



Universidade do Minho

Escola de Engenharia

Departamento de Informática

Lógica

LICENCIATURA EM ENGENHARIA INFORMÁTICA
MESTRADO integrado EM ENGENHARIA INFORMÁTICA

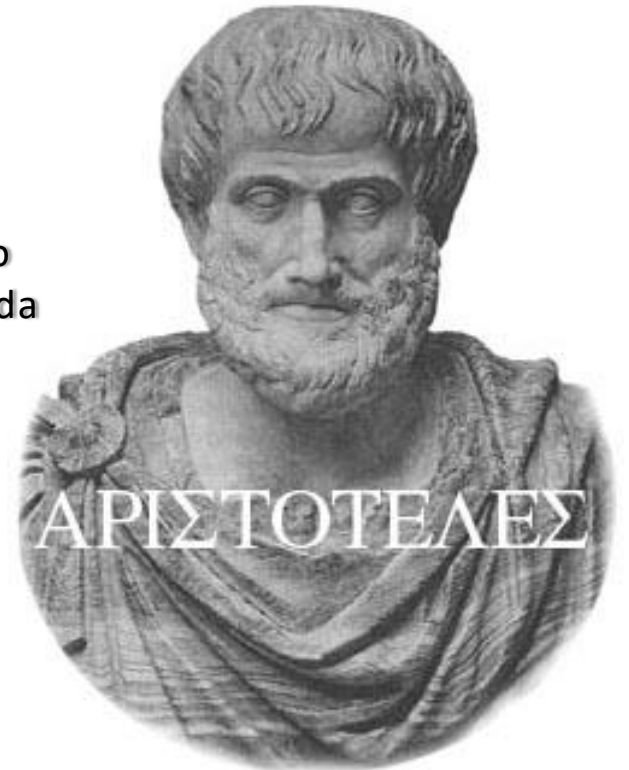
Inteligência Artificial

2025/26

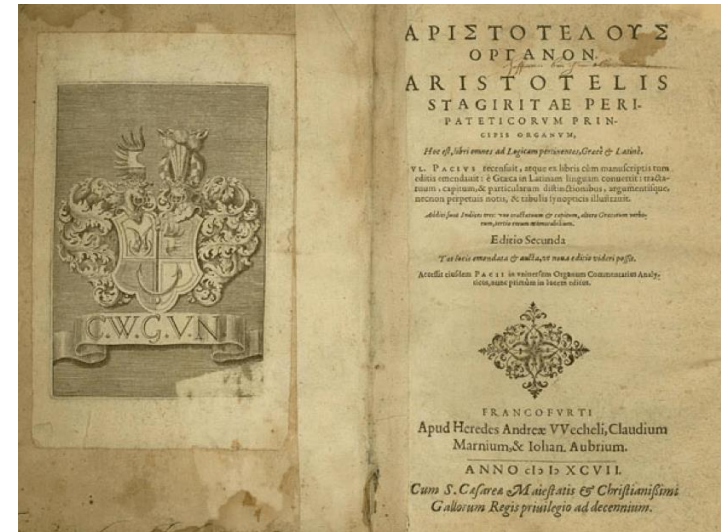
Representação do Conhecimento

- Conhecimento e Raciocínio;
- Lógica e Programação em Lógica;
- Regras de Produção;
- Programação Dirigida aos Padrões;
- Estruturas hierárquicas:
 - Redes semânticas;
 - Frames;
- Scripts;
- Sistemas Baseados em Conhecimento.

- **Lógica:**
 - Do grego «logos» (λογική)
 - [Dicionário Priberam da Língua Portuguesa](#)
 - [Infopédia - dicionários Porto Editora](#)
- A Lógica tem como objeto de estudo as leis gerais do pensamento (raciocínio) e o modo de as aplicar corretamente na investigação da verdade (mecanização do raciocínio);



- Aristóteles (384 a.C. – 322 a.C.)
 - Filósofo grego;
 - Sistematizou os conhecimentos sobre Lógica na obra “Organon”, conjunto de 6 textos;
 - “Organon”: instrumento; ferramenta para o correto pensar;



- Estudou o modo como o encadeamento de observações permite obter novo conhecimento: raciocínio;
 - Termo: coisa; representação de algo;
 - Mortal;
 - Bandeira;
 - Proposição: afirmação passível de avaliação lógica;
 - A bandeira portuguesa é redonda;
 - Silogismo: mecanismo de interpretação que obtém uma conclusão a partir de duas afirmações:
 - Qualquer homem é mortal;
 - Sócrates é homem;
 - Logo, Sócrates é mortal.

