



ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN



CƠ SỞ TRÍ TUỆ NHÂN TẠO

BÁO CÁO ĐỒ ÁN 2

**BIỂU DIỄN TRI THỨC
BẰNG LOGIC BẬC NHẤT**

Thành viên nhóm:

Đào Thành Đạt – 18120014

Trần Quốc Bảo – 18120111

Lê Minh Khoa – 18120415

Mục Lục

PHẦN MỞ ĐẦU:	4
PHẦN 1: TÌM HIỂU NGÔN NGỮ PROLOG	6
1.1. Các đặc điểm cơ bản của Prolog	6
1.1.1. Đối tượng	7
1.1.2. Quan hệ giữa các đối tượng	7
1.1.3. Sự kiện (Facts), luật (Terms), câu hỏi(queries)	7
1.2. Cấu trúc của một chương trình Prolog	7
1.2.1. Cú pháp của các thành phần cơ bản:	7
1.2.2. Phần Domains	7
1.2.3. Phần Predicates	8
1.2.4. Phần Clause	8
1.2.5. Phần Goal	8
1.3. Các nguyên tắc của ngôn ngữ Prolog	8
1.3.1. Đồng nhất	8
1.3.2. Hợp giải, quay lui	8
1.4. Các kiểu dữ liệu	9
1.5. Phép toán số học	9
1.6. Phép toán so sánh	9
1.7. Các hàm nhập xuất	10
1.7.1. Nhập xuất kí tự:	10
1.7.2. Nhập xuất luật(Terms) và tệp(files):	10
1.8. Kỹ thuật đệ qui trong Prolog	10
PHẦN 2: TÌM HIỂU MÔI TRƯỜNG SWI-PROLOG	12
2.1. Giới thiệu:	12
2.2. Cách sử dụng SWI-Prolog:	12
2.2.1. Cài đặt và khởi động SWI-prolog	12
2.2.2. Chu trình test-edit-reload	15
2.2.3. PceEmacs	16
2.2.3.1. Edit modes	16
2.2.3.2. Các lệnh cơ bản	16
2.2.4. Graphical Debugger	16
2.3. Ví dụ minh họa	17

2.3.1.	<i>Ví dụ 1:</i>	17
2.3.2.	<i>Ví dụ 2:</i>	18
2.3.3.	<i>Ví dụ 3:</i>	19
2.3.4.	<i>Ví dụ 4:</i>	19
2.3.5.	<i>Ví dụ 5:</i>	20
PHẦN 3: XÂY DỰNG CƠ SỞ TRI THỨC MÔ TẢ QUAN HỆ TRONG GIA ĐÌNH HOÀNG GIA ANH & VUA CHÚA CÁC ĐỜI NHÀ TRẦN		21
3.1.	Gia đình hoàng gia Anh.....	21
3.2.	Nhà Trần.....	24
PHẦN 4: CÁC PHÉP SUY DIỄN LOGIC BẬC NHẤT		29
4.1.	Suy diễn tiến	29
4.2.	Suy diễn lùi	30
4.3.	Hợp giải.....	32
TÀI LIỆU THAM KHẢO		34

PHẦN MỞ ĐẦU:

Mô tả đề án:

Biểu diễn tri thức (Knowledge representation) là một phần trong lĩnh vực Công nghệ tri thức và máy học – một nhánh của trí tuệ nhân tạo trong đó nghiên cứu về cách lưu trữ các tri thức về một đối tượng về một lĩnh vực nào đó, thông qua các quá trình xử lý phức tạp nhằm giải quyết những bài toán khó đòi hỏi có trình độ chuyên môn cao trong lĩnh vực đó một cách tự động.

Biểu diễn tri thức, tìm cách vận dụng nó để giải quyết vấn đề và tìm cách “bổ sung” thêm những tri thức mới từ những tri thức đã có là nhiệm vụ tối quan trọng. Nói một cách đơn giản là phát triển các phương pháp suy diễn tự động để ứng dụng vào những hệ thống tự động hoá, “con người” hoá.

Trong đề án này, nhóm thực hiện tìm hiểu về cách biểu diễn tri thức dưới dạng logic mệnh đề và logic bậc nhất. Sau đó thực hiện xây dựng cơ sở tri thức dựa trên 2 sơ đồ cụ thể là cây phả hệ Hoàng gia Anh và vua chúa các đời nhà Trần. Môi trường lập trình sử dụng trong đề án này chủ yếu là SWI-Prolog và Python.

Đánh giá mức độ hoàn thành:

Đánh giá tổng thể: 98%

Chi tiết từng yêu cầu:

Tên yêu cầu	Mức độ hoàn thành	Người thực hiện
Tìm hiểu ngôn ngữ prolog. Nêu lên các đặc điểm chính của ngôn ngữ.	100%	Bảo
Tìm hiểu môi trường lập trình SWI-Prolog.	100%	Khoa
Giải quyết bài toán xây dựng cây phả hệ cho gia đình Hoàng gia Anh và biểu diễn bằng SWI-Prolog.	100%	Đạt
Xây dựng cơ sở tri thức bằng công cụ Prolog trên chủ đề tự chọn(vua chúa các đời nhà Trần).	100%	Đạt, Khoa, Bảo
Cài đặt phương pháp logic suy diễn tiến	95%	Đạt
Cài đặt phương pháp logic suy diễn lùi	95%	Bảo
Cài đặt phương pháp logic hợp giải	95%	Khoa

Tổ chức chương trình:

Thư mục Script:

Thư mục Tran_Clan:

- Tran_clan_Knowledge_base: cơ sở tri thức ban đầu dựa trên sơ đồ ở file Tran_family_tree.pdf.
- Query_test.pl và Query_Description.txt: các câu truy vấn + mô tả bằng ngôn ngữ tự nhiên
- Result_query: kết quả sau khi truy vấn.

Thư mục Royal_Family:

- Royal_tree_Knowledge_base: cơ sở tri thức ban đầu dựa trên sơ đồ ở yêu cầu đề bài
- Query_test.txt: các câu truy vấn
- Result_query.txt: kết quả sau khi truy vấn.

Thư mục Source Code:

Thư mục backward_chaining:

- Backward_chaining.py: source code biểu diễn phương pháp suy diễn lùi bằng python.
- Hoanggia.kb: cơ sở tri thức Hoàng gia ban đầu.
- Question.txt: các câu truy vấn (đã được chỉnh sửa từ các truy vấn ở thư mục Royal_Family theo format cần thiết của source code python).

Thư mục forward_chaining:

- Forward_chaining.py: source code biểu diễn phương pháp suy diễn tiến bằng python.
- Royal_tree_Knowledge_base: cơ sở tri thức.

Thư mục resolution:

- resolution.py: Source code biểu diễn phương pháp suy diễn bằng hợp giải.
- test: thư mục chứa các tập tin test: gồm thư mục Royal chứa thông tin về hoàng gia Anh, thư mục Tran chứa thông tin về nhà Trần. Mỗi thư mục gồm các tập tin:
 - kb.txt: chứa cơ sở dữ liệu tri thức
 - query.txt: chứa câu hỏi
 - answer: chứa câu trả lời được suy diễn từ chương trình python.
 - prolog_answer: chứa câu trả lời được suy diễn từ Prolog.

PHẦN 1: TÌM HIỂU NGÔN NGỮ PROLOG

Tổng quan về ngôn ngữ Prolog

Prolog là ngôn ngữ được sử dụng phổ biến trong lĩnh vực trí tuệ nhân tạo. Nguyên lý lập trình logic dựa trên các mệnh đề Horn (Horn logic). Là một ngôn ngữ cấp cao, có đặc điểm gần với ngôn ngữ tự nhiên, từ những người mới học đến những lập trình viên chuyên nghiệp đều có thể tiếp cận một cách nhanh chóng, viết ra một chương trình ứng dụng hữu ích. Khác với những ngôn ngữ cấu trúc như Pascal, hay C mà ta đã làm quen, Prolog là một ngôn ngữ mô tả, với một số sự kiện và quy luật suy diễn đã mô tả, Prolog sẽ suy luận cho ta các kết quả. Pro

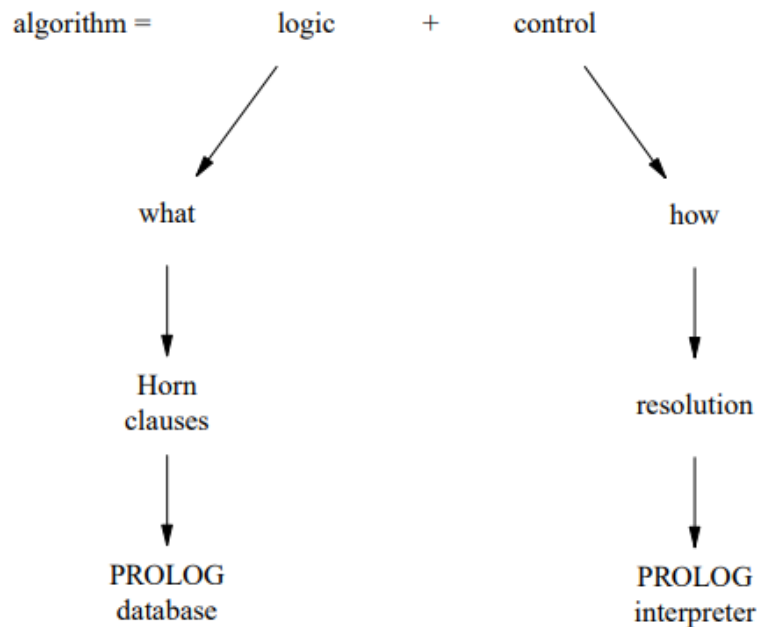


Figure 1: The relationship between PROLOG and logic programming.

