

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN



LAB 2 – Logic bậc nhất
MÔN: CƠ SỞ TRÍ TUỆ NHÂN TẠO

| Giáo viên hướng dẫn |

Thầy: Dương Nguyễn Thái Bảo

Sinh viên thực hiện:

CAO TẤT CƯỜNG – 18120296

Huỳnh Long Nam – 18120212

Lê Nhựt Nam - 18120061

Chuyên ngành: Khoa học máy tính

Thành phố Hồ Chí Minh – 2020

MỤC LỤC

MỤC LỤC	2
Phần I: Phân công công việc	3
Phần II: Làm quen với công cụ Prolog	4
2.1. Tìm hiểu ngôn ngữ Prolog	4
2.1.1 Tổng quát ngôn ngữ Prolog	4
2.1.2 Cấu trúc chính của một chương trình trong Prolog	5
2.1.3 Các nguyên tắc của ngôn ngữ Prolog	8
2.1.4 Các kiểu dữ liệu	8
2.1.5 Ứng dụng ngôn ngữ lập trình Prolog trong một số bài toán Trí tuệ nhân tạo.....	11
2.2 Tìm hiểu một môi trường lập trình Prolog (SWI-Prolog)	13
2.2.1 Giới thiệu về SWI-Prolog	13
2.2.2 Cách sử dụng SWI-Prolog	13
2.3 Xây dựng cây phả hệ Hoàng gia Anh	22
2.3.1 Nhóm vị từ định nghĩa sẵn.....	22
2.3.2 Nhóm vị từ được suy diễn.....	22
2.3.3 Bộ câu hỏi truy vấn và câu trả lời	23
Phần III: Xây dựng cơ sở tri thức với công cụ Prolog	27
3.1 Tóm tắt cơ sở tri thức về chủ đề cơ cấu trường KHTN.....	27
3.2 Các quan hệ chính	27
3.3 Các quan hệ mở rộng:	28
3.4 Bộ test cho hệ cơ sở tri thức	29
Phần IV: Cài đặt hệ thống suy diễn logic bằng ngôn ngữ lập trình	39
4.1 Các thuật toán sử dụng	39
4.2 Kiểm chứng kết quả	41
PHỤ LỤC.....	43
TÀI LIỆU THAM KHẢO	44

Phần I: Phân công công việc

Họ tên	Công việc	Hoàn thành
Cao Tất Cường	Làm quen với công cụ Prolog	100%
Lê Nhựt Nam	Xây dựng cơ sở tri thức với công cụ Prolog	100%
Huỳnh Long Nam	Cài đặt hệ thống suy diễn logic bằng ngôn ngữ lập trình	100%

Phần II: Làm quen với công cụ Prolog

2.1. Tìm hiểu ngôn ngữ Prolog

2.1.1 Tổng quát ngôn ngữ Prolog

Prolog = Programming Logic

Một ngôn ngữ lập trình logic được sử dụng rất phổ biến

Các khái niệm chính trong Prolog là: vị từ, đối tượng, biến, sự kiện, qui tắc, câu hỏi, danh sách.

Vài kỹ thuật thường được dùng trong Prolog là: quay lui (mặc định là tìm kiếm theo chiều sâu trên cây suy diễn), đệ qui, lát cắt.

Trong ngôn ngữ Prolog

- Các luật và các sự kiện là các tiên đề (axioms)
- Câu hỏi, được đưa ra bởi người dùng, là định lý cần chứng minh

Để chứng minh $P(a)$

- Tìm sự kiện $P(t)$ hoặc luật $(Q1 \wedge Q2 \wedge \dots \wedge Qn \rightarrow P(t))$
- Nếu tìm được sự kiện $P(t)$, thay thế $t = a$, định lý được chứng minh
- Nếu tìm được luật $(Q1 \wedge Q2 \wedge \dots \wedge Qn \rightarrow P(t))$, thì cần tiếp tục chứng minh các giả thiết
- Nếu có các biến, tìm cách thay thế các biến này bằng các giá trị, sao cho đích chứng minh được thỏa mãn

Ví dụ: Để mô tả các sự kiện: “Chó và Mèo là động vật”, qui tắc logic: “bất cứ con gì là động vật thì nó phải thở”, ta dùng hai vị từ chính: LàĐộngVật(Con vật), Thở(Con vật) như sau:

- LàĐộngVật (“Chó”).
- LàĐộngVật (“Mèo”).
- Thở(Con vật) :- LàĐộngVật (Con vật).

Dựa trên các tri thức vừa mô tả, ta có thể yêu cầu Prolog trả lời các câu hỏi liên quan sau:

- LàNgười(“Chó”) \rightarrow yes
- LàNgười(“Cái bát”) \rightarrow no
- Chết(“Chó”) \rightarrow yes
- Chết(“Cái bát”) \rightarrow no

hoặc:

- $(Ai) \rightarrow Ai = \text{“Socrate”}, Ai = \text{“Tèo”}$ (2 đáp án)
- Chết(Ai) $\rightarrow Ai = \text{“Socrate”}, Ai = \text{“Tèo”}$ (2 đáp án)

2.1.2 Cấu trúc chính của một chương trình trong Prolog

Một chương trình của Prolog thường bao gồm một số các phần chính và theo thứ tự sau:

1. Constants: Đoạn khai báo các đối tượng hằng
2. Domains: Đoạn khai báo kiểu các đối tượng riêng của người dùng
3. Database: Đoạn khai báo kiểu cho các vị từ của cơ sở dữ liệu
4. Predicates: Đoạn khai báo kiểu cho vị từ sẽ dùng trong các đoạn sau
5. Clauses: Đoạn liệt kê sự kiện và định nghĩa qui tắc đã khai báo trong phần Predicates
6. Goal: Đoạn đưa vào các vị từ câu hỏi, có thể đặt trước phần Clauses

Chú ý: Một chương trình có thể thiếu một số phần trên.

Phần Constants

Hằng số biểu thị các đối tượng, mục, yếu tố, giá trị, hiện tượng cụ thể.... Tên hằng bắt đầu bằng các chữ cái thường. Số nguyên, số hữu tỉ và chuỗi cũng hợp lệ (ví dụ: abc, 1, 'Hello').

Phần Domains

Đây là phần định nghĩa kiểu mới dựa vào các kiểu đã biết. Các kiểu được định nghĩa ở đây sẽ được sử dụng cho các đối số trong các vị từ. Nếu các vị từ sử dụng đối số có kiểu cơ bản thì có thể không cần phải định nghĩa lại các kiểu đó. Tuy nhiên để cho chương trình sáng sủa, người ta sẽ định nghĩa lại cả các kiểu cơ bản.

Cú pháp: <danh sách kiểu mới> = <kiểu đã biết> hoặc <danh sách kiểu mới> = <danh sách kiểu đã biết>

Trong đó các kiểu mới phân cách nhau bởi dấu phẩy, còn các kiểu đã biết phân cách nhau bởi dấu chấm phẩy.

Ví dụ:

- giao_vien, bo_mon, khoa = string
- so_luong_cv, so_de_tai = integer
- luong_gv = real
- ngay_nhan_chuc = ngay(ngay, thang, nam)
- cong_viec(giao_vien, bo_mon, khoa)
- nha(dia_chi, dien_tich)

Trong ví dụ trên, ta đã định nghĩa các kiểu mới, trong đó các kiểu mới giao_vien, bo_mon, khoa dựa vào cùng một kiểu đã biết là string; các kiểu mới so_luong_cv, so_de_tai dựa vào cùng một kiểu đã biết là integer; kiểu mới luong_gv dựa vào kiểu đã biết là real; kiểu mới ngay_nhan_chuc dựa vào kiểu ngày được xây dựng từ các kiểu đã biết là ngay, thang, nam; còn kiểu cong_viec dựa vào các kiểu đã biết.

Phần Predicates

Đây là phần bắt buộc phải có. Trong phần này chúng ta cần phải khai báo đầy đủ các vị từ sử dụng trong phần Clauses, ngoại trừ các vị từ mà Turbo Prolog đã xây dựng sẵn.

Cú pháp: <Tên vị từ> (<danh sách các kiểu>)

Các kiểu là các kiểu cơ bản hoặc là các kiểu đã được định nghĩa trong phần domains và được viết phân cách nhau bởi dấu phẩy.

Ví dụ:

- Thich(ten, keo)
- so_nguyen_duong(integer)

Trong ví dụ trên ta khai báo hai vị từ. Trong đó vị từ Thich(ten, keo) để chỉ một người có tên là ten thích kẹo. Còn vị từ so_nguyen_duong(integer) để xét xem một số integer nào đó có phải là số nguyên dương hay không.

Phần Clauses

Đây là phần bắt buộc phải có dùng để mô tả các sự kiện và các luật, sử dụng các vị từ đã khai báo trong phần predicates.

Cú pháp:

<Tên vị từ>(<danh sách các tham số>) <kí hiệu>

<Tên vị từ 1>(<danh sách các tham số 1>) <kí hiệu>

... ..

<Tên vị từ N>(<danh sách các tham số N>) <kí hiệu>

Trong đó: Tên vị từ phải là các tên vị từ đã được khai báo trong phần predicates. Các tham số có thể là các hằng hoặc biến có kiểu tương thích với các kiểu tương ứng đã được khai báo trong các vị từ ở trong phần predicates; các tham số được viết cách nhau bởi dấu phẩy. Các kí hiệu bao gồm:

:- (điều kiện nếu).

, (điều kiện và).

; (điều kiện hoặc).

. (kết thúc vị từ)

A bất kỳ từ nào được viết hoa đều là một biến

() để bao gồm dữ liệu / thông tin

Ví dụ:

- chẵn(X):- X chia_dư 2 = 0.
- đèn_sáng(X):- công_tắc_bật(X), có_điện.
- so_huu("Nguyen Van A", nha(dia_chi, dien_tich)).

Chú ý: Nếu trong các tham số của một vị từ có biến thì biến này phải xuất hiện ít nhất 2 lần trong vị từ đó hoặc trong các vị từ dùng để suy diễn ra vị từ đó. Nếu chỉ xuất hiện một lần thì bắt buộc phải dùng biến tự do.

Phần Goal

Bao gồm các mục tiêu mà ta yêu cầu Turbo Prolog xác định và tìm kết quả. Đây là phần không bắt buộc phải có. Nếu ta viết sẵn trong chương trình thì đó gọi là goal nội; Nếu không, khi chạy chương trình Turbo Prolog sẽ yêu cầu ta nhập goal vào, lúc này gọi là goal ngoại.