- A partir de observações avaliadas como verdadeiras, o lugar da Lógica é na formulação de leis gerais de encadeamento que permitem a descoberta de novas verdades.

- **Lógica:**
 - Relacionamento de proposições

- **Proposição:**
 - Afirmação (facto)

- **Predicado:**
 - Algo que se diz sobre uma coisa



- **Lógica:**

- Ciência de raciocinar

in Dicionário Priberam da Língua Portuguesa

- Disciplina normativa, tradicionalmente vinculada à filosofia, que se propõe determinar as condições da verdade nos diferentes domínios do saber

in Infopédia – Dicionários Porto Editora

- **Proposição:**

- Enunciado verbal suscetível de ser declarado verdadeiro ou não;

in Dicionário Priberam da Língua Portuguesa

- **Predicado:**

- Qualidade positiva; virtude; mérito;

In Infopédia – Dicionários Porto Editora

- Propriedade característica de algo ou de alguém (atributo);

in Dicionário Priberam da Língua Portuguesa



- A Lógica Matemática adota dois princípios fundamentais de pensamento:
 - Princípio da não negação:
 - Uma proposição não pode ser verdadeira e falsa ao mesmo tempo;
 - Princípio do terceiro excluído:
 - Qualquer proposição ou é verdadeira ou é falsa;
 - Verifica-se sempre um destes casos e nunca um terceiro.
- Operadores de conexão, ou conectivas lógicas:
 - Expressões ou palavras usadas para compor novas proposições (e, ou, não, se ... então, ... se e só se ..., etc.);
- Valores lógicos:
 - O valor lógico de uma proposição é verdade se a proposição for verdadeira;
 - O valor lógico de uma proposição é falsidade se a proposição for falsa.

1. O céu está nublado.
2. Cristiano Ronaldo é presidente da República Portuguesa.
3. Que horas são?
4. São 7 e meia da tarde.
5. O Vitória é o melhor clube do mundo.
6. O Vitória é o único clube que existe.
7. Qual Vitória?
8. Pouco barulho!
9. Quem me dera ser do Vitória.
10. Portugal é Campeão Europeu de Futebol.
11. Portugal é Campeão do Mundo de Futebol?
12. Portugal é Campeão do Mundo de Futebol.

Proposições

1. O céu está nublado.
2. Cristiano Ronaldo é presidente da República Portuguesa.
- ~~3. Que horas são?~~
4. São 7 e meia da tarde.
5. O Vitória é o melhor clube do mundo.
6. O Vitória é o único clube que existe.
- ~~7. Qual Vitória?~~
- ~~8. Pouco barulho!~~
- ~~9. Quem me dera ser do Vitória.~~
10. Portugal é Campeão Europeu de Futebol.
- ~~11. Portugal é Campeão do Mundo de Futebol?~~
12. Portugal é Campeão do Mundo de Futebol.

- Proposição:
 - é uma afirmação que pode ser classificada como verdadeira ou falsa, ou seja, que tem um **Valor de Verdade**.
- Não-proposição:
 - É uma frase (ordem, exclamação, pergunta, desejo, conselho) que não pode ser classificada como verdadeira ou falsa, portanto, que **não tem** um **Valor de Verdade**.
- Proposições simples:
 - Proposições que não se decompõem noutras proposições:
 - Guimarães é uma cidade.
- Proposições compostas:
 - Proposições que se podem decompor em proposições mais simples:
 - Guimarães foi a primeira capital de Portugal **e** Lisboa é a capital atual.