Khái niệm: 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 theo lô gíc". Xuất hiện từ năm 1972 (do Alain Colmerauer và Robert Kowalski thiết kế), mục tiêu của Prolog là giúp người dùng mô tả lại bài toán trên ngôn ngữ của logic, dựa trên đó, máy tính sẽ tiến hành suy diễn tự động dựa vào những cơ chế suy diễn có sẵn (hợp nhất, quay lui và tìm kiếm theo chiều sâu) để tìm câu trả lời cho người dùng. Prolog được sử dụng nhiều trong các ứng dụng của trí tuệ nhân tạo và ngôn ngữ học trong khoa học máy tính (đặc biệt là trong ngành xử lý ngôn ngữ tự nhiên vì đây là mục tiêu thiết kế ban đầu của nó). Cú pháp và ngữ nghĩa của Prolog đơn giản và sáng sủa, nó được người Nhật coi là một trong những nền tảng để xây dựng máy tính thế hệ thứ năm mà ở đó, thay vì phải mô tả cách giải quyết một bài toán trên máy tính, con người chỉ cần mô tả bài toán và máy tính sẽ hỗ trợ họ nốt phần còn lại.

1.1. Các đặc điểm cơ bản của Prolog

1.1.1. Đối tượng

Các đối tượng trong Prolog có thể là sự việc, sự vật, con người,... được biểu thị bởi các hằng và biến trong chương trình. Hằng là giá trị không thể thay đổi, còn các biến có giá trị thay đổi được gán trong khi chạy chương trình.

Ví dụ: con mèo, Bảo, xe hơi, giá vé tàu....

1.1.2. Quan hệ giữa các đối tượng

Được dùng dưới hình thức vị từ, bao gồm tên của các vị từ và các đối số của nó, các đối số được đặt trong dấu ngoặc đơn, cách nhau bởi dấu phẩy.

Ví dụ:

Ngôn ngữ tự nhiên	Prolog
Jonh là cha của susan.	father(john,susan).
Bảo mệt.	tired(Bao)
John ăn pizza.	eats(john,pizza)

1.1.3. Sự kiện (Facts), luật (Terms), câu hỏi(queries)

- Sự kiện: là một vị từ diễn tả một đặc điểm của một đối tượng, một sự vật nào đó.
Ví dụ: 5 là số tự nhiên.
- Luật: được trình bày dưới dạng mệnh đề, biểu thị quy tắc liên quan đến đối tượng đó.
Ví dụ: "Scott là ông nội của Susan" là đúng nếu "Scott là cha của John" đúng và "John là cha của Susan" đúng.

- Câu hỏi: dùng để hỏi máy tính về các dữ kiện đã được đưa ra dưới dạng một mệnh đề.

Ví dụ: (sử dụng dữ liệu ở bảng 2.2)

Câu hỏi	Prolog (dạng prompt ?-)	Prolog phản hồi
Is John the father of Susan?	father(john,susan).	yes (or(true))
Is Bao tired?	tired(Bao).	yes
Is Dung tired?	tired(Dung).	no (of false)
Who is tired?	Tired(X)	X = Bao
Who eats pizza?	eats(X, pizza).	X = John

1.2.Cấu trúc của một chương trình Prolog

1.2.1. Cú pháp của các thành phần cơ bản:

Sự kiện	<clause>
Luật	<clause> :- <clause>
Câu hỏi	<?-> + <clause> Ví dụ: ?- eats(X, pizza) .
Biến	Một ký tự hoa hoặc một chuỗi ký tự, bắt đầu bằng một ký tự hoa. Có thể dùng _ (dấu gạch dưới) để đặt cho biến tự do. Ví dụ: X, Abc, _,....
Hằng	Gồm các số, bắt đầu bằng chữ thường và đặt trong dấu ngoặc kép.

1.2.2. Phần Domains

Đây là phần khai báo kiểu các đối tượng mà ta sử dụng, ta có thể định nghĩa kiểu dữ liệu mới dựa trên những kiểu dữ liệu đã biết.

Các kiểu dữ liệu này sử dụng cho các đối số của vị từ. Ta có thể sử dụng những kiểu dữ liệu có sẵn như để chương trình mạch lạc, ta thường định nghĩa lại các kiểu cơ bản.

Cú pháp: <danhsách kiểu mới> = <kiểu đã biết>

Hoặc: <danhsách kiểu mới> = <danhsách 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:

Domains

ten, tacgia, diachi = string

namsox = integer

dientich = real

dovat = sach(ten, tacgia, namsox);nha(dia_chi, dien_tich)

Trong ví dụ trên, ta đã định nghĩa các kiểu mới, trong đó có:

- ten, tacgia, diachi: dựa vào kiểu dữ liệu đã biết string.
- Namsox, dientich: dựa vào kiểu dữ liệu đã biết integer, real.
- Dovat: dựa vào kiểu dữ liệu mới sach, nha.

1.2.3. Phần Predicates

Đoạn khai báo kiểu cho vị từ sẽ dùng trong các đoạn sau

1.2.4. Phần Clause

Đoạn liệt kê sự kiện và định nghĩa qui tắc đã khai báo trong phần Predicates

1.2.5. Phần Goal

Đoạn đưa vào các vị từ câu hỏi, có thể đặt trước phần Clauses

1.3. Các nguyên tắc của ngôn ngữ Prolog

1.3.1. Đồng nhất

Một quan hệ bất kì đồng nhất với một quan hệ cùng tên với cùng số lượng tham số, với các đại lượng này đồng nhất theo từng cặp.

Một biến cũng có thể đồng nhất với biến, hằng khác.

Ví dụ:

Ta có thể thay thế tired(Bao), tired(Dat), tired(Khoa) bằng tired(X).

Khi đó thay vì trả về kết quả yes, tired(X) sẽ trả về 3 kết quả: Bao, Dat, Khoa.

1.3.2. Hợp giải, quay lui

Ví dụ: Ta có các Facts:

All men are mortal

Socrates is a man.

Thì từ đó ta có:

Socrates is mortal.

Nghĩa là, theo tam đoạn luận, ta có “Tất cả những người đàn ông đều phải chết”,

“Socrates là một người đàn ông” nên từ đó suy ra “Socrates chết”.

Prolog sử dụng **hợp giải** để xâu chuỗi các mệnh đề một cách máy móc và chứng minh một truy vấn. Thuật toán này thường bắt nguồn từ nguyên tắc phân giải của Robinson (1965), được gọi là SLD resolution.

Giả sử hệ thống đang chứng minh goal g , trong đó g được mô tả như sau:

$g :- g_1, g_2, \dots, g_{j-1}, g_j, \dots, g_n$. Khi các g_i kiểm chứng từ trái sang phải, đến g_j là sai thì hệ thống sẽ **quay lui** lại g_{j-1} để tìm lời giải khác.

1.4. Các kiểu dữ liệu

Prolog cung cấp các kiểu dữ liệu chuẩn là: char, integer, real, string và symbol.

Char: Kiểu kí tự. Nằm trong dấu nháy đơn. Ví dụ: 'a', '+'.
Integer: Là kiểu số nguyên, tập giá trị từ -32768 đến 32767. Ví dụ: 123, 21.
Real: Kiểu số thực. Ví dụ: 1.1.
String: Kiểu chuỗi kí tự, nằm trong dấu ngoặc kép. Ví dụ: "Bao".
Symbol: Kiểu dữ liệu sơ cấp.

1.5. Phép toán số học

Toán tử số học trong Prolog bao gồm một tập hợp các toán tử tiền tố, tiền tố và hậu tố.

Ký hiệu số học: "+", "-", "*", "/".

Trình dịch Prolog coi toán tử như các hàm và biến đổi các biểu thức thành các terms.

Ví dụ: $2 * 3 + 4 * 2$ sẽ tương đương với $+(*(2, 3), *(4, 2))$.

Bảng độ ưu tiên thứ tự toán tử (từ 1-1200).

Priority	Specifier	Operators
1,200	xfx	:- -->
1,200	fx	:- ?-
1,100	xfy	;
1,050	xfy	->
1,000	xfy	', ,'
900	fy	\+
700	xfx	= \=
700	xfx	== \== @< @=< @> @>=
700	xfx	=..
700	xfx	is == \= < =< > >=
550	xfy	:
500	yfx	+ - # /\ \/
400	yfx	* / // rem mod << >>
200	xfx	**
200	xfy	^
200	fy	+ - \

1.6. Phép toán so sánh

Các phép toán so sánh trong Prolog xử lý biểu thức số học và chữ. Sau khi tính toán biểu thức ở cả 2 vế thì Prolog mới đưa ra kết quả cuối cùng.

Ví dụ:

```
?- 1 + 2 < 3 + 4
```

True

Kí tự được so sánh bằng mã ASCII.

