

LÓGICA APLICADA À COMPUTAÇÃO:

# **Lógica probabilística**

---

Um trabalho por Matheus Ryan, Caio César e Rubens Matheus

# INTRODUÇÃO

---

**01**

DEFINIÇÃO

**02**

SINTAXE

**03**

ESTUDO DE CASO



# 01



## DEFINIÇÃO

---

O que é o ProbLog propriamente dito?



# CONTEXTO HISTÓRICO DO PROBLOG

---

**SURTIU EM 1990**

**EXTENSÃO DO PROLOG**

**ADIÇÃO DE FATOS PROBABILÍSTICOS**

**BASEADO NA LÓGICA DE PRIMEIRA ORDEM**




# 02



## SINTAXE

---

Estrutura do ProbLog



# SINTAXE

|                            |                            |
|----------------------------|----------------------------|
| FATO                       | a.                         |
| FATO PROBABILÍSTICO        | $0.5 :: a.$                |
| CLÁUSULA                   | $a :- x.$                  |
| CLÁUSULA<br>PROBABILÍSTICA | $0.5 :: a :- x.$           |
| ANNOTATED DISJUNCTION      | $0.5 :: a; 0.5 :: b.$      |
| ANNOTATED DISJUNCTION      | $0.5 :: a; 0.5 :: b :- x.$ |

# SINTAXE

---

```
1  0.1::perder.
```

```
2  0.9::alarm :- hora.
```

```
1  1/6::die(D, 1); 1/6::die(D, 2); 1/6::die(D, 3);
```

```
2  1/6::die(D, 4); 1/6::die(D, 5); 1/6::die(D, 6).
```

# SINTAXE

```
0.5::weather(0,sun); 0.5::weather(0,rain).
```

```
0.8::weather_after_sun(T,sun); 0.2::weather_after_sun(T,rain).
```

```
weather(T, sun) :- T > 0, T1 is T - 1, weather(T1, sun), weather_after_sun(T, sun).
```

```
weather(T, rain) :- T > 0, T1 is T - 1, weather(T1, sun), weather_after_sun(T, rain).
```

```
0.4::weather_after_rain(T,sun); 0.6::weather_after_rain(T,rain).
```

```
weather(T, sun) :- T > 0, T1 is T - 1, weather(T1, sun), weather_after_rain(T, sun).
```

```
weather(T, rain) :- T > 0, T1 is T - 1, weather(T1, sun), weather_after_rain(T, rain).
```







**03**

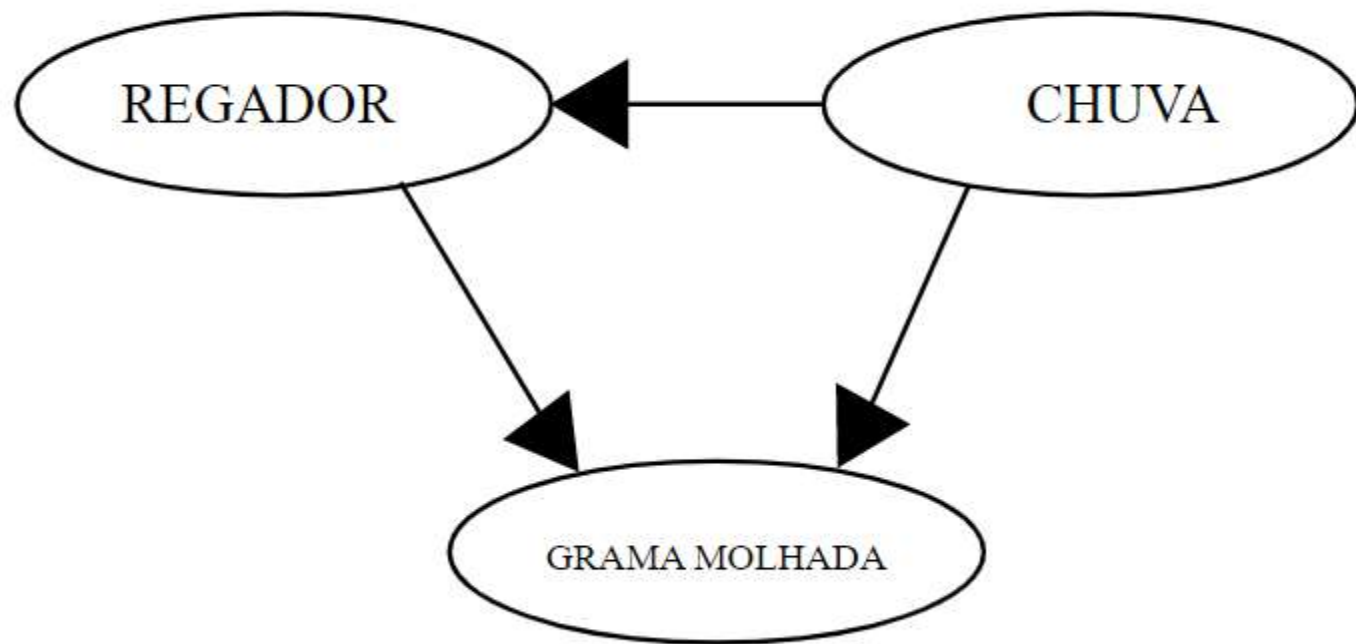


# **ESTUDO DE CASO**

---

Aplicação do ProbLog





# **ESTUDO DE CASO**

---

**A LÓGICA PROBABILÍSTICA PODE SER APLICADA EM DIVERSAS OCASIÕES COTIDIANAS. TAIS COMO JOGAR UM DADO, PREVER O CLIMA OU ESCOLHER CARTAS.**

