

AULA 06

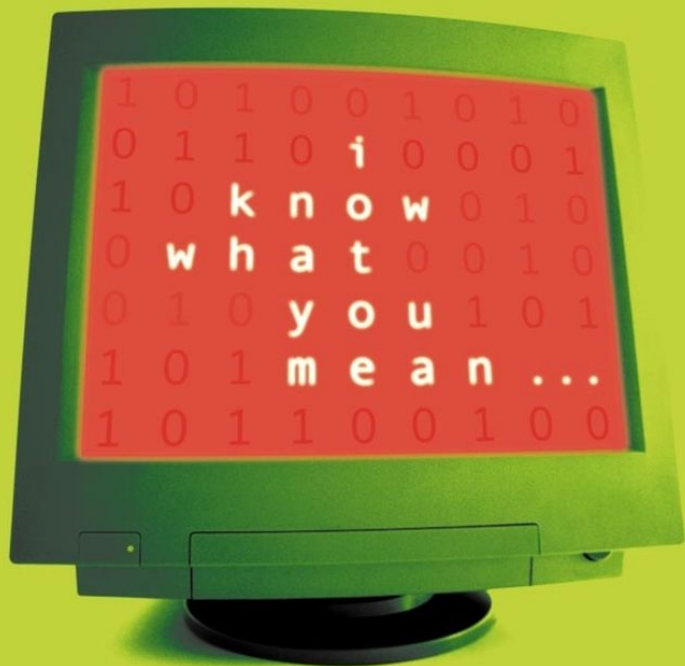
Lógica

CAMILA LARANJEIRA

mila.laranjeira@gmail.com



PUC Minas



Agenda

- Introdução à Lógica
- Ontologias
- Lógica Proposicional
- Lógica de 1ª Ordem

Você sabe lógica! Mas você sabe ensinar computadores a ter raciocínio lógico?

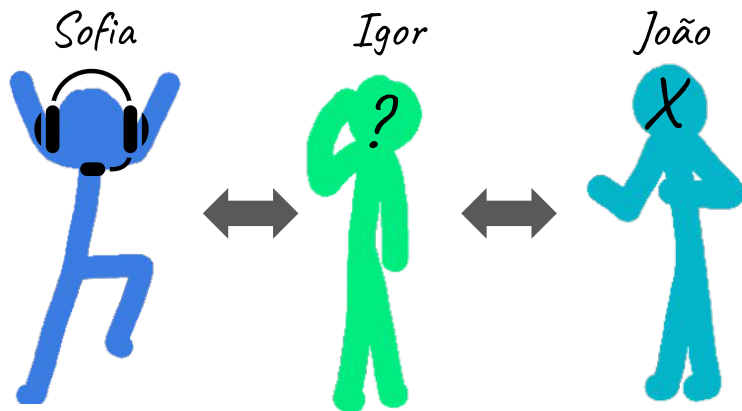
Sobre os slides

Esses slides usam material de:

- Week 9: Logic | CS221 Stanford (Autumn 2021)
 - https://stanford-cs221.github.io/autumn2021-extra/workouts/week9_slides.pdf
- Vários slides do curso CS227: Knowledge Representation and Reasoning | Stanford
 - <https://web.stanford.edu/class/cs227/>
- Lectures 7 and 8: Propositional Logic | Berkeley CS188: AI (Spring 2022)
 - <https://inst.eecs.berkeley.edu/~cs188/sp22/assets/slides/Lecture7.pdf>
 - <https://inst.eecs.berkeley.edu/~cs188/sp22/assets/slides/Lecture8.pdf>

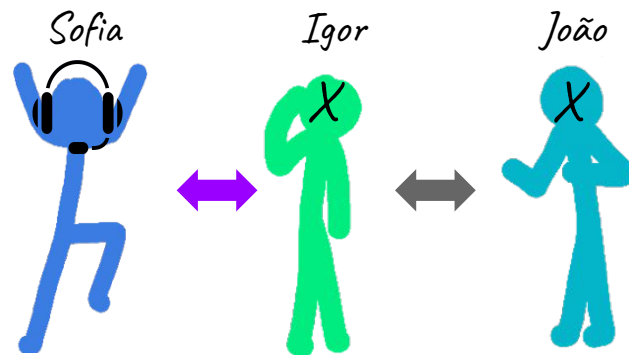
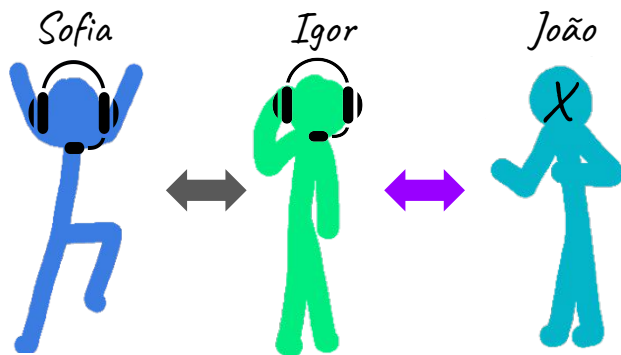
Lógica

- Sofia é amiga de Igor. Igor é amigo de João. Sofia joga Fortnite, João não joga, e não sabemos se Igor joga. Considerando essas relações, **existe amizade entre uma pessoa que joga Fortnite e outra que não joga?**



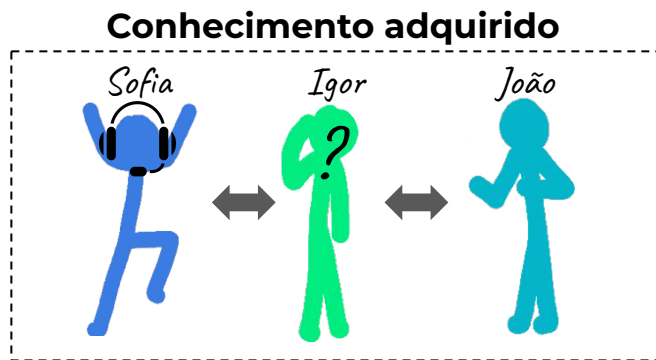
Lógica

- Sofia é amiga de Igor. Igor é amigo de João. Sofia joga Fortnite, João não joga, e não sabemos se Igor joga. Considerando essas relações, existe amizade entre uma pessoa que joga Fortnite e outra que não joga?
 - A resposta é sempre sim! Duas possibilidades: Igor joga e é amigo de João, ou Igor não joga e é amigo de Sofia.



Lógica

- Alguns problemas podem ser modelados da seguinte forma

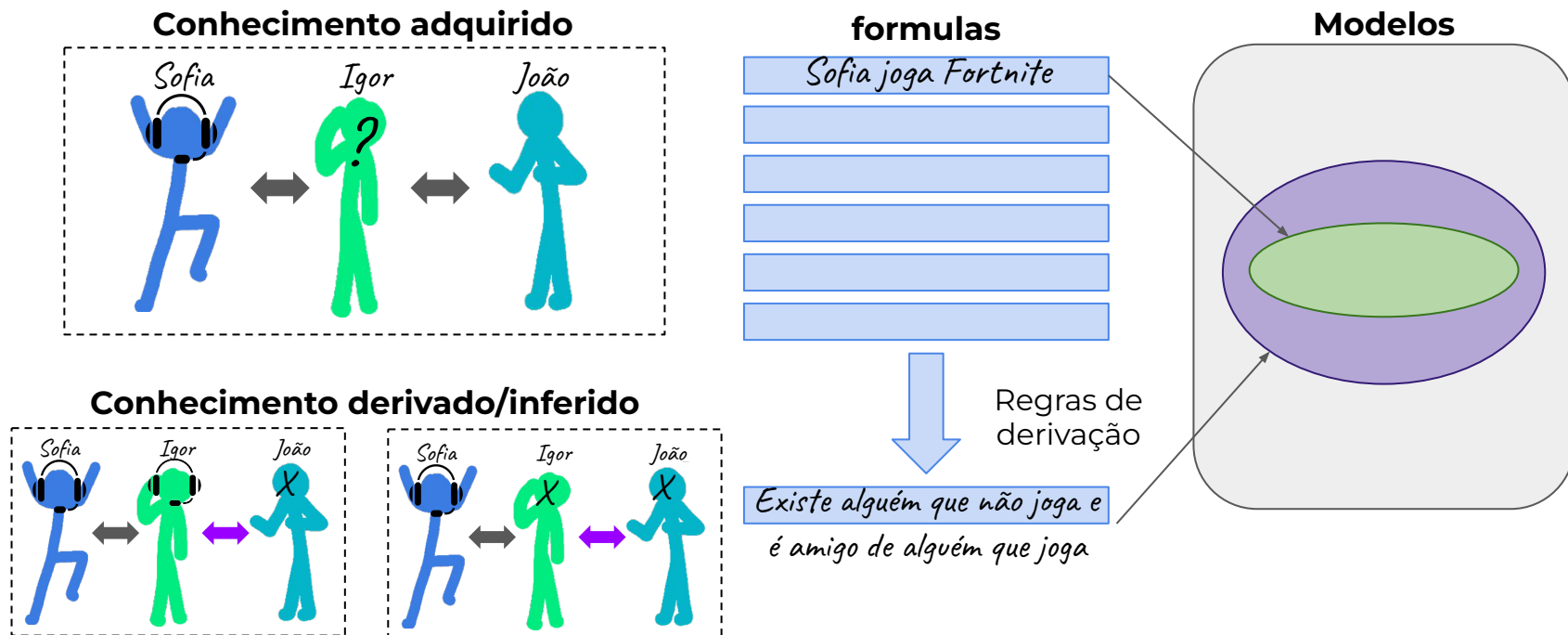


formulas

Sofia joga Fortnite

Lógica

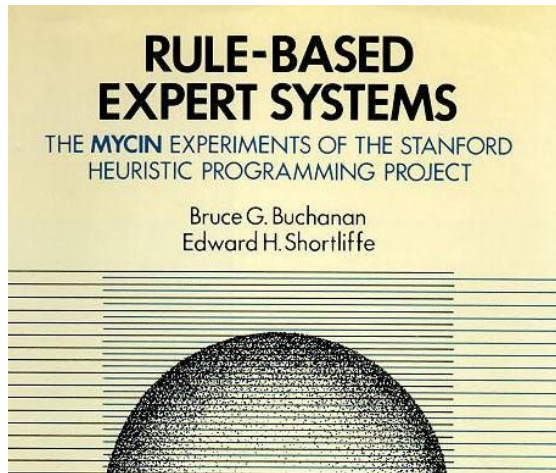
- Alguns problemas podem ser modelados da seguinte forma



Lógica

- Pensar racionalmente requer **representar conhecimento especialista** e **derivar soluções** lógicas dentro daquele escopo.
 - Primeira abordagem de I.A. a ser aplicada na indústria.

MYCIN: Diagnóstico de doenças



XCON: Recomendação de especificações de computador



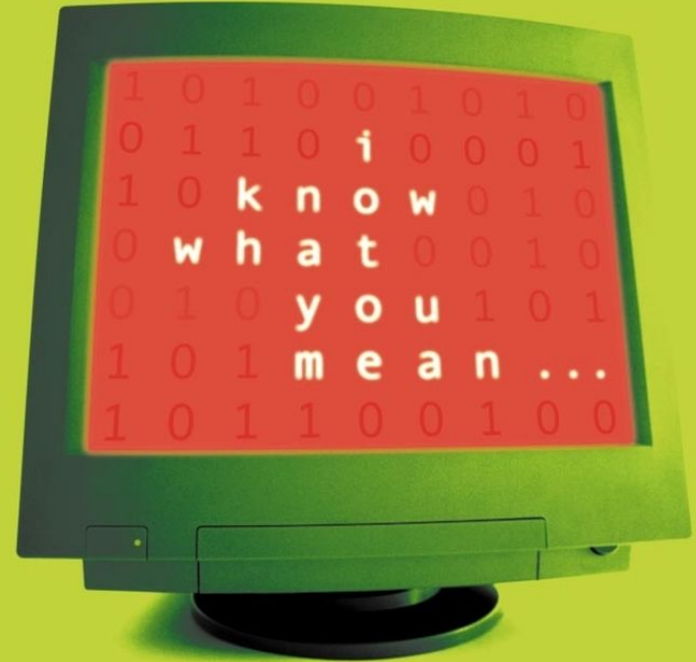
Semantic Web

SEMANTIC WEB - 2001

<https://www.jstor.org/stable/pdf/26059207.pdf>

"Machine-readable content"

Siri: proposta em 2007, no iPhone 4S em 2011



Google Knowledge Graph

SEARCH

Introducing the Knowledge Graph: things, not strings

<https://blog.google/products/search/introducing-knowledge-graph-things-not/>

May 16, 2012 · 4 min read



Amit Singhal
SVP, Engineering

Share

Google Knowledge Graph

All Images Videos News Shopping More Tools

About 1,780,000,000 results (0.55 seconds)

<https://developers.google.com/knowledge-graph>

Google Knowledge Graph Search API

Dec 10, 2021 — The **Knowledge Graph** has millions of entries that describe real-world entities like people, places, and things. These entities form the nodes of ...