Ví dụ:

```
?- a @< b.
```

true.

Ngoài ra còn một số phép so sánh khác:

	Arithmetic comparison	Literal term comparison
Equality operator	= : =	==
Inequality operator	= \ =	\ ==
Less than	<	@<
Less than or equal	=<	@=<
Greater than	>	@>
Greater than or equal	>=	@>=

1.7. Các hàm nhập xuất

1.7.1. Nhập xuất kí tự:

Nhập xuất mặc định sẽ được thực hiện dưới hai luồng chuẩn (standard I/O stream) thường là nhập vào từ bàn phím và xuất ra màn hình.

get_char(?Char)	Đọc kí tự tiếp theo của input stream hiện tại.
put_char(+Char)	Ghi kí tự vào output stream hiện tại.
nl/0	Ghi một dòng mới vào output stream.

1.7.2. Nhập xuất luật (Terms) và tệp (files):

read(?Term)	Đọc một term từ input stream hiện tại.
write(?Term)	Ghi một term vào output stream hiện tại.
open(+SourceSink, +Mode, -Stream)	Mở tệp SourceSink ở Mode đầu vào hoặc đầu ra. Luồng (Stream) hợp nhất với luồng đã mở và sử dụng cho các hoạt động tiếp theo.
close(+Stream)	Đóng luồng Stream.

1.8. Kỹ thuật đệ qui trong Prolog

Đệ qui theo cách đã biết tốn rất nhiều tài nguyên về bộ nhớ, bộ nhớ tăng theo cấp số mũ tùy thuộc vào số lần gọi đệ qui. Tuy nhiên có một số trường hợp Prolog có thể biến đổi đệ qui về phép lặp gọi là đệ qui đuôi.

Ví dụ:

Bài toán Tháp Hà Nội: Có 3 cọc A, B, C. Trên cọc A có n đĩa tròn chồng lên nhau với đường kính khác nhau, sao cho đĩa bé luôn nằm trên đĩa lớn. Ta có thêm 2 cọc B, C trống. Tìm cách chuyển n đĩa từ A đến C với số lần di chuyển đĩa ít nhất ($2^n - 1$), sao cho: mỗi lần chỉ di chuyển 1 đĩa bé nhất trên cùng của một cọc tùy ý sang một cọc khác để đĩa bé luôn nằm trên đĩa lớn.

Domain	<code>i = integer</code> <code>s = symbol</code>
Predicates	<code>HaNoi(i, i)</code> <code>ChuyenDia(i, i, s, s, s, i)</code>
Clauses	<code>HaNoi(N, SoLanDiChuyen) :- ChuyenDia(N, N, a, c, b, SoLanDiChuyen).</code> <code>ChuyenDia(1, Nhan, Tu, Den, _, 1) :- !, nl, write("Chuyen dia: ", Nhan, " tu ", Tu, " den ", Den).</code> <code>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.</code>
Goal	<code>makewindow(1, 3, 7, "Thap Ha Noi", 0, 0, 25, 80), write("Nhap so dia n (1<=n<=12): "), readint(N), HaNoi(N, Tong), write("Tong so lan di chuyen dia: ", Tong) ..</code>
Kết quả với số đĩa là 2	<code>Chuyen dia: 1 tu a den b</code> <code>Chuyen dia: 2 tu a den c</code> <code>Chuyen dia: 1 tu b den c</code> <code>Tong so lan di chuyen dia: 3</code>

Kết quả của chương trình với n=2 là:

<code>Chuyen dia: 1 tu a den b</code> <code>Chuyen dia: 2 tu a den c</code> <code>Chuyen dia: 1 tu b den c</code> <code>Tong so lan di chuyen dia: 3</code>
--

PHẦN 2: TÌM HIỂU MÔI TRƯỜNG SWI-PROLOG

2.1. Giới thiệu:

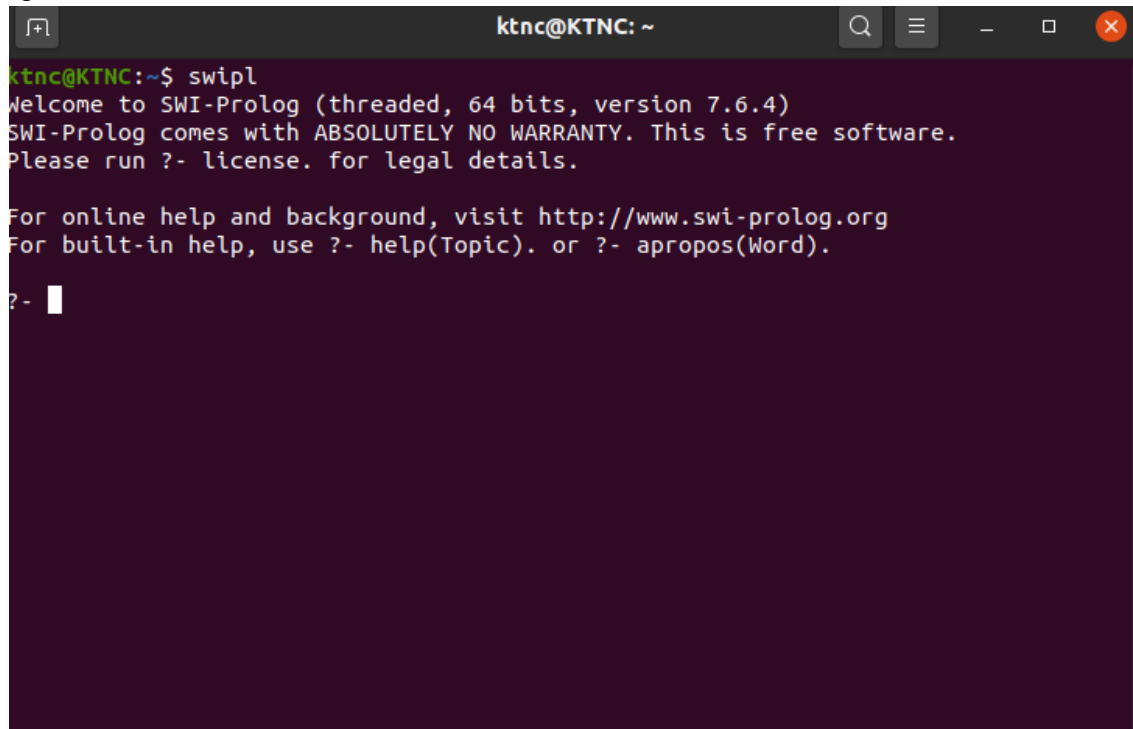
SWI-Prolog là một môi trường lập trình Prolog miễn phí, thường được sử dụng cho mục đích giáo dục và các ứng dụng mạng ngữ nghĩa. Prolog có ưu điểm có nhiều tính năng, thư viện cho lập trình logic ràng buộc, đa luồng, GUI, interface cho Java, ODBC, và các công cụ cho lập trình viên (IDE với debug bằng GUI). SWI-Prolog chạy được trên nhiều nền tảng như Windows, Unix, Macintosh, Linux. SWI là viết tắt của Sociaal-Wetenschappelijke Infomatica ("Social Science Infomatics").

2.2. Cách sử dụng SWI-Prolog:

2.2.1. Cài đặt và khởi động SWI-prolog

Để cài đặt SWI-Prolog, ta truy cập trang chủ, tải về bản cài đặt. Ngoài ra, ta có thể cài đặt trên nền Docker.

Trên môi trường Windows, SWI-Prolog cung cấp cho người dùng giao diện sử dụng. Trên Linux, ta chỉ có thể dùng SWI-Prolog thông qua dòng lệnh. Để mở SWI-Prolog bằng dòng lệnh, ta sử dụng câu lệnh **swipl**. Khi đó, giao diện SWI-Prolog hiện ra với các thông tin giới thiệu. Ngoài ra ta có thể mở một tập tin trong lúc khởi động bằng cách thêm tên vào argument.



```
ktnc@KTNC: ~  
ktnc@KTNC:~$ swipl  
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)  
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.  
Please run ?- license. for legal details.  
  
For online help and background, visit http://www.swi-prolog.org  
For built-in help, use ?- help(Topic). or ?- apropos(Word).  
?- 
```

Hình 2.2.1.a. Mở giao diện prolog từ terminal.

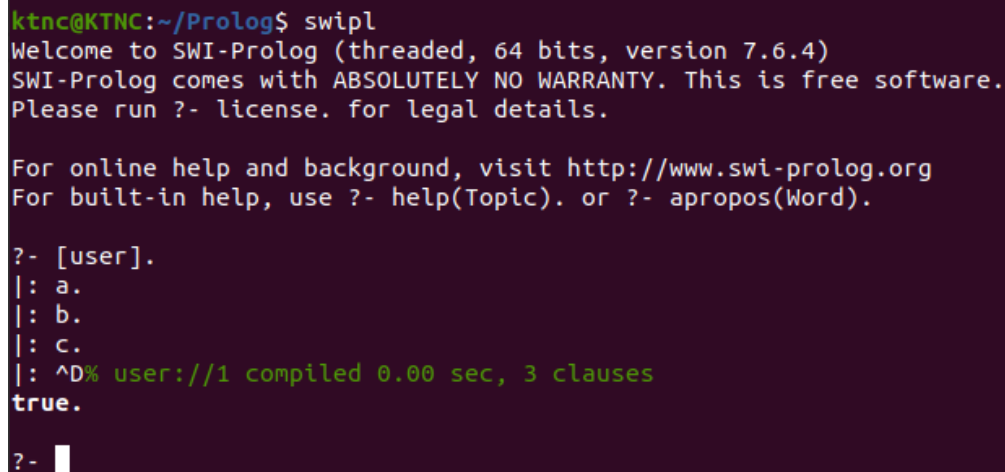
Các tập tin chương trình Prolog có phần mở rộng “.pl” hoặc “.pro” khi có xung đột về tên mở rộng tập tin giữa các phần mềm.