Tabelas de verdade

- Por assunção do Princípio do terceiro excluído, qualquer proposição tem o valor lógico verdadeiro (V) ou falso (F);
- No caso das proposições compostas, o seu valor lógico depende unicamente dos valores lógicos das proposições simples que a compõem;
- Para determinar o valor lógico de proposições compostas utiliza-se a construção de **Tabelas de Verdade**:

p
V
F

p	q
V	V
V	F
F	V
F	F

p	q	r
V	V	V
V	V	F
V	F	V
V	F	F
F	V	V
F	V	F
F	F	V
F	F	F

- Negação (\neg): chama-se negação de uma proposição p à proposição representada por $\neg p$, cujo valor lógico é verdade (V) quando p é falsa e falsidade (F) quando p é verdadeira;

p	$\neg p$
F	V
V	F

Operações lógicas

- **Conjunção (\wedge):** chama-se conjunção de duas proposições p e q à proposição representada por $p \wedge q$, cujo valor lógico é verdade (V) quando ambas as proposições p e q são verdadeiras e falsidade (F) nos restantes casos;

p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

- Disjunção (\vee): chama-se disjunção de duas proposições p e q à proposição representada por $p \vee q$, cujo valor lógico é verdade (V) quando pelo menos uma das proposições é verdadeira e falsidade (F) quando ambas as proposições são falsas;

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

- Disjunção exclusiva (\vee'): chama-se disjunção exclusiva de duas proposições p e q à proposição representada por $p \vee' q$, cujo valor lógico é verdade (V) quando somente uma das proposições é verdadeira e falsidade (F) quando ambas as proposições são falsas ou ambas são verdadeiras;

p	q	$p \vee' q$
V	V	F
V	F	V
F	V	V
F	F	F

- Implicação (\rightarrow): chama-se implicação à proposição representada por $p \rightarrow q$ (leia-se “se p então q ”), cujo valor lógico é falsidade (F) quando p é verdadeira e q é falsa e verdade (V) nos restantes casos;

p	q	$p \rightarrow q$
V	V	V
V	F	F
F	V	V
F	F	V

- Equivalência (\leftrightarrow): chama-se equivalência à proposição representada por $p \leftrightarrow q$ (leia-se “p é equivalente a q”), cujo valor lógico é verdade (V) quando ambas as proposições são verdadeiras ou são falsas e falsidade (F) nos restantes casos;

p	q	$p \leftrightarrow q$
V	V	V
V	F	F
F	V	F
F	F	V

▪ Assumir que:

○ p : Lisboa é a capital do Brasil; q : Carlos é advogado; r : Está a chover;

1. $\neg p$: Lisboa não é a capital do Brasil

2. $\neg q$: Carlos não é advogado

3. $p \wedge q$: Lisboa é a capital do Brasil e Carlos é advogado

4. $p \vee r$: _____

5. $\neg q \wedge \neg r$: _____

6. $p \vee q \wedge r$: _____

7. $(p \vee q) \wedge r$: _____

8. $\neg(\neg r \wedge \neg q) \vee p$: _____

▪ Assumir que:

○ p : Lisboa é a capital do Brasil; q : Carlos é advogado; r : Está a chover;

1. $\neg p$: Lisboa **não** é a capital do Brasil
2. $\neg q$: Carlos **não** é advogado
3. $p \wedge q$: Lisboa é a capital do Brasil **e** Carlos é advogado
4. $p \vee r$: Lisboa é a capital do Brasil **ou** está a chover
5. $\neg q \wedge \neg r$: Carlos **não** é advogado **e** **não** está a chover
6. $p \vee q \wedge r$: Lisboa é a capital do Brasil **ou** Carlos é advogado **e** está a chover
7. $(p \vee q) \wedge r$: Lisboa é a capital do Brasil **e** está a chover **ou** Carlos é advogado **e** está a chover
8. $\neg (\neg r \wedge \neg q) \vee p$: Lisboa é a capital do Brasil **ou não é verdade que não** está a chover **e**
Carlos **não** é advogado