Reference · Knowledge Graph Search... · Usage Limits · Prerequisites

People also ask


- How do I get a Google Knowledge Graph?
- Does Google still use Knowledge Graph?
- Is Google Knowledge Graph free?
- What is the Google Knowledge Graph and how it works?

Feedback

<https://support.google.com/knowledgepanel/answer>

How Google's Knowledge Graph works - Knowledge Panel Help

The **Knowledge Graph** allows us to answer factual questions such as "How tall is the Eiffel Tower?" or "Where were the 2016 Summer Olympics held." Our goal with ...



Google Knowledge Graph

The Google Knowledge Graph is a knowledge base used by Google and its services to enhance its search engine's results with information gathered from a variety of sources. The information is presented to users in an infobox next to the search results. [Wikipedia](#)

Size

Date

May 16, 2012

Painel de conhecimento

Knowledge panel

Google Knowledge Graph

Equipe do Google em BH coloca no ar o Health Search

The screenshot shows a Google Health Search interface. At the top, the search bar contains 'sintomas gripe'. Below the search bar, there are tabs for 'All', 'Images', 'News', 'Videos', 'Shopping', and 'More'. The search results show 'About 112,000,000 results (0.48 seconds)'. The main content area displays a snippet from 'https://butantan.gov.br' with the title 'Saiba como diferenciar os sintomas da gripe e da Covid-19 ...'. Below this, there is a section titled '7 principais sintomas de gripe - Tua Saúde' with a list of symptoms: 'Febre, normalmente entre 38 e 40°C; Calafrios; Dor de cabeça; Tosse, espirros e nariz escorrendo; Dor de garganta; Dor ...'. At the bottom, there is a 'Common questions' section with four questions: 'Diarreia pode ser sintoma de COVID-19?', 'Como ocorre a transmissão da COVID-19?', 'O que é o coronavírus?', and 'Devemos ficar em casa para nos proteger do coronavírus (COVID-19)?'. On the right side, there is a 'Flu' panel with tabs for 'OVERVIEW', 'SYMPTOMS', 'TREATMENTS', and 'SPECI'. The 'SYMPTOMS' tab is selected, showing 'Usually self-diagnosable' and 'People may experience:' sections. The 'Usually self-diagnosable' section lists symptoms: 'Symptoms include fever, chills, muscle aches, cough, congestion, runny nose, headaches and fatigue.' The 'People may experience:' section lists various symptoms: 'Pain areas: in the joints; Cough: can be dry, with phlegm, mild, or severe; Whole body: chills, dehydration, fatigue, fever, flushing, loss of appetite, malaise, body ache, or sweating; Nasal: congestion, runny nose, or sneezing; Throat: irritation or soreness; Also common: chest pressure, headache, muscle weakness, nausea, shortness of breath, or swollen lymph nodes'. At the bottom of the panel, there is a disclaimer: 'For informational purposes only. Consult your local medical authority for advice. Sources: Hospital Israelita A. Einstein and others. Learn more' and a 'Feedback' link.

sintomas gripe

About 112,000,000 results (0.48 seconds)

Os **sintomas** clássicos da **gripe** sazonal são febre súbita, tosse (geralmente seca), dor de cabeça, dores musculares e articulares, mal-estar, dor de garganta e coriza. A tosse pode ser forte e durar duas ou mais semanas, segundo a Organização Mundial de Saúde (OMS). Dec 24, 2021

<https://butantan.gov.br> > notícias > saiba-como-diferenciar...
Saiba como diferenciar os sintomas da gripe e da Covid-19 ...

About featured snippets • Feedback

<https://www.tuasaude.com> > sintomas... • Translate this page
7 principais sintomas de gripe - Tua Saúde
7 principais **sintomas** de **gripe** • Febre, normalmente entre 38 e 40°C; • Calafrios; • Dor de cabeça; • Tosse, espirros e nariz escorrendo; • Dor de garganta; • Dor ...

COVID-19 >

Common questions

Diarreia pode ser sintoma de COVID-19?

Como ocorre a transmissão da COVID-19?

O que é o coronavírus?

Devemos ficar em casa para nos proteger do coronavírus (COVID-19)?

For informational purposes only. Consult your local medical authority for health advice

Flu
Also called: influenza

OVERVIEW SYMPTOMS TREATMENTS SPECI

Usually self-diagnosable
Symptoms include fever, chills, muscle aches, cough, congestion, runny nose, headaches and fatigue.

People may experience:
Pain areas: in the joints
Cough: can be dry, with phlegm, mild, or severe
Whole body: chills, dehydration, fatigue, fever, flushing, loss of appetite, malaise, body ache, or sweating
Nasal: congestion, runny nose, or sneezing
Throat: irritation or soreness
Also common: chest pressure, headache, muscle weakness, nausea, shortness of breath, or swollen lymph nodes

For informational purposes only. Consult your local medical authority for advice.
Sources: Hospital Israelita A. Einstein and others. [Learn more](#)

Feedback

Painel de
conhecimento

Knowledge panel

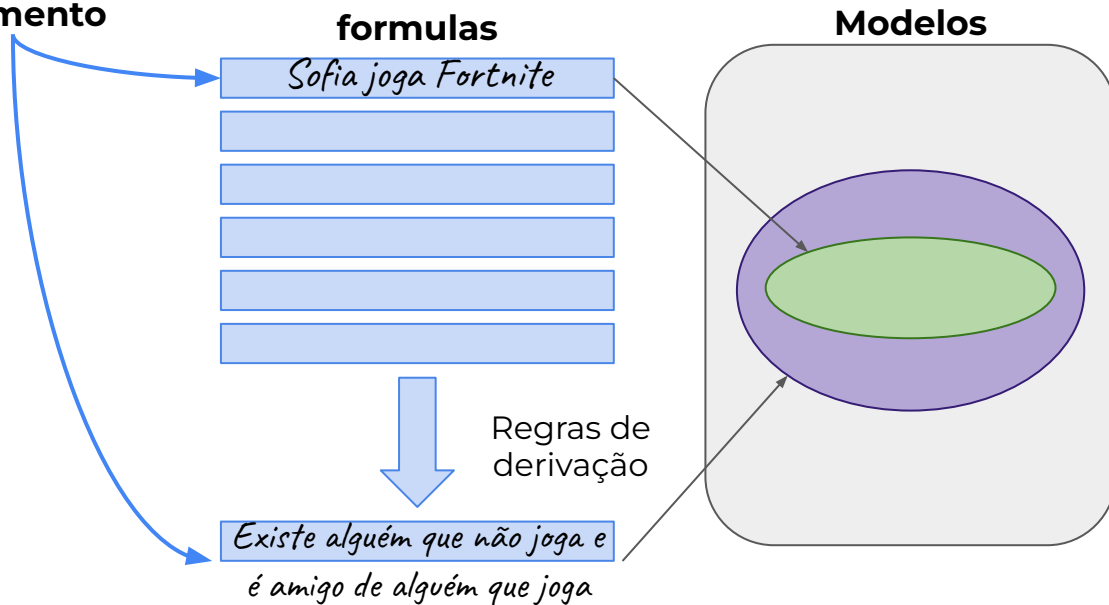
Lógica

- Pensar racionalmente requer

- **Representar conhecimento**

- Derivar soluções

Ontologias



Ontologia

- Ramo da filosofia que estuda a natureza do ser, da existência e da própria realidade
 - Formalização e estrutura da realidade
 - Sistema que define primitivas de entidades e relacionamentos
 - Define significados e restrições, garantindo a consistência da aplicação de primitivas

Ontologia

THE GENE ONTOLOGY RESOURCE

The mission of the GO Consortium is to develop a comprehensive, **computational model of biological systems**, ranging from the molecular to the organism level, across the multiplicity of species in the tree of life.

The Gene Ontology (GO) knowledgebase is the world's largest source of information on the functions of genes. **This knowledge is both human-readable and machine-readable,** and is a foundation for computational analysis of large-scale molecular biology and genetics experiments in biomedical research.

Search GO term or Gene Product in AmiGO ...



☒ Any ☐ Ontology ☐ Gene Product

GO Enrichment Analysis ?

Powered by PANTHER

Your gene IDs here...

biological process



Homo sapiens

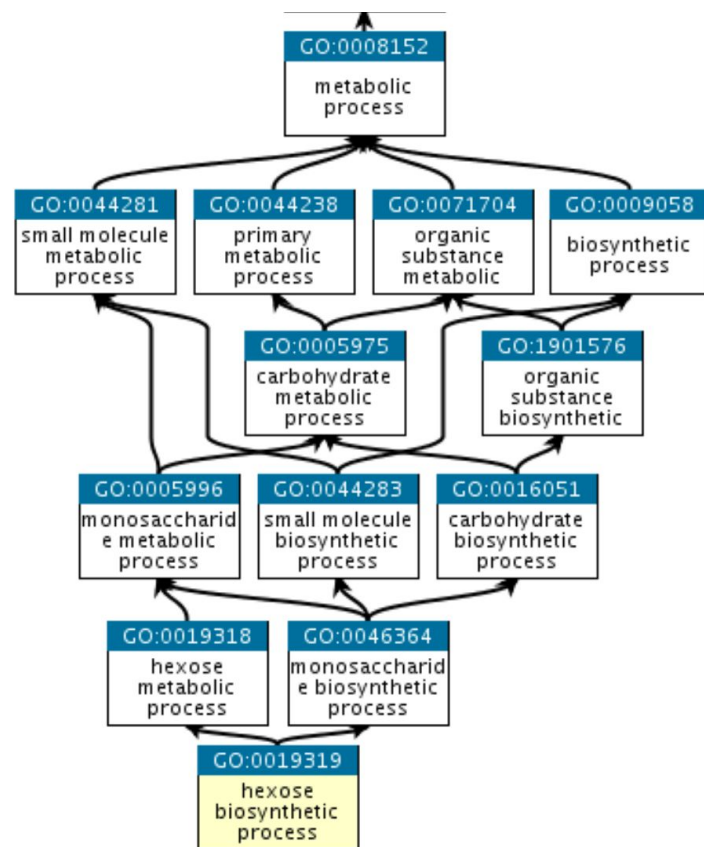
Examples

Launch

Hint: can use UniProt ID/AC, Gene Name, Gene Symbols, MOD IDs

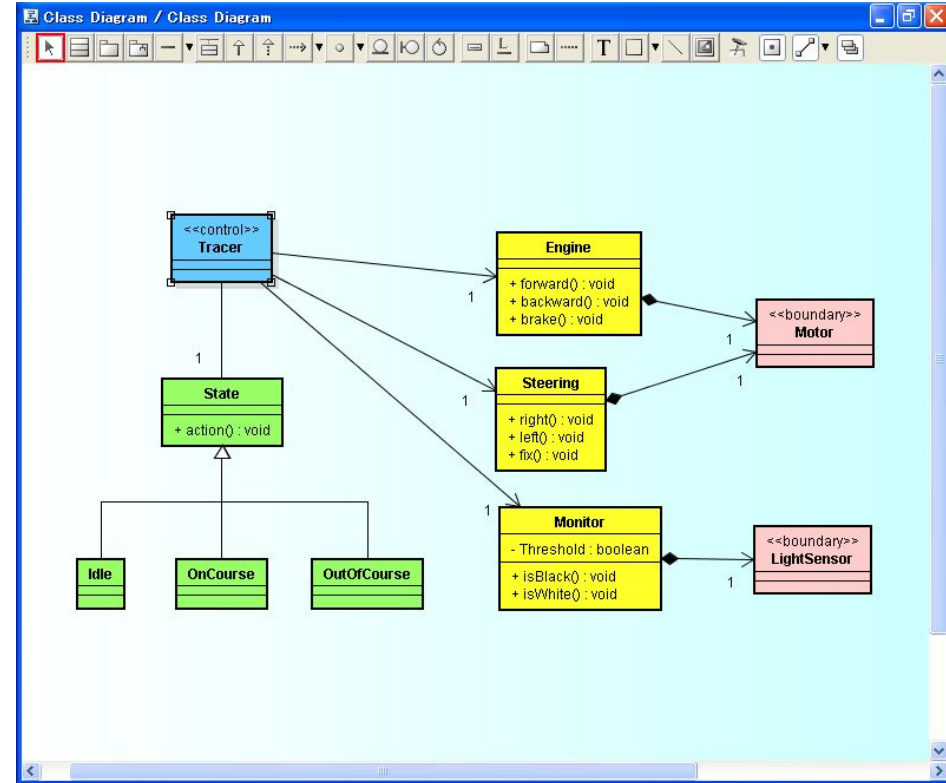
Ontologia

- Exemplo: GeneOntology
 - Função molecular
 - Componente celular
 - Processo biológico



Ontologia

- Unified Modeling Language
 - *Machine-readable information*



Ontologia

- Exemplo: iCalendar
 - Internet Calendaring and Scheduling Core Object Specification (.ics)



 mila.laranjeira@gmail.com.ics - Bloco de Notas

Arquivo Editar Formatar Exibir Ajuda

```
BEGIN:VCALENDAR
PRODID:-//Google Inc//Google Calendar 70.9054//EN
VERSION:2.0
CALSCALE:GREGORIAN
METHOD:PUBLISH
X-WR-CALNAME:mila.laranjeira@gmail.com
X-WR-TIMEZONE:America/Sao_Paulo
BEGIN:VTIMEZONE
TZID:America/Sao_Paulo
X-LIC-LOCATION:America/Sao_Paulo
BEGIN:STANDARD
TZOFFSETFROM:-0300
TZOFFSETTO:-0300
TZNAME:-03
DTSTART:19700101T000000
END:STANDARD
END:VTIMEZONE
BEGIN:VTIMEZONE
TZID:America/Bahia
```

Níveis de Ontologia

C

CACADA – Gargalhada. No dialecto noroiteiro da Índia também se diz *cacada*, com a mesma significação.

