Prop1	Prop2	Conjunção	Disjunção	Negação	Implicação	Equivalência
p	q	$p \wedge q$	$p \vee q$	$\sim p$	$p \rightarrow q$	
V	V	V	V	F	V	
V	F	F	V	F	F	
F	V	F	V	V	V	
F	F	F	F	V	V	

Aula07: Prolog

Disciplina: Matemática Discreta
Profa. Kênia Arruda
kenia.costa@uniube.br

Prolog

- Foi criado em 1970 por R. Kowalski e Maarten van Emden em Edimburgo e Alain Colmerauer em Marsailles

  algoritmo = lógica + controle

- O termo Prolog é derivado da expressão “Programming in Logic”, uma vez que é baseado em Lógica de Predicados de 1a ordem, ou seja, uma Linguagem de programação lógica e declarativa
- Propósito da criação: criar programas para tradução de linguagem natural (= linguagens faladas, como português, inglês).
- Junto com Lisp, são as linguagens de programação simbólica mais usadas em Inteligência Artificial

Aplicações

- Principais aplicações se dão na área de computação simbólica:
  - Lógica matemática, prova automática de teoremas e semântica;
  - Solução de equações simbólicas;
  - Bancos de dados relacionais;
  - Linguagem Natural;
  - Sistemas Especialistas;
  - Planejamento Automático de Atividades;
  - Aplicações de “General Problem Solving”, como jogos (Xadrez, Damas, Jogo da Velha, etc.);
  - Compiladores;
  - Análise Bioquímica e projetos de novas drogas.

Prolog

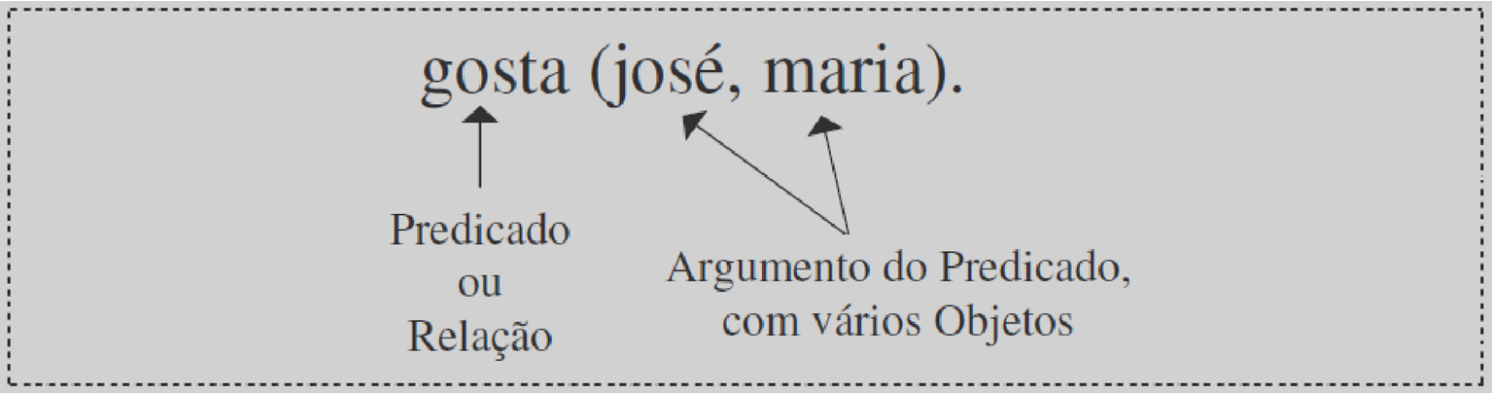
- Existe LPs inteiramente baseada na lógica de predicados!
- A mais famosa é Prolog.
- Um programa Prolog é um conjunto de proposições (“fatos”) e (“regras”) em lógica de predicados.
- A entrada ao programa é uma proposição de consulta.
  - Que queremos saber se é True ou False.
- O interpretador Prolog realiza algumas deduções automáticas para determinar quando a pergunta segue dos fatos.

Prolog

- É uma linguagem de programação que é utilizada para resolver problemas que envolvam objetos e relações entre objetos.
- Um programa em Prolog consiste basicamente de:
  - declaração de fatos (facts) sobre objetos e suas relações;
  - definições de regras (rules)sobre os objetos e suas relações;
  - questões (goals) que são feitas sobre os objetos e suas relações.

