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:

1. TRẦN NHẬT HUY - 1612272
2. TRẦN ĐÌNH KHẢI – 1612282

1. Tổng quát ngôn ngữ 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 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.

2. Các kiểu dữ liệu của ProLog

Clause<mệnh đề>: Các mệnh đề cấu trúc nên một chương trình Prolog .

Predicat<vị từ>: Mỗi mệnh đề được xây dựng từ các vị từ. Một vị từ là một phát biểu về các đối tượng có giá trị là đúng (true) hoặc sai (false). Một vị từ có thể chứa có nguyên tử logic.

Các *kí hiệu* bao gồm:

:- (điều kiện nếu).

, (điều kiện và).

; (điều kiện hoặc).

. (kết thúc vị từ)

Logic atom: < nguyên tử logic>: biểu diễn quan hệ giữa các hạng (term) => Hạng và quan hệ của các hạng tạo thành một mệnh đề

Term: đối tượng dữ liệu của prolog. Hạng sơ cấp: hằng (constant), biến (variable) . Hạng phức hợp: biểu diễn các đối tượng phực tạp của bài toán. Hạng phức hợp là một hàm tử functor chứa các đối số argument

Functor : Tên_hàm _tử(Đối_1, Đối_2,…, Đối_n)

Tên hàm tử là một chuỗi chữ cái và/ hoặc chữ số , bắt đầu bằng một chữ cái thường. dùng xây dựng các cấu trúc

Luật: <…>:-<…>

Câu hỏi: ?-<…>

Chú thích: /* chú thích */ để chú thích nhiều dòng hoặc % để chú thích theo dòng

Kiểu hằng số: Prolog sử dụng cả số nguyên và số thực:

Ví dụ: 1 1.23 -0.005 -1000

Kiểu hàng logic: true và false. Được dùng như các mệnh đề và kết quả trả về.

Kiểu hằng chuỗi ký tự: được đặt trong dấu nháy kép

Ví dụ: "Đại học khoa học tự nhiên": chuỗi

"": chuỗi rỗng

"\"": chuỗi chỉ có nháy kép

Biến: Tên biến là một chuỗi ký tự gồm chữ cái, chữ số, bắt đầu bởi chữ hoa hoặc dấu gạch dưới dòng.

X, Y, A Result, List_of_members _x23, _X, _, ... Phép Toán và Số học của Prolog

Nếu biến chỉ xuất hiện một lần trong mệnh đề thì không cần đặt tên chi nó, ta sử dụng biến nặc danh vơi tên biến là _ dấu gạch dưới.

Ví dụ: co_con(X):-chame(X,_)

Tức là X có con khi là cha mẹ của một ai đó, mà ta không cần quan tâm người con đó là ai

Nếu trong mệnh đề có nhiều biến nặc danh thì các biến này là khác nhau. Nếu biến nặc danh xuất hiện trong câu hỏi thì kết quả không hiển thị giá trị tìm được của biến nặc danh đó.

Các biến chỉ có tác dụng trong một mệnh đề. Tức là ở hai mệnh đề khác nhau thì hai biến cùng tên là khác nhau. Còn hằng thì thể hiện một đối tượng duy nhất trong suốt chương trình.

Mệnh đề Horn:< Head>:-<Body>. Dùng để xây dựng các vị từ và được kết thúc bằng dấu chấm

Head:-Body .

Ví dụ: ThiRot :- KhongHocBai.

Sự kiện: là những mệnh đề Horn mà phần thân Body là rỗng (true). Dùng để mô tả dữ kiện của bài toán: sinhVien(TranVanA): mô tả TranVanA là sinh viên hay sinhVien(TranVanA):-true

Luật là mệnh đề Horn đầy đủ, có đủ head và body. Thể hiện một mệnh đề nếu..thì ngược (<-), thường thể hiện những phát biểu logic trong bài toán. Dấu phẩy trong body tương đương với phép logic and (và). N Phần head của Luật là một là một vị từ(hàm tử) với tên hàm và các đối số dùng khi đặt câu hỏi. Phần thân là các vị từ, các luật khác, hay dữ kiện được các nhau bởi dấu phẩy ",".