# **JOGAR UMA MOEDA**

---

**CONSIDERANDO DUAS MOEDAS, UM NORMAL E OUTRO VICIADO À 60% CARA, E AO JOGARMOS DUAS MOEDAS, EM QUE PELO MENOS UMA NÃO SAIU CARA, QUAL A CHANCE DE CADA UMA TER CAÍDO COROA?**

# JOGAR UMA MOEDA

---

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}$$

# JOGAR UMA MOEDA

```
%Fatos Probabilisticos
0.5::cara1. 0.6::cara2.

%Regras Lógicas
duasCaras :- cara1, cara2.

%Criamos a nossa Evidencia
evidence(duasCaras,false).

query(cara1).
query(cara2).
```

# ALARME

---

Suponha que uma casa tenha um alarme que dispare se ocorrer um assalto ou um terremoto. Os donos dessa casa estão viajando mas os seus vizinhos Paulo e Maria estão nas casas ao lado e prometeram ligar para o dono se ouvirem que o alarme disparou. A probabilidade de assalto é de 0.1, terremoto 0.2, e algum dos vizinhos ouvir o alarme caso ele dispare é de 0.7.

# ALARME

```
%Fatos Probabilísticos:
0.1::assalto.
0.2::terremoto.

%Regras Lógicas:
pessoa(paulo).
pessoa(maria).

0.7::escuta_alarme(X) :- pessoa(X).
%Interpretamos como:
%A pessoa(X) escuta o alarme com probabilidade 0.7

%O alarme dispara com um assalto ou terremoto
alarme :- assalto.
alarme :- terremoto.

%A pessoa(X) liga para o dono da casa
%Se o alarme disparar e essa pessoa escutar o alarme
liga(X) :- alarme, escuta_alarme(X).

query(liga(X)).
```






# PRÁTICA



Demonstrando um recomendador de  
filmes usando o ProbLog

---



# ESCOLHA DE FILMES

---

- Filme: ET, Diretor: Steven Spielberg(SS), Gênero: Sci-Fi, Ano de Produção: 1982.
- Filme: Inteligência Artificial, Diretor: Steven Spielberg(SS), Gênero: Sci-Fi, Ano de Produção: 2001.
- Filme: Pulp Fiction, Diretor: Quentin Tarantino(QT), Gênero: Crime, Ano de Produção: 1994.
- Filme: Cães de Aluguel, Diretor: Quentin Tarantino(QT), Gênero: Ação, Ano de Produção: 1992.
- Filme: Batman Begins, Diretor: Christopher Nolan(CN), Gênero: Ação, Ano de Produção: 2005
- Filme: O Show de Truman, Diretor: Peter Weir(PW), Gênero: Comedia, Ano de Produção: 1998.

# ESCALA DO PROGRAMA

---

$$\mathbb{P}(SS|F6) := \frac{\#(SS \in F6)}{\#F6} = \frac{\text{Quantidade de Filmes de SS que estão em } F6}{\text{Quantidade Total de filmes em } F6} = \frac{2}{6} = \frac{1}{3}$$

# ESCALA DO PROGRAMA

```
%Modelando os fatos do Conjunto F6:
```

```
%Diretores:
```

```
2/6::diretor(ss); 2/6::diretor(qt);
```

```
1/6::diretor(cn); 1/6::diretor(pw).
```

```
%Gêneros:
```

```
2/6::genero(scifi); 1/6::genero(crime);
```

```
2/6::genero(acao); 1/6::genero(comedia).
```

```
%Ano:
```

```
0::ano(antigo); 4/6::ano(classico); 2/6::ano(novo).
```

# ESCALA DO PROGRAMA

---

```
%Regra  
%Recomendador1:  
rec(Titulo,Diretor,Genero,Ano):- filme(Titulo,Diretor,Genero,Ano),  
                                   diretor(Diretor),  
                                   genero(Genero),  
                                   ano(Ano).
```

# PROBLEMAS DO PROGRAMA

---


```
1) rec(de_volta_para_o_futuro,ss,scifi,classico) 0.074074074
2) rec(kill_bill_1,qt,acao,novo) 0.037037037
3) rec(o_cavaleiro_das_trevas,cn,acao,novo) 0.018518519

4) rec(uma_guerra_muito_louca,ss,comedia,antigo) 0
5) rec(um_drink_no_inferno,qt,terror,classico) 0
6) rec(amnesia,cn,suspense,novo) 0
7) rec(star_wars,gl,scifi,classico) 0
```

# CONCLUSÃO

---

Fatores probabilísticos podem ou não depender de infinitos fatores probabilísticos mesmo dentro da programação.



# REFERÊNCIAS

- [LucasSilvaArensteinV2.pdf \(usp.br\)](#)
- [2. ProbLog models — ProbLog 2.1 documentation](#)
- [Introduction. — ProbLog: Probabilistic Programming \(kuleuven.be\)](#)
- [ML-KULeuven/problog: ProbLog is a Probabilistic Logic Programming Language for logic programs with probabilities. \(github.com\)](#)





# THANKS !

---

Matheus Ryan, Caio César e Rubens Matheus