Ta cũng có thể sử dụng lệnh pwd và cd để kiểm tra vùng làm việc hiện tại và thay đổi thư mục làm việc.

Để mở một tập tin vào SWI-Prolog, ta sử dụng lệnh [**'path/to/ten-tap-tin.pl'**]. Có một vài tập tin đặc biệt trong một project SWI-Prolog mà ta cần lưu ý:

- Load.pl: được dùng để cài đặt môi trường và mã nguồn. Ngoài ra nó còn được dùng để parse lệnh command line và chạy chương trình.
- Run.pl: được dùng để thực thi chương trình. Thông thường, tập tin này sẽ gọi load.pl và thực thi chương trình từ load.pl.
- Save.pl: dùng để tạo và lưu trạng thái hiện tại của chương trình cùng với lệnh `qsave_program/2` với các tùy chọn khác nhau.
- Debug.pl: dùng cho việc debug. Ngoài ra ta có thể tùy biến khả năng in ra màn hình các thông tin trong lúc debug.

Mặc dù ta nên cài đặt các rule trong file nhưng trong một vài trường hợp cần thiết, ta có thể thêm các luật trực tiếp trong chương trình bằng cách sử dụng lệnh [**user**]., sau đó nhập lần lượt từng luật vào và khi kết thúc ta nhấn Ctrl + D.



```
ktnc@KTNC:~/Prolog$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [user].
|: a.
|: b.
|: c.
|: ^D% user://1 compiled 0.00 sec, 3 clauses
true.
?- 
```

Hình 2.2.1.b. Nhập các luật(rule) vào prolog.

Để thực hiện truy vấn trong chương trình, ta sử dụng các biến trong Prolog. Hệ thống sẽ thực hiện truy vấn và trả về kết quả $X = \langle \text{giá trị} \rangle$ nếu có. Khi đó người dùng có thể nhấn nút “;” hoặc dấu cách để tìm kết quả khác, ngược lại nhấn Enter để trở về. Một câu truy vấn kết thúc bằng dấu chấm khi không còn hoặc trả về false. khi không có kết quả.

```
ktnc@KTNC:~/Prolog$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- ['first.pl'].
true.

?- giaovien(A).
A = a .

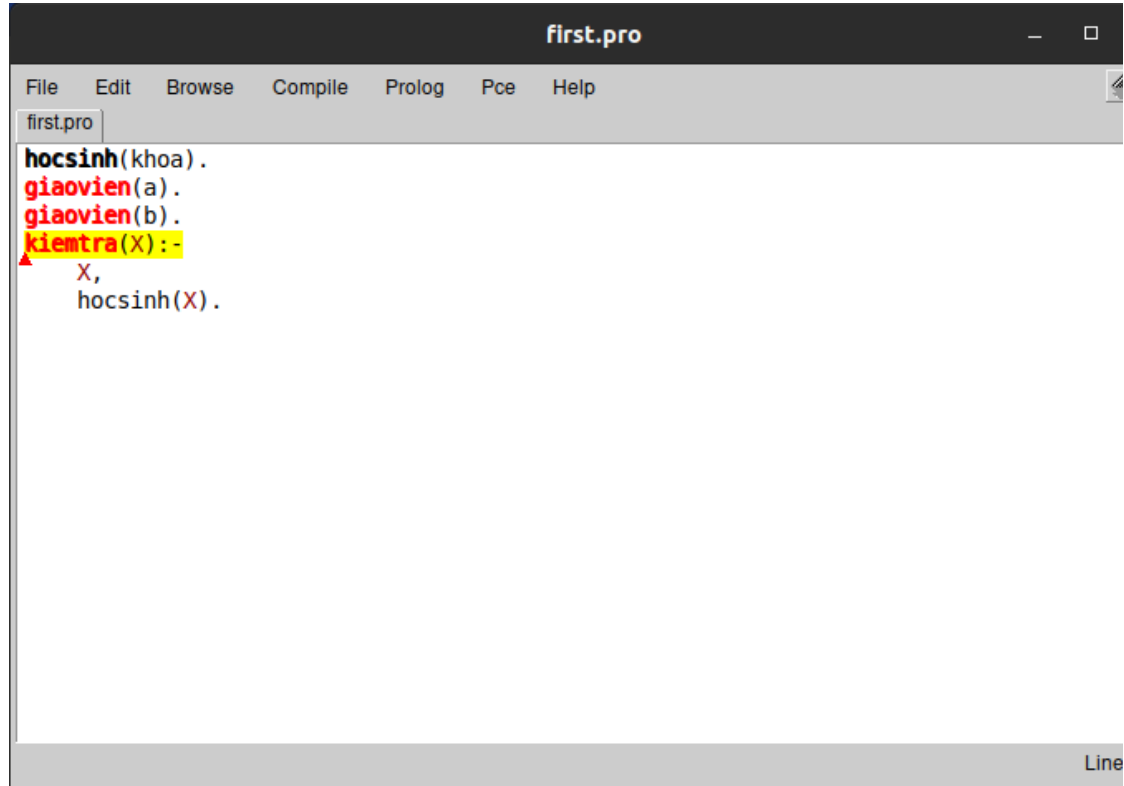
?- giaovien(x).
false.

?-
```

Hình 2.2.1.c. Thực hiện các truy vấn trong Prolog.

2.2.2. Chu trình test-edit-reload

SWI-Prolog không yêu cầu người dùng sử dụng một trình editor cụ thể. Ta có thể xác định trình editor cho Prolog bằng Prolog flag editor. Tuy nhiên ta vẫn có thể sử dụng trình Editor có sẵn của SWI-Prolog là PceEmac.



Hình 2.2.2.a. Demo Sử dụng Editor PceEmac – Editor có sẵn của Prolog

Một điểm đặc biệt của Prolog là việc cho phép người dùng chỉnh sửa mã nguồn ngay trong lúc chạy. Để xác định vị trí cần chỉnh sửa, ta dùng lệnh `edit(X)` (với X là tên file, tên module hoặc tên phương thức). Khi có nhiều hơn một kết quả, ta có thể chọn nơi cần chỉnh sửa.

Ví dụ:

```
?- edit(country).  
Please select item to edit:  
1 chat:country/10'/home/jan/.config/swi-prolog/lib/chat/countr.pl':16  
2 chat:country/1'/home/jan/.config/swi-prolog/lib/chat/world0.pl':72  
Your choice?
```

Khi thực hiện test chương trình và gặp lỗi, ta có thể sửa trực tiếp bằng lệnh `edit` và reload thực hiện chạy lại chương trình. Dùng lệnh **make** ta sẽ kiểm tra được phần nào đã được chỉnh sửa và sau đó tiến hành reload lại phần đó.

```

?- ['Prolog/first.pro'].
true.

?- edit(kiemtra).
true.

?- make.
% /home/ktnc/Prolog/first.pro compiled 0.00 sec, 0 clauses
% Found new meta-predicates in iteration 1 (0.001 sec)
% :- meta_predicate user:kiemtra(0).
% Restarting analysis ...
true.

?- █

```

Hình 2.2.2.b. Sửa chương trình trực tiếp từ terminal, dùng lệnh *edit*, *make*

2.2.3. PceEmacs

Đây là một công cụ editor có sẵn trong SWI-Prolog được xây dựng dựa trên GNU-Emacs, thích hợp cho người dùng mới.

2.2.3.1. Edit modes

Modes là trái tim của Emacs, mỗi modes khác nhau tương ứng với từng loại loại tập tin nguồn. Để chỉnh sửa Prolog, ta cần sử dụng Prolog mode.

Để sử dụng Prolog mode trong Emacs, ta có thể dùng các cách:

- Mở tập tin với tên mở rộng là .pl, .pro. Emacs sẽ tự động chuyển sang Prolog mode.
- Chèn vào dòng đầu tập tin chuỗi ký tự “*-*- Prolog -*-*”.
- Vào File/Mode chọn Prolog.

2.2.3.2. Các lệnh cơ bản

Cut/Copy/Paste:

- Copy: double click chuột trái vào từ cần copy hoặc tô đen vào phần cần copy và nhấn tổ hợp Ctr + Shift + C.
- Cut: tô đen phần cần cắt và nhấn phím Del.
- Paste: dùng chuột giữa hoặc chọn Edit/Paste.