Cú pháp phần goal giống như cú pháp phần clauses. Tức là ta đưa vào một hoặc một số các vị từ.

Nếu tất cả các tham số của vị từ là hằng thì kết quả nhận được là Yes (đúng) hoặc No (sai). Nếu trong các tham số của vị từ có biến thì kết quả trả về sẽ là các giá trị của biến.

Ngoài các phần chủ yếu nói trên, ta có thể đưa vào các phần liên quan đến khai báo hằng, các tập tin liên quan hoặc chỉ thị dịch.

Câu hỏi: dùng để tra cứu một điều gì đó.

Ví dụ: Muốn tìm con nào là con mèo: ?-mèo(X).

Sau khi hiển thị câu trả lời đầu tiên, Prolog sẽ lần lượt tìm kiếm dữ kiện thoả mãn và lần lượt hiển thị kết quả nếu chừng nào người còn yêu cầu cho đến khi không còn kết quả lời giải nào nữa (kết thúc bởi Yes) .

Ví dụ:

?:-Cat(X)

X=tôm ;

X=abc;

Ở kết quả đầu tiên, để tiếp tục nhận các kết quả khác, người dùng tiếp tục yêu cầu bằng dấu chấm phẩy(;). Nhấn enter hoặc dấu chấm . để kết thúc luồng trả lời.

Khi hỏi các câu hỏi một lần và cách nhau dấu , tức là tìm đáp án thoả mãn tất cả các câu hỏi đó(tức là phép và (and))

Ví dụ cần hỏi. Lan và Quang có cùng cha không?

?- cha(X, Lan),cha(X,Quang)

Tức là prolog sẽ tìm một người X thoả mãn vừa là cha của Lan và của Quang. Không có thì sẽ trả lời là no. câu trả lời “no” tức là không tìm ra câu trả lời thoả mãn câu hỏi (mà khác với các câu trả lời trước đó).

Việc trả lời câu trong prolog là đi tìm các dữ kiện thế vào các luật thoả mãn.

Ví dụ: Ta đặt câu hỏi: ?:-father(X,Y)

Lúc này prolog sẽ đi tìm định nghĩa của luật này hay dữ kiện về father. Nếu đó là một luật thì sẽ bắt đầu tìm câu trả lời cho phần thân của luật đó. Nếu nó là một dữ kiện thì X,Y là những hạng được xác định quan hệ parent trước đó. Tóm lại là để trả lời một câu hỏi thì prolog tìm và thay thế, trả lời những câu hỏi nhỏ để trả lời có tồn tại hay không.

Nếu trong các vị từ là các hạng thì kết quả là có (yes) hay không (no) các hạng đó thoả mãn phần thân của luật hay là dữ kiện được xác định trước đó.

2.1.3 Các nguyên tắc của ngôn ngữ Prolog

Việc giải quyết vấn đề trong ngôn ngữ Prolog chủ yếu dựa vào hai nguyên tắc sau: Đồng nhất, quay lui.

Đồng nhất

Một quan hệ có thể đồng nhất với một quan hệ nào đó cùng tên, cùng số lượng tham số, các đại lượng con cũng đồng nhất theo từng cặp.

Một hằng có thể đồng nhất với một hằng.

Một biến có thể đồng nhất với một hằng nào đó và có thể nhận luôn giá trị hằng đó.

- Đồng nhất ($\text{Contra}(An), \text{Contra}(y)) = \{y/An\}$
- Đồng nhất ($\text{Yeu}(An,x), \text{Yeu}(y,Binh)) = \{x/Binh; y/An\}$
- Đồng nhất ($\text{Yeu}(An,x), \text{Yeu}(y, \text{Emgai}(Hoa)) = \{x/\text{Emgai}(Hoa); y/An\}$
- Đồng nhất ($\text{Yeu}(An,x), \text{Yeu}(An,y)) = \{x, y/x\}$
- Đồng nhất ($\text{Ban}(An,x), \text{Ban}(y, \text{Emgai}(y)) = \{x/\text{Emgai}(An); y/An\}$
- Đồng nhất ($\text{P}(a,X), \text{P}(X,b)) = \text{failure}$
- Đồng nhất ($\text{parents}(x, \text{father}(x), \text{mother}(Jane)), \text{parents}(Bill, \text{father}(y), \text{mother}(y))) = \text{failure}$

Quay lui

Để xác định quan hệ tổ tiên trong cây phả hệ. Ta cần nhiều luật theo nhiều bậc của cây phả hệ

Bậc 1: Ông bà: $\text{ongba}(X,Y):-\text{chame}(X,Z),\text{chame}(Z,Y).$

Bậc 2: Ông bà có: $\text{totien}(X,Y):-\text{chame}(X,Z),\text{chame}(Z,U),\text{chame}(U,Y).$

Cây phả hệ càng lớn thì các định nghĩa càng dễ nhiều. Ta sử dụng sức mạnh của prolog là Đệ qui. Ta có định nghĩa đệ qui của quan hệ tổ tiên như sau: Tổ tiên nếu là cha mẹ hay tổ tiên của cha mẹ. Ta có quan hệ viết bằng prolog như sau

$\text{totien}(X,Y):-\text{chame}(X,Y).$

$\text{totien}(X,Y):-\text{totien}(X,Z),\text{chame}(Z,Y).$

2.1.4 Các kiểu dữ liệu

Trong prolog có kiểu dữ liệu chuẩn và kiểu do người lập trình định nghĩa.

Kiểu dữ liệu chuẩn

Phép toán số học

Phép toán	Ý nghĩa	Kiểu của đối số	Kiểu kết quả
+	Cộng hai số	Integer, real	giống kiểu đối số
-	Trừ hai số	Integer, real	giống kiểu đối số
*	Nhân hai số	Integer, real	giống kiểu đối số

/	Chia hai số	Integer, real	giống kiểu đối số
Mod	Phép chia lấy phần dư	Integer	Integer
Div	Phép chia lấy phần nguyên	Integer	Integer

Phép toán quan hệ

Phép toán	Ý nghĩa	Kiểu của đối số	Kết quả
<	Nhỏ hơn	Char, integer, real, string	Yes hoặc No
<=	Nhỏ hơn hay bằng	Char, integer, real, string	Yes hoặc No
=	Bằng	Char, integer, real, string	Yes hoặc No
>	Lớn hơn	Char, integer, real, string	Yes hoặc No
>=	Lớn hơn hay bằng	Char, integer, real, string	Yes hoặc No
<> hay ><	Khác	Char, integer, real, string	Yes hoặc No

Các vị từ như các hàm toán học

Vị từ	Ý nghĩa	Kiểu của đối số	Kiểu kết quả
Sin(X)	Tính sin của X	real	real
Tan(X)	Tính tang của X	real	real
Arctan(X)	Tính arctang của X	real	real
Exp(X)	Tính eX	real	real
Ln(X)	Tính logarit cơ số e của X	real	real
Log(X)	Tính Logarit cơ số 10 của X	real	real
SQRT(X)	Tính căn bậc hai của X	real	real
ROUND(X)	Cho ta số nguyên là số X được làm tròn, dấu là dấu của X	real	integer
TRUNC(X)	Cho phần nguyên của số X, dấu là dấu của X	real	integer
ABS(X)	Cho ta trị tuyệt đối của X	real	real

Random(X)	Cho ta số thực X nằm trong khoảng [0, 1)	real	real
Random(Y, X)	Cho ta số nguyên X nằm trong khoảng [0, Y)	real	integer

Toán tử NOT(X) : Nếu X là Yes thì NOT(X) là No và ngược lại.

Các kiểu dữ liệu do người lập trình định nghĩa

Kiểu mẫu tin:

Cú pháp: <tên kiểu mẫu tin> = tên mẫu tin (danh sách các kiểu phần tử)

Ví dụ:

Domains

giao_vien, bo_mon, khoa = string

so_luong_cv, so_de_tai = integer

luong_gv = real

ngay_nhan_chuc = ngay(ngay, thang, nam)

predicates

cong_viec(giao_vien, bo_mon, khoa)

nha(dia_chi, dien_tich)

clauses

phan_cong("Nguyen Van A", cong_viec ("Hoang Thi B", "KHMT", "CNTT")).

Kiểu danh sách

Cú pháp: <tên kiểu danh sách> = <tên kiểu phần tử>*

Các hàm xuất nhập chuẩn

Xuất ra màn hình

- write(Arg1, Arg2, ... ,Argn) in ra màn hình giá trị của các đối số.
- writef(định_dang, Arg1, Arg2, ... ,Argn) in ra màn hình giá trị của các đối số theo định dạng
- Các định dạng
 - "%d": In số thập phân bình thường; đối số phải là char hoặc integer
 - "%c": Đối số là một số integer, in ký tự có mã Ascii là đối số đó, chẳng hạn writef("%c",65) được A
 - "%e": In số thực dưới dạng lũy thừa của 10
 - "%x": In số Hexa; đối số phải là char hoặc integer
 - "%s": In một chuỗi hoặc một symbol

Nhập vào từ bàn phím

- Readln(X): Nhập một chuỗi ký tự vào biến X
- ReadInt(X): Nhập một số nguyên vào biến X
- ReadReal(X): Nhập một số thực vào biến X
- ReadChar(X): Nhập vào một ký tự vào biến X

2.1.5 Ứng dụng ngôn ngữ lập trình Prolog trong một số bài toán Trí tuệ nhân tạo

Ví dụ 1: Xét xem một số N có phải là số nguyên tố hay không.

```
domains
so_nguyen = integer
predicates
so_nguyen_to(so_nguyen)
Clauses
so_nguyen_to(2):- !.
so_nguyen_to(N):- N>0,
so_nguyen_to(M),
M<N,
N MOD M <>0.
goal
so_nguyen_to(13).
```

Ví dụ 2: Tính n giai thừa.

```
Predicates
Facto (integer, integer)
Clauses
Facto(0,1):- !.
Facto(N, FactN) :- N > 0, M = N - 1, facto(M, factM), factN = N*factM.
```

Ở ví dụ trên ta đã định nghĩa một vị từ dùng để tính giá trị giai thừa của một số tự nhiên, đối số thứ nhất là số cần tính giai thừa và đối số thứ hai dùng để nhận giá trị trả về.

Trường hợp dùng ở đây được xác định bởi sự kiện 0 giai thừa là 1.