CACAI – Zarolho; vesgo.

CACHÍ – Morder; mastigar. Cortar com os dentes.

D

DALE – Sovar; bater; agredir. Ingerir; tomar com gosto. Arriscar (no jogo). *Pegá dale*: dar uma sova. «Dar-lhe»

science noun

Synonyms for science

[knowledge](#), [lore](#), [wisdom](#)

Near Antonyms for science

[ignorance](#), [inexperience](#), [innocence](#), [nescience](#)

$$(\forall x)(A \rightarrow B) \rightarrow (\forall x)A \rightarrow (\forall x)B$$

Glossário

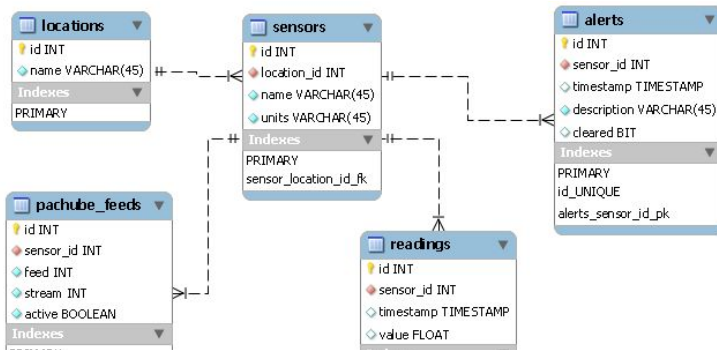
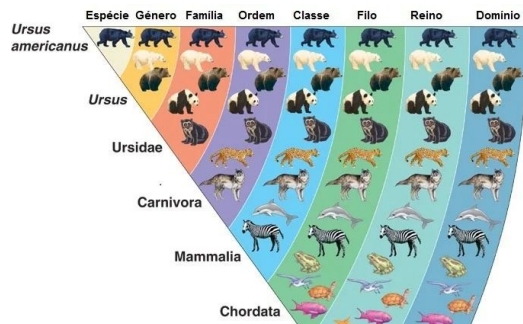
Dicionário de
Sinônimos

Teoria
Axiomática

Precisão da Ontologia

Taxonomia

DB/OO



Lógica

Sintaxe

- Quais expressões e símbolos são válidos?

Semântica

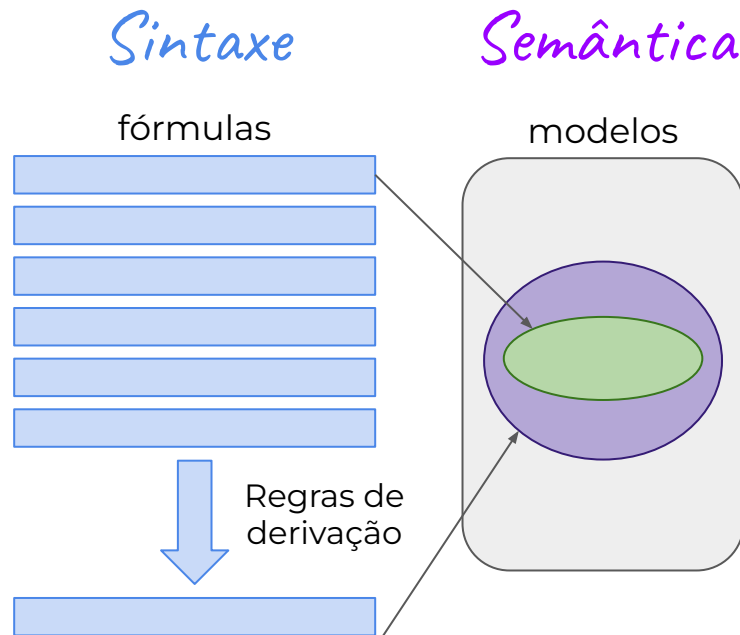
- O que essas expressões significam?

Ex1: Sintaxe diferente, mesma semântica

$$2 + 3 \Leftrightarrow 3 + 2$$

Ex2: Mesma sintaxe, semântica diferente

$$3 / 2 \text{ (Python 2.7)} \not\Leftrightarrow 3 / 2 \text{ (Python 3)}$$



Lógica

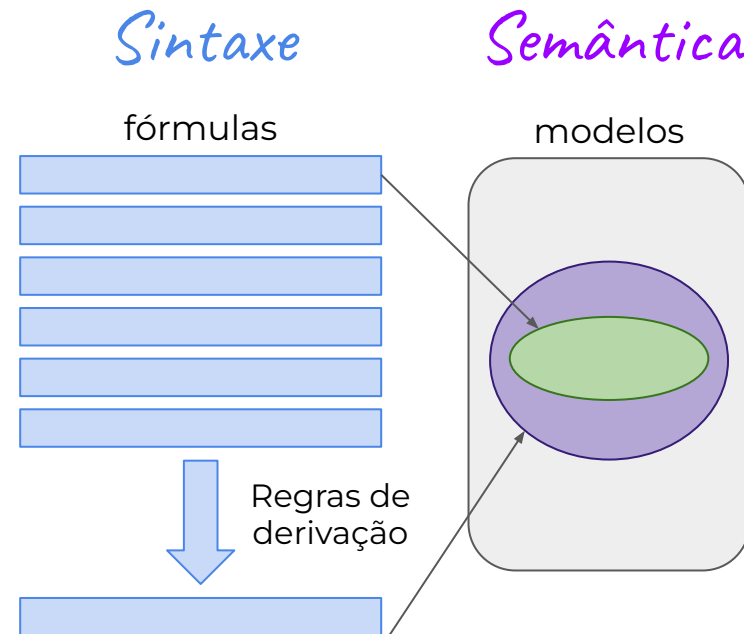
Conheceremos dois tipos

- **Lógica Proposicional**

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

- **Lógica de 1º Ordem**

$$\forall x \exists y P(x,y)$$



Lógica Proposicional

Sintaxe

- Símbolos de proposição: A, B, C
- Conectivos lógicos: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

- Fórmulas

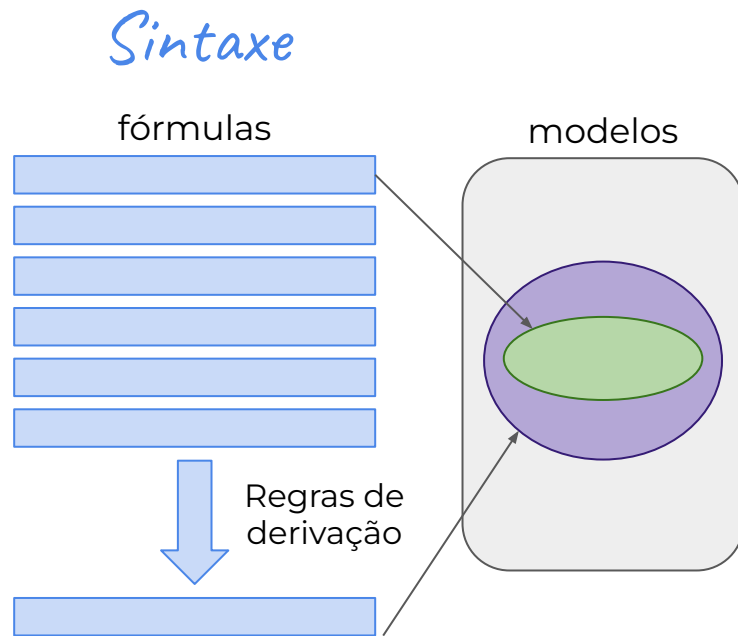
Negação: $\neg A$

Conjunção: $A \wedge B$

Disjunção: $A \vee B$

Implicação: $A \rightarrow B$

Equivalência: $A \leftrightarrow B$



Lógica Proposicional

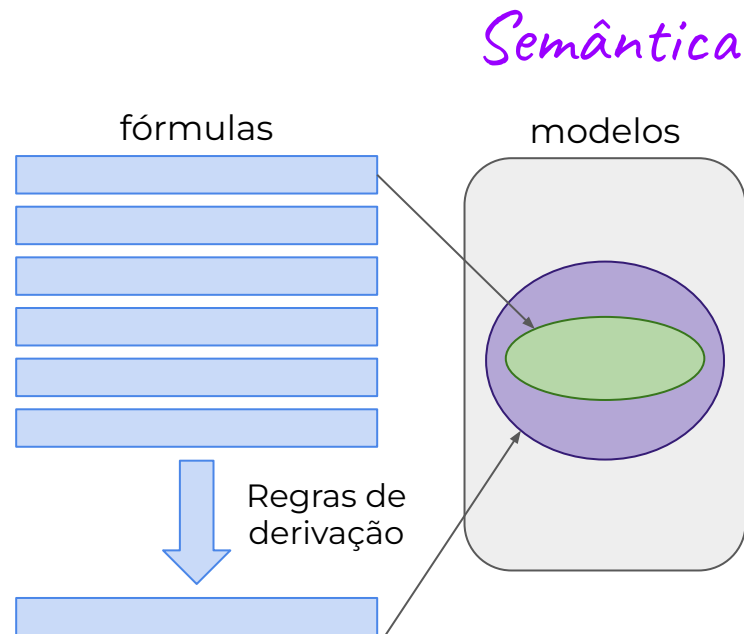
Semântica

- Um modelo é uma associação de **valor-verdade** a símbolos de proposição

└→ **V ou F**

Exemplo

- Fórmula $f = (\neg A \wedge B) \leftrightarrow C$
- Modelo $w = \{A:1, B:1, C:0\}$



Lógica Proposicional

Semântica

- Um modelo é uma associação de **valor-verdade** a símbolos de proposição

└→ **V ou F**

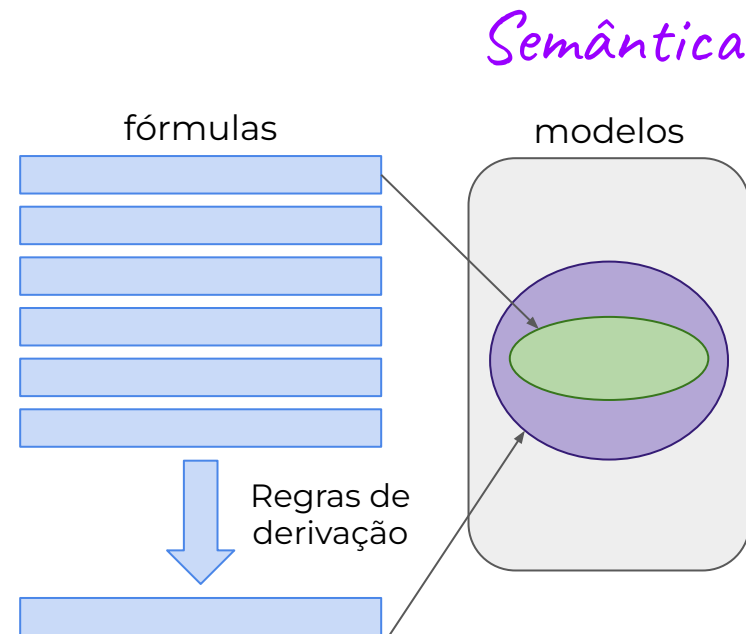
Exemplo

- Fórmula $f = (\neg A \wedge B) \leftrightarrow C$
- Modelo $w = \{A:1, B:1, C:0\}$