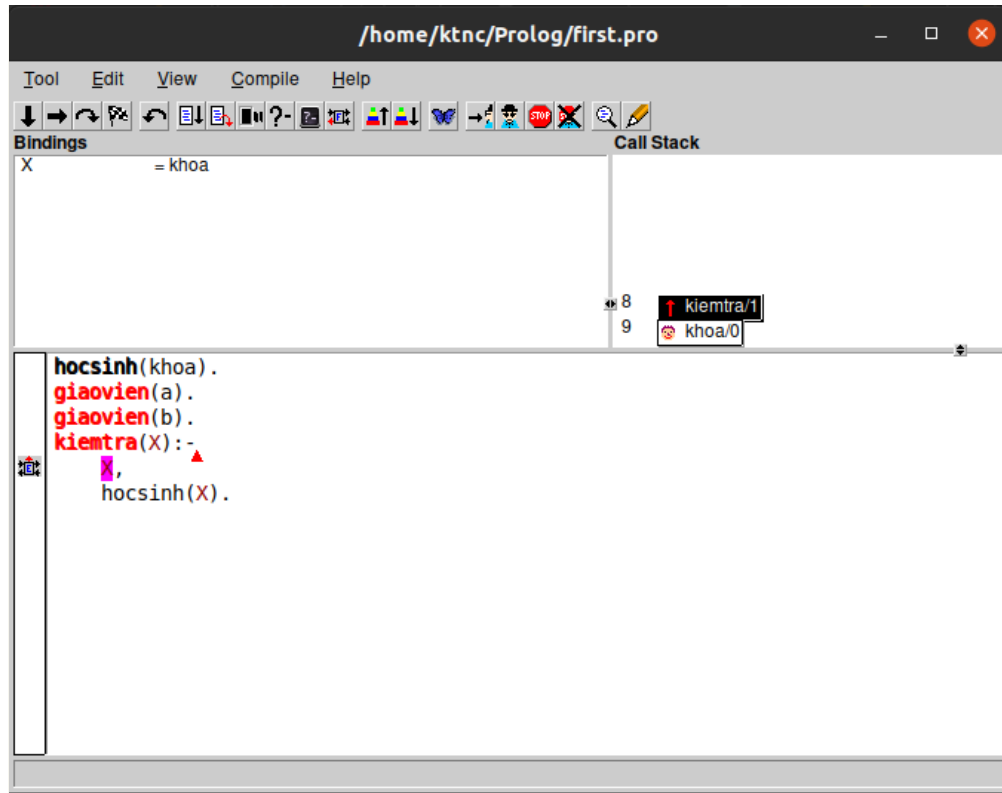
Undo: Sử dụng tổ hợp phím Ctr + Z.

Tìm kiếm: dùng tổ hợp phím Ctr + S (tìm kiếm tới) hoặc Ctr + R (tìm kiếm lui), sau đó gõ nội dung cần tìm kiếm.

2.2.4. Graphical Debugger

SWI-Prolog cung cấp cho người dùng hai trình Debugger khác nhau: console debugger và graphical debugger. Để dễ dàng hơn trong lúc debug, ta nên sử dụng graphical debugger.

Ta sử dụng lệnh **guitracer**. Để cài đặt chế độ thực thi debug, ta dùng lệnh **trace**. hoặc **spy**.



Hình 2.2.4. Debug trên môi trường SWI-Prolog

2.3. Ví dụ minh họa

2.3.1. Ví dụ 1:

- Tiến hành truy vấn trên cơ sở dữ liệu sau:
- `hocsinh(khoa).`
- `hocsinh(dat).`
- `hocsinh(bao).`
- `giaovien(a).`
- `giaovien(b).`
- `kiem_tra.`
- `ra_ve`

```

?- ['first.pl'].
true.

?- hocsinh(Who).
Who = khoa ;
Who = dat ;
Who = bao.

?- giaovien(Who).
Who = a ;
Who = b.

?- kiem_tra.
true.

?- ra_choi.
ERROR: Undefined procedure: ra_choi/0 (DWIM could not correct
?- 

```

Hình 2.3.1. Minh họa ví dụ 1

2.3.2. Ví dụ 2:

So sánh hai số:

- `sosanh(A, B, KQ):-`
- `A == B,`
- `KQ = bang.`
- `sosanh(A, B, KQ):-`
- `A < B,`
- `KQ = nho_hon.`
- `sosanh(A, B, KQ):-`
- `A > B,`
- `KQ = lon_hon.`

```

?- ['first.pl'].
true.

?- sosanh(4,3,KQ).
KQ = lon_hon.

?- sosanh(3,3, KQ).
KQ = bang .

?- sosanh(1,3,KQ).
KQ = nho_hon .

?- 

```

Hình 2.3.2. Minh họa ví dụ so sánh số tự nhiên.

2.3.3. Ví dụ 3:

Tính a^b : ta có thể sử dụng đệ quy để tính:

- $\text{mu}(A, 1, R):- R \text{ is } A.$
- $\text{mu}(A, 0, R):- R \text{ is } 1.$
- $\text{mu}(A, B, R):-$
- $B > 1,$
- $N1 \text{ is } B - 1,$
- $\text{mu}(A, N1, R1),$
- $R \text{ is } A * R1.$

```
?- ['first.pl'].
true.

?- mu(2,3,W).
W = 8 ;
false.

?- mu(9,3, H).
H = 729
```

Hình 2.3.3. Minh họa ví dụ a^b .

2.3.4. Ví dụ 4:

Tìm nghiệm của phương trình $ax + b = c$

- $\text{mu}(0, B, C, X):-$
- $B == C,$
- $X = \text{vo_so}.$
- $\text{mu}(0, B, C, X):-$
- $B \neq C,$
- $X = \text{vo_nghiem}.$
- $\text{mu}(A, B, C, X):-$
- $D \text{ is } C - B,$
- $X \text{ is } D/A.$

```

first.pl
home > ktnc > Prolog > first.pl
1  mu(0, B, C, X):-
2      B == C,
3      X = vo_so.
4
5  mu(0, B, C, X):-
6      B \== C,
7      X = vo_nghiem.
8
9  mu(A, B, C, X):-
10     D is C - B,
11     X is D/A.
12

?- ['first.pl'].
true.
?- mu(0,1,1, X).
X = vo_so .
?- mu(0,1,2,X).
X = vo_nghiem .
?- mu(3, 4,5, X).
X = 0.3333333333333333.
?- 

```

Hình 2.3.4. Minh họa ví dụ tìm nghiệm của phương trình bậc nhất $ax + b = c$

2.3.5. Ví dụ 5:

Cho cơ sở tri thức như sau: Câu điều kiện dangky(X) cho biết người học có được đăng ký một môn X hay không. Câu điều kiện tienquyet(A, B) cho biết môn học A là môn học tiên quyết của môn học B. Câu điều kiện quamon(X) cho biết người học đã qua môn X chưa.

- dangky(Monhoc):-
- not(quamon(Monhoc)),
- tienquyet(MHTQ, Monhoc),
- quamon(MHTQ).
- tienquyet(nmlt, ktlr).
- tienquyet(ktlr, ctdlvgt).
- quamon(nmlt).
- quamon(ktlr).

```

?- ['first.pl'].
true.

?- dangky(nmlt).
false.

?- dangky(ktlr).
false.

?- dangky(ctdlvgt).
true.

?- 

```

Hình 2.3.4. Minh họa ví dụ dùng Prolog trên cơ sở dữ liệu.

PHẦN 3: XÂY DỰNG CƠ SỞ TRI THỨC MÔ TẢ QUAN HỆ TRONG GIA ĐÌNH HOÀNG GIA ANH & VUA CHÚA CÁC ĐỜI NHÀ TRẦN

3.1. Gia đình hoàng gia Anh

Định nghĩa vị từ và các phép suy diễn:

Nhóm vị từ định nghĩa sẵn:

gồm parent(Parent,Child), male(Person), female(Person), divorced(Person, Person) và married(Person, 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).

Nhóm vị từ được suy diễn:

husband(Person, Wife): Person là chồng của Wife

wife(Person,Husband): Person là vợ của Husband

father(Parent,Child): Parent là cha của Child

mother(Parent,Child): Parent là mẹ của Child

child(Child,Parent): Child là con của Parent

son(Child,Parent): Child là con trai của Parent

daughter(Child,Parent): Child là con gái của Parent

grandparent(GP,GC): GP là ông bà của GC

grandmother(GM,GC): GM là bà của GC

grandfather(GF,GC): GF là ông của GC

grandchild(GC,GP): GC là cháu của ông bà GP

grandson(GS,GP): GC là cháu trai của ông bà GP

granddaughter(GD,GP): GC là cháu gái của ông bà GP

sibling(Person1,Person2): Person1 là anh chị em của Person2

brother(Person,Sibling): Person1 là anh em của Person2

sister(Person,Sibling): Person1 là chị em của Person2

aunt(Person,NieceNephew): Person là dì của NieceNewpew

uncle(Person,NieceNephew): Person là chú của NieceNewpnew

niece(Person,AuntUncle): Person là cháu gái của AuntUncle

nephew(Person,AuntUncle): Person là cháu trai của AuntUncle

Các câu truy vấn:

1. Ai là chồng của Queen Elizabeth II?

?- husband(X, queenElizabethII).

2. Sarah Ferguson là vợ của Prince Andrew?

?- wife(sarahFerguson, princeAndrew).

3. Ai là cha của Princess Beatrice?

?- father(X, princessBeatrice).

4. Kate Middleton là mẹ của Princess Charlotte?

?- mother(kateMiddleton, princessCharlotte).