- Como utilizar a lógica e a informatização, para automatizar os procedimentos de raciocínio?
- Existem duas aproximações para desenvolver a Lógica como um mecanismo computacional (de programação):
 - A Teoria dos Modelos
 - Hodges, Wilfrid – Model theory. Cambridge University Press, Cambridge 1997.
 - (<https://doi.org/10.1017/CBO9780511551574>)
 - A Teoria da Prova
 - A. S. Troelstra, H. Schwichtenberg (1996). Basic Proof Theory. In series Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, ISBN 0-521-77911-1.
 - (<https://doi.org/10.2307/2586674>)

- Teoria dos Modelos
 - Examina as relações entre fórmulas lógicas, quando interpretadas
 - Associa-lhes domínios de valores
 - Atribui-lhes valores de verdade
 - Utiliza termos como:
 - Verdadeiro
 - Falso
 - Interpretação
 - Satisfação
 - Modelo
 - Implicação
 - Consequência semântica

▪ Teoria dos Modelos

- Examina as relações entre fórmulas lógicas, quando interpretadas
- Associa-lhes domínios de valores
- Atribui-lhes valores de verdade
- Utiliza termos como:
 - Verdadeiro
 - Falso
 - Interpretação
 - Satisfação
 - Modelo
 - Implicação
 - Consequência semântica

▪ Teoria da Prova

- Examina as relações entre as fórmulas lógicas através da derivabilidade a partir de outras fórmulas
- Usa regras que atuam, apenas, na estrutura das fórmulas
- Utiliza termos como:
 - Axioma
 - Regra de inferência
 - Teorema
 - Prova
 - Consistência
 - Consequência sintática

▪ Teoria dos Modelos

- O que tencionamos que seja verdadeiro
- As respostas que um programa implica

▪ Teoria da Prova



- O que somos capazes de provar



- As respostas que são computacionalmente obtidas de um programa

▪ Teoria dos Modelos

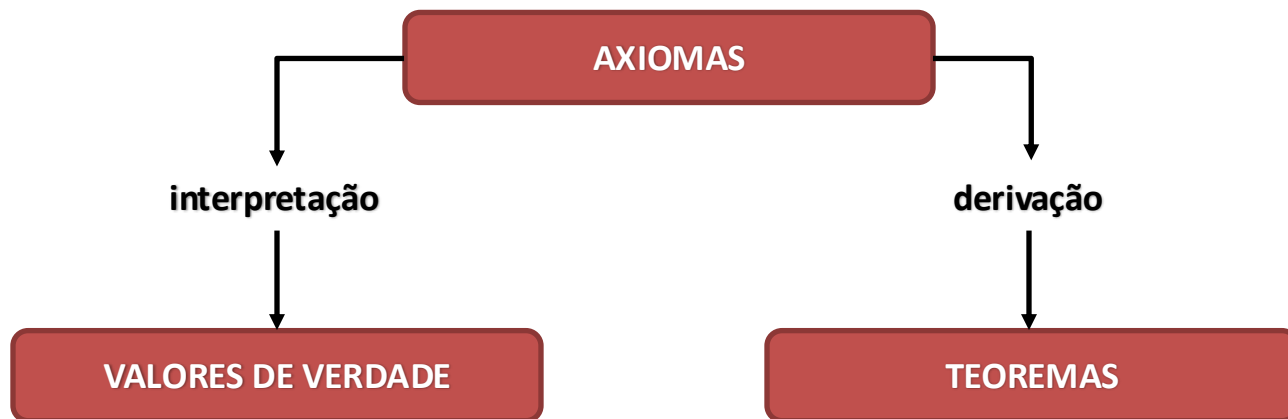
- O que tencionamos que seja verdadeiro
- As respostas que um programa implica

▪ Teoria da Prova



- O que somos capazes de provar

- As respostas que são computacionalmente obtidas de um programa



- A Teoria dos Modelos considera a atribuição do significado a fórmulas lógicas
- A relação de Implicação Lógica:
 - É uma relação sobre uma linguagem \mathcal{L} , tal que seja:
 - \mathcal{L} uma linguagem lógica (predicativa) de primeira ordem
 - S um subconjunto de \mathcal{L}
 - s um membro de \mathcal{L}
 - Tem-se que:
 - $\models = \{ \langle S, s \rangle \text{ em que qualquer modelo para } S \text{ é um modelo para } s \}$
 - $S \models s$
 - Recorrendo à noção de valores de verdade sobre \mathcal{L}