Để tính $N!$ ta tính $M!$ với $M = N-1$. Yếu tố dẫn đến trường hợp dùng là biến M có giá trị bằng N-1.

Ví dụ 3: Bài toán “Tháp Hà Nội”

%Program Thap_HaNoi

Domains

i = integer

s = symbol

Predicates

HaNoi(i, i)

ChuyenDia(i, i, s, s, s, i)

Clauses

HaNoi(N, SoLanDiChuyen) :- ChuyenDia(N, N, a, c, b, SoLanDiChuyen).

ChuyenDia(1, Nhan, Tu, Den, _, 1) :- !, nl,

write(“Chuyen dia: “, Nhan, “ tu “, Tu, “ den “, Den).

ChuyenDia(N, Nhan, Tu, Den, Tgian, Tong):-

N1=N-1, NhanPhu=Nhan-1,

ChuyenDia(N1, NhanPhu, Tu, TGian, Den, Tong1),

ChuyenDia(1, Nhan, Tu, Den, Tgian, 1),

ChuyenDia(N1, NhanPhu, TGian, Den, Tu, Tong2),

Tong = Tong1 + Tong2 + 1.

Goal %trong

makewindow(1, 3, 7, “Thap Ha Noi”, 0, 0, 25, 80),

write(“Nhap so dia n ($1 \leq n \leq 12$): “), readint(N), HaNoi(N, Tong),

write(“Tong so lan di chuyen dia: “, Tong)..

Kết quả của chương trình với $n=2$ là:

Chuyen dia: 1 tu a den b

Chuyen dia: 2 tu a den c

Chuyen dia: 1 tu b den c

Tong so lan di chuyen dia: 3

2.2 Tìm hiểu một môi trường lập trình Prolog (SWI-Prolog)

2.2.1 Giới thiệu về SWI-Prolog

Prolog là một ngôn ngữ lập trình. Tên gọi Prolog được xuất phát từ cụm từ tiếng Pháp Programmation en logique, nghĩa là lập trình logic. Prolog được sử dụng nhiều trong các ứng dụng trí tuệ nhân tạo và ngôn ngữ học trong khoa học máy tính.

SWI Prolog là một môi trường lập trình Prolog rất phổ biến, có các phiên bản chạy trên các hệ điều hành khác nhau như Windows, MacOS, Linux. SWI Prolog là một môi trường mã nguồn mở và thường được sử dụng trong dạy học.

SWI Prolog đã được phát triển mạnh mẽ do nhu cầu của các ứng dụng thế giới thực, được sử dụng rộng rãi trong nghiên cứu và giáo dục cũng như các ứng dụng thương mại. SWI Prolog hoạt động theo hệ thống đơn lẻ, có giao diện trao đổi 2 chiều linh hoạt



2.2.2 Cách sử dụng SWI-Prolog

2.2.2.1 Chương trình Prolog

Chương trình Prolog là một tập hợp các vị từ .

Vị từ xác định mối quan hệ giữa các đối số của chúng. Về mặt logic, một chương trình Prolog chỉ rõ những gì được lưu giữ .

Có một vài quy ước để viết các chương trình Prolog và các cách đọc chúng khác nhau .

Để chạy chương trình Prolog:

(Với hệ điều hành Windows , kích đúp lên tập tin chương trình, hoặc

Chạy phần mềm SWI-Prolog, và tham vấn (consult) tới tập tin chương trình

```
?- consult('C:\\PrologPrograms\\myPrologProg.pl').
```

Sau đó, đưa ra câu hỏi mong muốn

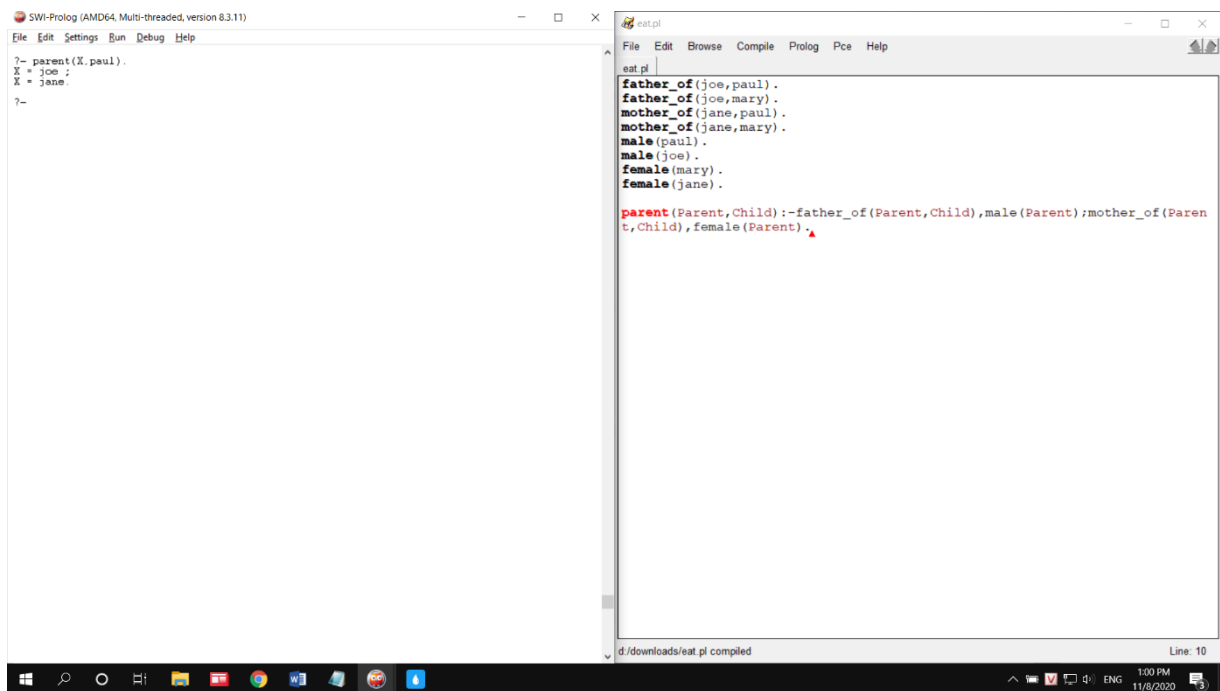
```
?- parent(X,paul).
```

Prolog đưa ra câu trả lời

```
X = joe;
```

```
X = jane.
```

Prolog cung cấp các giá trị cho các biến khi cần để có thể hoàn chỉnh một chứng minh



The screenshot shows the SWI-Prolog IDE with two windows. The left window displays a Prolog query: `?- parent(X,paul).` with bindings `X = joe ;` and `X = jane.`. The right window shows the source file `eat.pl` containing the following code:

```
father_of(joe,paul).
father_of(joe,mary).
mother_of(jane,paul).
mother_of(jane,mary).
male(paul).
male(joe).
female(mary).
female(jane).

parent(Parent,Child):-father_of(Parent,Child),male(Parent);mother_of(Parent,Child),female(Parent).
```

The status bar at the bottom indicates the file is `d:/downloads/eat.pl compiled` and the current line is 10.

Ta truy vấn câu các câu sau:

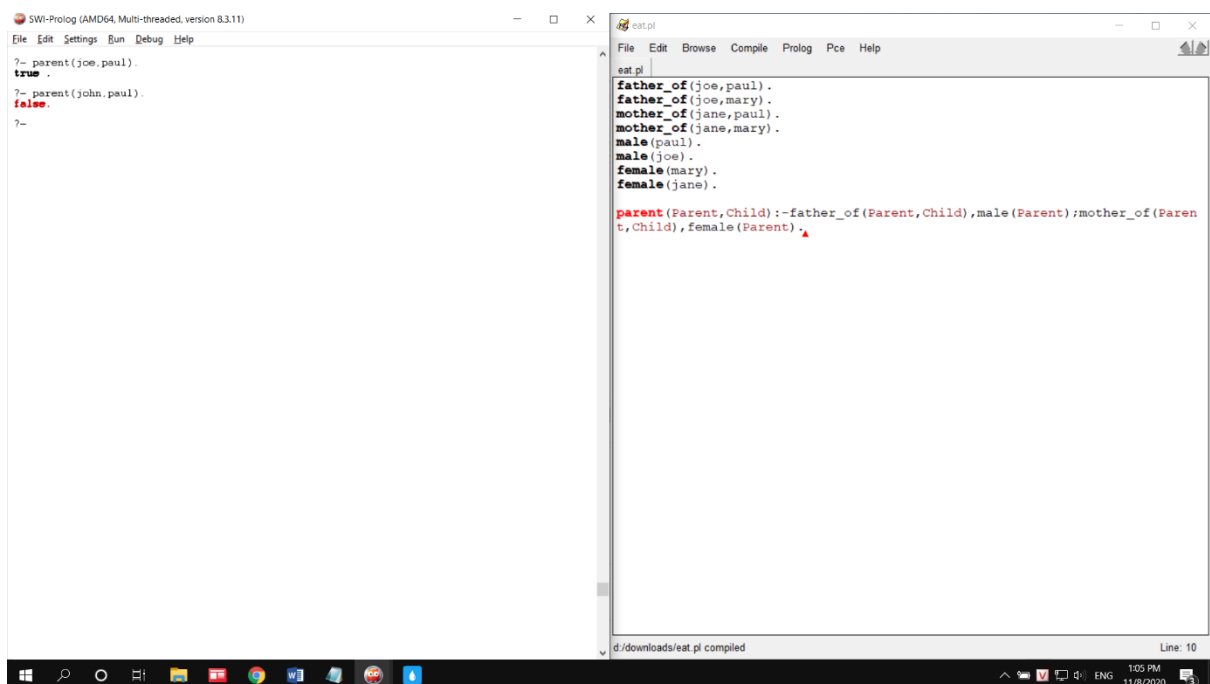
`?-parent(joe,paul).`

`true.`

`?-parent(john,paul).`

`false.`

Prolog trả về giá trị là `true` có nghĩa là chứng minh được, `false` có nghĩa là không thể chứng minh.



This screenshot shows the same SWI-Prolog IDE after executing the queries. The left window now displays the results: `?- parent(joe,paul).` returns `true.` and `?- parent(john,paul).` returns `false.`. The right window remains the same, showing the `eat.pl` source file.