5. Ai là con của Prince Andrew?

?- child(X, princeAndrew).

6. Prince George là con trai của Kate Middleton?

?- son(princeGeorge, kateMiddleton).

7. Ai là con gái của Peter Phillips?

?- daughter(X, peterPhillips).

8. Ai là ông bà của Mia Grace Tindall?

?- grandparent(X, miaGraceTindall).

9. Camilla Parker Bowles là bà của Princess Beatrice?

?- grandmother(camillaParkerBowles, princessBeatrice).

10. Prince Charles là ông của Princess Charlotte?

?- grandfather(princeCharles, princessCharlotte).

11. Ai là cháu của Captain Mark Phillips

?- grandchild(X, captainMarkPhillips).

12. Princess Eugenie là cháu trai của Prince Phillip?

?- grandson(princessEugenie, princePhillip).

13. Zara Phillips là cháu gái của Prince Phillip?

?- granddaughter(zaraPhillips, princePhillip).

14. ai là anh em của Princess Eugenie?

?- sibling(X, princessEugenie).

15. ai là anh em trai của Zara Phillips?

?- brother(X, zaraPhillips).

16. Savannah Phillips là chị em gái của Mia Grace Tindall?

?- sister(savannahPhillips, miaGraceTindall).

17. ai là dì/ cô của Peter Phillips?

?- aunt(X, peterPhillips).

18. Prince Charles là chú của Princess Eugenie?

?- uncle(princeCharles, princessEugenie).

19. ai là cháu gái của cô/chú Peter Phillips?

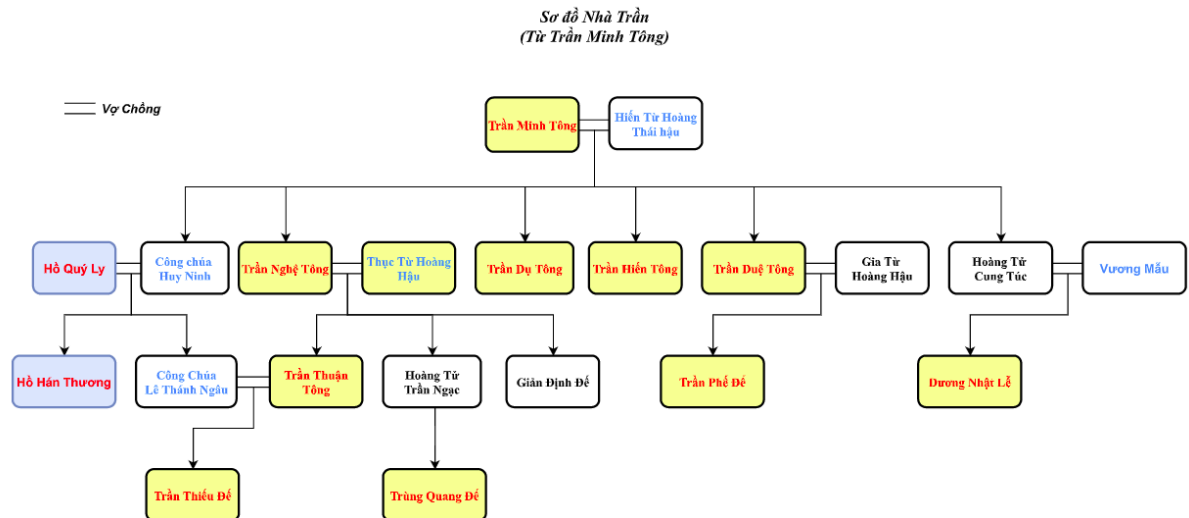
?- niece(X, peterPhillips).

20. Prince George là cháu gái của cô/chú Prince Harry?

?- nephew(princeGeorge, princeHarry).

3.2. Nhà Trần

Sơ đồ mô tả:



Nhóm vị từ định nghĩa sẵn: nam(Person), nu(Person), vochong(Chong, Vo), cha(Cha, Con), me(Me, Con), anhtrai(Anh, Em), vua(Vua).

Nhóm vị từ được suy diễn:

phuhuynh(Phuhuynh, Con): Phuhuynh là cha hoặc mẹ của Con.

con(Con, Nguoi): Nguoi là cha hoặc mẹ của Con.

contrai(Con, Nguoi): Con là con trai của Nguoi.

congai(Con, Nguoi): Con là con gái của Nguoi.

anhemruot(ConA, ConB): ConA và ConB có cùng cha hoặc mẹ.

chigai(Chi, Em): Chi là chị ruột của Em.

anhemho(ConA, ConB): phụ huynh của ConA và ConB là anh em ruột của nhau.

ongba(OngBa, Chau): cha hoặc mẹ của Chau là con của OngBa.

ongbangoai(OngBa, Chau): mẹ của Chau là con của OngBa.

ongbanoi(OngBa, Chau): cha của Chau là con của OngBa.

ongnoi(Ong, Chau): Ong là cha của cha của Chau.

banoi(Ba, Chau): Ba là mẹ của cha của Chau.

ongngoai(Ong, Chau): Ong là cha của mẹ của Chau.

bangoai(Ba, Chau): Ba là mẹ của mẹ của Chau.

chau(Chau, OngBa): OngBa là ông bà của Chau.

chaanoi(Chau, OngBa): OngBa là ông hoặc bà nội của Chau.

chaungoai(Chau, OngBa): OngBa là ông hoặc bà ngoại của Chau.

chu(Chu, Chau): Chu là em của cha của Chau.

bac(Bac, Chau): Bac là anh của cha của Chau.

cau(Cau, Chau): Cau là anh em ruột của mẹ của Chau.

co(Co, Chau): Co là nữ, là anh em ruột của mẹ của Chau.

co(Co, Chau): Co là nữ, là em của cha của Chau.

hoanghau(Nguoi, Vua): Nguoi là vợ của Vua và Vua là vua.

hoangthai(nguoi, Vua): Nguoi là mẹ của Vua và Vua là vua và chồng của Nguoi đã từng làm vua.

quocmau(Nguoi, Vua): Nguoi là mẹ của Vua và Vua là vua và chồng của Nguoi chưa từng làm vua.

quocla(nguoi, Vua): Nguoi là cha của Vua và Vua là vua và Nguoi chưa từng làm vua.

thaituonghoang(Nguoi, Vua): Nguoi là cha của Vua, Vua là vua và Nguoi đã từng làm vua.

thaimau(Nguoi, Vua): Nguoi là bà nội của Vua và Vua là vua.

hoanghuynh(Nguoi, Vua): Nguoi là anh trai của Vua và Vua là vua.

hoangde(Nguoi, Vua): Nguoi là em trai của Vua và Vua là vua.

hoangty(Nguoi, Vua): Nguoi là chị gái của Vua và Vua là vua.

hoangmuoi(Nguoi, Vua): Nguoi là em gái của Vua và Vua là vua.

hoangtu(Nguoi, Vua): Nguoi là con trai của Vua và Vua là vua.

thaitu(Nguoi, Vua): Nguoi là con trai của Vua, sẽ làm vua và Vua là vua.

congchua(Nguoi, Vua): Nguoi là con gái của Vua và Vua là vua.

hoangtuc(Nguoi, Vua): Nguoi là vợ của hoàng tử của Vua và Vua là vua.

hoangphi(Nguoi, Vua): Nguoi là vợ vua thái tử của Vua và Vua là vua.

phoma(Nguoi, Vua): Nguoi là chồng của công chúa của Vua và Vua là vua.

phuhoang(Con, Vua): Con là con trai của Vua và Vua là vua.

hoangthuc(Nguoi, Vua): Nguoi là chú của Vua và Vua là vua.

quoctruong(Nguoi, Vua): Nguoi là cha của vợ của Vua và Vua là vua.

hoangba(Nguoi, Vua): Nguoi là bác của Vua và Vua là vua.

Các câu truy vấn:

1. Ai là vợ của vua Trần Minh Tông?

?- vochong(tranminhtong, X).

2. Ai là vua?

?- vua(X).

3. Ai là mẹ của Hoàng Tử Cung Túc?

?- me(X, cungtuc).

4. Vua Trần Minh Tông có phải là ông nội của Trần Dụ Tông Không?

?- ongnoi(tranminhtong, trandutong).

5. Công chúa Huy Ninh có phải là mẹ của vua Hồ Quý Ly không?

?- me(huyninh, hoquily).

6. Công chúa Huy Ninh có phải là chị gái của vua Trần Nghệ Tông không?-

?- chigai(huyninh, trannghetong).

7. Thục tử Hoàng Hậu có phải là vợ của vua Trần Dụ Tông không?

?- vochong(trandutong, thuctu).

8. Cung Túc có phải là con gái của vua Trần Minh Tông không?

?- congai(cungtuc, tranminhtong).