Fatos

- Os fatos são os elementos fundamentais da programação em Prolog, pois determinam as relações que existem entre os objetos conhecidos.



João gosta de Maria

Fatos

- Características dos fatos:
  - Os nomes dos predicados e dos objetos devem começar com letra minúscula. Por exemplo: joão, casa, gosta.
  - Os predicados são escritos primeiro e os objetos são escritos depois, separados por vírgulas.
  - Os objetos são escritos dentro de parênteses.
  - Todo fato é terminado com um ponto final.
  - A ordem dos objetos é importante: gosta (maria, josé). [1] gosta (josé, maria).
  - Uma coleção de fatos é chamada de “banco de conhecimento” ou “banco de dados”.
  - Os fatos podem ter um número arbitrário de objetos como argumento.

Fatos

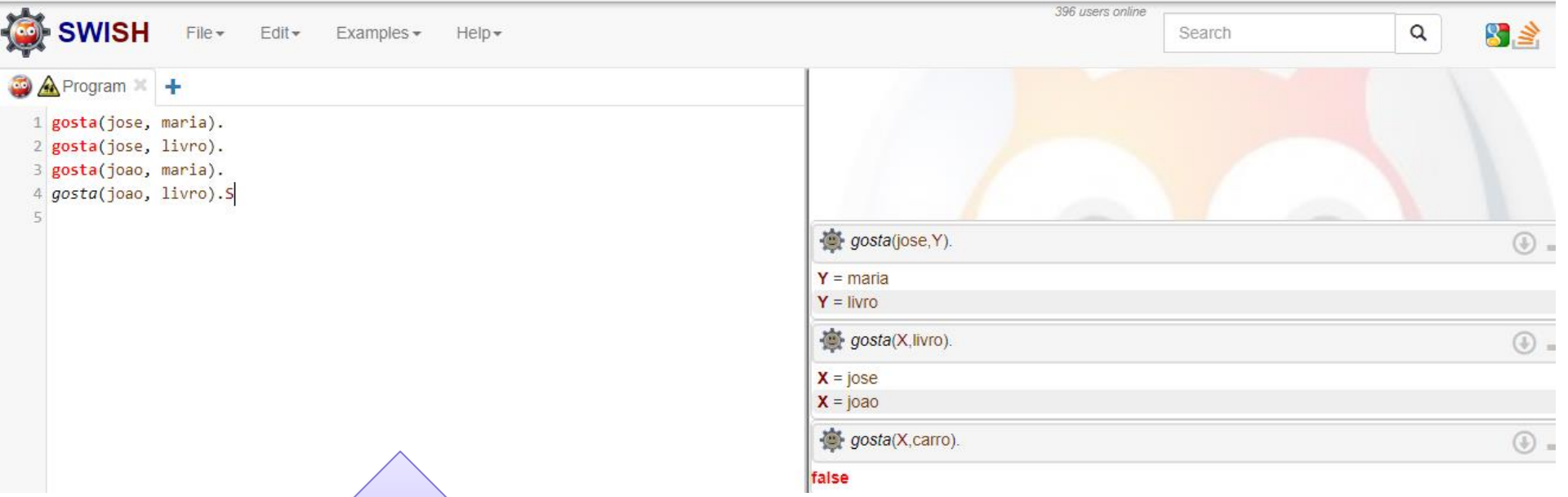
- Quando uma questão é feita, o Prolog realiza uma busca na Base de Conhecimento, procurando um Fato que seja igual ao da questão.
- Dois fatos (ou um fato e uma questão) são unificados (são iguais) se:
  - 1. seus predicados são os mesmos,
  - 2. eles possuem o mesmo número de argumentos e,
  - 3. os argumentos são iguais.
  - Se o Prolog encontra um fato que se iguala a questão, ele retorna “YES”, indicando que a questão tem resposta verdadeira; caso contrário, ele retorna “NO”.
  - OBS: O Prolog retornará “NO” toda vez que não conseguir igualar a questão a um fato da Base de Conhecimento. Isso implica que se nada for declarado sobre um determinado problema, qualquer questão relacionada a ele retornara como “NO”.

Vamos testar...

Acesse o site

- https://swish.swi-prolog.org/example/kb.pl

- Baixar e instalar:
  - https://www.swi-prolog.org/download/stable



Onde colocamos as regras e os fatos!!

Onde colocamos os as questões!!