2.2.2.2 Cú pháp trong SWI-Prolog

Một số ký hiệu cơ bản:

Ký hiệu	Ký hiệu trong Prolog
and	,
or	;
if	:-
not	not

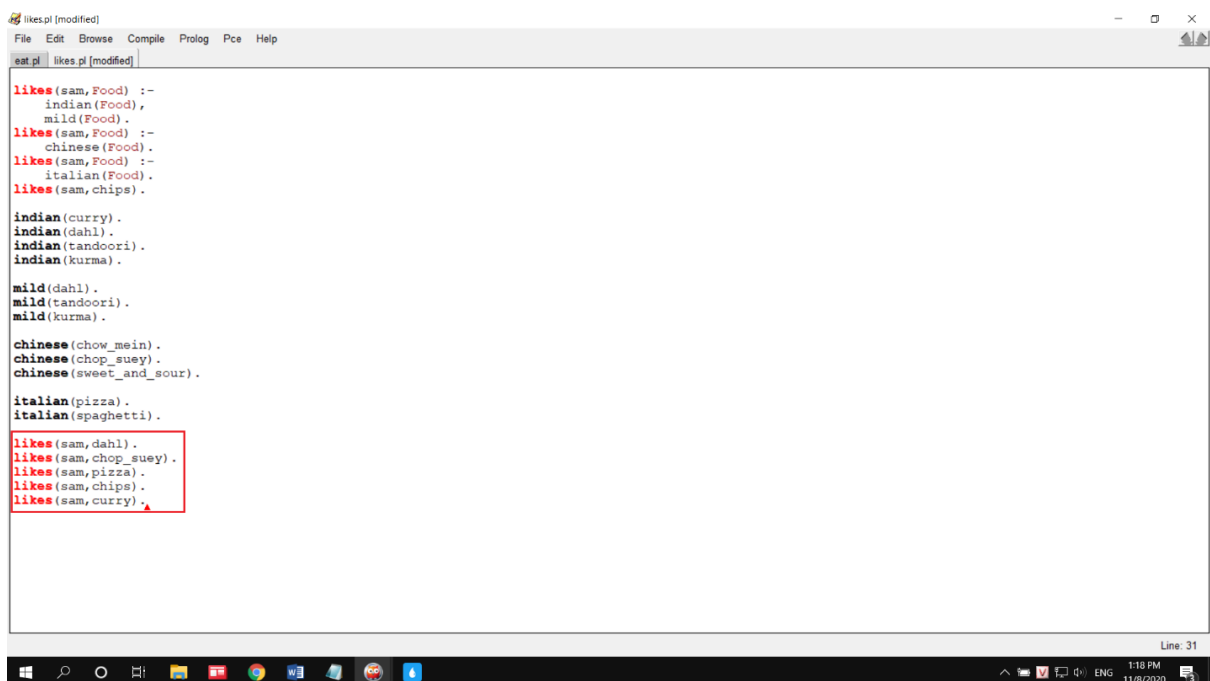
- Các cấu trúc
 - Một cấu trúc (structure) gồm một tên (name) và không, một, hoặc nhiều tham số (arguments) (arguments)
 - Bỏ đi cặp dấu ngoặc (), nếu như không có tham số
 - Các ví dụ của cấu trúc
 - dog.
 - monkey.
 - banana.
 - meat.
 - eat(dog,meat).
 - eat(monkey,banana).

```

eat.pl [modified]
File Edit Browse Compile Prolog Pce Help
eat.pl [modified]
dog.
monkey.
banana.
meat.
eat(dog,meat).
eat(monkey,banana).
  
```

- Tên biến trong Prolog bắt đầu bằng chữ in hoa. Tên vị từ và tên hàm bắt đầu bằng chữ thường. Tất cả các mệnh đề đều phải kết thúc bằng dấu chấm.

- Khi định nghĩa một vị từ trong Prolog, nếu không khai báo biến cụ thể thì mặc định chương trình hiểu biến đó mang giá trị “với mọi”.
 - Ví dụ: `eat(X, rice).` /* Mọi người đều thích ăn cơm */
- Luật (Rule) trong Prolog có bao gồm:
 - Một cấu trúc biểu diễn mệnh đề kết luận của luật
 - Ký hiệu: :-
 - Một danh sách các cấu trúc biểu diễn mệnh đề giả thiết của luật ngăn cách bởi dấu ‘,’. Dấu ‘,’ giữa các cấu trúc có nghĩa như toán tử logic AND
 - Ví dụ:
 - `likes(sam,dahl).`
 - `likes(sam,chop_suey).`
 - `likes(sam,pizza).`
 - `likes(sam,chips).`
 - `likes(sam,curry).`



```
likes.pl [modified]
File Edit Browse Compile Prolog Pcs Help

eat.pl likes.pl [modified]

likes(sam, Food) :-
    indian(Food),
    mild(Food).
likes(sam, Food) :-
    chinese(Food).
likes(sam, Food) :-
    italian(Food).
likes(sam, chips).

indian(curry).
indian(dahl).
indian(tandoori).
indian(kurma).

mild(dahl).
mild(tandoori).
mild(kurma).

chinese(chow_mein).
chinese(chop_suey).
chinese(sweet_and_sour).

italian(pizza).
italian(spaghetti).

likes(sam, dahl).
likes(sam, chop_suey).
likes(sam, pizza).
likes(sam, chips).
likes(sam, curry).
```

- Biến và hằng:
 - Các biến bắt đầu bằng chữ cái in hoa hoặc ký tự đặc biệt
Ví dụ: X, Dog, _parent
 - Các hằng không bắt đầu bằng các chữ cái in hoa hoặc các ký tự đặc biệt
Ví dụ: x, dog, parent
- Kiểu atom
 - Một chuỗi các ký tự được tạo thành từ các chữ hoa, chữ thường, chữ số và dấu gạch dưới và phải bắt đầu bằng một chữ thường
 - Một chuỗi các ký tự đặc biệt. Ví dụ @=, @==>, :-, ...
- Kiểu số
 - Các số nguyên được sử dụng nhiều trong Prolog
 - Cú pháp của nó rất đơn giản: 365, -1000, 0, 1, ...

- Chú thích
 - Được đặt trong cặp dấu: /*...*/
 - Hoặc được đặt sau dấu %
 - Ví dụ parent(x,y) %. sự kiện mô tả x là cha mẹ của y

2.2.2.3 Cách suy diễn lùi trong Prolog

Giả sử chúng ta có cơ sở tri thức (chương trình Prolog):

```
father_of(joe,paul).
father_of(joe,mary).
mother_of(jane,paul).
mother_of(jane,mary).
male(paul).
male(joe).
female(mary).
female(jane).
parent(Parent,Child):-
  father_of(Parent,Child),male(Parent);mother_of(Parent,Child),female(Parent).
```

Giả sử người dùng đặt câu hỏi

```
?-parent(X,paul).
```

Quá trình suy diễn lùi (Back chaining) của Prolog:

- father_of(X,paul) = father_of(joe,paul), X = joe
male(joe) => Thành công
- mother_of(X,paul) = mother_of(jane,paul), X = jane
female(joe) => Thành công

2.2.2.4 Các ví dụ minh họa

Ví dụ 1:

Ta có hệ cơ sở tri thức sau:

```
woman(mia).
woman(jody).
woman(yolanda).
playsAirGuitar(jody).
party.
```

Các câu hỏi:

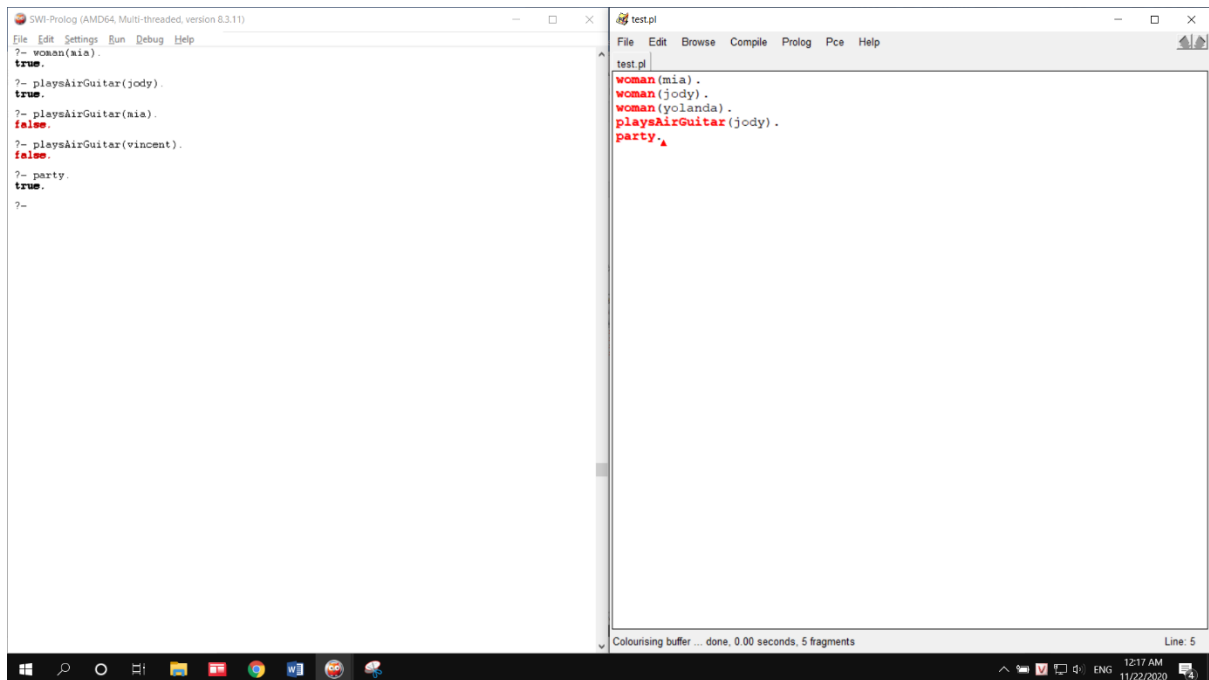
```
?- woman(mia).
```

```
?- playsAirGuitar(jody).
```

?- playsAirGuitar(mia).

?- playsAirGuitar(vincent).

?- party.



Ví dụ 2:

Ta có hệ cơ sở tri thức sau:

studies(charlie, csc135).

studies(olivia, csc135).

studies(jack, csc131).

studies(arthur, csc134).

teaches(kirke, csc135).

teaches(collins, csc131).

teaches(collins, csc171).

teaches(juniper, csc134).

professor(X, Y):-teaches(X, C), studies(Y, C).