9. Vua Trần Dụ Tông có phải là anh trai của vua Trần Hiến Tông không?
?- anhtrai(trandutong, tranhientong).
10. Vua Trần Thuận Tông có phải là anh em của Công Chúa Lê Thánh Ngâu không?
?- anhemho(tranthuantong, lethanhngau).
11. Công Chúa Lê Thánh Ngâu có phải là vợ của vua Trần Thuận Tông không?-
?- vochong(tranthuantong, lethanhngau).
12. Vua Trần Minh Tông có phải là ông nội của Hoàng Tử Trần Ngạc không?
?- ongnoi(tranminhtong, tranngac).
13. Có những ai là công chúa của vua Trần Minh Tông?
?- congchua(X, tranminhtong).
14. Ai là anh trai của vua Trần Thuận Tông?
?- anhtrai(X, tranthuantong).
15. Ai là bà nội của vua Trần Thuận Tông?
?- banoi(X, tranthuantong).
16. Những ai là ông bà nội của Hoàng tử Trần Ngạc?
?- ongbanoi(X, tranngac).
17. Ai là ông bà ngoại của Giản Định Đế?
?- ongbangoai(X, giandinhde).
18. Ai là ông ngoại của Hồ Hán Thương?
?- ongngoai(X, hohanthuong).
19. Hiến từ Hoàng Thái Hậu có phải là bà ngoại của Công chúa Lê Thánh Ngâu không?
?- bangoai(hientu, lethanhngau).
20. Vua Trần Dụ Tông có phải là chú của Giản Định Đế Không?
?- chu(trandutong, giandinhde).
21. Vua Trần Duệ Tông có phải là bác của Hoàng Tử Trần Ngạc Không?

?- bac(tranduetong, tranngac).

22. Trần Nghệ Tông có phải là cậu của Trần Phế Đế không?

?- cau(trannghetong, tranphede).

23. Ai là cô của Giản Định Đế?

?- co(X, giandinhde).

24. Ai là Hoàng Thái Hậu của vua Trần Dụ Tông?

?- hoangthaihou(X, trandutong).

25. Công Chúa Huy Ninh có phải là Hoàng Hậu của vua Hồ Quý Ly không?

?- hoanghou(huyninh, hoquyly).

26. Ai là Thái Thượng Hoàng của vua Trần Nghệ Tông?

?- thaithuonghoang(X, trannghetong).

27. Trần Ngạc có phải là hoàng tử của vua Trần Nghệ Tông không?

?- hoangtu(tranngac, trannghetong).

28. Ai là Hoàng Đế của vua Trần Dụ Tông?

?- hoangde(X, trandutong).

29. Ai là Hoàng Muội của vua Trần Hiến Tông?

?- hoangmuoi(X, tranhientong).

30. Lê Thánh Ngâu có phải là công chúa của Vua Trần Nghệ Tông?

?- congchua(lethanngau, trannghetong).

PHẦN 4: CÁC PHÉP SUY DIỄN LOGIC BẬC NHẤT

4.1.Suy diễn tiến

Xây dựng và cài đặt phương pháp suy diễn tiến(forward chaining). Sử dụng thuật toán trong sách Artificial intelligence: a modern approach 3rd edition. Kết quả chạy của thuật toán chưa đúng hoàn toàn với 2 cơ sở tri thức được xây dựng.

```

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
       $(p_1 \wedge \dots \wedge p_n \Rightarrow q) \leftarrow \text{STANDARDIZE-VARIABLES}(\text{rule})$ 
      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
  
```

Figure 9.3 A conceptually straightforward, but very inefficient, forward-chaining algorithm. On each iteration, it adds to KB all the atomic sentences that can be inferred in one step from the implication sentences and the atomic sentences already in KB . The function STANDARDIZE-VARIABLES replaces all variables in its arguments with new ones that have not been used before.

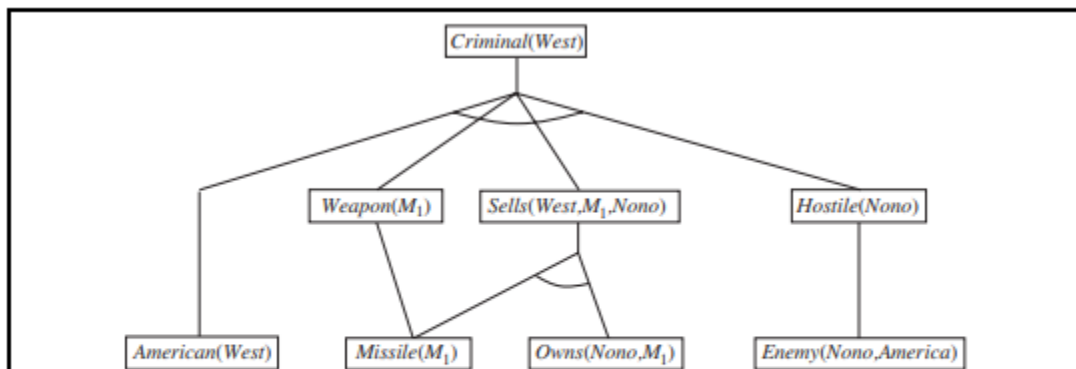


Figure 9.4 The proof tree generated by forward chaining on the crime example. The initial facts appear at the bottom level, facts inferred on the first iteration in the middle level, and facts inferred on the second iteration at the top level.

Cách sử dụng chương trình:

- Đầu vào bao gồm: file cơ sở tri thức, câu hỏi của người dùng.
- Đầu ra: câu trả lời của hệ thống, kết quả sau quá trình suy diễn.
- Cú pháp biểu diễn vị từ tương tự như cú pháp của Prolog.
- Cú pháp thực thi chương trình:

```
python3 forward_chaining.py <tên file cơ sở tri thức> "<câu truy vấn>"
```

- Câu truy vấn được biểu diễn giống so với Prolog: được đặt trong 2 dấu “ ”
- Ví dụ: husband(X, queenelizabethii) sẽ được biểu diễn thành “husband(X, queenelizabethii).”
- Câu trả lời sẽ có dạng tương tự như câu truy vấn, nếu câu truy vấn là câu hỏi True/False thì chương trình sẽ trả về True/False. Ngược lại chương trình sẽ trả về danh sách những vị từ là kết quả suy ra từ cơ sở tri thức.

Ví dụ:

```
python3 forward_chaining.py Royal_tree_Knowledge_base.pl
"husband(X, queenElizabethII)."
husband(X, queenelizabethii).
X = princephillip.
```

Ví dụ:

```
python3 forward_chaining.py Royal_tree_Knowledge_base.pl
"husband(princephillip, queenelizabethii)."
"husband(princephillip, queenelizabethii)."
True.
```

4.2.Suy diễn lùi

Xây dựng và cài đặt phương pháp suy diễn lùi(backward chaining). Sử dụng thuật toán trong sách Artificial intelligence: a modern approach 3rd edition. Kết quả chạy của thuật toán chưa đúng hoàn toàn với 2 cơ sở tri thức được xây dựng.

```

function FOL-BC-ASK(KB, query) returns a generator of substitutions
  return FOL-BC-OR(KB, query, { })

```

```

generator FOL-BC-OR(KB, goal,  $\theta$ ) yields a substitution
  for each rule (lhs  $\Rightarrow$  rhs) in FETCH-RULES-FOR-GOAL(KB, goal) do
    (lhs, rhs)  $\leftarrow$  STANDARDIZE-VARIABLES((lhs, rhs))
    for each  $\theta'$  in FOL-BC-AND(KB, lhs, UNIFY(rhs, goal,  $\theta$ )) do
      yield  $\theta'$ 

```

```

generator FOL-BC-AND(KB, goals,  $\theta$ ) yields a substitution
  if  $\theta = \text{failure}$  then return
  else if LENGTH(goals) = 0 then yield  $\theta$ 
  else do
    first, rest  $\leftarrow$  FIRST(goals), REST(goals)
    for each  $\theta'$  in FOL-BC-OR(KB, SUBST( $\theta$ , first),  $\theta$ ) do
      for each  $\theta''$  in FOL-BC-AND(KB, rest,  $\theta'$ ) do
        yield  $\theta''$ 

```

Figure 9.6 A simple backward-chaining algorithm for first-order knowledge bases.