Prática

- mulher(ana).
- mulher(maria).
- mulher(joana).
- homem(joao).
- homem(mario).
- homem(jose).

Pergunta

- mulher(jose).
- mulher(ana).
- Homem(X).

Prática

- mulher(ana).
- mulher(maria).
- mulher(joana).
- homem(joao).
- homem(mario).
- homem(jose).
- ama(joao,ana).
- ama(jose,ana).

Pergunta

- ama(X,ana).
- ama(jose,X).
- ama(jose,maria).

Cláusulas

- Nesta parte são declarados todos os fatos e todas as regras do programa.
  - Ex.
    - gosta (joão, maria). /* um fato */
    - gosta (joão, X) :- X = mulher. /* uma regra */
- São escritas na forma de uma implicação invertida:
  - H :- B
  - em que H é uma conclusão (átomo ou literal positivo chamado de cabeça), B é um conjunto de premissas (corpo), i.e., é uma conjunção de átomos e :- é a implicação clássica invertida (←)
  - H :- (cláusula unária ou fato) e :- B (cláusula objetivo ou consulta)

Prática

- mulher(ana).
- mulher(maria).
- mulher(joana).
- homem(joao).
- homem(mario).
- homem(jose).
- ama(joao,ana).
- ama(jose,ana).
- ciume(X,Y):- ama(X,Z),ama(Y,Z).

Pergunta

- ciume(jose,Z).
- ciume(jose,ana).

Prática

- mulher(ana).
- mulher(maria).
- mulher(joana).
- homem(joao).
- homem(mario).
- homem(jose).
- ama(joao,ana).
- ama(jose,ana).
- ciume(X,Y):- ama(X,Z),ama(Y,Z).
- ciume2(X,Y):- ama(X,Y), ama(X,Z).

Pergunta

- Quem tem ciúme de quem na regra?

Conjunção e Disjunção

Prática

- Conjunção: ‘,’ e a Disjunção: ‘;’
  - grande(urso).
  - grande(elefante).
  - pequeno(gato).
  - marrom(urso).
  - preto(gato).
  - cinza(elefante).
  - escuro(Z) :- preto(Z).
  - escuro(Z) :- marrom(Z).

  - Ou escuro(Z) :- preto(Z); marrom(Z).

```
?- escuro(X); grande(X).
```

Unificação

Prática

- Sentenças
  - amigo(joão, marcos) %João é amigo de Marcos.
  - amigo(Y, daniel)
  - amigo(Y, mãe(Y))
  - amigo(X, aline)
- Unificação
  - amigo(joão, X) , amigo(joão, marcos) [(marcos, X)]
  - amigo(joão, X) , amigo(Y, daniel) [(daniel, X), (joão, Y)]
  - amigo(joão, X) , amigo(Y, mãe(Y)) [(joão, Y), (mãe(joão), X)]
  - amigo(joão, X) , amigo(X, aline) Falha a unificação pois X não pode receber dois valores diferentes.

Aplicando

- Suponha a seguinte relação:
  - casou(joao, maria, dia(5, maio, 1980)).
  - casou(andre, fernanda, dia(11, agosto, 1950)).
  - casou(adriano, claudia, dia(15, outubro, 1973)).
  - Questione:
    - Qual a data do casamento de Maria?
    - Qual o mês do casamento de Andre?
    - Quem casou antes de Adriano, considerando somente o ano de casamento?
    - Quem casou a menos de 30 anos (considerando apenas o ano)?

Retrocesso

- é um procedimento dentro da linguagem Prolog. Uma busca inicial em um programa nesta linguagem segue o padrão Busca em profundidade (depth-first search), ou seja, a árvore é percorrida sistematicamente de cima para baixo e da esquerda para direita.
- Quando essa pesquisa falha, ou é encontrado um nó terminal da árvore, entra em funcionamento o mecanismo de backtracking. Esse procedimento faz com que o sistema retorne pelo mesmo caminho percorrido com a finalidade de encontrar soluções alternativas.