- A Teoria da Prova considera a geração de fórmulas lógicas a partir de outras fórmulas lógicas;
- A Teoria da Prova usa a noção de Derivabilidade de uma fórmula através da aplicação de um conjunto de regras de derivação;
- A relação de Derivabilidade:
 - É uma relação sobre uma linguagem \mathcal{L} , tal que seja:
 - \mathcal{L} uma linguagem lógica (predicativa) de primeira ordem
 - S um subconjunto de \mathcal{L}
 - \mathcal{R} um conjunto de regras de derivação
 - s um membro de \mathcal{L}
 - Tem-se que:
 - $\vdash = \{ \langle S, s \rangle \mid s \text{ é derivável de } S \text{ usando } \mathcal{R} \}$
 - $S \vdash s$

- 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* $\{ (A \text{ se } B), B \} \vdash A$ (*sound* – válida)
 - *Modus tollens* $\{ (A \text{ se } B), \neg A \} \vdash \neg B$ (*sound* – válida)
 - *Modus mistakens* $\{ (A \text{ se } B), A \} \vdash B$ (*unsound* – não válida)
- Sistema de Inferência: união dos axiomas e das regras de derivação \mathcal{R}
- Prova: sequência $\langle s_1, s_2, \dots \rangle$ 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!!!

A prova de teoremas em Lógica

- Pretende-se que:
 - s seja logicamente implicado por S
- Ou seja, que se obtêm respostas corretas em qualquer interpretação que use regras de inferência válidas (*modus ponens* ou *modus tollens*)
 - $\vdash S, s : S \vdash s$ se $S \vdash s$ (a derivabilidade é um subconjunto da implicação lógica)
- Ainda, pretende-se que:
 - s seja derivável de S , sempre que S implique s
- Para que não existam respostas corretas que não possam ser obtidas através de uma prova
 - $\vdash S, s : S \vdash s$ se $S \vdash s$ (a implicação lógica é um subconjunto da derivabilidade)
- Tal acontece quando \mathcal{L} (linguagem):
 - É a Lógica Proposicional
 - É a Lógica Predicativa de primeira ordem
 - Obedece à notação clausal

- 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)

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).

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).

p se q.

Caracterização da Programação em Lógica

PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

$\circ p \text{ se } q$

ou seja

$p \models q$

Caracterização da Programação em Lógica

PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

○ $p \text{ se } q$

ou seja

○ ou, admitindo a expansão de q

○ $p \Rightarrow q$

$p \Rightarrow q_1 \wedge q_2 \wedge \dots \wedge q_n$

Caracterização da Programação em Lógica PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

○ $p \text{ se } q$

ou seja

- ou, admitindo a expansão de q
- usando conectiva implicação lógica (\Rightarrow)

○ $p \Rightarrow q$

$p \Rightarrow q_1 \wedge q_2 \wedge \dots \wedge q_n$

$q_1 \wedge q_2 \wedge \dots \wedge q_n \Rightarrow p$

Caracterização da Programação em Lógica PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

○ $p \text{ se } q$

ou seja

- ou, admitindo a expansão de q
- usando conectiva implicação lógica (\Rightarrow)
- aplicando a regra de equivalência de introdução da implicação
- passamos a ter

$p \Rightarrow q$

$p \Rightarrow q_1 \wedge q_2 \wedge \dots \wedge q_n$

$q_1 \wedge q_2 \wedge \dots \wedge q_n \Rightarrow p$

$A \Rightarrow B \Leftrightarrow \neg A \vee B$

$\neg (q_1 \wedge q_2 \wedge \dots \wedge q_n) \vee p$

Caracterização da Programação em Lógica PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

○ p se q

ou seja

- ou, admitindo a expansão de q
- usando conectiva implicação lógica (\Rightarrow)
- aplicando a regra de equivalência de introdução da implicação
- passamos a ter
- aplicando a Lei de Morgan
- teremos