Possíveis modelos:

2^3 combinações

$\{A:0, B:0, C:0\}$
$\{A:0, B:0, C:1\}$
...
$\{A:1, B:1, C:1\}$



Lógica Proposicional

Semântica

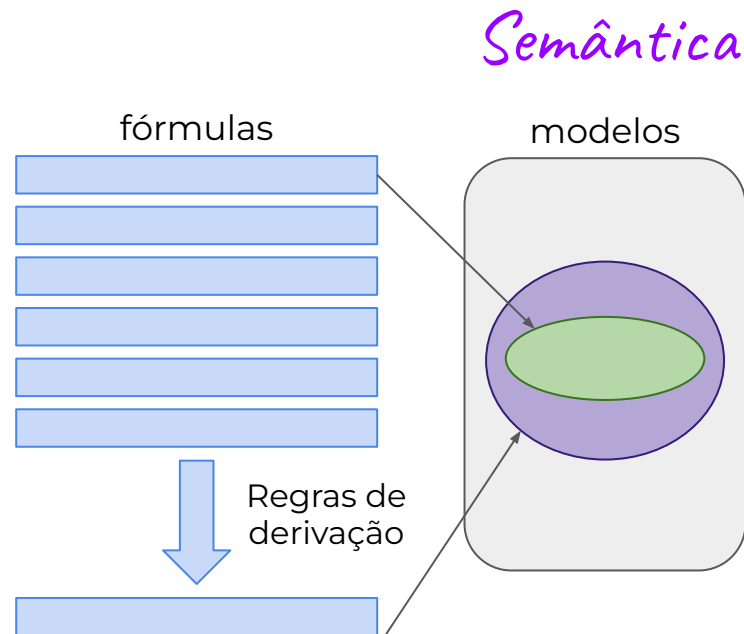
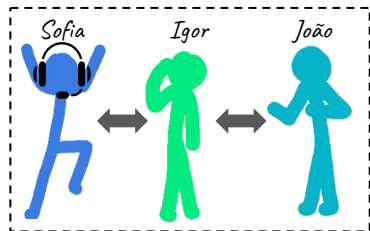
- Na prática, vamos reduzir afirmações a proposições e lhes considerar verdadeiras.

Sofia joga Fortnite: A

Sofia é amiga de Igor: B

Sofia é amiga de João: C

$w = \{A: 1, B: 1, C: 0\}$



Lógica Proposicional

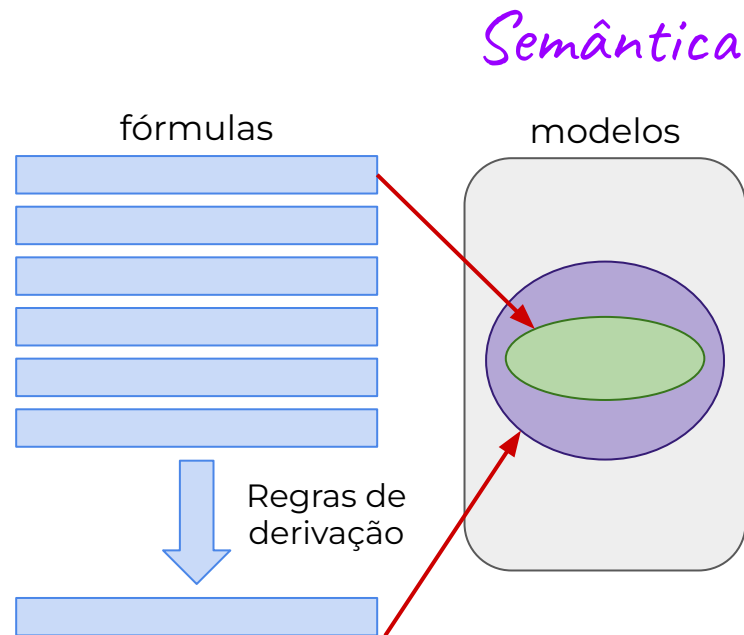
Semântica

- Um modelo é uma associação de **valor-verdade** a símbolos de proposição

└→ **V ou F**

Exemplo

- Fórmula $f = (\neg A \wedge B) \leftrightarrow C$
- Modelo $w = \{A:1, B:1, C:0\}$
- Função de Interpretação $I(f, w)$ retorna
 - **V** (1) se w satisfaz f
 - **F** (0) se w não satisfaz f



Lógica Proposicional

Semântica

- Um modelo é uma associação de valor-verdade a símbolos de proposição
 - Caso base (f é uma proposição) : $I(f, w) = w(f)$
- Caso recursivo. Para qualquer par de fórmulas f e g, pode-se definir:

Com n proposições,
temos 2^n possíveis
modelos

$I(f, w)$	$I(g, w)$	$I(\neg f, w)$	$I(f \vee g, w)$	$I(f \wedge g, w)$	$I(f \rightarrow g, w)$	$I(f \leftrightarrow g, w)$
0	0					
0	1					
1	0					
1	1					

Lógica Proposicional

Semântica

- Algumas propriedades para relembrar

$$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge$$

$$(\alpha \vee \beta) \equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee$$

$$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge$$

$$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee$$

$$\neg(\neg\alpha) \equiv \alpha \quad \text{double-negation elimination}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) \quad \text{implication elimination}$$

$$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination}$$

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

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

$$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee$$

$$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge$$

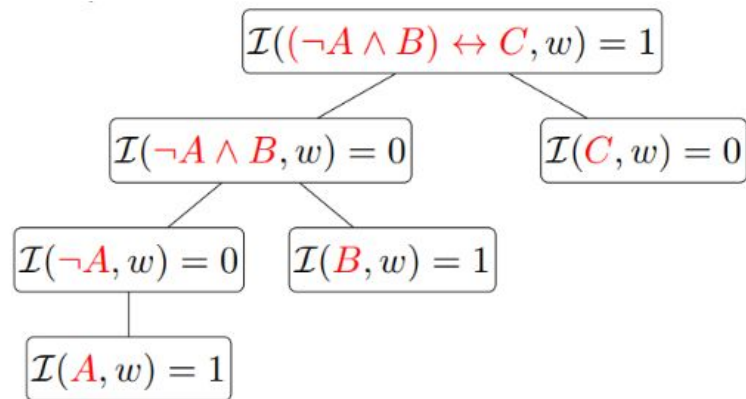
Lógica Proposicional

- Fórmula $(\neg A \wedge B) \leftrightarrow C$
- Modelo $w = \{A:1, B:1, C:0\}$
- Função de Interpretação $I(f, w)$ retorna
 - V (1) se w satisfaz f
 - F (0) se w não satisfaz f

$$I((\neg A \wedge B) \leftrightarrow C, w) = 1$$

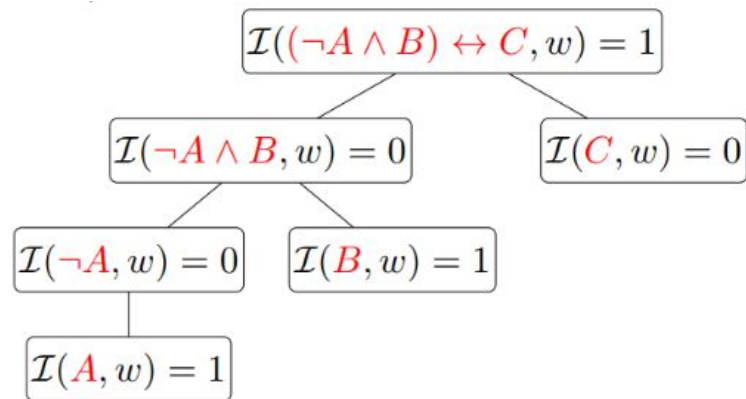
Lógica Proposicional

- Fórmula $(\neg A \wedge B) \leftrightarrow C$
- Modelo $w = \{A:1, B:1, C:0\}$
- Função de Interpretação $I(f, w)$ retorna
 - **V (1)** se w satisfaz f
 - **F (0)** se w não satisfaz f



Lógica Proposicional

- Fórmula $(\neg A \wedge B) \leftrightarrow C$
- Modelo $w = \{A:1, B:1, C:0\}$
- Função de Interpretação $I(f, w)$ retorna
 - **V** (1) se w satisfaz f
 - **F** (0) se w não satisfaz f



function PL-TRUE?(α ,model) **returns** true or false

if α is a symbol **then return** Lookup(α , model)

if Op(α) = \neg **then return** not(PL-TRUE?(Arg1(α),model))

if Op(α) = \wedge **then return** and(PL-TRUE?(Arg1(α),model),
PL-TRUE?(Arg2(α),model))

etc.

Base de conhecimento

- *Knowledge base*
- Fórmulas + ontologia de tipos e relações
- Intersecção de um conjunto de fórmulas
- Exl: KB = {chuva, chuva \rightarrow molhado}

		Wet	
		0	1
Rain	0		
	1		

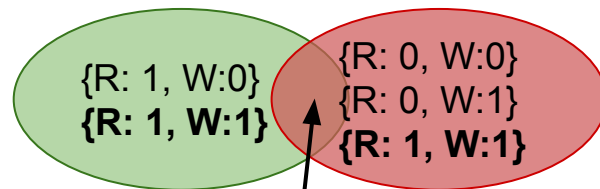
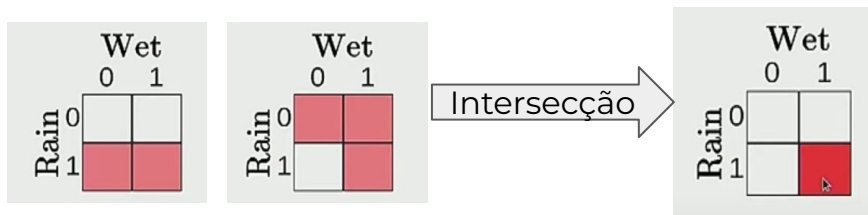


		Wet	
		0	1
Rain	0		
	1		



Base de conhecimento

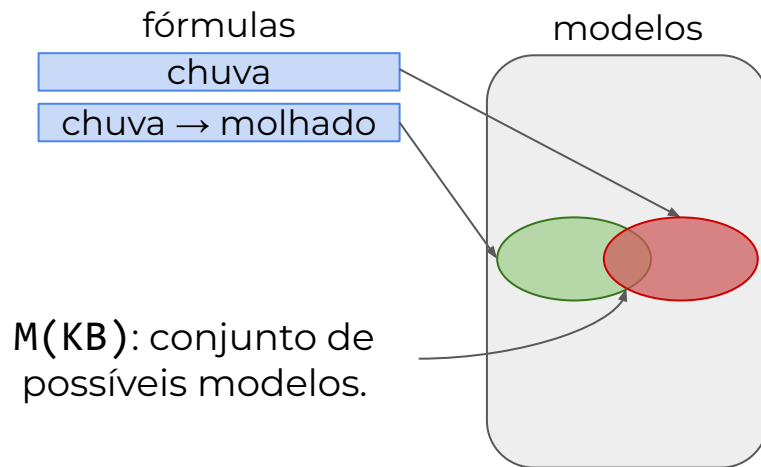
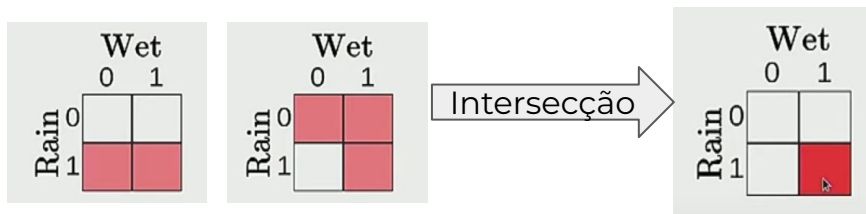
- *Knowledge base*
- Fórmulas + ontologia de tipos e relações
- Intersecção de um conjunto de fórmulas
- Exl: KB = {chuva, chuva \rightarrow molhado}



$M(KB)$: conjunto de possíveis modelos.

Base de conhecimento

- *Knowledge base*
- Fórmulas + ontologia de tipos e relações
- Intersecção de um conjunto de fórmulas
- Exl: $KB = \{chuva, chuva \rightarrow molhado\}$



$M(KB)$: conjunto de possíveis modelos.

- Quanto mais fórmulas forem adicionadas, menor é o espaço de possíveis modelos

Até agora...

- Ontologia: Definição de primitivas (tipos e relações)
 - Sintaxe: conjunto de símbolos e expressões válidas
 - Semântica: significado de cada símbo (proposição com valor-verdade; função de interpretação)
- Base de conhecimento: intersecção de fórmulas conhecidas; espaço de possíveis modelos.