Retrocesso

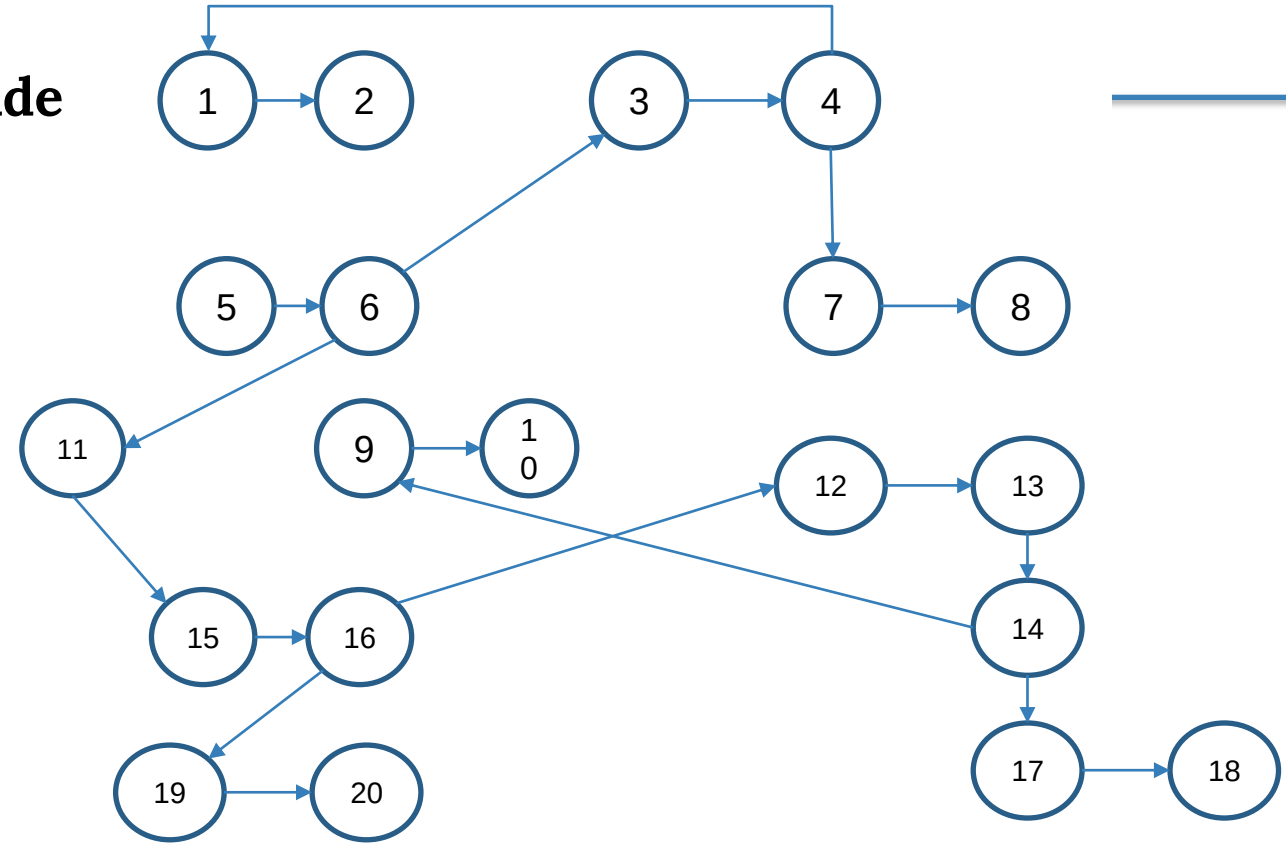
- Exemplo:
  - Considerando uma base de dados família, fazemos a seguinte consulta:
    - ?- pai(roberto,X), mae(vera,X)

Recursividade

- É uma função que chama a si mesma diretamente (recursividade direta) ou ainda indireta (para resolver um problema).
- Ex. Fatorial
  - n! = n * (n–1) * (n–2).... (1)
  - 5! = 5*4*3*2*1
- Outro exemplo é a série de Fibonacci.
  - 0,1,1,2,3,5,8,13,21,34,...