Nếu trời mưa thì X không đi học được viết thành luật sau: KhongDiHoc(x):-TroiMua.

Toán tử :- được dịch thôi là nếu

Ví dụ: father(X,Y):-parent(X,Y), male(X)

grandparent(GP,GC):-parent(X,GC),parent(GP,X).

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 hai chấm (;). 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 thoã 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.

Đệ qui trong prolog

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

Chẳng hạn: Bậc 1: Ông b
totien(X,Y):-chame(X,Z),chame(Z,Y).

Ông bà cố: totien(X,Y):- chame(X,Z),chame(Z,U),chame(U,Y).

Cây phả hệ càng lớn thì các định nghĩa càng để 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

totien(X,Y):-chame(X,Y)

totien(X,Y):-totien(X,Z),chame(Z,X).

3. Phép toán trong prolog

1.1. Số học:

Cộng : +

Trừ: -

*: Nhân

Chia số thực: /

Chia lấy số nguyên: //

Mod: chia lấy phần dư

Biểu thức số học được xây dựng bằng vị từ is. Đối số nằm bên trái là một đối tượng sơ cấp. Đối số bên phải là một biểu thức toán học.

Lưu ý:

```
?- X=1+2.
X = 1+2.
```

Vì Prolog hiểu đây là hàm tử + với hai đối số 1 và 2.

Phần biểu thức toán học là các phép tính giữa các số hay các hàm (function (sin, cos,…))

Prolog có sẵn các hàm số học như : sin, cos, tan, atan, sqrt, pi, e, exp, log, ...

```
?- X is 5*2.
X = 10.

?- is(X, 5*2)
|    .
X = 10.

?- 1.0 is sin(pi/2)
|    .
true.

?- 1 is sin(pi/2)
|    .
false.
```

```
?- X is 1+Z.
ERROR: Arguments are not sufficiently instantiated
```

```
?- X is 1+z.
ERROR: Arithmetic: 'z' is not a function

?- X is cos(1).
X = 0.5403023058681398.
```

Các phép so sánh

>(lớn), <(nhỏ), =<(nhỏ hơn hoặc bằng), >=(lớn hơn hoặc bằng), =\=(khác), =:=(bằng), between(a,b,Z) (a<Z<b), succ(Int1,Int2)(thành công khi Int2=Int1+1 và Int1>=0), plus(Int1,Int2,Int3)(thành công khi Int3=Int2+Int1)

1.2. Các vị từ xác định

Nonvar(X) : X không phải là biến?

Atom(A): A là một nguyên tử?

Integer(I): I là một số nguyên?

float(R) R là một số thực (dấu chấm động) ?

number(N) N là một số (nguyên hoặc thực) ?

atomic(A) A là một nguyên tử hoặc một số ?

compound(X) X là một hạng có cấu trúc ?

ground(X) X là một hạng đã hoàn toàn ràng buộc ?

functor(T, F, N) T là một hạng với F là hạng tử và có N đối (arity) T =..L Chuyển đổi hạng T thành danh sách L clause(Head, Term) Head :- Term là một luật trong chương trình ? arg(N, Term, X) Thế biến X cho tham đối thứ N của hạng Term name(A, L) Chuyển nguyên tử A thành danh sách L gồm các mã ASCII (danh sách sẽ được trình bày trong chương sau).

Ví dụ:

?- functor(t(a, b, c), F, N).

F = t N = 3 Yes

?- functor(father(jean, isa), F, N).

F = father, N = 2.

Yes

?- functor(T, father, 2).

T = father(_G346, _G347). % _G346 và _G347 là hai biến của Prolog

?- t(a, b, c) =..L.

L = [t, a, b, c]

Yes

?- T =..[t, a, b, c, d, e].

T = t(a, b, c, d, e)

Yes

?- arg(1, father(jean, isa), X).

X = jean ?- name(toto, L).

L = [116, 111, 116, 111].

Yes