UNIVERSIDADE ESTADUAL VALE DO ACARAÚ

Centro de Ciências Exatas e Suas Tecnologias (CCET)

Ciência da Computação

**MAXWELL ALVES TEIXEIRA**

**LUCAS DE LIMA**

**3° Trabalho de Inteligência Artificial**

**SOBRAL - CE**

**2018**

1ª. O programa a seguir associa a cada pessoa seu esporte preferido.

joga(ana,volei).

joga(bia,tenis).

joga(ivo,basquete).

joga(eva,volei).

joga(leo,tenis).

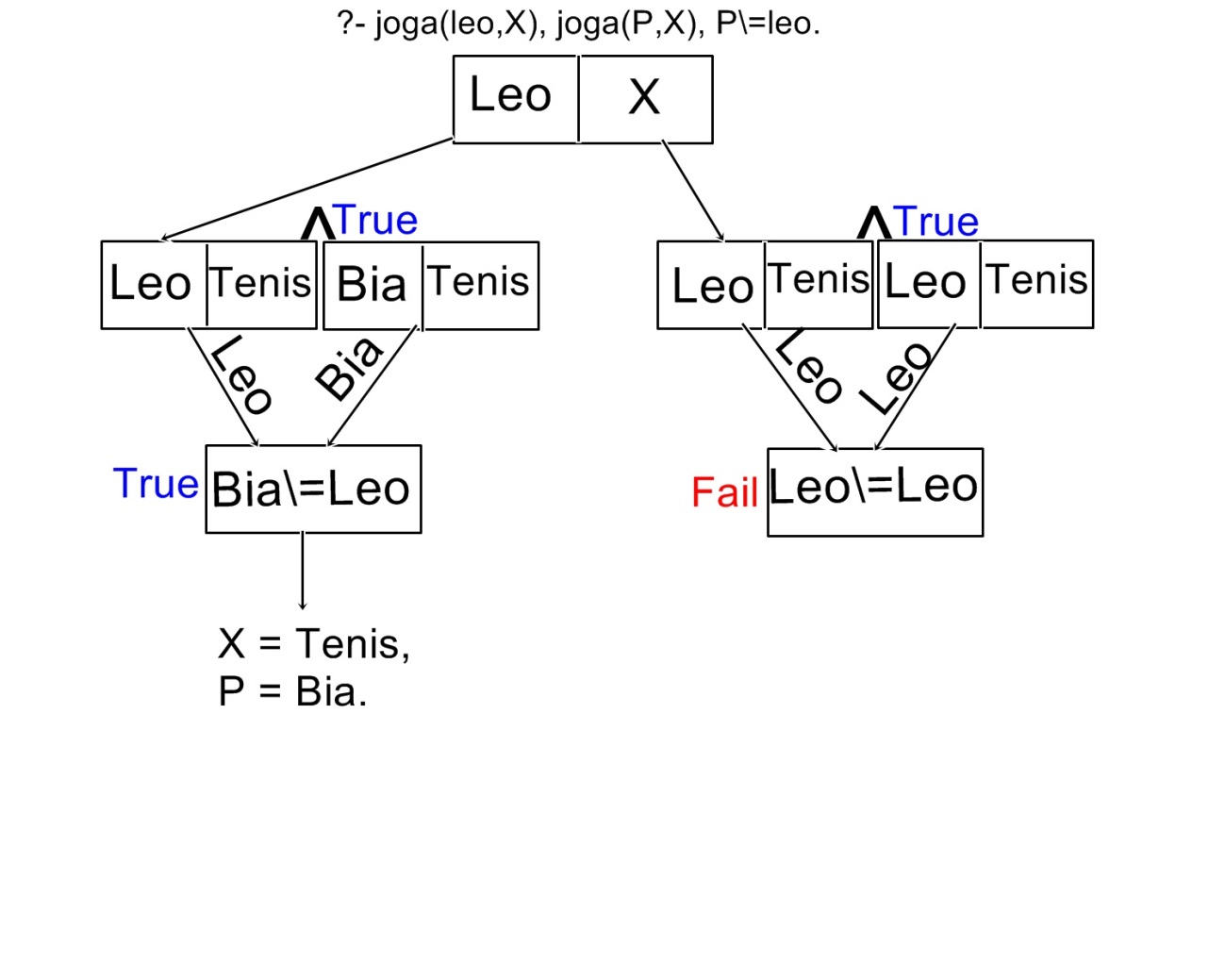