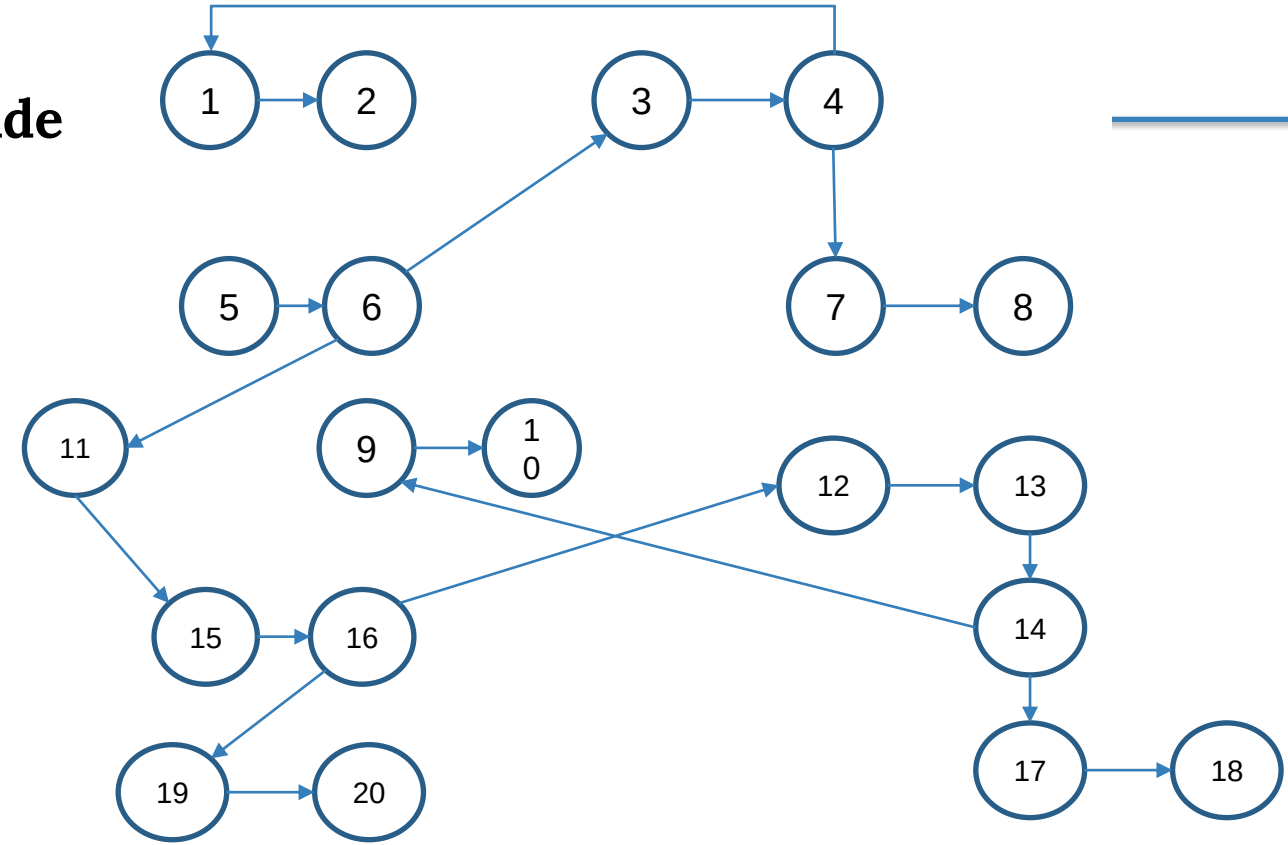
Recursividade

```
conectado(1,2).
conectado(3,4).
conectado(5,6).
conectado(7,8).
conectado(9,10).
conectado(12,13).
conectado(13,14).
conectado(15,16).
conectado(17,18).
conectado(19,20).
conectado(4,1).
conectado(6,3).
conectado(4,7).
conectado(6,11).
conectado(14,9).
conectado(11,15).
conectado(16,12).
conectado(14,17).
conectado(16,19).
caminho( X, Y) :- conectado( X, Y). % conexão direta
caminho( X, Y) :- conectado( X, Z), caminho(Z, Y). % conexão indireta
```



