BÁO CÁO LAB2
TÌM HIỂU NGÔN NGỮ PROLOG

Thông tin thành viên nhóm
Trần Quốc Khương - 18120427
Lê Văn Vũ - 1712919

1. Giới thiệu về ngôn ngữ Prolog

- Prolog là một ngôn ngữ lập trình, tên gọi Prolog xuất phát từ tiếng Pháp Programmation en logique, nghĩa là lập trình theo logic. Prolog xuất hiện từ năm 1972 được thiết kế bởi Alain Colmerauer và Robert Kowanski, mục tiêu của Prolog là giúp người dùng mô tả lại bài toán trên ngôn ngữ logic, dựa trên đó máy tính sẽ tiến hành suy diễn tự động dựa vào những cơ chế suy diễn có sẵn để tìm câu trả lời của người dùng.
- Prolog có nhiều ứng dụng trong các lĩnh vực của trí tuệ nhân tạo và ngôn ngữ học trong khoa học máy tính (đặc biệt là ngành xử lí ngôn ngữ tự nhiên). Cú pháp và ngữ nghĩa của Prolog đơn giản và sáng sủa. Để giải quyết một số vấn đề ta thấy sử dụng ngôn ngữ Prolog cho ta chương trình gọn nhẹ hơn nhiều so với các ngôn ngữ khác.

2. Các yếu tố cơ bản trong Prolog

Trong một chương trình Prolog chúng ta cần khai báo các yếu tố sau đây: đối tượng, quan hệ giữa các đối tượng, sự kiện và các luật

* Đối tượng:

Gồm có các hằng và biến. Hằng mang giá trị cho sẵn ở đầu chương trình hoặc trong quá trình viết ta đưa vào; Các biến có giá trị thay đổi sẽ được gán giá trị khi chạy chương trình. Tên biến là một ký tự hoa hoặc một chuỗi ký tự, bắt đầu bằng một ký tự hoa.

Có một loại biến đặc biệt gọi là biến tự do, biến này không có tên và người ta dùng ký hiệu _ (dấu gạch dưới) thay cho tên biến.

* Quan hệ giữa các đối tượng:

Quan hệ giữa các đối tượng được dùng dưới hình thức vị từ.

Các vị từ sẽ bao gồm tên của vị từ và các đối số của nó. Các đối số được đặt trong ngoặc đơn và được ngăn cách bởi dấu phẩy.

Ví dụ:

Yeu(A,B) là vị từ diễn tả cho câu “A Yeu B”

Yellow(Cat) là vị từ diễn tả cho câu “Cat is yellow”

* Sự kiện và các luật:

Sự kiện là vị từ diễn tả một sự thật

Ví dụ: “10 là một số chẵn” là một sự kiện vì nó diễn tả sự thật 10 là số chẵn.

Luật là vị từ diễn tả quy luật suy diễn mà ta công nhận là đúng. Luật được trình bày dưới dạng một mệnh đề.

Ví dụ: Để suy diễn một số nguyên N bất kì là một số nguyên dương chẵn ta viết:

“N là một số nguyên dương chẵn nếu N > 0 và N chia hết cho 2”

1. Các kiểu dữ liệu của Prolog

Trong Prolog sẽ có các kiểu dữ liệu chuẩn hoặc kiểu dữ liệu do người lập trình định nghĩa.

* Kiểu dữ liệu chuẩn: gồm char, integer, real, string và symbol

char	Kiểu ký tự. Hằng ký tự phải nằm giữa 2 dấu nháy đơn.
integer	Kiểu số nguyên, tập giá trị bao gồm các số từ -32768 đến 32767
real	Kiểu số thực.
string	Kiểu chuỗi ký tự. Hằng chuỗi ký tự phải nằm giữa dấu nháy kép.
symbol	Là một kiểu sơ cấp, có hình thức giống kiểu chuỗi ký tự. Hằng symbol có 2 dạng: Dãy các chữ, số và dấu gạch dưới viết liên tiếp, ký tự đầu phải viết thường Dãy các ký tự ở giữa hai nháy kép.

* Kiểu dữ liệu do người lập trình định nghĩa:

- Kiểu mẫu tin:

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

Ví dụ:

ngay, thang, nam = integer

ngay_sx = nha_sx(ngay, thang, nam)

- Kiểu danh sách:

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

Cấu trúc của danh sách bao gồm 2 phần: Phần đầu là phần tử đầu tiên của danh sách và phần đuôi là một danh sách các phần tử còn lại.

Ví dụ:

integerlist = integer* (Danh sách là một dãy các phần tử cách nhau bởi dấu phẩy và đặt trong cặp dấu ngoặc vuông)

* Các phép toán học số học:

Phép toán	Ý nghĩa	Kiểu của đối số	Kiểu kết quả
+	Cộng hai số	integer, real	Giống đối số
-	Trừ hai số	integer, real	Giống đối số
*	Nhân hai số	integer, real	Giống đối số
/	Chia hai số	integer, real	Giống đối số
mod	Chia lấy phần dư	integer	integer
div	Chia lấy nguyên	integer	integer

* Các 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 hoặc 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 hoặc bằng	char, integer, real,	Yes hoặc No

		string	
<> hoặc ><	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 tan của X	real	real
Arctan(X)	Tính arctan của X	real	real
Exp(X)	Tính ex	real	real
Ln(X)	Tính ln của X	real	real
Log(X)	Tính log 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 X, dấu là dấu của X	real	integer
ABS(X)	Tính trị tuyệt đối của X	real	real
Random(X)	Cho số thực X ngẫu nhiên nằm trong khoảng [0,1)	real	real
Random(Y,X)	Cho số nguyên X ngẫu nhiên nằm trong khoảng [0,Y)	real	integer

2. Cấu trúc của một chương trình Prolog

Một chương trình Prolog thường có 3 hoặc 4 phần gồm: Clauses, Predicates, Domains, Goal. Phần Goal có thể bỏ đi, nếu ta không thiết kế trong chương trình thì khi thực hiện hệ thống sẽ yêu cầu chúng ta nhập.

