



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

---

*CƠ SỞ TRÍ TUỆ NHÂN TẠO*

*BÁO CÁO ĐỒ ÁN 2 – LOGIC BẬC NHẤT*

***TÌM HIỂU NGÔN NGỮ PROLOG***

---

**Thành viên nhóm:**

Đào Thành Đạt – 18120014

Trần Quốc Bảo – 18120111

Lê Minh Khoa – 18120415

## Mục Lục

1.	Các đặc điểm cơ bản của Prolog.....	4
1.1.	Đối tượng .....	4
1.2.	Quan hệ giữa các đối tượng .....	4
1.3.	Sự kiện (Facts), luật (Terms), câu hỏi(queries).....	4
2.	Cấu trúc của một chương trình Prolog.....	5
2.1.	Cú pháp của các thành phần cơ bản: .....	5
2.2.	Phần Domains .....	5
2.3.	Phần Predicates .....	5
2.4.	Phần Clause.....	5
2.5.	Phần Goal.....	6
3.	Các nguyên tắc của ngôn ngữ Prolog.....	6
3.1.	Đồng nhất.....	6
3.2.	Phân giải, quay lui.....	6
4.	Các kiểu dữ liệu .....	6
5.	Phép toán số học .....	6
6.	Phép toán so sánh.....	7
7.	Các hàm nhập xuất.....	7
7.1.	Nhập xuất kí tự:.....	8
7.2.	Nhập xuất luật(Terms) và tệp(files): .....	8
8.	Kỹ thuật đệ qui trong Prolog.....	8
9.	Tài liệu tham khảo: .....	9

## 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

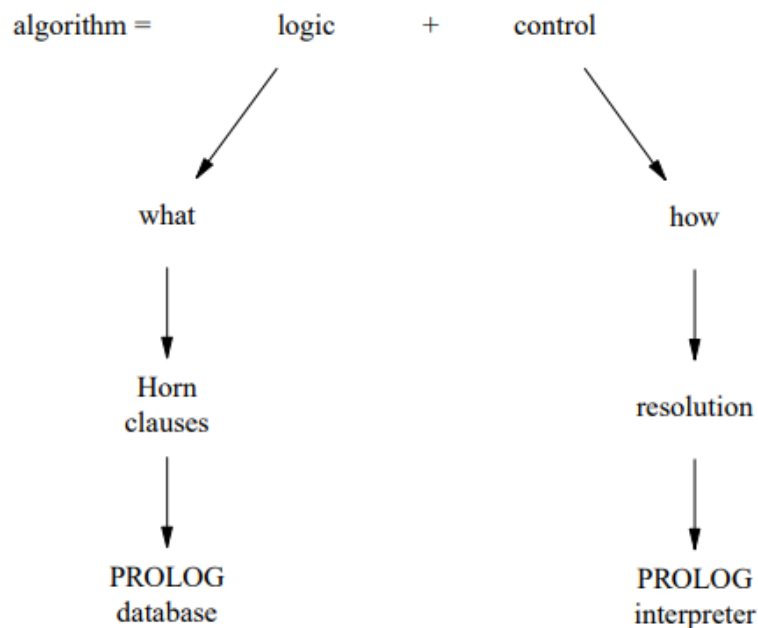


Figure 1: The relationship between PROLOG and logic programming.

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

nhien 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 thể 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. Các đặc điểm cơ bản của Prolog

### 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....

### 1.2. 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. Bảo mệt. John ăn pizza.	father(john,susan). tired(Bao) eats(john,pizza)

### 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

## 2. Cấu trúc của một chương trình Prolog

### 2.1. Cú pháp của các thành phần cơ bản:

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

### 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: `<danhsach kiểu mới> = <kiểu đã biết>`

Hoặc: `<danhsach kiểu mới> = <danhsach 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`

`namsx = integer`

`dientich = real`

`dovat = sach(ten, tacgia, namsx); 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`.
- `Namsx, dientich`: dựa vào kiểu dữ liệu đã biết `integer, real`.
- `Dovat`: dựa vào kiểu dữ liệu mới `sach, nha`.

### 2.3. Phần Predicates

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

### 2.4. Phần Clause

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

## 2.5. Phần Goal

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

## 3. Các nguyên tắc của ngôn ngữ Prolog

### 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.

### 3.2. Phân 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 phải chết”.

Prolog sử dụng một thuật toán phân 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.

## 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.

## 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: “+”, “-”, “\*” và “/”.

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	+ - \

#### 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	>=	@>=

#### 7. Các hàm nhập xuất

### 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.

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

### 7.2. Nhập xuất luật (Terms) và tệp (files):

<code>read(?Term)</code>	Đọc một term từ input stream hiện tại.
<code>write(?Term)</code>	Ghi một term vào output stream hiện tại.
<code>open(+SourceSink, +Mode, -Stream)</code>	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.
<code>close(+Stream)</code>	Đóng luồng Stream.

## 8. Kỹ thuật đệ quy trong Prolog

Đệ quy 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 quy. Tuy nhiên có một số trường hợp Prolog có thể biến đổi đệ quy về phép lặp gọi là đệ quy 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>



	<pre> ChuyenDia(1, Nhan, Tu, Den, _, 1) :- !, n1, 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. </pre>
Goal	<pre> makewindow(1, 3, 7, "Thap Ha Noi", 0, 0, 25, 80), write("Nhap so dia n (1&lt;=n&lt;=12): "), readint(N), HaNoi(N, Tong), write("Tong so lan di chuyen dia: ", Tong).. </pre>
Kết quả với số địa là 2	<pre> 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 </pre>

Kết quả của chương trình với n=2 là:

<pre> 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 </pre>
--

#### 9. 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/prototinker/prolog\\_intro.php](http://www.mind.ilstu.edu/curriculum/prototinker/prolog_intro.php)
- <https://www.cs.ru.nl/~peterl/teaching/CS3510/intro-prol.pdf>
- Appendix A - An Introduction to Prolog.