Các câu hỏi:

?- studies(charlie, What).

?- professor(kirke, Students).

The screenshot shows the SWI-Prolog IDE with two windows. The left window is the REPL (Read-Eval-Print Loop) showing the execution of a Prolog program. The right window is the editor showing the source code file 'eat.pl'.

```

?~ professor(kirke, Students).
Students = charlie ;
Students = olivia.
?~

```

```

% eat.pl
studies(charlie, csc135).
studies(olivia, csc135).
studies(jack, csc131).
studies(arthur, csc134).

teaches(kirke, csc135).
teaches(collins, csc131).
teaches(collins, csc171).
teaches(juniper, csc134).

professor(X, Y):-teaches(X, C), studies(Y, C).

```

Ví dụ 3: Tính giai thừa

factorial(0,1).

factorial(N,F) :-

$N > 0$,

$N1$ is $N-1$,

factorial($N1$, $F1$),

F is $N * F1$.

The screenshot shows the SWI-Prolog IDE with two windows. The left window is the REPL showing the execution of the factorial program. The right window is the editor showing the source code file 'eat.pl'.

```

?~ factorial(3, F).
F = 6.
?~ factorial(3, 6).
true.
?~

```

```

% eat.pl
factorial(0, 1).

factorial(N, F) :-
    N > 0,
    N1 is N-1,
    factorial(N1, F1),
    F is N * F1.

```

Ví dụ 4:

Ta có hệ cơ sở tri thức sau:

male(james1).
male(charles1).
male(charles2).
male(james2).
male(george1).
female(catherine).
female(elizabeth).
female(sophia).
parent(charles1, james1).
parent(elizabeth, james1).
parent(charles2, charles1).
parent(catherine, charles1).
parent(james2, charles1).
parent(sophia, elizabeth).
parent(george1, sophia).

Các câu hỏi:

?- parent(charles1, george1).
?- parent(charles1,X).
?- parent(X,charles1).
?- male(john).
?- female(sophia).

```

SWI-Prolog (AMD64 Multi-threaded, version 8.3.11)
File Edit Settings Run Debug Help
% c:/Users/Admin/OneDrive/Documents/Prolog/test.pl compiled 0.00 sec, 10 clauses
?- parent(charles1, george1).
false.
?- parent(charles1, X).
X = james1.
?- parent(X, charles1).
X = charles2 ;
X = catherine ;
X = james2.
?- male(john).
false.
?- female(sophia).
true.
?-

test.pl
File Edit Browse Compile Prolog Pce Help
test.pl
male(james1).
male(charles1).
male(charles2).
male(james2).
male(george1).

female(catherine).
female(elizabeth).
female(sophia).

parent(charles1, james1).
parent(elizabeth, james1).
parent(charles2, charles1).
parent(catherine, charles1).
parent(james2, charles1).
parent(sophia, elizabeth).
parent(george1, sophia).

```

Ví dụ 5: Đổi một đô la

change([H,Q,D,N,P]) :-

```

member(H,[0,1,2]),          /* Half-dollars */
member(Q,[0,1,2,3,4]),      /* quarters  */
member(D,[0,1,2,3,4,5,6,7,8,9,10]) , /* dimes    */
member(N,[0,1,2,3,4,5,6,7,8,9,10,    /* nickels   */
11,12,13,14,15,16,17,18,19,20]),

S is 50*H + 25*Q +10*D + 5*N,

S =< 100,

P is 100-S.

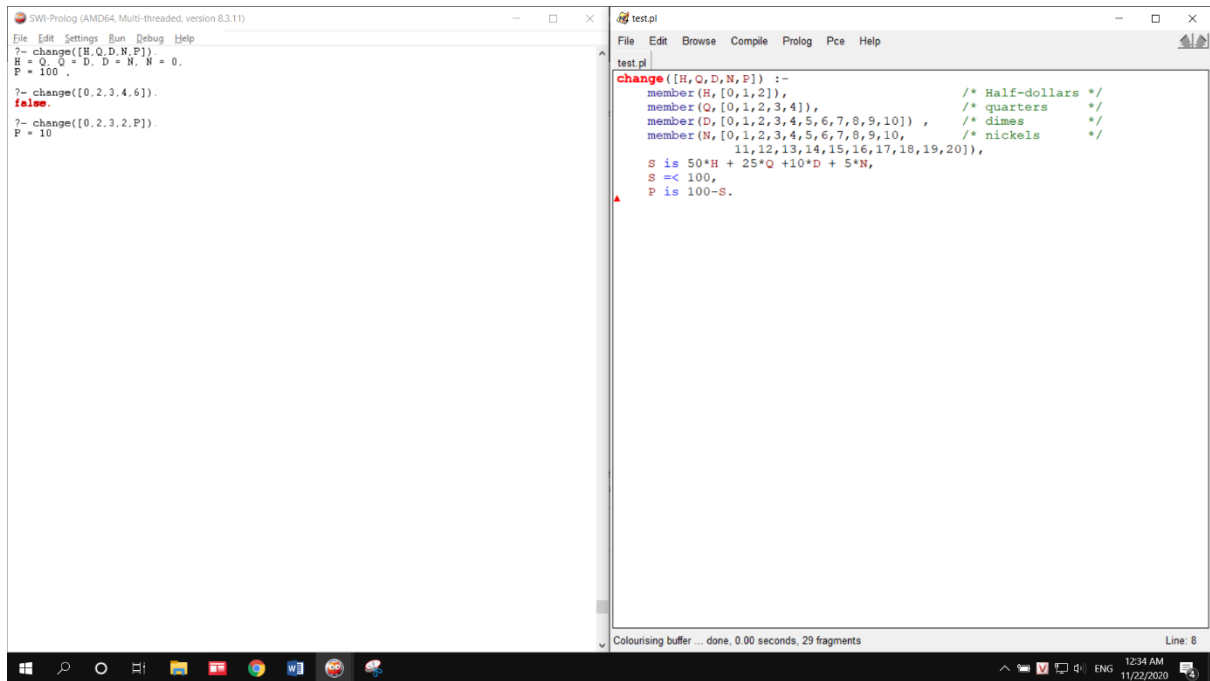
```

Các câu hỏi:

?- change([H,Q,D,N,P]).

?- change([0,2,3,4,6]).

?- change([0,2,3,2,P]).



2.3 Xây dựng cây phả hệ Hoàng gia Anh

2.3.1 Nhóm vị từ định nghĩa sẵn

`parent(Parent, Child)`: Parent là bố mẹ của Child

`male(Person)`: Person là nam

`female(Person)`: Person là nữ

`married(Person, Person)`: Person kết hôn với Person

`divorced(Person, Person)`: Person ly hôn với Person

Với nhóm vị từ này, lưu ý mối quan hệ tình trạng hôn nhân cần định nghĩa cả hai fact: `married(X, Y)` và `married(Y, X)`; `divorced(X, Y)` và `divorced(Y, X)`.

2.3.2 Nhóm vị từ được suy diễn

`husband(Person, Wife)`: Person là nam và đang kết hôn với Wife

`wife(Person, Husband)`: Person là nữ và đang kết hôn với Husband.

`father(Parent, Child)`: Parent là nam và là bố của Child.

`mother(Parent, Child)`: Parent là nữ và là mẹ của đối tượng Child.

`child(Child, Parent)`: Child có bố mẹ là Parent.

`son(Child, Parent)`: Child là nam và là con trai của Parent.

`daughter(Child, Parent)`: Child là nữ và là con gái của Parent.

grandparent(GP, GC): GP là bố mẹ của của bố mẹ của GC.

grandmother(GM, GC): GP là nữ và bà của GC.

grandfather(GF, GC): GP là nam và ông của GC.

grandchild(GC, GP): GC là có ông/bà là GP.

grandson(GS, GP): GS là nam và là cháu trai của GP.

granddaughter(GD, GP): GD là nữ và là cháu gái của GP.

sibling(Person1, Person2): Person1 khác Person2, Person1 và Person2 có cùng bố mẹ

brother(Person, Sibling): Person là nam và là anh/em ruột với Sibling.

sister(Person, Sibling): Person là nữ và là chị/em ruột với Sibling.

aunt(Person, NieceNephew): Person là nữ, là chị/em của ba/mẹ hoặc là vợ của chú của NieceNephew.

uncle(Person, NieceNephew): Person là nam, là anh/em của ba/mẹ hoặc là chồng của dì của NieceNephew.

niece(Person, AuntUncle): Person là nữ và có dì/chú là AuntUncle.

nephew(Person, AuntUncle): Person là nam và có dì/chú là AuntUncle.

2.3.3 Bộ câu hỏi truy vấn và câu trả lời

1. Ai là chồng của Camilla Parker Bowles?

?- husband(X,camillaParkerBowles).

X = princeCharles.

2. Princess Anne có phải là vợ của Captain Mark Phillips không?

?- wife(princessAnne, captainMarkPhillips).

false.

3. Ai là bố của Mia Grace Tindall?

?- father(X,miaGraceTindall).

X = mikeTindall.

4. Ai là mẹ của Prince Charles?

?- mother(X,princeCharles).

X = queenElizabethII.

5. *Prince Harry có phải là con của Princess Diana không?*

?- child(princeHarry, princessDiana).

true.

6. *Ai là con trai của Prince William*

?- son(X,princeWilliam).

X = princeGeorge.

7. *Ai là con gái của Peter Phillips?*

?- daughter(X,peterPhillips).

X = savannahPhillips ;

X = islaPhillips.

8. *Ai ông bà của Mia Grace Tindall là ai?*

?- grandparent(X,miaGraceTindall).

X = captainMarkPhillips ;

X = princessAnne.

9. *Ai là bà ngoại của Zara Phillips?*

?- grandmother(X,zaraPhillips).

X = queenElizabethII.

10. *Timothy Laurence có phải là ông của Isla Phillips không?*

?- grandfather(timothyLaurence,islaPhillips).

false.

11. *Ai là cháu của Prince Phillip?*

?- grandchild(X,princePhillip).

X = princeWilliam ;

X = princeHarry ;

X = peterPhillips ;

X = zaraPhillips ;

X = princessBeatrice ;

X = princessEugenie ;

X = jamesViscountSevern ;

X = ladyLouiseMountbatten_Windsor.

12. Prince George có phải là cháu trai của Princess Diana không?

?- grandson(princeGeorge,princessDiana).

true.