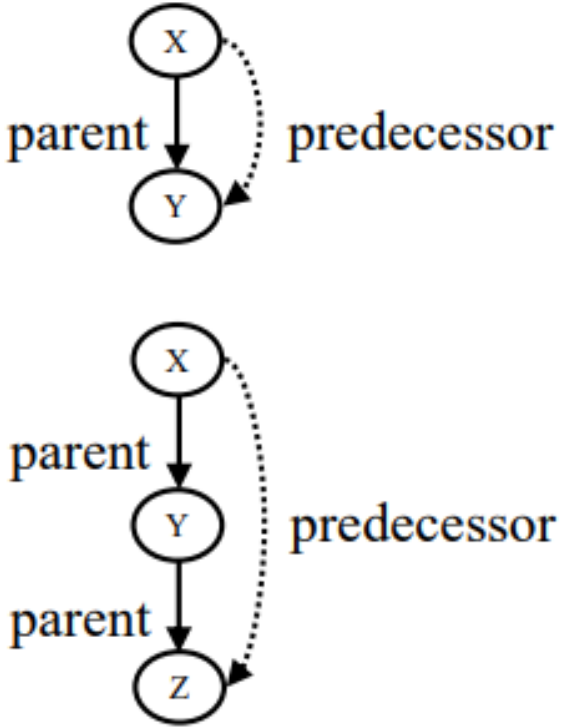
Recursividade

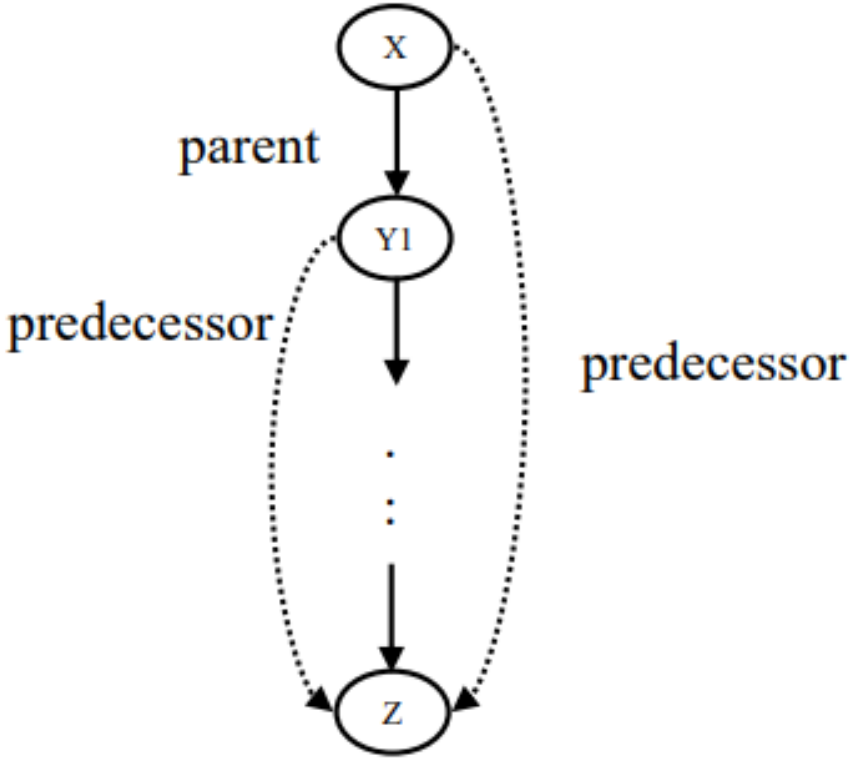
conectado(1,2).
conectado(3,4).
conectado(5,6).
conectado(7,8).
conectado(9,10).
conectado(12,13).
conectado(13,14).
conectado(15,16).
conectado(17,18).
conectado(19,20).
conectado(4,1).
conectado(6,3).
conectado(4,7).
conectado(6,11).
conectado(14,9).
conectado(11,15).
conectado(16,12).
conectado(14,17).
conectado(16,19).
caminho(X, Y) :- conectado(X, Y). % conexão direta
caminho(X, Y) :- conectado(X, Z), caminho(Z, Y). % conexão indireta



Pergunte:
caminho(5,10).
caminho(1,Y).
caminho(13,Y).

