ordem: Afirmam verdades e usam regras/implicações para calcular as consequências dessas afirmações.

- Consistem em fatos (suposições) e regras de inferência.

Introdução à Prolog

- Podemos executar programas Prolog tentando provar coisas por meio de consultas.

- A programação em Prolog é declarativa, ou seja, descreve o que deve ser feito (não como), e a execução se dá por meio de consultas que o sistema tenta resolver com base em fatos e regras lógicas.

Aplicações

- A lógica formal, especificamente a lógica de primeira ordem, forma a base de diversos sistemas computacionais e áreas de estudo em inteligência artificial (IA).

- Com o uso do Prolog, uma linguagem declarativa, é possível estruturar soluções que vão desde a resolução de problemas complexos até aplicações cotidianas que envolvem inferência lógica.

Exemplo 1

- Prolog tem sido amplamente utilizado devido à sua capacidade de modelar problemas com regras e fatos. O exemplo ao lado demonstra fatos e regras sobre relações familiares.

```
% Definicao de fatos
pai(joao, maria).
pai(joao, jose).
pai(carlos, ana).

mae(maria, ana).
mae(maria, pedro).

% Definicao de regras
filho(X, Y) :- pai(Y, X).
filho(X, Y) :- mae(Y, X).

% Inicializacao
:- initialization(main).

main :-
    % Realiza a consulta e exibe o resultado
    (pai(joao, maria) ->
        write('Joao e pai de Maria: Sim.')
    ;
        write('Joao e pai de Maria: Nao.')),
    nl,  % Adiciona uma nova linha para melhor formatação

    (mae(maria, pedro) ->
        write('Maria e mae de Pedro: Sim.')
    ;
        write('Maria e mae de Pedro: Nao.')),
    nl.  % Adiciona uma nova linha para melhor formatação
```

```
Joao e pai de Maria: Sim.
Maria e mae de Pedro: Sim.
```

Exemplo 2

- Estabelecendo regras para verificar se a paciente (Ana) possui Gripe, Resfriado ou estar apenas estressada.

```
sintoma(ana, febre).
sintoma(ana, tosse).
sintoma(ana, coriza).
sintoma(ana, dor_de_cabeca).

doenca(gripe) :- sintoma(_, febre), sintoma(_, tosse), sintoma(_, dor_de_cabeca), sintoma(_, coriza).
doenca(resfriado) :- sintoma(_, febre), sintoma(_, dor_de_cabeca), sintoma(_, coriza).
doenca(estresse) :- sintoma(_, dor_de_cabeca).

% Inicialização
:- initialization(main).

main :-
    (doenca(gripe) ->
        write('Ana tem gripe.'), nl
    ;
    (doenca(resfriado) ->
        write('Ana tem resfriado.'), nl
    ;
    (doenca(estresse) ->
        write('Ana tem estresse.'), nl
    ;
        write('Ana não tem nenhuma doença conhecida.'), nl
    ))).
```

```
Ana tem gripe.
```

Conclusão

- A lógica formal, com seus fundamentos e a representação lógica do conhecimento, é a base para o desenvolvimento de sistemas inteligentes.

- Ela permite estruturar e formalizar argumentos, garantindo raciocínio preciso e inferências válidas, fundamentais para a construção de algoritmos eficientes e a tomada de decisões consistentes em computação e IA.

Referências

Prolog-tutorial.ppt. Disponível em: https://homepage.cs.uri.edu/faculty/hamel/courses/2018/fall2018/csc501/lecture-notes/prolog-tutorial.pdf. Acesso em: 12 jan. 2025.

AWATI, Rahul. Lógica de primeira ordem. Disponível em: https://www.techtarget.com/whatis/definition/first-order-logic. Acesso em: 11 jan. 2025.

MUNDIUM, Roberto Patrus. Lógica Formal - princípios elementares. Economia & Gestão, Belo Horizonte. Disponível em: https://periodicos.pucminas.br/index.php/economiaegestao/article/view/113/104

BASIC, B.D.; Snajder, J. Knowledge Representation Using Formal Logic. Faculty of Electrical Engineering and Computing, University of Zagreb, 2018. Disponível em: https://www.fer.unizg.hr/_download/repository/AI-2018-05-KnowledgeRepresentation.pdf.