Operações

Proposição A: está chovendo

Tell(A): está chovendo

Possíveis respostas

- Já sabia disso
- Não acredito nisso
- Aprendi algo novo

Ask(A): está chovendo?

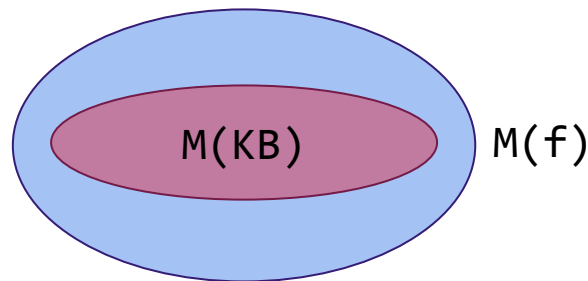
Possíveis respostas

- Sim
- Não
- Não sei

Entailment

- Implicação, vínculo
- A informação de f já era conhecida.

$KB \models f$ se e somente se $M(KB) \subseteq M(f)$

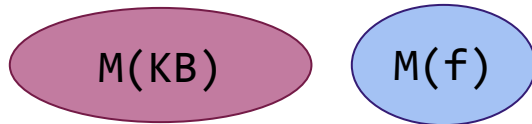


- Exemplo: $(\text{está chovendo} \wedge \text{está frio}) \models (\text{está frio})$
 - $\text{tell}(\text{está chovendo} \wedge \text{está frio})$
Aprendi algo novo
 - $\text{tell}(\text{está frio})$
Já sabia disso

Contradição

- f contradiz o que já era conhecido

KB contr. f se e somente se $M(KB) \cap M(f) = \emptyset$

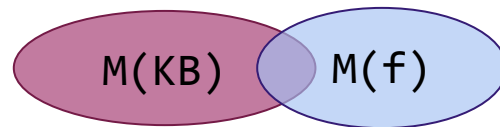


- Exemplo: (está chovendo \wedge está frio) contr. (\neg está frio)
 - tell(está chovendo \wedge está frio)
Aprendi algo novo
 - tell(\neg está frio)
Não acredito nisso

Contingência

- f adiciona informação à base de conhecimento

$$\{\emptyset \subsetneq M(KB) \cap M(f) \subsetneq M(KB)\}$$



- Exemplo:
 - tell(está chovendo)
Aprendi algo novo
 - tell(está frio)
Aprendi algo novo

Operações

Proposição A: está chovendo

Tell(A): está chovendo

Possíveis respostas

- Já sabia disso
- Não acredito nisso
- Aprendi algo novo

Ask(A): está chovendo?

Possíveis respostas

- Sim
- Não
- Não sei

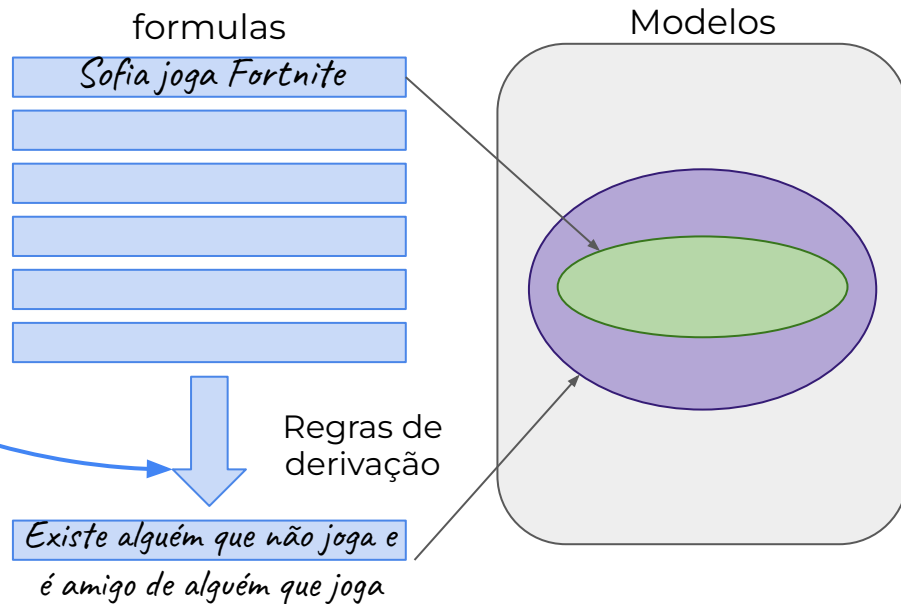
PropositionalLogic.ipynb

Lógica Proposicional

- Pensar racionalmente requer
 - Representar conhecimento
 - **Derivar soluções**

Model checking

Regras de Inferência

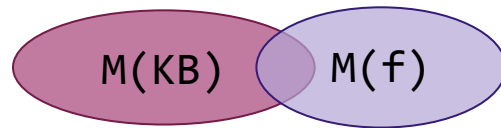
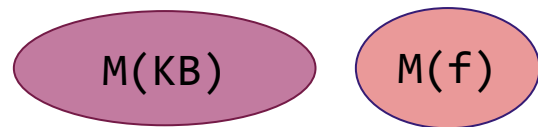
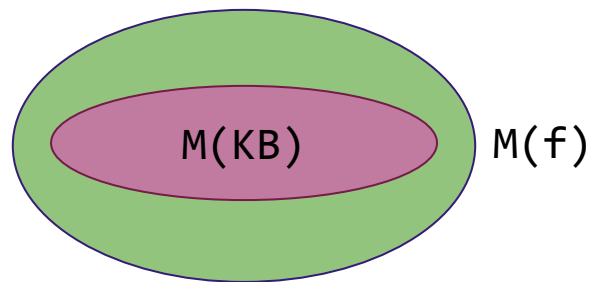
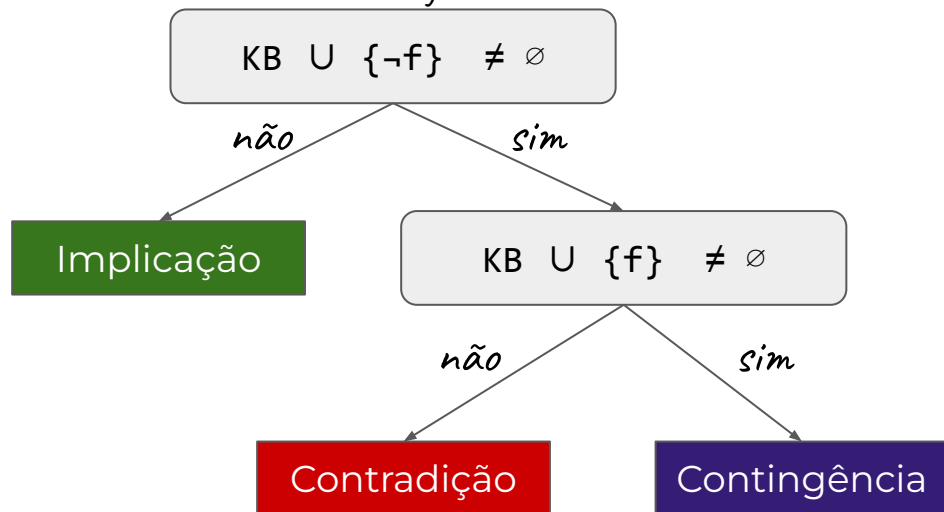


Satisfatibilidade (*satisfação*)

- Propriedade:

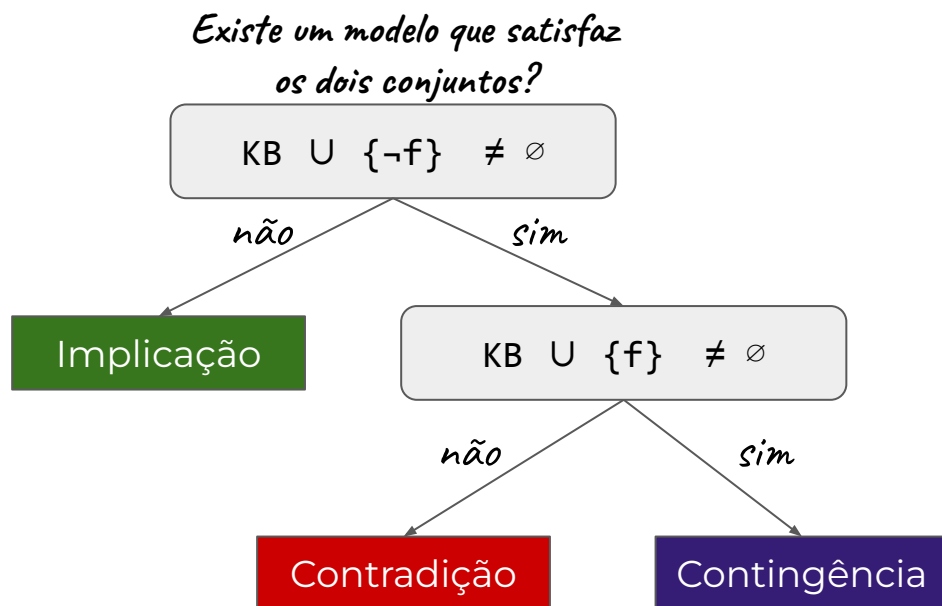
KB *implica* em f se e somente se *contradiz* $\neg f$

*Existe um modelo que satisfaz
os dois conjuntos?*



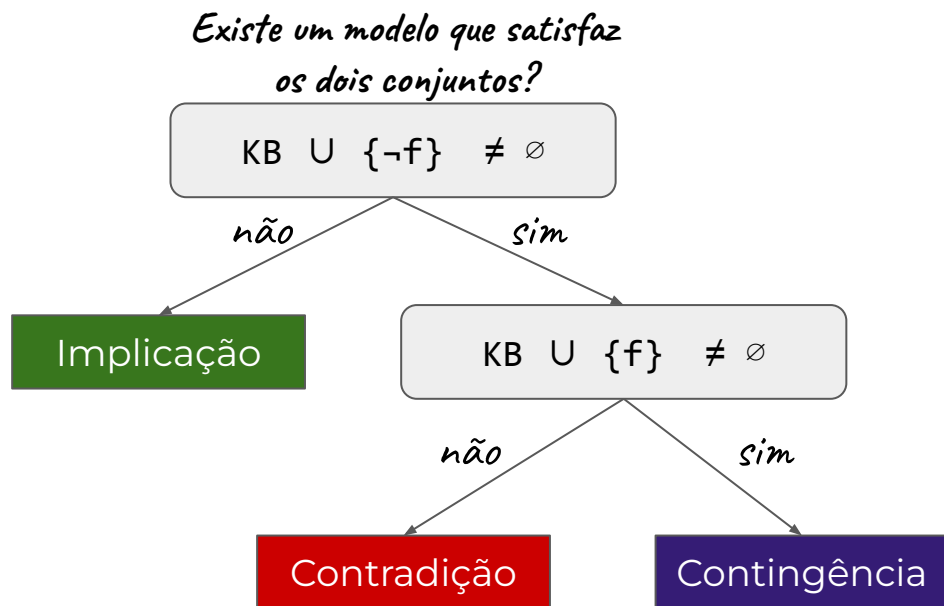
Satisfatibilidade (*satisfação*)

- Como saber se tal modelo existe?



Satisfatibilidade (*satisfação*)

- Como saber se tal modelo existe?



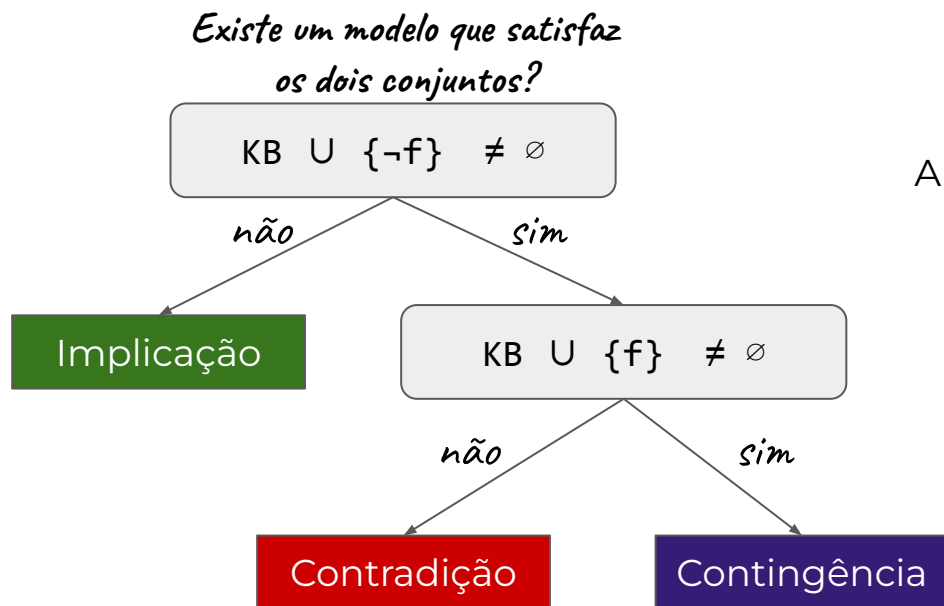
Teste os possíveis modelos!