* 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:

ten, dia_chi, nha_san_xuat = string.

ngay, thang, nam = integer

dien_tich = real
ngay_sx = nha_sx(ngay, thang, nam)
Trong đó ten, dia_chi, nha_san_xuat là các kiểu mới dựa vào cùng một kiểu đã biết là string; ngày, tháng, năm dựa vào kiểu đã biết là integer, dien_tich dựa vào kiểu đã biết là real. Còn ngay_sx là kiểu dữ liệu dựa vào kiểu nha_sx được xây dựng từ kiểu ngay, thang, nam.
* 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.
Cú pháp: <Tên vị từ> (<Danh sách các kiểu>)
Các kiểu là các kiểu cơ bản hoặc các kiểu được định nghĩa trong phần Domains và được viết cách nhau bởi dấu phẩy.
Ví dụ:
so_chan(interger): xét xem một số integer nào đó có phải là số chẵn không
nam_giu(ten, chuc_vu): để chỉ một người có tên là "ten" nắm giữ một chức vụ "chuc_vu" nào đó.
*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 ở phần Predicates
Cú pháp:
<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 ở phần Predicates.
Các tham số có thể là 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 cách nhau bởi dấu phẩy.
Các kí hiệu bao gồm:
:- (điều kiện nếu)
; (điều kiện hoặc)
, (điều kiện và)
. (kết thúc vị từ)
Lưu 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 1 lần thì phải dùng biến tự do.
Ví dụ
so_chan(10):-!.
d(X,X,1):-!. (Tính đạo hàm: d x dx = 1)
s(*,X,Y,Z):- integer(X), integer(Y), X is X*Y. (Tính x * y = z)
combination(_,0,1). (dùng để biểu diễn tổ hợp chập 0 của số N bất kì là 1, trong trường hợp này không thể viết là combination(N,0,1) vì biến N chỉ xuất hiện đúng 1 lần trong vị từ này)
*Goal:
Đây là phần không bắt buộc phải có. Nếu không viết trong phần chương trình thì khi chạy chương trình sẽ yêu cầu nhập goal vào.
Cú pháp phần Goal giống như cú pháp phần Clauses (đưa vào một hoặc một số vị từ).

Nếu tất cả các tham số của vị từ là hằng thì kết quả nhận được là ĐÚNG hoặc SAI. Nếu trong các tham số của vị từ có biến thì kết quả trả về sẽ là các giá trị biến.

3. Các nguyên tắc của Prolog

Việc giải quyết vấn đề trong Prolog chủ yếu dựa trên 2 nguyên tắc là ĐỒNG NHẤT và 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.

* Quay lui:

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

g :- g1, g2, …, gj-1, gj, …, gn.

Khi các gi kiểm chứng từ trái sang phải, đến gj là sai thì hệ thống sẽ quay lui lại gj-1 để tìm lời giải khác.

4. Các hàm xuất nhập chuẩn

* Xuất ra màn hình:

Write(Arg1, Arg2, … ,Argn) in ra màn hình giá trị của các đối số.

Writef(Formatstring, Arg1, Arg2, … ,Argn) in ra màn hình giá trị của các đối số theo định dạng được chỉ định trong Formastring.

Trong đó Formastring là một chuỗi có thể là:

“%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 Ascci 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 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.

5. Bộ ký tự, từ khóa

* Prolog sử dụng bộ ký tự: các chữ cái và chữ số (A-Z,a-z,0-9) và các toán tử (+,-,*,/,<,>,=) và các ký tự đặc biệt.

* Một số từ khóa:

- Trace: Khi có từ khóa này ở đầu chương trình, thì chương trình được thực hiện từng bước để theo dõi, nhấn F10 để tiếp tục.

6. Đệ qui trong Prolog

Đệ quy là kỹ thuật lập trình được sử dụng trong nhiều ngôn ngữ. Trong Prolog ta sử dụng đệ quy khi một vị từ được định nghĩa nhờ vào chính vị từ đó.Như đã nói trong chương lập trình hàm, trong chương trình đệ quy phải có ít nhất một trường hợp dừng và lời gọi đệ quy phải chứa yếu tố dẫn đến trường hợp dừng. Trong Prolog, trường hợp dừng được thể hiện bằng một sự kiện, yếu tố dẫn đến trường hợp dừng thể hiện bằng một biến, liên hệ với biến ban đầu bởi một công thức.

Ví dụ: Tính giai thừa

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

Trong ví dụ này điều kiện dừng xác định bởi sự kiện 0 giai thừa là 1. Để tính N! ta tính M! với M=N-1, yếu tố dẫn các trường hợp dừng là biến M có giá trị bằng N-1.