$p \Rightarrow q$

$p \Rightarrow q_1 \wedge q_2 \wedge \dots \wedge q_n$

$q_1 \wedge q_2 \wedge \dots \wedge q_n \Rightarrow p$

$A \Rightarrow B \Rightarrow \neg A \vee B$

$\neg(q_1 \wedge q_2 \wedge \dots \wedge q_n) \vee p$

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

$\neg q_1 \vee \neg q_2 \vee \dots \vee \neg q_n \vee p$

Caracterização da Programação em Lógica PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

○ p se q

ou seja

- ou, admitindo a expansão de q
- usando conectiva implicação lógica (\Rightarrow)
- aplicando a regra de equivalência de introdução da implicação
- passamos a ter
- aplicando a Lei de Morgan
- teremos
- que podemos reduzir a
- em que

$i \in 0$

$0 \leq j \leq 1$

$p \Rightarrow q$

$p \Rightarrow q_1 \wedge q_2 \wedge \dots \wedge q_n$

$q_1 \wedge q_2 \wedge \dots \wedge q_n \Rightarrow p$

$A \Rightarrow B \Rightarrow \neg A \vee B$

$\neg(q_1 \wedge q_2 \wedge \dots \wedge q_n) \vee p$

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

$\neg q_1 \vee \neg q_2 \vee \dots \vee \neg q_n \vee p$

$\neg q_i \vee p_j$

Caracterização da Programação em Lógica PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

o p se q

ou seja

o ou, admitindo a expansão de q

o usando conectiva implicação lógica (\Rightarrow)

o aplicando a regra de equivalência de introdução da implicação

o passamos a ter

o aplicando a Lei de Morgan

o teremos

o que podemos reduzir a

o em que

$i \in 0$

$0 \leq j \leq 1$

o ou seja

$p \Rightarrow q$

$p \Rightarrow q_1 \wedge q_2 \wedge \dots \wedge q_n$

$q_1 \wedge q_2 \wedge \dots \wedge q_n \Rightarrow p$

$A \Rightarrow B \Rightarrow \neg A \vee B$

$\neg(q_1 \wedge q_2 \wedge \dots \wedge q_n) \vee p$

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

$\neg q_1 \vee \neg q_2 \vee \dots \vee \neg q_n \vee p$

$\neg q_i \vee p_j$

$q_i \Rightarrow p_j$

Caracterização da Programação em Lógica PROLOG

- Utilizando a linguagem da Lógica para a representação do conhecimento usamos regras do tipo:

○ p se q ou seja

○ ou, admitindo a expansão de q

○ usando conectiva implicação lógica (\Rightarrow)

○ aplicando a regra de equivalência de introdução da implicação

○ passamos a ter

○ aplicando a Lei de Morgan

○ teremos

○ que podemos reduzir a

○ em n ou

○ ou seja

cláusulas de Horn

$i \geq 0$

$0 \leq j \leq 1$

$p \Rightarrow q$

$p \Rightarrow q_1 \wedge q_2 \wedge \dots \wedge q_n$

$q_1 \wedge q_2 \wedge \dots \wedge q_n \Rightarrow p$

$A \Rightarrow B \Rightarrow \neg A \vee B$

$\neg(q_1 \wedge q_2 \wedge \dots \wedge q_n) \vee p$

$\neg(\bigwedge B) \Rightarrow \neg A \vee \neg B$

$\neg q_1 \vee \neg q_2 \vee \dots \vee \neg q_n \vee p$

$\neg q_i \vee p_j$

$q_i \Rightarrow p_j$

- 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

- 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

$$i \in \{0, \dots, n-1\}$$

$$0 \leq j \leq m-1$$

Bibliografia Recomendada

- Stuart Russell and Peter Norvig, Artificial Intelligence - A Modern Approach, 4rd edition, ISBN: 978-0134610993, 2020, Chapter 7.
- Ivan Bratko, PROLOG: Programming for Artificial Intelligence, 3rd Edition, Addison-Wesley Longman Publishing Co., Inc., 2000.



Universidade do Minho

Escola de Engenharia

Departamento de Informática

Lógica

LICENCIATURA EM ENGENHARIA INFORMÁTICA
MESTRADO integrado EM ENGENHARIA INFORMÁTICA

Inteligência Artificial

2025/26