- Model checking
- **NP-completo**

PropositionalLogic.ipynb

Satisfatibilidade (*satisfação*)

- Como saber se tal modelo existe?



Teste os possíveis modelos!

- *Model checking*
- **NP-completo** (SAT)

Algoritmos populares (SAT Solvers)

- **DPLL**: backtracking + poda
- **WalkSat**: busca local aleatória

Regras de Inferência

- A alternativa à checagem de todos os possíveis modelos é a **prova de teoremas**
- A partir de regras de inferência pré-definidas é possível partir de **premissas** a **conclusões**
- Se $f_1, f_2, f_3, \dots, f_k$ e g são fórmulas, uma regra de inferência é definida por:

$$\frac{f_1, f_2, f_3, \dots, f_k}{g}$$

(premissas) (conclusão)

Regras de Inferência

- Se $f_1, f_2, f_3, \dots, f_k$ e g são fórmulas, uma regra de inferência é definida por:

$$\frac{f_1, f_2, f_3, \dots, f_k}{g} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

- **Modus ponens:** para quaisquer símbolos p e q

$$\frac{p, p \rightarrow q}{q} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

Regras de Inferência

- Se $f_1, f_2, f_3, \dots, f_k$ e g são fórmulas, uma regra de inferência é definida por:

$$\frac{f_1, f_2, f_3, \dots, f_k}{g}$$

(premissas) (conclusão)

- **Modus ponens:** para quaisquer símbolos p e q

horn clauses

$$\frac{p, p \rightarrow q}{q}$$

(premissas) (conclusão)

Se aplica a cláusulas definidas

Conjunção de símbolos \rightarrow símbolo

$$\frac{f_1 \wedge f_2 \wedge \dots \wedge f_k \rightarrow}{g}$$

(premissas) (conclusão)

Regras de Inferência

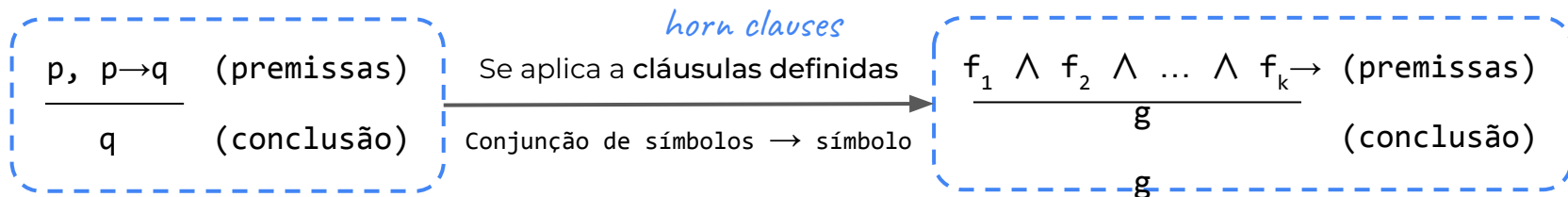
Modus Ponens — *Horn clauses*

chifre

conjunção:

\wedge

- Modus ponens: para quaisquer símbolos p e q



Modus Ponens (forward inference/chaining)

- Exemplo

kb = {chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio}

chuva	molhado	escorregadio
1	?	?

p, p \rightarrow q	(premissas)
<hr/>	
q	(conclusão)

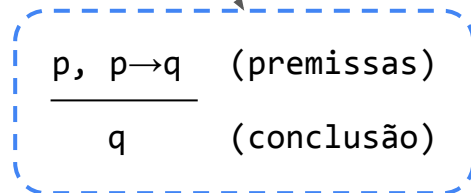
Modus Ponens (forward inference/chaining)

- Exemplo

$kb = \{chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio\}$

- Repita até que nenhuma mudança aconteça em kb
 - Escolha um conjunto de fórmulas $f_1, f_2, \dots, f_k \in kb$
 - Se existe uma regra correspondente ao modus ponens, adicione a conclusão a kb

chuva	molhado	escorregadio
1	?	?



Modus Ponens (forward inference/chaining)

- Exemplo

kb = {chuva, chuva→molhado, molhado→escorregadio}

- Repita até que nenhuma mudança aconteça em kb
 - Escolha um conjunto de fórmulas $f_1, f_2, \dots, f_k \in kb$
 - Se existe uma regra correspondente ao modus ponens, adicione a conclusão a kb

kb = {chuva, chuva→molhado, molhado→escorregadio,
molhado}

chuva	molhado	escorregadio
1	?	?
1	1	?

p, p→q	(premissas)
<hr/>	
q	(conclusão)

Modus Ponens (forward inference/chaining)

- Exemplo

kb = {chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio}

- Repita até que nenhuma mudança aconteça em kb
 - Escolha um conjunto de fórmulas $f_1, f_2, \dots, f_k \in kb$
 - Se existe uma regra correspondente ao modus ponens, adicione a conclusão a kb

kb = {chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio, molhado}

kb = {chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio, molhado, escorregadio}

chuva	molhado	escorregadio
1	?	?
1	1	?
1	1	1

p, p \rightarrow q	(premissas)
q	(conclusão)

Modus Ponens (forward inference/chaining)

- Exemplo

$kb = \{chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio\}$

- Repita até que nenhuma mudança aconteça em kb
 - Escolha um conjunto de fórmulas $f_1, f_2, \dots, f_k \in kb$
 - Se existe uma regra correspondente ao modus ponens, adicione a conclusão a kb

$kb = \{chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio, molhado\}$

$kb = \{chuva, chuva \rightarrow molhado, molhado \rightarrow escorregadio, molhado, escorregadio\}$

chuva	molhado	escorregadio
1	?	?
1	1	?
1	1	1

$p, p \rightarrow q$	(premissas)
<hr/>	
q	(conclusão)

kb **deriva/prova** f , ou $kb \vdash f$, se f é adicionado a kb

Modus Ponens (forward inference/chaining)

- Roda em tempo linear graças a alguns truques
 - Cada símbolo sabe em que regras ele aparece
 - Cada regra registra quantas premissas ainda não foram satisfeitas

```
function PL-FC-ENTAILS?(KB, q) returns true or false
  count ← a table, where count[c] is the number of symbols in c's premise
  inferred ← a table, where inferred[s] is initially false for all s
  agenda ← a queue of symbols, initially symbols known to be true in KB
  while agenda is not empty do
    p ← Pop(agenda)
    if p = q then return true
    if inferred[p] = false then
      inferred[p] ← true
      for each clause c in KB where p is in c.premise do
        decrement count[c]
        if count[c] = 0 then add c.conclusion to agenda
  return false
```

Inferência: Propriedades

- **São** (*sound*, sanidade): nada mais que a verdade

$$\{f : \text{KB} \vdash f\} \subseteq \{f : \text{KB} \models f\}$$

Tudo que deriva é uma implicação

- **Completo**: toda a verdade

$$\{f : \text{KB} \vdash f\} \supseteq \{f : \text{KB} \models f\}$$

Deriva todas as implicações

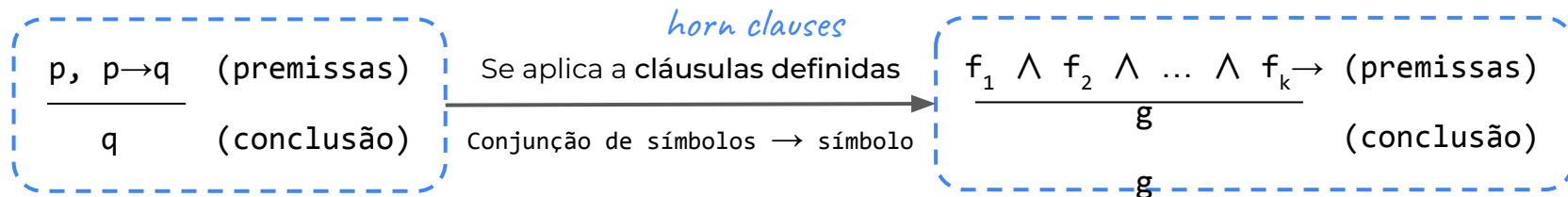


Modus Ponens

Não é loucura
sonhar

Se kb tem apenas cláusulas definidas,
forward inference com modus ponens
é **são** e **completo**.

- Modus ponens: para quaisquer símbolos p e q



Modus Ponens

- Exemplo

$kb = \{chuva, chuva \vee molhado \rightarrow escorregadio\}$

$f = escorregadio$

chuva	molhado	escorregadio
1	?	?

- Semanticamente, $kb \models f$ (kb implica em f)
- Sintaticamente, não conseguimos derivar com Modus Ponens

$p, p \rightarrow q$	(premissas)
<hr/>	
q	(conclusão)

Modus Ponens

- Exemplo

$kb = \{chuva, chuva \vee molhado \rightarrow escorregadio\}$

$f = escorregadio$

chuva	molhado	escorregadio
1	?	?

- Semanticamente, $kb \models f$ (kb implica em f)
- Sintaticamente, não conseguimos derivar com Modus Ponens

Se alguma cláusula não é definida, forward inference com modus ponens é são mas **incompleto**.

Sem chifres, modus ponens é incompleto :(



$p, p \rightarrow q$	(premissas)
<hr/>	
q	(conclusão)

Regras de Inferência

- Se $f_1, f_2, f_3, \dots, f_k$ e g são fórmulas, uma regra de inferência é definida por:

$$\frac{f_1, f_2, f_3, \dots, f_k}{g} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

- **Modus Ponens** reescrito

$$\frac{p, p \rightarrow q}{q} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

Forma normal conjuntiva

$$\frac{p, \neg p \vee q}{q} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

p	q	$p \rightarrow q$	$\neg p \vee q$
0	0	1	1
0	1	1	1
1	0	0	0
1	1	1	1

Regras de Inferência

- Se $f_1, f_2, f_3, \dots, f_k$ e g são fórmulas, uma regra de inferência é definida por:

$$\frac{f_1, f_2, f_3, \dots, f_k}{g} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

CNF (conjunctive normal form)

Forma normal conjuntiva

$$\frac{p, p \rightarrow q}{q} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

$$\frac{p, \neg p \vee q}{q} \quad \begin{array}{l} \text{(premissas)} \\ \text{(conclusão)} \end{array}$$

- Expressões proposicionais podem ser convertidas para CNF
- Existe uma regra mais robusta que atua em expressões CNF
 - Resolução

Forma Normal Conjuntiva

- Conjunção de uma ou mais cláusulas, onde cada cláusula é uma disjunção de literais
 - ANDs de vários ORs
 - Negações apenas em literais

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

cláusula *literal*
(proposição)

Forma Normal Conjuntiva

- Conjunção de uma ou mais cláusulas, onde cada cláusula é uma disjunção de literais
 - ANDs de vários ORs
 - Negações apenas em literais

$$A \leftrightarrow (B \vee C)$$

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

Forma Normal Conjuntiva

- Conjunção de uma ou mais cláusulas, onde cada cláusula é uma disjunção de literais
 - ANDs de vários ORs
 - Negações apenas em literais

$$A \leftrightarrow (B \vee C)$$

$$(A \rightarrow (B \vee C)) \wedge ((B \vee C) \rightarrow A) \text{ \# elimina equivalência}$$

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

Forma Normal Conjuntiva

- Conjunção de uma ou mais cláusulas, onde cada cláusula é uma disjunção de literais
 - ANDs de vários ORs
 - Negações apenas em literais

$$A \leftrightarrow (B \vee C)$$

$$(A \rightarrow (B \vee C)) \wedge ((B \vee C) \rightarrow A) \quad \# \text{ elimina equivalência}$$

$$(\neg A \vee B \vee C) \wedge (\neg(B \vee C) \vee A) \quad \# \text{ elimina implicação}$$