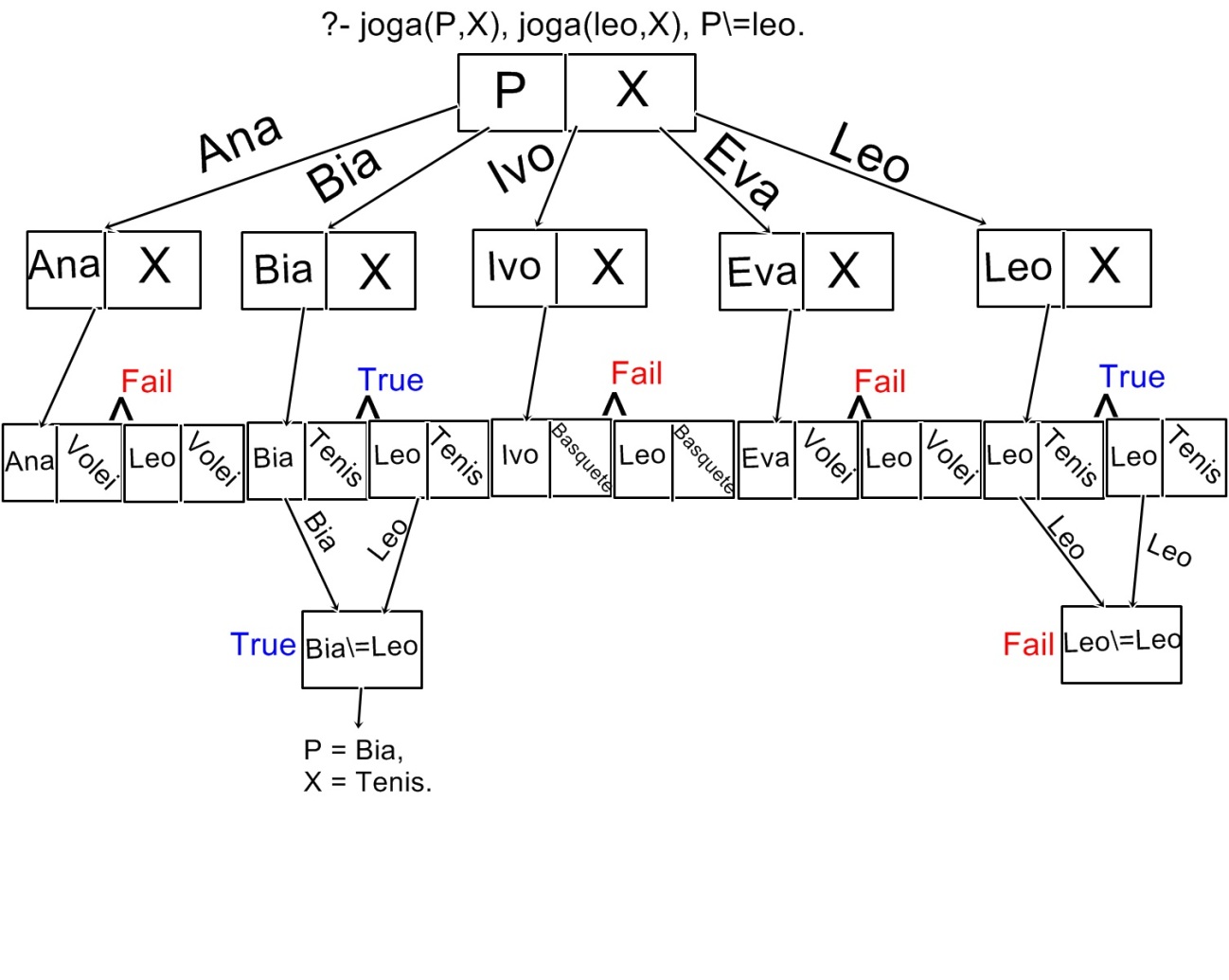
Suponha que desejamos consultar esse programa para encontrar um parceiro P para jogar com Leo. Então, podemos realizar essa consulta de duas formas:

a) ?- joga(P,X), joga(leo,X), P\=leo.

b) ?- joga(leo,X), joga(P,X), P\=leo.

Desenhe as árvores de busca construídas pelo sistema ao responder cada uma dessas consultas. Qual consulta é mais eficiente, por quê?