13. Ai là cháu gái của Sarah Ferguson?

?- granddaughter(X, sarahFerguson).

false.

14. Ai là anh chị em của Prince Charles?

?- sibling(X,princeCharles).

X = princessAnne ;

X = princeAndrew ;

X = princeEdward.

15. Prince Andrew có phải là anh em trai của Prince Edward không?

?- brother(princeAndrew,princeEdward).

true.

16. Ai là chị em gái của Princess Beatrice?

?- sister(X,princessBeatrice).

X = princessEugenie.

17. Ai là cô/dì của Prince William?

?- aunt(X, princeWilliam).

X = princessAnne ;

X = sophieRhys-jones.

18. Peter Phillips có phải là chú của Princess Charlotte

?- uncle(peterPhillips, princessCharlotte).

false.

19. Ai là cháu gái của Kate Middleton(cô/dì)?

?- niece(X,kateMiddleton).

false.

20. Ai là cháu trai của PrincessAnne

?- nephew(X,princessAnne).

X = princeWilliam ;

X = princeHarry ;

X = jamesViscountSevern.

21. Ai là anh chị em của James Viscount Severn?

?- sibling(jamesViscountSevern,X).

X = ladyLouiseMountbatten_Windsor.

22. Ai là con của của Autumn Kelly?

?- child(X,autumnKelly).

X = savannahPhillips ;

X = islaPhillips.

23. Chồng của Kate Middleton?

?- husband(X,kateMiddleton).

X = princeWilliam.

24. Prince Charles có phải là ông của Princess Charlotte không?

?- grandparent(princeCharles,princessCharlotte).

true.

25. Zara Phillips có phải là cháu gái của Queen Elizabeth II không?

?- granddaughter(zaraPhillips,queenElizabethII).

true.

Phần III: Xây dựng cơ sở tri thức với công cụ Prolog

3.1 Tóm tắt cơ sở tri thức về chủ đề cơ cấu trường KHTN

- to_chuc: DHQGHCM.
- truong: KHTN.
- khoa: có thể là một trong 9 khoa của trường như là CNTT, Toán – Tin học, Vật lý – Vật lý kỹ thuật, Sinh học – Công nghệ sinh học, Điện tử – Viễn thông, Hóa học, Địa chất, Môi trường, Khoa học và Công nghệ vật liệu.
- bo_mon: có thể là bộ môn của khoa CNTT như: Hệ thống thông tin, Mạng máy tính và Viễn thông, Công nghệ phần mềm, Thị giác máy tính và điều khiển học thông minh, Khoa học máy tính, Công nghệ tri thức; có thể là bộ môn của khoa Toán – Tin học: Ứng dụng tin học, Xác suất thống kê, Đại số, Giải tích, Tối ưu và Hệ thống, Giáo dục Toán học, Tài chính Dinh lương; có thể là bộ môn của khoa Vật lý – Vật lý kỹ thuật: Vật lý Tin học, Vật lý Lý thuyết, Vật lý Ứng dụng, Vật lý Địa cầu, Vật lý Điện tử, Vật lý Chất rắn, Vật lý Hạt nhân, Hai dương học – Khí tượng – Thủy văn; có thể là bộ môn của khoa Sinh học – Công nghệ sinh học: CNSH Phân tử và MT, Sinh lý Thực vật, Sinh hóa, Di truyền, CNSH Thực vật và Chuyển hóa SH, Vi sinh, Sinh lý học và Công nghệ Sinh học học Động vật, Sinh thái và Sinh học Tiến hóa; có thể là bộ môn của khoa Hóa học: Hóa phân tích, Vô cơ và Ứng dụng, Hữu cơ, Hóa lý, Hóa học Polymer; có thể là bộ môn của khoa Địa chất: Địa chất biển và Dầu khí, Địa chất cơ sở, Địa chất thủy văn – Địa chất công trình, Thạch ngọc và Khoáng sản; có thể là bộ môn của khoa Môi trường: Khoa học môi trường, Công nghệ môi trường, Quản lý môi trường, Tin học môi trường; có thể là bộ môn của khoa Khoa học và Công nghệ vật liệu: Vật liệu Polymer và Composite, Vật liệu Nano và Mạng mong, Vật liệu tu và Y sinh
- ngành: có thể là một ngành của khoa CNTT: Khoa học máy tính, Kỹ thuật phần mềm, Hệ thống thông tin, Công nghệ thông tin.
- chuyên ngành: có thể là một chuyên ngành của khoa CNTT: Kỹ thuật phần mềm, Hệ thống thông tin, Mạng máy tính, Công nghệ thông tin, Khoa học máy tính, Thị giác máy tính, Công nghệ tri thức, Khoa học dữ liệu, An toàn thông tin.
- giang_vien: là giáo viên giảng dạy tại trường KHTN.
- nam: giới tính nam
- nu: giới tính nữ

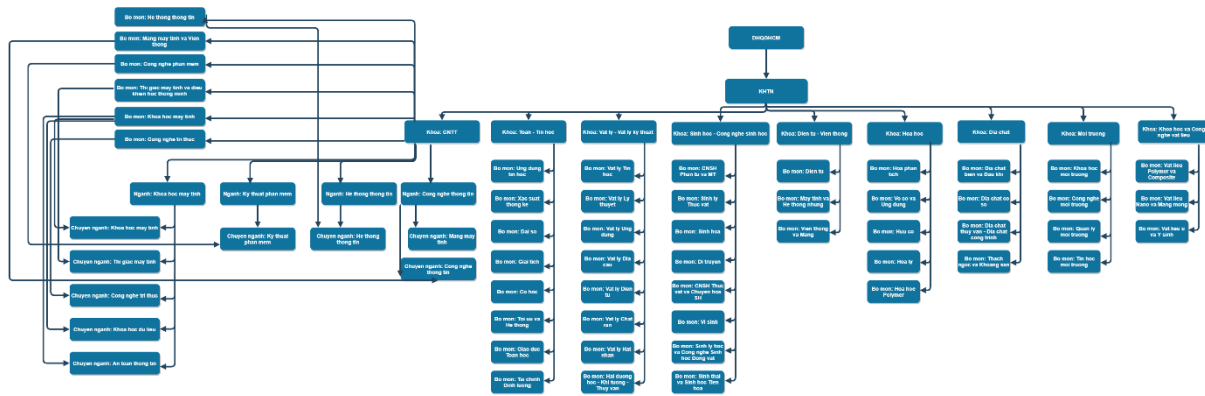
3.2 Các quan hệ chính

- truong_khoa(_giangVien, _khoa): _giangVien giữ chức trưởng khoa của _khoa hay không.
- pho_khoa(_giangVien, _khoa): _giangVien giữ chức phó khoa của một _khoa hay không.
- truong_bo_mon(_giangVien, _boMon): _giangVien giữ chức trưởng bộ môn của _boMon hay không.
- pho_bo_mon(_giangVien, _boMon): _giangVien giữ chức phó bộ môn của _boMon hay không.

- `hoc_ham(_giangVien, _hocHam)`: `_giangVien` có học hàm là `_hocHam` hay không.
- `hoc_vi(_giangVien, _hocVi)`: `_giangVien` có học vị `_hocVi` hay không.
- `truc_thuoc(A,B)`: A có trực thuộc B hay không
- `phong(_boMon, _phong)`: bộ môn `_boMon` ở phòng `_phong` hay không.
- `nam_thanh_lap(A,B)`: A được thành lập năm B hay không

3.3 Các quan hệ mở rộng:

- `giang_vien_thuoc_bo_mon(_giangVien, _boMon)`: `_giangVien` thuộc bộ môn `_boMon`
- `giang_vien_thuoc_khoa(_giangVien, _khoa)`: `_giangVien` thuộc khoa `_khoa`
- `giang_vien_thuoc_truong(_giangVien, _truong)`: `_giangVien` thuộc trường `_truong`
- `bo_mon_thuoc_khoa(_boMon, _khoa)`: bộ môn `_boMon` thuộc khoa `_khoa`
- `bo_mon_thuoc_truong(_boMon, _truong)`: bộ môn `_boMon` thuộc `_truong`
- `khoa_thuoc_truong(_khoa, _truong)`: khoa `_khoa` có thuộc trường `_truong` hay không.
- `truong_thuoc(_truong, _tochuc)`: trường `_truong` có thuộc tổ chức `_tochuc` hay không
- `nganh_cung_khoa(_nganh01, _nganh02)`: ngành `_nganh01` và ngành `_nganh02` có cùng khoa hay không.
- `chuyen_nganh_cung_nganh(_chuyenNganh01, _chuyenNganh02)`: chuyên ngành `_chuyenNganh01` và chuyên ngành `_chuyenNganh02` có cùng ngành hay không.
- `giang_vien_cung_bo_mon(_giangVien1, _giangVien2)`: `_giangVien1` và `_giangVien2` có cùng bộ môn hay không
- `giang_vien_cung_khoa(_giangVien1, _giangVien2)`: `_giangVien1` và `_giangVien2` có cùng khoa hay không
- `giang_vien_cung_truong(_giangVien1, _giangVien2)`: `_giangVien1` và `_giangVien2` có cùng trường hay không
- `bo_mon_cung_khoa(_boMon1, _boMon2)`: `_boMon1` và `_boMon2` có cùng khoa hay không
- `cung_la_giao_su(_giangVien01, _giangVien02)`: `_giangVien01` và `_giangVien02` có cùng là Giáo sư hay không
- `cung_la_pho_giao_su(_giangVien01, _giangVien02)`: `_giangVien01` và `_giangVien02` có cùng là Phó Giáo sư hay không
- `cung_la_tien_si(_giangVien01, _giangVien02)`: `_giangVien01` và `_giangVien02` có cùng là Tiến sĩ hay không
- `cung_la_thac_si(_giangVien01, _giangVien02)`: `_giangVien01` và `_giangVien02` có cùng là Thạc sĩ hay không



*Do ảnh lưu vào Word bị mờ

Link drive ảnh: https://drive.google.com/file/d/1HL1KLdnykvWiDc-vRyXc958xdI9U_2dx/view?usp=sharing

Link file gốc sơ đồ: https://drive.google.com/file/d/1HL1KLdnykvWiDc-vRyXc958xdI9U_2dx/view?usp=sharing

3.4 Bộ test cho hệ cơ sở tri thức

1. Thầy Lê Ngọc Thành có phải là giảng viên không?

?- giang_vien(le_Ngoc_Thanh).

true.