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

p	q	$p \rightarrow q$	$\neg p \vee q$
0	0	1	1
0	1	1	1
1	0	0	0
1	1	1	1

Forma Normal Conjuntiva

- Conjunção de uma ou mais cláusulas, onde cada cláusula é uma disjunção de literais
 - ANDs de vários ORs
 - Negações apenas em literais

$$A \leftrightarrow (B \vee C)$$

Atenção para o OR convertido em AND

$$(A \rightarrow (B \vee C)) \wedge ((B \vee C) \rightarrow A) \text{ \# elimina equivalência}$$

$$(\neg A \vee B \vee C) \wedge (\neg(B \vee C) \vee A) \text{ \# elimina implicação}$$

$$(\neg A \vee B \vee C) \wedge ((\neg B \wedge \neg C) \vee A) \text{ \# regra de De Morgan (distribui o not)}$$

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

Forma Normal Conjuntiva

- Conjunção de uma ou mais cláusulas, onde cada cláusula é uma disjunção de literais
 - ANDs de vários ORs
 - Negações apenas em literais

$$A \leftrightarrow (B \vee C)$$

$$(A \rightarrow (B \vee C)) \wedge ((B \vee C) \rightarrow A) \text{ \# elimina equivalência}$$

$$(\neg A \vee B \vee C) \wedge (\neg(B \vee C) \vee A) \text{ \# elimina implicação}$$

$$(\neg A \vee B \vee C) \wedge ((\neg B \wedge \neg C) \vee A) \text{ \# regra de De Morgan (distribui o not)}$$

$$(\neg A \vee B \vee C) \wedge (\neg B \vee A) \wedge (\neg C \vee A) \text{ \# regra distributiva}$$

Forma Normal Conjuntiva

- Remover equivalência e implicações

$$\frac{f \leftrightarrow g}{(f \rightarrow g) \wedge (g \rightarrow f)}$$

$$\frac{f \rightarrow g}{\neg f \vee g}$$

- Negações apenas em literais

$$\frac{\neg(f \wedge g)}{\neg f \vee \neg g}$$

$$\frac{\neg(f \vee g)}{\neg f \wedge \neg g}$$

$$\frac{\neg \neg f}{f}$$

- Distribuir disjunções e conjunções

$$\frac{f \vee (g \wedge h)}{(f \vee g) \wedge (f \vee h)}$$

Resolução

- Regra de inferência mais robusta capaz de derivar a partir de expressões CNF
- Implementada na demo de Lógica Proposicional
- Resolve também para Lógica de 1ª Ordem (estamos chegando lá)

Resolução

Ela é mais complicada, mas em sua essência:

- A partir de duas cláusulas contendo o mesmo literal (com valores opostos)
- É possível derivar a unificação das cláusulas “cancelando” aquele literal
- Esse processo é aplicado iterativamente

Premissa 1: $\neg P \vee Q$

Premissa 2: P

Conclusão: Q

Resolução

Ela é mais complicada, mas em sua essência:

- A partir de duas cláusulas contendo o mesmo literal (com valores opostos)
- É possível derivar a unificação das cláusulas “cancelando” aquele literal
- Esse processo é aplicado iterativamente

Premissa 1: $\neg P \vee Q$

Premissa 2: P

Conclusão: Q

https://pt.wikipedia.org/wiki/Princ%C3%ADpio_da_resolu%C3%A7%C3%A3o

A resolução na lógica de primeira ordem

A resolução na [Lógica de primeira ordem](#) condensa os [silogismos](#) tradicionais de inferência lógica. resolução funciona, considere o seguinte exemplo de silogismo da lógica aristotélica:

Todos os gregos são europeus.
Homero é grego.
Então, Homero é europeu.

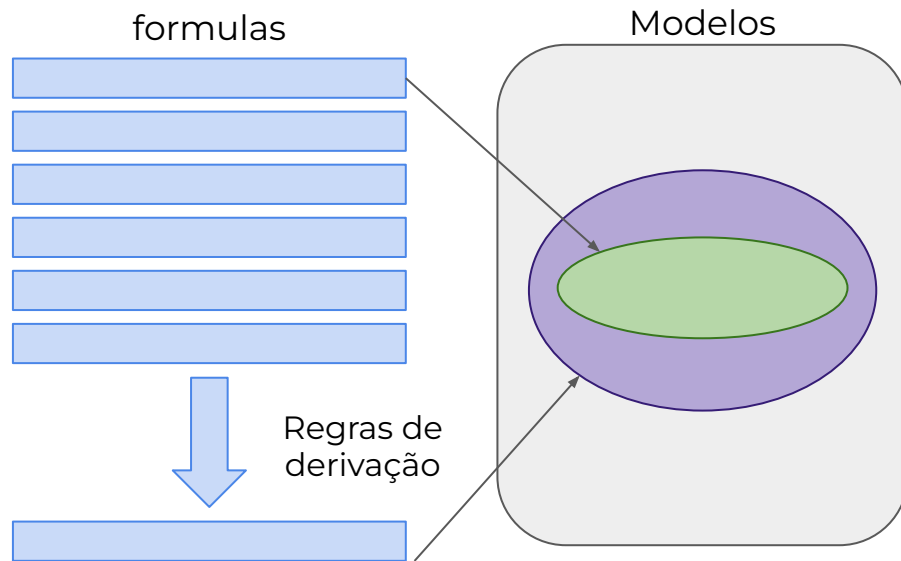
https://youtu.be/_Iz83hfkFds?t=1903

Lógica Proposicional

Para derivar conclusões / responder perguntas a partir de premissas, vimos:

- Model checking
 - SAT solvers
- Regras de inferência
 - Modus Ponens
 - Resolução

Soluções mais eficientes atuam em CNF



Lógica de 1ª Ordem

Quem derrubou a Internet?

- Lógica Proposicional

A: Maria_derrubou_internet

B: Gustavo_derrubou_internet

C: July_derrubou_internet

D: Henrique_derrubou_internet

Lógica de 1ª Ordem

Quem derrubou a Internet?

- Lógica Proposicional

A: Maria_derrubou_internet

B: Gustavo_derrubou_internet

C: July_derrubou_internet

D: Henrique_derrubou_internet

- Lógica de 1ª Ordem

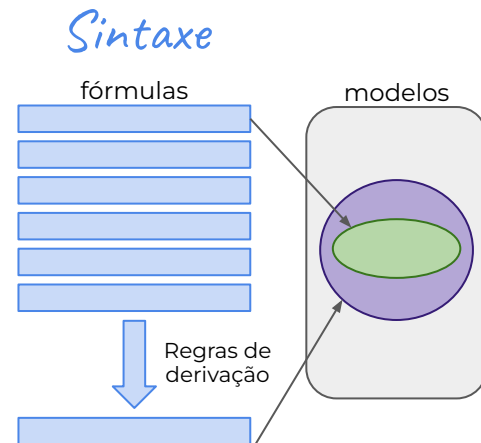
$\exists x \text{ Suspeito}(x) \wedge \text{DerrubouInternet}(x)$

Muito mais expressiva!

Lógica de 1ª Ordem

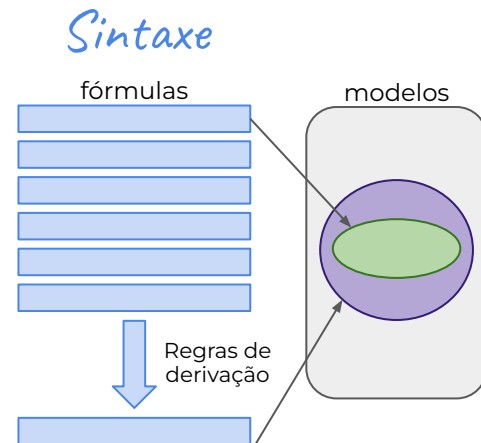
$$\forall x \text{ Estudante}(x) \rightarrow \text{Sabe}(x, \text{matemática})$$

- Termos
 - Constantes: `matemática`
 - Variáveis: `x`
 - Funções*: `F(x)`
- Fórmulas
 - Átomos/Predicados aplicados a termos: `Estudante(x)`
 - Quantificadores (universal e de existência): $\forall \exists$
 - Conectivos: $\neg \wedge \vee \rightarrow \leftrightarrow$



Quantificadores

- Universal (para todo x)
 - Conjunção
 - $\forall xP(x)$ é como $P(A) \wedge P(B) \wedge P(C) \wedge \dots$
- Existência (existe um x)
 - Disjunção
 - $\exists xP(x)$ é como $P(A) \vee P(B) \vee P(C) \vee \dots$

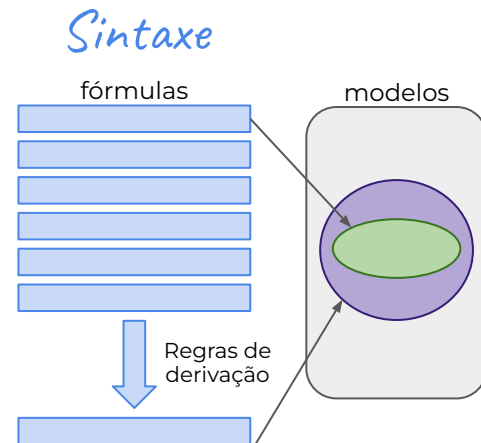


Quantificadores

- Universal (para todo x)
 - $\forall xP(x)$ é como $P(A) \wedge P(B) \wedge P(C) \wedge \dots$
- Existência (existe um x)
 - $\exists xP(x)$ é como $P(A) \vee P(B) \vee P(C) \vee \dots$

Exemplos:

- Todos os estudantes sabem cálculo
- Alguns estudantes sabem cálculo

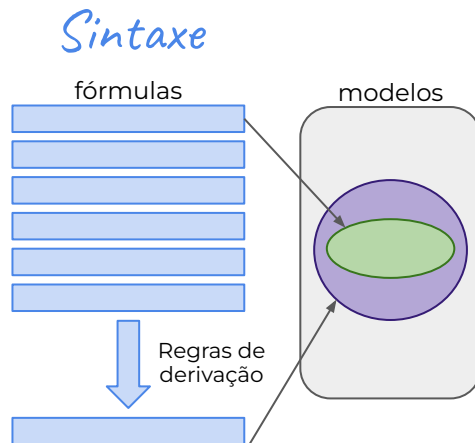


Quantificadores

- Universal (para todo x)
 - $\forall x P(x)$ é como $P(A) \wedge P(B) \wedge P(C) \wedge \dots$
- Existência (existe um x)
 - $\exists x P(x)$ é como $P(A) \vee P(B) \vee P(C) \vee \dots$

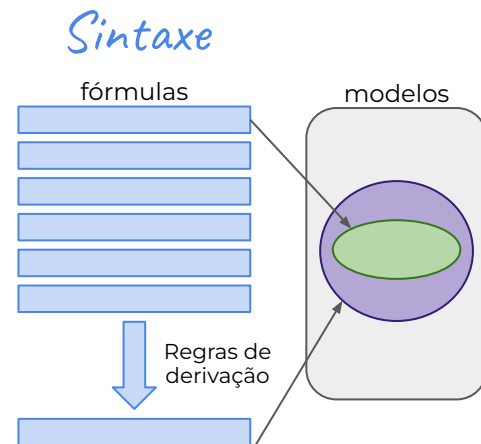
Exemplos:

- Todos os estudantes sabem cálculo $\forall x \text{ Estudante}(x) \rightarrow \text{Sabe}(x, \text{cálculo})$
- Alguns estudantes sabem cálculo $\exists x \text{ Estudante}(x) \wedge \text{Sabe}(x, \text{cálculo})$



Quantificadores

- Universal (para todo x)
 - $\forall x P(x)$ é como $P(A) \wedge P(B) \wedge P(C) \wedge \dots$
- Existência (existe um x)
 - $\exists x P(x)$ é como $P(A) \vee P(B) \vee P(C) \vee \dots$
- Algumas propriedades
 - $\neg \forall x P(x)$ equivale a $\exists x \neg P(x)$
 - $\forall x \exists y (Px, y)$ é diferente de $\exists y \forall x (Px, y)$



Lógica de 1º Ordem

Vamos fazer algumas frases

- Todo mundo conhece alguém
- Existe alguém que todo mundo conhece
- Existe um curso que todos os estudantes fazem
- Todo mundo no Brasil conhece Anitta

Vamos fazer algumas frases

- Todo mundo conhece alguém

$$\forall x \text{ Pessoa}(x) \rightarrow \exists y \text{ Pessoa}(y) \wedge \text{Conhece}(x, y)$$

- Existe alguém que todo mundo conhece

$$\exists y \text{ Pessoa}(y) \wedge \forall x \text{ Pessoa}(x) \rightarrow \text{Conhece}(x, y)$$

- Existe um curso que todos os estudantes fazem

$$\exists y \text{ Curso}(y) \wedge \forall x \text{ Estudante}(x) \rightarrow \text{Cursou}(x, y)$$

- Todo mundo no Brasil conhece Anitta

$$\forall x (\text{Pessoa}(x) \wedge \text{Origem}(x, \text{Brasil})) \rightarrow \text{Conhece}(x, \text{Anitta})$$