R: A consulta b é mais eficiente, pois encontra o P com mais rapidez sem a necessidade de vários retrocessos por causa de falha ou se encontrou um nó terminal da árvore, escolhendo a consulta b a raiz da arvore se divide em poucos ramos facilitando a busca.



2ª. O predicado num classifica números em três categorias: positivos, nulos e negativos. Esse predicado, da maneira como está definido, realiza retrocesso desnecessário. Explique por que isso acontece e, em seguida, utilize cortes para eliminar esse retrocesso.

num(N,positivo) :- N>0.

num(0,nulo).

num(N,negativo) :- N<0.

R: O retrocesso acontece porque a linguagem de programação prolog segue o padrão busca em profundidade, ou seja, a árvore é percorrida de cima para baixo e da esquerda para a direita e quando a consulta falha ou encontra um nó terminal começa a funcionar o retrocesso retornando assim pelo mesmo caminho percorrido em busca de soluções alternativas.

num(N,positivo) :- N>0, !.

num(0,nulo):- !.

num(N,negativo) :- N<0.

3ª. Suponha que o predicado fail não existisse em Prolog. Qual das duas definições a seguir poderia ser corretamente usada para causar falhas?

a) falha :- (1=1).

b) falha :- (1=2).

R: Para causar corretamente uma falha, a definição do item b é a certa.

Considere o programa a seguir:

animal(cão).

animal(canário).

animal(cobra).

animal(morcego).

animal(gaivota).

voa(canário).

voa(morcego).

voa(gaivota).

dif(X,X) :- !, fail.

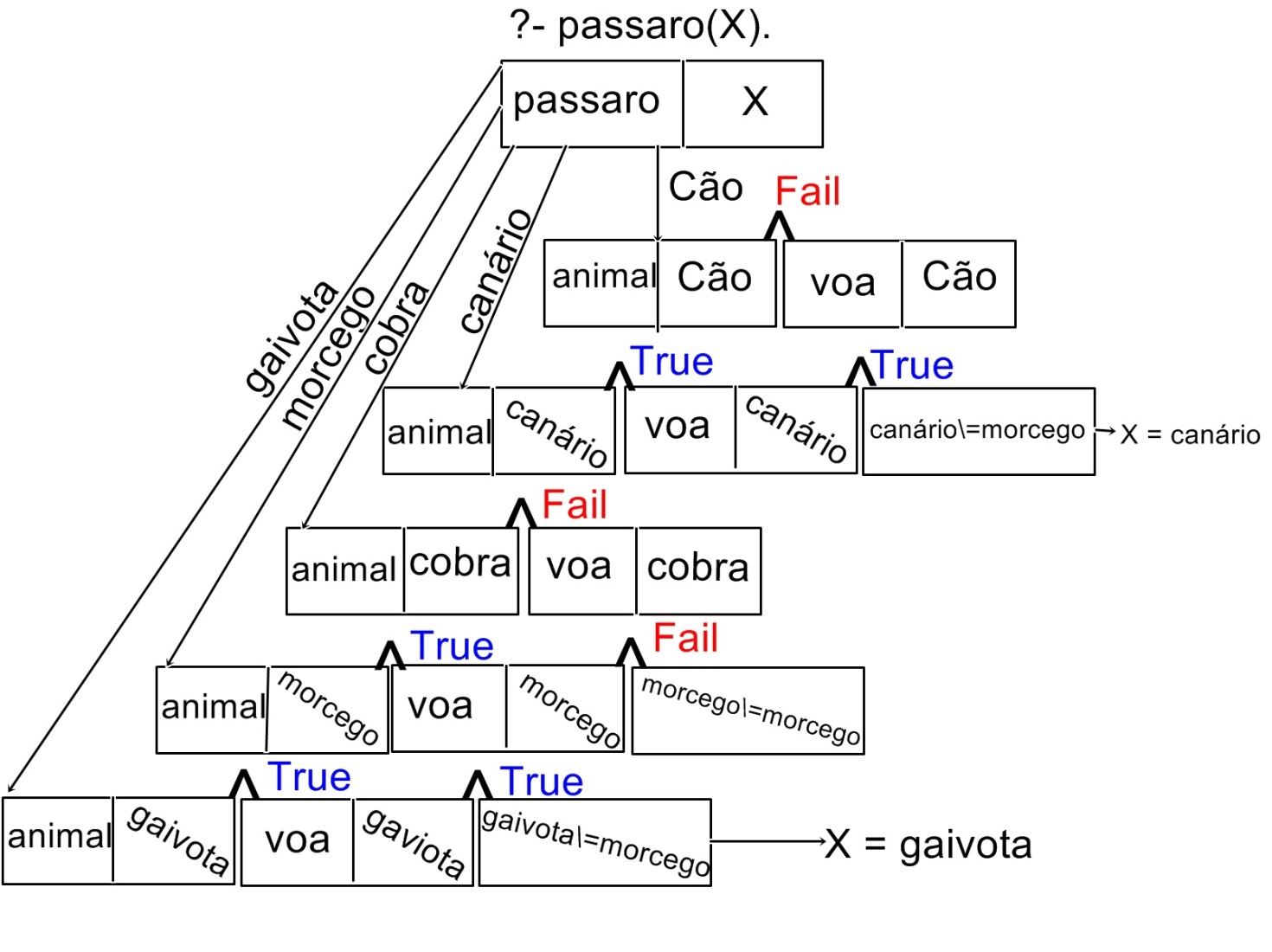
dif(\_,\_).