Recursividade: árvore genealogica

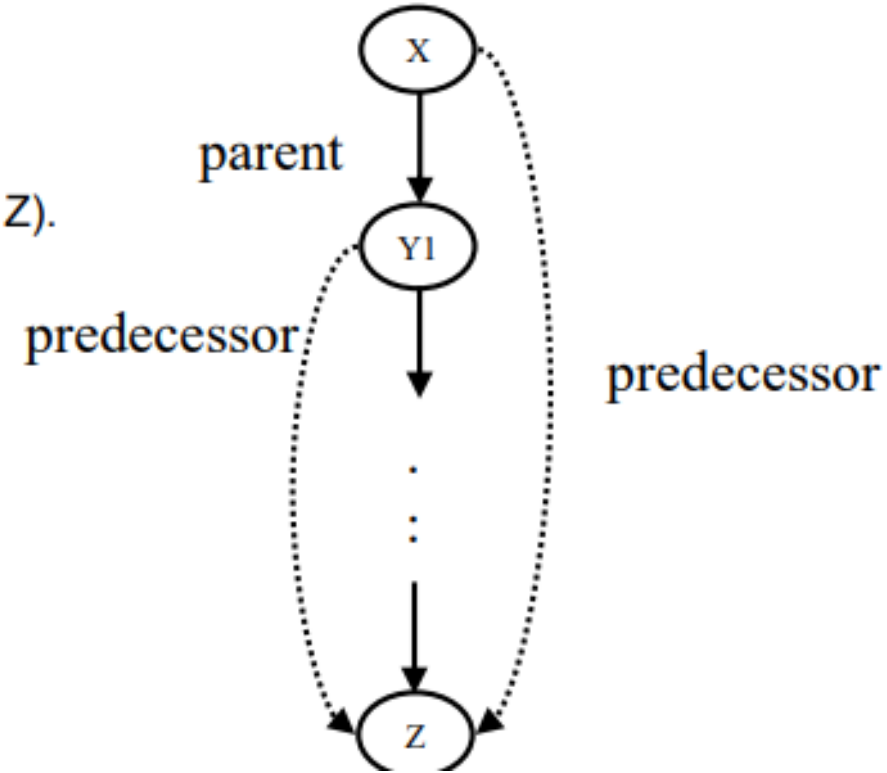




Regras Recursivas

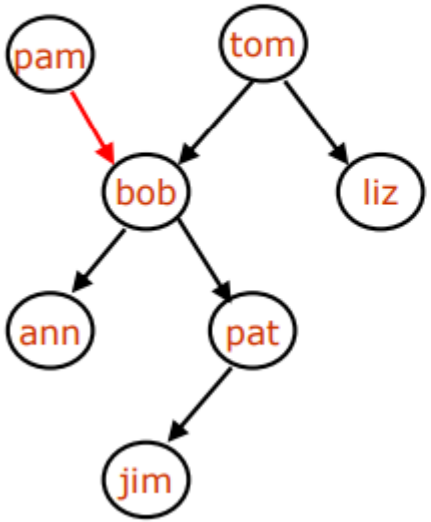
```
predecessor( X, Z):- parent( X, Z).
predecessor( X, Z):-
      parent( X, Y), predecessor( Y, Z).
```

- Interpretação:
  - Para todo *X* e *Z*,
    X é um predecessor de *Z* se
    existe um *Y* tal que
    (1) *X* é um pai de *Y* e
    (2) *Y* é um predecessor de *Z*



Recursividade

- **Definindo relações por meio de fatos**




```
% Parent example

parent(tom,bob).
parent(pam,bob).
parent(bob,ann).
parent(bob,pat).
parent(pat,jim).
parent(tom,liz).
```

Recursividade

- Bob é um dos pais de Pat?

  ?- parent(bob, pat).

  ?- parent(liz, pat).

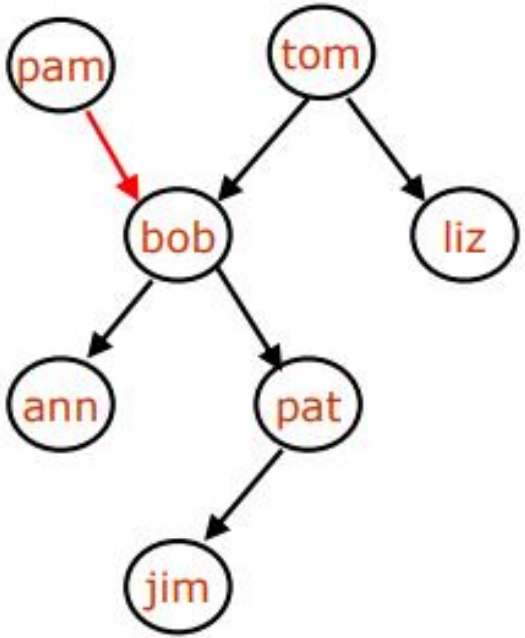
  ?- parent(tom, ben).

- Quem é um dos pais de Liz?

  ?- parent(X, liz).

- Quem são os filhos de Bob?

  ?- parent(bob, X).



Recursividade

- **Questões**

  - Quem é pai de quem?
    - Ache *X* and *Y* de modo que *X* é um dos pais de *Y*

    ?- parent(X, Y).

  - Quem são os avós de Jim?

    ?- parent(Y, jim),
    parent(X, Y).