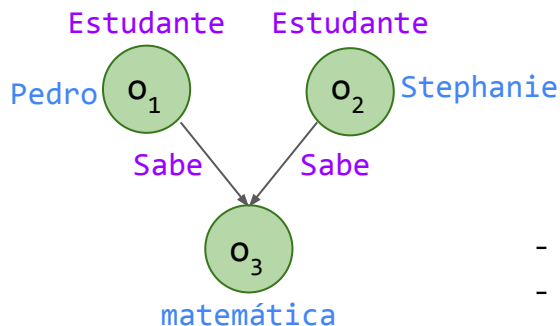
Modelos

- Lógica Proposicional

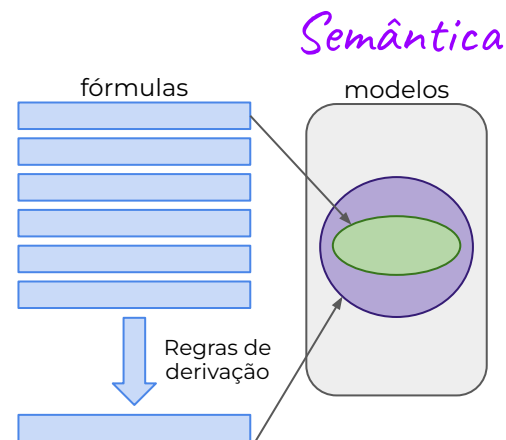
{PedroSabeMatemática: 1, StephanieSabeMatemática: 1}

- Lógica de 1ª Ordem

- Representação em grafos para predicados unários e binários



- Nós são **objetos** rotulados com **constantes**
- Arestas são **predicados** binários
- **Predicados** unários são rótulos adicionais do nó



Modelos: Lógica de 1ª Ordem

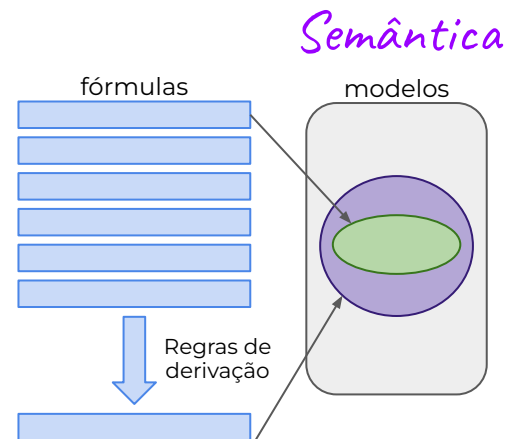
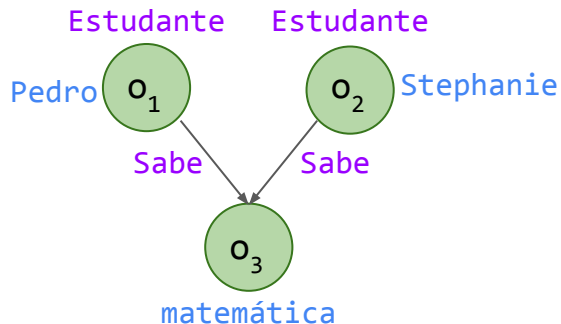
- Um modelo w na lógica de 1ª ordem mapeia:

- Símbolos constantes para objetos

$w(\text{Pedro}) = o_1$, $w(\text{Stephanie}) = o_2$, $w(\text{matemática}) = o_3$

- Predicados para tuplas de objetos

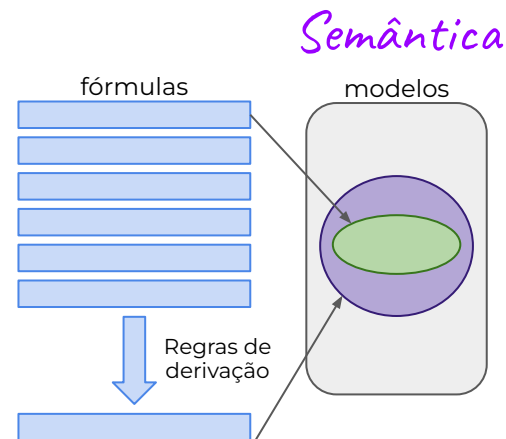
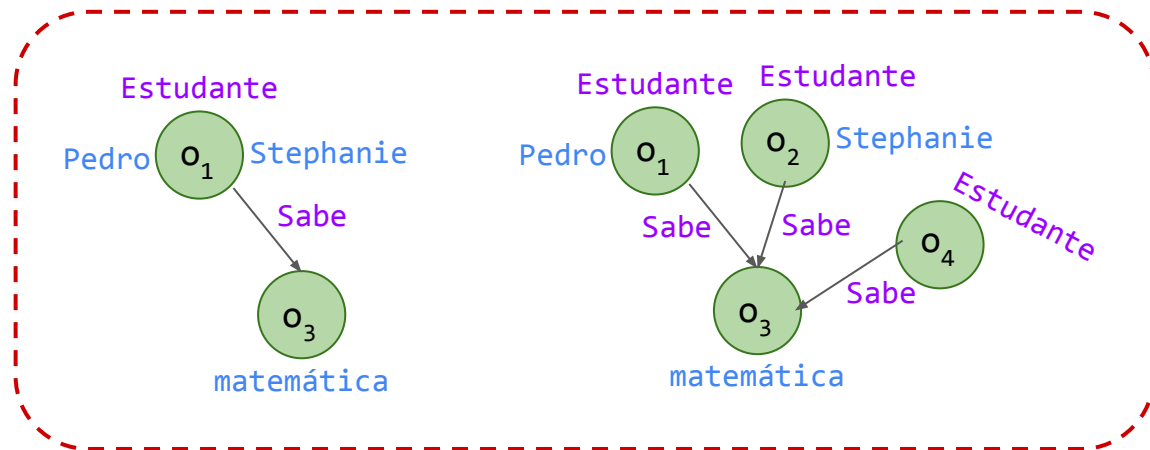
$w(\text{sabe}) = \{(o_1, o_3), (o_2, o_3)\}$



Modelos: Lógica de 1ª Ordem

- Em uma base de conhecimento kb **todos os objetos tem uma e apenas uma constante associada**

- Exemplos de bases **inconsistentes**:



Modelos: Lógica de 1ª Ordem

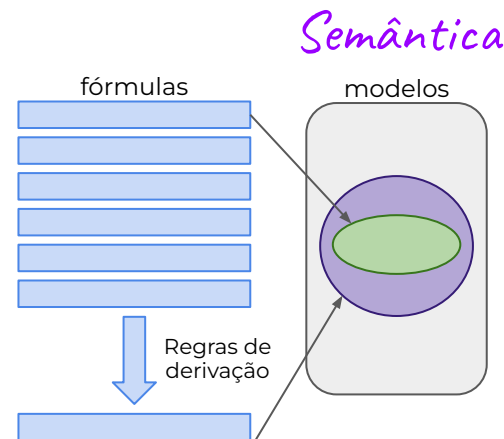
- Em uma base de conhecimento kb todos os objetos tem uma e apenas uma constante associada;
- Com isso em mente, podemos converter expressões de 1ª ordem para lógica proposicional.

Knowledge base in first-order logic

$\text{Student}(\text{alice}) \wedge \text{Student}(\text{bob})$
 $\forall x \text{ Student}(x) \rightarrow \text{Person}(x)$
 $\exists x \text{ Student}(x) \wedge \text{Creative}(x)$

Knowledge base in propositional logic

$\text{Studentalice} \wedge \text{Studentbob}$
 $(\text{Studentalice} \rightarrow \text{Personalice}) \wedge (\text{Studentbob} \rightarrow \text{Personbob})$
 $(\text{Studentalice} \wedge \text{Creativealice}) \vee (\text{Studentbob} \wedge \text{Creativebob})$



Modelos: Lógica de 1ª Ordem

- Em uma base de conhecimento kb todos os objetos tem uma e apenas uma constante associada;
- Com isso em mente, podemos converter expressões de 1ª ordem para lógica proposicional.

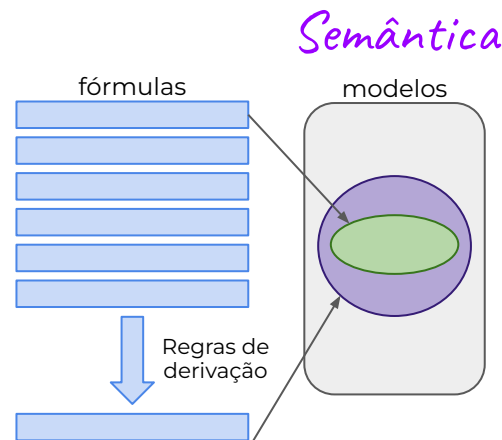
Knowledge base in first-order logic

$\text{Student}(\text{alice}) \wedge \text{Student}(\text{bob})$
 $\forall x \text{ Student}(x) \rightarrow \text{Person}(x)$
 $\exists x \text{ Student}(x) \wedge \text{Creative}(x)$

Divirta-se com as regras de inferência e os SAT solvers da lógica proposicional!

Knowledge base in propositional logic

$\text{Studentalice} \wedge \text{Studentbob}$
 $(\text{Studentalice} \rightarrow \text{Personalice}) \wedge (\text{Studentbob} \rightarrow \text{Personbob})$
 $(\text{Studentalice} \wedge \text{Creativealice}) \vee (\text{Studentbob} \wedge \text{Creativebob})$



Lógica de 1ª Ordem: Substituição/Unificação

- Existem nuances nos processos de substituição e unificação para transformar expressões de 1ª ordem em proposições.
- Não entraremos em detalhes, mas recomendo o material de referência para aprofundar.

<https://inst.eecs.berkeley.edu/~cs188/sp22/assets/slides/Lecture9.pdf>
https://inst.eecs.berkeley.edu/~cs188/sp22/assets/notes/sp21_notes/note03.pdf

https://youtu.be/_Iz83hfkFds?t=3579

- Convert $(KB \wedge \neg\alpha)$ to PL, use a PL SAT solver to check (un)satisfiability
 - Trick: replace variables with ground terms, convert atomic sentences to symbols
 - $\forall x \text{ Knows}(x, \text{Obama})$ and $\text{Democrat}(\text{Feinstein})$
 - $\text{Knows}(\text{Obama}, \text{Obama})$ and $\text{Knows}(\text{Feinstein}, \text{Obama})$ and $\text{Democrat}(\text{Feinstein})$
 - $\text{Knows_Obama_Obama} \wedge \text{Knows_Feinstein_Obama} \wedge \text{Democrat_Feinstein}$
 - and $\forall x \text{ Knows}(\text{Mother}(x), x)$
 - $\text{Knows}(\text{Mother}(\text{Obama}), \text{Obama})$ and $\text{Knows}(\text{Mother}(\text{Mother}(\text{Obama})), \text{Mother}(\text{Obama}))$
 - Real trick: for $k = 1$ to infinity, use all possible terms of function nesting depth k
 - If entailed, will find a contradiction for some finite k (Herbrand); if not, may continue for ever;
semidecidable



Example: modus ponens in first-order logic

Premises:

$\text{Takes}(\text{alice}, \text{cs221})$

$\text{Covers}(\text{cs221}, \text{mdp})$

$\forall x \forall y \forall z \text{ Takes}(x, y) \wedge \text{Covers}(y, z) \rightarrow \text{Knows}(x, z)$

Conclusion:

$\theta = \{x/\text{alice}, y/\text{cs221}, z/\text{mdp}\}$

Derive $\text{Knows}(\text{alice}, \text{mdp})$

Expressividade da Lógica de 1ª Ordem

- Agentes Lógicos

https://inst.eecs.berkeley.edu/~cs188/sp22/assets/notes/sp21_notes/note03.pdf

$$F^{t+1} \Leftrightarrow \text{ActionCauses}F^t \vee (F^t \wedge \neg \text{ActionCausesNot}F^t)$$

In our world, the transition could be formulated as $\text{Hot}^{t+1} \Leftrightarrow \text{StepCloseToLava}^t \vee (\text{Hot}^t \wedge \neg \text{StepAwayFromLava}^t)$.

- Representação robusta de conhecimento

<https://youtu.be/mmQl6VGvX-c>



Resumo

- Muitos problemas de otimização podem ser formulados como busca local
 - Basta que o caminho até a solução não seja relevante
- Muitos algoritmos de aprendizado de máquina são buscas locais (e falaremos mais deles)
- Famílias gerais de algoritmos
 - Hill climbing/Gradient descent
 - Determinístico
 - Simulated annealing e outros métodos estocásticos
 - Beam search
 - Algoritmos genéticos