pássaro(X) :- animal(X), voa(X), dif(X,morcego).

Desenhe a árvore de busca necessária para responder a consulta

?- pássaro(X).

Em seguida, execute o programa para ver se as respostas do sistema correspondem àquelas que você encontrou.

R:

Trace:

[trace] ?- passaro(X).

Call: (8) passaro(\_2954) ? creep

Call: (9) animal(\_2954) ? creep

Exit: (9) animal(cão) ? creep

Call: (9) voa(cão) ? creep

Fail: (9) voa(cão) ? creep

Redo: (9) animal(\_2954) ? creep

Exit: (9) animal(canário) ? creep

Call: (9) voa(canário) ? creep

Exit: (9) voa(canário) ? creep

Call: (9) dif(canário, morcego) ? creep

Exit: (9) dif(canário, morcego) ? creep

Exit: (8) passaro(canário) ? creep

X = canário ;

Redo: (9) animal(\_2954) ? creep

Exit: (9) animal(cobra) ? creep

Call: (9) voa(cobra) ? creep

Fail: (9) voa(cobra) ? creep

Redo: (9) animal(\_2954) ? creep

Exit: (9) animal(morcego) ? creep

Call: (9) voa(morcego) ? creep

Exit: (9) voa(morcego) ? creep

Call: (9) dif(morcego, morcego) ? creep

Call: (10) fail ? creep

Fail: (10) fail ? creep

Fail: (9) dif(morcego, morcego) ? creep

Redo: (9) animal(\_2954) ? creep

Exit: (9) animal(gaivota) ? creep

Call: (9) voa(gaivota) ? creep

Exit: (9) voa(gaivota) ? creep

Call: (9) dif(gaivota, morcego) ? creep

Exit: (9) dif(gaivota, morcego) ? creep

Exit: (8) passaro(gaivota) ? creep

X = gaivota.

4ª. O que está errado no programa abaixo? Rode-o com trace, para ? max(3,4,M) e ?- max(4,3,M).

max(X,Y,M):-!, X>Y, M=X.

max(X,Y,M):-!, X<=Y, M=Y.

R: O erro estava no corte (!), na linha 1 ele corta antes dele fazer o retrocesso e verificar o outro ramo da arvore que seria a linha 2, impossibilitando assim que o programa verificasse a consulta, ?- max(3,4,M), corretamente.

Forma correta:

max(X,Y,M):- X>Y, !, M=X.

max(X,Y,M):-!, X=<Y, M=Y.

5ª. Seja o seguinte programa Prolog:

p(1).

p(2) :- !.

p(3).

Escreva todas as respostas do sistema Prolog para as seguintes consultas:

(a) ?-p(X).

R:

X = 1 ;

X = 2.

(b) ?-p(X), p(Y).

R:

X = Y, Y = 1 ;

X = 1,

Y = 2 ;

X = 2,

Y = 1 ;

X = Y, Y = 2.

(c) ?-p(X), !, p(Y).

R:

X = Y, Y = 1 ;

X = 1,

Y = 2.

6ª. Seguinte relação classifica números em três classes: positivo, nulo ou negativo:

classe(N, positivo) :- N > 0.

classe(0, nulo).

classe(N, negativo) :- N < 0.

Defina este procedimento de uma forma mais eficiente usando cuts.

R:

classe(N, positivo) :- N > 0, !.

classe(0, nulo) :- !.

classe(N, negativo) :- N < 0.