2. Thị giác máy tính và điều khiển học thông minh có trực thuộc khoa CNTT hay không?

?- truc_thuoc(thi_giac_may_tinh_va_dieu_khien_hoc_thong_minh,cNTT).

true.

3. Trưởng bộ môn Thị giác máy tính và điều khiển học thông minh có phải là thầy Lý Quốc Ngọc?

?- truong_bo_mon(X,thi_giac_may_tinh_va_dieu_khien_hoc_thong_minh).

X = ly_Quoc_Ngoc.

4. Ngành/ Bộ môn trực thuộc khoa Vật lý - Vật lý kỹ thuật

?- truc_thuoc(X,vatly-vatlykythuat).

X = vat_ly_Tinhoc ;

X = vat_ly_Ly_thuyet ;

X = vat_ly_Ung_dung ;

X = vat_ly_Dia_cau ;

X = vat_ly_Dien_tu ;

X = vat_ly_Chat_ran ;

X = vat_ly_Hat_nhan ;

X = hai_duong_hoc-khi_tuong-thuy_van.

5. Thầy Lê Vũ Tuấn Hùng có phải là trưởng khoa Vật lý - Vật lý kỹ thuật hay không?

?- truong_khoa(le_Vu_Tuan_Hung,vatly-vatlykythuat).

true.

6. Bộ môn Tối ưu và Hệ thống có trực thuộc trong khoa Toán hay không?

?- truc_thuoc(toi_uu_va_He_thong,toan-tinhoc).

true.

7. Những giảng viên nào đang có học vị Tiến sĩ

?- hoc_vi(X,tS).

X = dinh_Ba_Tien ;

X = lam_Quang_Vu ;

X = nguyen_Van_Vu ;

X = le_Ngoc_Thanh ;

X = le_Hoai_Bac ;

X = dinh_Dien ;

X = nguyen_Dinh_Thuc ;

X = nguyen_Tran_Minh_Thu ;

X = pham_Nguyen_Cuong ;

X = ly_Quoc_Ngoc ;

X = tran_Thai_Son ;

X = nguyen_Thi_Minh_Tuyen ;

X = tran_Trung_Dung ;

X = huynh_Thi_Bao_Tran ;

X = le_Vu_Tuan_Hung ;

X = huynh_Truc_Phuong ;
X = huynh_Van_Tuan ;
X = nguyen_Quoc_Khanh ;
X = nguyen_Thanh_Van ;
X = nguyen_Van_Hieu ;
X = tran_Quang_Trung ;
X = chau_Van_Tao ;
X = truong_Thi_Hong_Loan ;
X = vo_Luong_Hong_Phuoc ;
X = huynh_Quang_Vu ;
X = nguyen_Le_Hoang_Anh ;
X = pham_The_Bao ;
X = dang_Duc_Trong ;
X = mai_Hoang_Bien ;
X = trinh_Anh_Ngoc ;
X = nguyen_Viet_Dong ;
X = dinh_Ngoc_Thanh ;
X = huynh_Huu_Thuan ;
X = bui_Trong_Tu ;
X = dang_Le_Kha ;
X = nguyen_Thi_Thanh_Mai ;
X = nguyen_Cong_Tranh ;
X = nguyen_Van_Dong ;
X = le_Tien_Khoa ;
X = nguyen_Trung_Nhan ;
X = ton_That_Quang ;
X = nguyen_Thai_Hoang ;

X = hoang_Ngoc_Cuong ;

X = nguyen_Tri_Nha ;

X = quach_Ngo_Diem_Phuong ;

X = tran_Van_Hieu ;

X = dang_Thi_Phuong_Thao ;

X = tran_Thanh_Huong ;

X = ngo_Dai_Nghiep ;

X = nguyen_Thuy_Vy ;

X = nguyen_Duc_Hoang ;

X = tran_Le_Bao_Ha ;

X = nguyen_Thi_Kim_Dung ;

X = pham_Trung_Hieu ;

X = tran_Thi_Hoang_Ha ;

X = le_Duc_Phuc ;

X = nguyen_Kim_Hoang ;

X = tran_Thi_Thanh_Van ;

X = ha_Thuc_Chi_Nhan ;

X = hoang_Thi_Dong_Quy ;

X = pham_Kim_Ngoc ;

X = le_Van_Hieu ;

X = to_Thi_Hien ;

X = dao_Nguyen_Khoi ;

X = le_Tu_Thanh ;

X = truong_Thi_Cam_Trang ;

X = nguyen_Bich_Ngoc ;

X = duong_Thi_Thuy_Nga.

8.Học vị của thầy Huỳnh Hữu Thuận có phải là Giáo sư?

?- hoc_vi(huynh_Huu_Thuan,gS).

false.

9. Phòng bộ môn Khoa học máy tính nằm ở I81 phải không?

?- phong(khoa_hoc_may_tinh,i81).

true.

10. Những ai đang có học hàm Phó Giáo Sư?

?- hoc_ham(X,pGS).

X = nguyen_Dinh_Thuc ;

X = dinh_Dien ;

X = ly_Quoc_Ngoc ;

X = le_Vu_Tuan_Hung ;

X = huynh_Truc_Phuong ;

X = huynh_Van_Tuan ;

X = nguyen_Quoc_Khanh ;

X = nguyen_Thanh_Van ;

X = nguyen_Van_Hieu ;

X = tran_Quang_Trung ;

X = vo_Luong_Hong_Phuoc ;

X = pham_The_Bao ;

X = dinh_Ngoc_Thanh ;

X = nguyen_Thi_Thanh_Mai ;

X = nguyen_Van_Dong ;

X = nguyen_Trung_Nhan ;

X = nguyen_Thai_Hoang ;

X = hoang_Ngoc_Cuong ;

X = quach_Ngo_Diem_Phuong ;

X = tran_Van_Hieu ;

X = dang_Thi_Phuong_Thao ;

X = tran_Thanh_Huong ;

X = ngo_Dai_Nghiep ;

X = nguyen_Duc_Hoang ;

X = tran_Le_Bao_Ha ;

X = pham_Trung_Hieu ;

X = tran_Thi_Thanh_Van ;

X = ha_Thuc_Chi_Nhan ;

X = hoang_Thi_Dong_Quy ;

X = to_Thi_Hien.

11. Những ai đang có học hàm Giáo Sư?

?- hoc_ham(X,gS).

X = le_Hoai_Bac ;

X = chau_Van_Tao ;

X = dang_Duc_Trong ;

X = le_Van_Hieu.

12. Trường ta có những khoa nào?

?- khoa(X).

X = cNTT ;

X = vatly-vatlykythuat ;

X = toan-tinhoc ;

X = dientu-vienthong ;

X = hoa_hoc ;

X = sinh_hoc-cong_nghe_sinh_hoc ;

X = dia_chat ;

X = khoa_hoc_va_Cong_nghe_vat_lieu ;

X = moi_truong.

13. Ngành/ bộ môn trực thuộc khoa Địa chất là những ngành/ bộ môn gì?

?- `truc_thuoc(X,dia_chat)`.

`X = dia_chat_bien_va_Dau_khi ;`

`X = dia_chat_co_so ;`

`X = dia_chat_Thuy_van-dia_chat_cong_trinh ;`

`X = thach_hoc_va_Khoang_san.`

14. Những ai là những giảng viên nam?

?- `nam(X)`.

`X = dinh_Ba_Tien ;`

`X = lam_Quang_Vu ;`

`X = ban_Chi_Nam ;`

`X = nguyen_Van_Vu ;`

`X = le_Hoai_Bac ;`

`X = le_Ngoc_Thanh ;`

`X = nguyen_Dinh_Thuc ;`

`X = dinh_Dien ;`

`X = pham_Nguyen_Cuong ;`

`X = ly_Quoc_Ngoc ;`

`X = tran_Thai_Son ;`

`X = tran_Trung_Dung ;`

`X = huynh_Huu_Thuan ;`

`X = cao_Tran_Bao_Thuong ;`

`X = bui_Trong_Tu ;`

`X = dang_Le_Kha ;`

`X = nguyen_Cong_Tranh ;`

`X = nguyen_Van_Dong ;`

`X = le_Tien_Khoa ;`

X = nguyen_Trung_Nhan ;

X = ton_That_Quang ;

X = nguyen_Thai_Hoang ;

X = ho_Pham_Anh_Vu ;

X = hoang_Ngoc_Cuong ;

X = nguyen_Tri_Nha ;

X = tran_Van_Hieu ;

X = ngo_Dai_Nghiep ;

X = nguyen_Duc_Hoang ;

X = pham_Trung_Hieu ;

X = le_Duc_Phuc ;

X = ngo_Minh_Thien ;

X = nguyen_Kim_Hoang ;

X = ha_Thuc_Chi_Nhan ;

X = le_Van_Hieu ;

X = dao_Nguyen_Khoi ;

X = le_Tu_Thanh.

15. Những ai là những giảng viên nữ?

?- nu(X).

X = nguyen_Tran_Minh_Thu ;

X = huynh_ThiBao_Tran ;

X = nguyen_Thi_Minh_Tuyen ;

X = nguyen_Thi_Thanh_Mai ;

X = nguyen_Thu_Huong ;

X = quach_Ngo_Diem_Phuong ;

X = dang_Thi_Phuong_Thao ;

X = tran_Thanh_Huong ;

X = nguyen_Thuy_Vy ;

X = tran_Le_Bao_Ha ;

X = nguyen_Thi_Kim_Dung ;

X = tran_Thi_Hoang_Ha ;

X = ngo_Thi_Phuong_Uyen ;

X = tran_Thi_Thanh_Van ;

X = tran_Thi_Minh_Thu ;

X = hoang_Thi_Dong_Quy ;

X = pham_Kim_Ngoc ;

X = ta_Thi_Kieu_Hanh ;

X = to_Thi_Hien ;

X = truong_Thi_Cam_Trang ;

X = nguyen_Bich_Ngoc ;

X = duong_Thi_Thuy_Nga.

16. Ngành/ bộ môn nào thuộc quản lý của khoa CNTT

?- truc_thuoc(X,cNTT).

X = cong_nghe_tri_thuc ;

X = khoa_hoc_may_tinh ;

X = cong_nghe_phan_mem ;

X = thi_giac_may_tinh_va_dieu_khien_hoc_thong_minh ;

X = mang_may_tinh_va_Vien_thong ;

X = he_thong_thong_tin ;

X = he_thong_thong_tin ;

X = khoa_hoc_may_tinh ;

X = ky_thuat_phan_mem ;

X = cong_nghe_thong_tin.