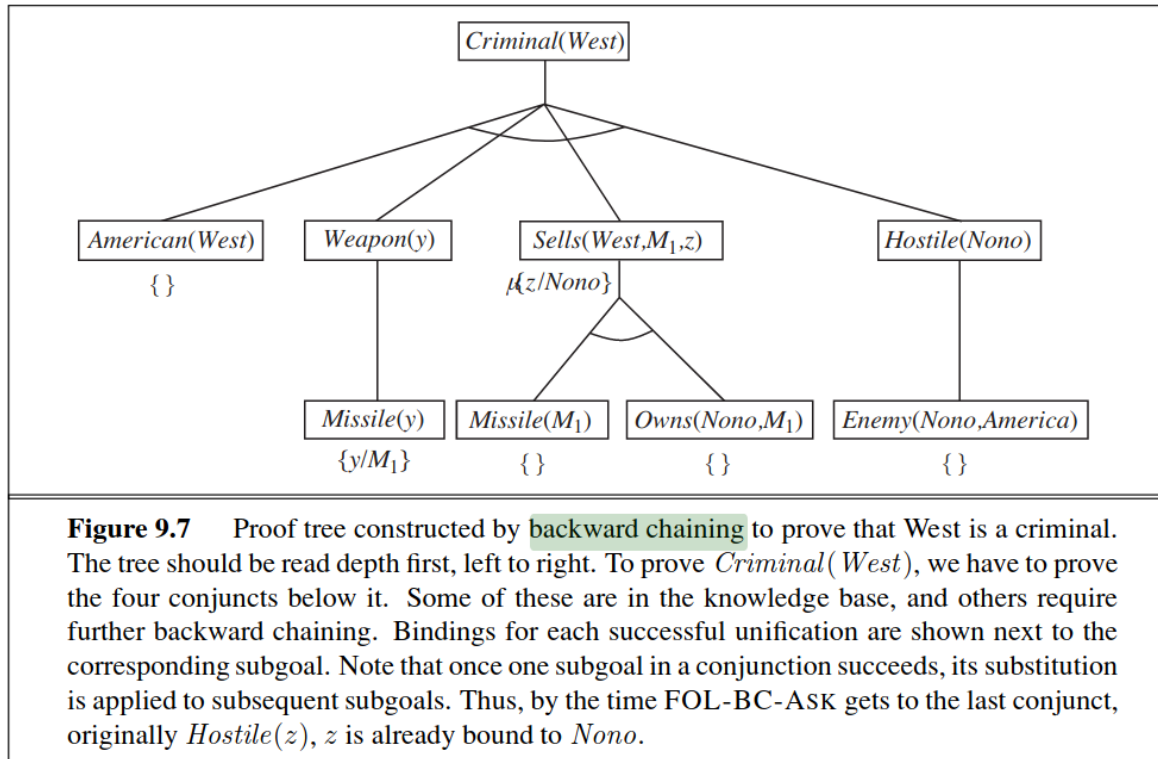


Figure 9.7 Proof tree constructed by **backward chaining** to prove that West is a criminal. The tree should be read depth first, left to right. To prove *Criminal(West)*, we have to prove the four conjuncts below it. Some of these are in the knowledge base, and others require further backward chaining. Bindings for each successful unification are shown next to the corresponding subgoal. Note that once one subgoal in a conjunction succeeds, its substitution is applied to subsequent subgoals. Thus, by the time FOL-BC-ASK gets to the last conjunct, originally *Hostile(z)*, *z* is already bound to *Nono*.

Cách sử dụng chương trình:

- Đầu vào bao gồm: file cơ sở tri thức, câu hỏi của người dùng.
- Đầu ra: câu trả lời của hệ thống, kết quả sau quá trình suy diễn.
- Cú pháp biểu diễn vị từ tương tự như cú pháp của Prolog.
- Cú pháp thực thi chương trình:

```
python3 backward_chaining <tên file cơ sở tri thức> "<câu truy vấn>"
```

- Câu truy vấn được biểu diễn hơi khác so với Prolog: câu truy vấn nằm trong 2 dấu ngoặc "(" và ")". Ta chỉ quan tâm đến các vị từ, các vị từ được đặt cách nhau 1 khoảng trắng, nếu một vị từ chứa biến thì trước đó sẽ có dấu "?"
- Ví dụ: husband(x, queenelizabethii) sẽ được biểu diễn thành ((husband ?x queenelizabethii))
- Câu trả lời sẽ có dạng tương tự như câu truy vấn, nếu câu truy vấn là câu hỏi True/False thì chương trình sẽ trả về True/False.
- Ngược lại chương trình sẽ trả về danh sách những vị từ là kết quả suy ra từ cơ sở tri thức.

Ví dụ:

```
python3 backward_chaining.py hoanggia.kb
"((husband ?x queenelizabethii))"
#Output: ((husband princephillip queenelizabethii))
```

Ví dụ:

```
python3 backward_chaining.py hoanggia.kb
"((husband princephillip queenelizabethii))"
#Output: True
```

4.3.Hợp giải

Xây dựng và cài đặt phương pháp suy diễn bằng hợp giải. Dựa vào thuật toán hợp giải của logic mệnh đề và đồng nhất các biến, ta xây dựng được hợp giải cho logic bậc nhất. Thuật giải có độ phức tạp quá cao, mất nhiều thời gian để thực thi. Do đó vấn đề kiểm thử chưa thực hiện được.

Proof by contradiction, i.e., show $KB \wedge \neg\alpha$ unsatisfiable

```

function PL-RESOLUTION( $KB, \alpha$ ) returns true or false
inputs:  $KB$ , the knowledge base, a sentence in propositional logic
          $\alpha$ , the query, a sentence in propositional logic

 $clauses \leftarrow$  the set of clauses in the CNF representation of  $KB \wedge \neg\alpha$ 
 $new \leftarrow \{ \}$ 
loop do
  for each  $C_i, C_j$  in  $clauses$  do
     $resolvents \leftarrow$  PL-RESOLVE( $C_i, C_j$ )
    if  $resolvents$  contains the empty clause then return true
     $new \leftarrow new \cup resolvents$ 
  if  $new \subseteq clauses$  then return false
   $clauses \leftarrow clauses \cup new$ 

```

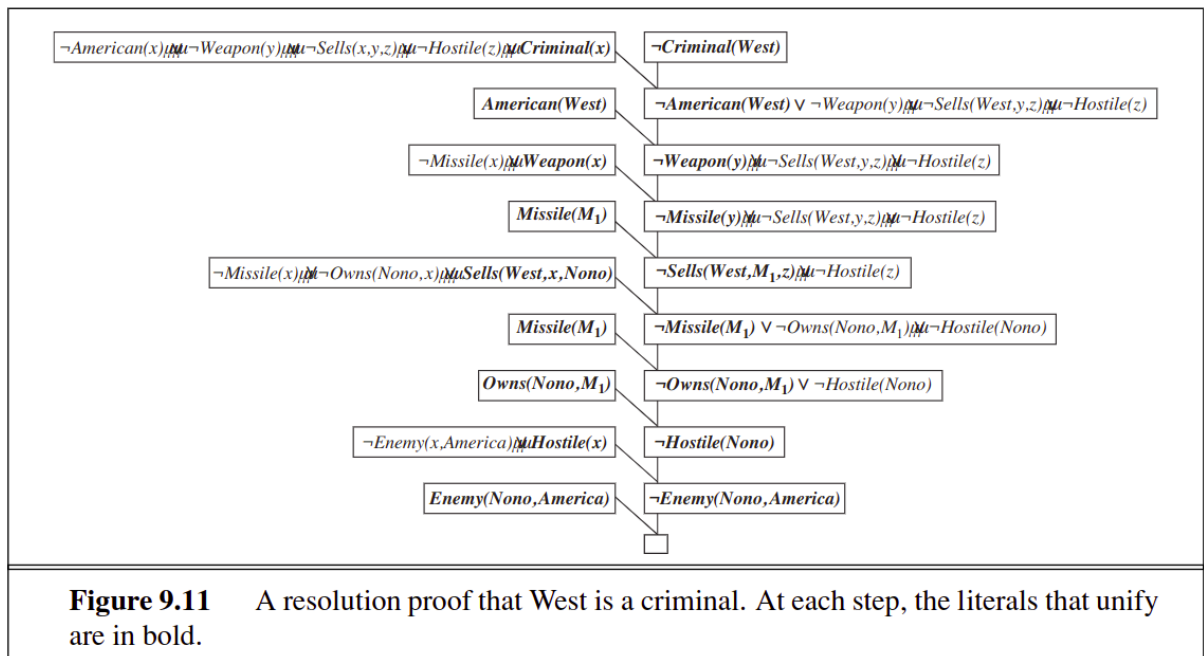


Figure 9.11 A resolution proof that West is a criminal. At each step, the literals that unify are in bold.

Các sử dụng chương trình:

- Đầu vào bao gồm: tập tin chứa cơ sở trí thức, tập tin chứa các câu hỏi.
- Đầu ra: câu trả lời của hệ thống, có định dạng gần tương tự như Prolog.
- Cú pháp biểu diễn vị từ: tương tự như Prolog.
- Cú pháp thực thi chương trình:

```
python3 resolution.py kb.txt "query"
```

- Câu truy vấn được biểu diễn giống so với Prolog: được đặt trong 2 dấu “ ”. Ví dụ: husband(X, queenelizabethii) sẽ được biểu diễn thành “husband(X, queenelizabethii).”
- Câu trả lời sẽ có dạng tương tự như câu truy vấn, nếu câu truy vấn là câu hỏi True/False thì chương trình sẽ trả về True/False. Ngược lại chương trình sẽ trả về danh sách những vị từ là kết quả suy ra từ cơ sở tri thức.

```
python3 resolution.py first.txt "me(X, cungtuc)."
me(X, cungtuc).
X = hientu

python3 resolution.py first.txt "me(hientu, cungtuc)."
me(hientu, cungtuc).
True.
```

TÀI LIỆU THAM KHẢO

- <https://vi.wikipedia.org/wiki/Prolog>
- <http://ent.htu.edu.vn/tin-tuc-su-kien/ngon-ngu-lap-trinh-prolog.html>
- http://www.mind.ilstu.edu/curriculum/protothinker/prolog_intro.php
- <https://www.cs.ru.nl/~peterl/teaching/CS3510/intro-prol.pdf>
- Appendix A - An Introduction to Prolog.
- <https://www.swi-prolog.org/>
- https://www.swi-prolog.org/pldoc/doc_for?object=manual
- https://www.doc.gold.ac.uk/~mas02gw/prolog_tutorial/prologpages/
- <https://github.com/zeralight>