17. Phó khoa CNTT là những ai/ ai?

?- pho_khoa(X,cNTT).

X = lam_Quang_Vu ;

X = van_Chi_Nam ;

X = nguyen_Van_Vu.

18. Phó khoa Hóa học là những ai/ ai?

?- pho_khoa(X,hoa_hoc).

X = nguyen_Cong_Tranh ;

X = nguyen_Thu_Huong.

19. Những bộ môn nào thuộc khoa sinh học?

?- truc_thuoc(X,sinh_hoc-cong_nghe_sinh_hoc).

X = cNSH_Phan_tu_va_MT ;

X = sinh_ly_Thuc_vat ;

X = sinh_hoa ;

X = di_truyen ;

X = cNSH_Thuc_vat_va_Chuyen_hoa_SH ;

X = vi_sinh ;

X = sinh_ly_hoc_va_Cong_nghe_Sinh_hoc_Dong_vat ;

X = sinh_thai_va_Sinh_hoc_Tien_hoa

20. Những bộ môn nào thuộc khoa Vật liệu?

?- truc_thuoc(X,khoa_hoc_va_Cong_nghe_vat_lieu).

X = vat_lieu_Polymer_va_Composite ;

X = vat_lieu_Nano_va_Mang_mong ;

X = vat_lieu_tu_va_Y_sinh.

Phần IV: Cài đặt hệ thống suy diễn logic bằng ngôn ngữ lập trình

4.1 Các thuật toán sử dụng

Sử dụng ngôn ngữ lập trình Python để cài đặt hệ thống suy diễn logic

Mô tả lại hoạt động của 2 hàm có mã giả như sau

Hàm đồng nhất Unify(x,y,theta):

```

function UNIFY( $x, y, \theta$ ) returns a substitution to make  $x$  and  $y$  identical
inputs:  $x$ , a variable, constant, list, or compound expression
           $y$ , a variable, constant, list, or compound expression
           $\theta$ , the substitution built up so far (optional, defaults to empty)

if  $\theta$  = failure then return failure
else if  $x = y$  then return  $\theta$ 
else if VARIABLE?( $x$ ) then return UNIFY-VAR( $x, y, \theta$ )
else if VARIABLE?( $y$ ) then return UNIFY-VAR( $y, x, \theta$ )
else if COMPOUND?( $x$ ) and COMPOUND?( $y$ ) then
    return UNIFY( $x$ .ARGS,  $y$ .ARGS, UNIFY( $x$ .OP,  $y$ .OP,  $\theta$ ))
else if LIST?( $x$ ) and LIST?( $y$ ) then
    return UNIFY( $x$ .REST,  $y$ .REST, UNIFY( $x$ .FIRST,  $y$ .FIRST,  $\theta$ ))
else return failure

function UNIFY-VAR( $var, x, \theta$ ) returns a substitution

if  $\{var/val\} \in \theta$  then return UNIFY( $val, x, \theta$ )
else if  $\{x/val\} \in \theta$  then return UNIFY( $var, val, \theta$ )
else if OCCUR-CHECK?( $var, x$ ) then return failure
else return add  $\{var/x\}$  to  $\theta$ 
  
```

Hàm này có nhiệm vụ như sau:

- Đồng nhất biến/hằng/danh sách/mệnh đề, sau đó trả về một phép thế theta phù hợp.

Input: biến/hằng/danh sách/mệnh đề Output: Phép thế theta phù hợp

Thuật toán

- Nếu theta trả về là một phép thế rỗng, tức 2 biến/hằng/danh sách/mệnh đề hoàn toàn giống nhau
- Nếu theta trả về False, nghĩa là không thể đồng nhất 2 biến/hằng/danh sách/mệnh đề được
- Nếu theta trả về 1 dictionary có dạng $\{X: 'value', \dots\}$ tức là để hợp nhất 2 biến/hằng/danh sách/mệnh đề ta sử dụng phép thế $X=value, \dots$

Hàm suy diễn tiến FOL-FC-ASK(KB,a) :

```

function FOL-FC-ASK( $KB, \alpha$ ) returns a substitution or false
inputs:  $KB$ , the knowledge base, a set of first-order definite clauses
          $\alpha$ , the query, an atomic sentence
local variables:  $new$ , the new sentences inferred on each iteration

repeat until  $new$  is empty
   $new \leftarrow \{ \}$ 
  for each  $rule$  in  $KB$  do
     $(p_1 \wedge \dots \wedge p_n \Rightarrow q) \leftarrow \text{STANDARDIZE-VARIABLES}(rule)$ 
    for each  $\theta$  such that  $\text{SUBST}(\theta, p_1 \wedge \dots \wedge p_n) = \text{SUBST}(\theta, p'_1 \wedge \dots \wedge p'_n)$ 
      for some  $p'_1, \dots, p'_n$  in  $KB$ 
         $q' \leftarrow \text{SUBST}(\theta, q)$ 
        if  $q'$  does not unify with some sentence already in  $KB$  or  $new$  then
          add  $q'$  to  $new$ 
           $\phi \leftarrow \text{UNIFY}(q', \alpha)$ 
          if  $\phi$  is not fail then return  $\phi$ 
  add  $new$  to  $KB$ 
return false

```

Hàm này có nhiệm vụ như sau:

- Làm giàu cơ sở tri thức(KB) bằng cách kích hoạt các luật(rule) và sinh ra các tri thức mới(q), bổ sung vào cơ sở tri thức đến khi nào câu hỏi được trả lời.
- Lặp lại các bước sau đến khi nào không có tri thức mới được tạo ra hoặc trả lời được câu hỏi yêu cầu:

Input: Cơ sở tri thức (KB) và câu truy vấn (query)

Output: Các tri thức thỏa mãn truy vấn

Thuật toán

- Với mỗi rule trong KB ($p_1, p_2, \dots, p_n \Rightarrow q$) các phép thế theta với $p_1=p'_1, p_2=p'_2, p_3=p'_3, \dots, p_n=p'_n$ với $p'_1, p'_2, p'_3, \dots, p'_n$ là các tri thức có sẵn trong KB.
- Tri thức q' mới được kiểm tra xem đã có trong KB chưa, nếu chưa thì thêm q' vào KB
- Kiểm tra xem q' đã trả lời được a chưa. Nếu hàm Unify trả về False nghĩa là chưa trả lời được, nếu hàm Unify trả về theta nghĩa là có thể trả lời được a bằng phép thế theta.

4.2 Kiểm chứng kết quả

Cách chạy chương trình

```
python main.py test/test1/knowledge.pl test/test1/query.pl test/test1/output.txt
```

Kiểm chứng kết quả của chương trình trên cây phả hệ của Hoàng gia Anh chính xác 100%

```
mother(X, princeCharles)
{"{'X': 'queenElizabethII'}"}

father(X, miaGraceTindall)
{"{'X': 'mikeTindall'}"}

sibling(jamesViscountSevern, X)
{"{'X': 'ladyLouiseMountbatten_Windsor'}"}

uncle(peterPhillips, princessCharlotte)
{'false'}

niece(princessCharlotte, peterPhillips)
{'false'}

sister(X, princessBeatrice)
{"{'X': 'princessEugenie'}"}

sister(X, ladyLouiseMountbatten_Windsor)
{'false'}

father(mikeTindall, savannahPhillips)
{'false'}

grandson(princeGeorge, princessDiana)
{'true'}

sibling(X, princeCharles)
{"{'X': 'princessAnne'}", "{'X': 'princeAndrew'}", "{'X': 'princeEdward'}"}

child(princeHarry, princessDiana)
{'true'}

granddaughter(zaraPhillips, queenElizabethII)
{'true'}
```

Kiểm chứng kết quả của chương trình trên cơ sở tri thức Trường đại học Khoa học Tự nhiên chính xác 100%

```
pho_khoa(X, CNTT)
{"X": 'van_Chi_Nam'}, {"X": 'lam_Quang_Vu'}, {"X": 'nguyen_Van_Vu'}

giang_vien(le_Ngoc_Thanh)
{'true'}

truc_thuoc(X, sinh_hoc-cong_nghe_sinh_hoc)
{"X": 'sinh_ly_hoc_va_Cong_nghe_Sinh_hoc_Dong_vat'}, {"X": 'vi_sinh'}, {"X": 'sinh_ly_Thuc_vat'}, {"X": 'sinh_thai_va_Sinh_hoc_Ti

phong(khoa_hoc_may_tinh, i81)
{'true'}

hoc_ham(X, GS)
{"X": 'le_Hoai_Bac'}, {"X": 'dang_Duc_Trong'}, {"X": 'chau_Van_Tao'}, {"X": 'le_Van_Hieu'}

truong_bo_mon(X, thi_giac_may_tinh_va_dieu_khien_hoc_thong_minh)
{"X": 'ly_Quoc_Ngoc'}

hoc_vi(X, TS)
{"X": 'lam_Quang_Vu'}, {"X": 'nguyen_Dinh_Thuc'}, {"X": 'truong_Thi_Hong_Loan'}, {"X": 'pham_Trung_Hieu'}, {"X": 'dao_Nguyen_Kh

hoc_ham(X, PGS)
{"X": 'nguyen_Dinh_Thuc'}, {"X": 'dinh_Dien'}, {"X": 'pham_Trung_Hieu'}, {"X": 'le_Vu_Tuan_Hung'}, {"X": 'tran_Thanh_Huong'},

truong_khoa(le_Vu_Tuan_Hung, vatly-vatlykythuat)
{'true'}

truc_thuoc(toi_uu_va_He_thong, toan-tinhoc)
{'true'}

truc_thuoc(X, CNTT)
{"X": 'cong_nghe_tri_thuc'}, {"X": 'cong_nghe_phan_mem'}, {"X": 'thi_giac_may_tinh_va_dieu_khien_hoc_thong_minh'}, {"X": 'mang_ma
```

PHỤ LỤC

TÀI LIỆU THAM KHẢO

- [1] <https://nguyenvanhieu.vn/huong-dan-prolog/>
- [2] <https://huuvinhfit.files.wordpress.com/2015/01/chuong-7-prolog.pdf>
- [3] https://www.cpp.edu/~jrfisher/www/prolog_tutorial/2.html
- [4] Giáo trình Artificial Intelligence A Modern Approach, 3rd Edition