Introduzione a Prolog

I ELEMENTI BASICI DI PROLOG

Notazione

Book

Prolog Programming for Artificial Intelligence (fourth edition)

Prof.Ivan Bratko

Installazione

SWI-Prolog applicazione open-source [www.*swi-prolog.org*](www.swi-prolog.org)

Listener

- *Richieste (query-goal)*
- *Risposte (results)*

Collegamenti con knowledge base file (database)

Fatte attenzione con la versione del vostro Sistema Operatorivo

Domain di SWI-Prolog

1.Come si college il listener con il knowledge base (db)?Consult

2.Verificare la direttoria dove dovremo lavorare (working _directory)

3.Sistemi operative diversi-diverso modo di lavorare

4.Compile I predicate specificati su knowledge base file.

Facts, rules and queries

Knowledge base

- Facts(arguments) /return true or false resultato esempio room(kitchen)-> true
- Rules(arguments) /return resultati dalle condizioni specificate nella rule

Listener

- Queries /return dei resultati

Unification (l’unificazione)

Le richieste (query) su Prolog includono **pattern matching**. La richiesta si chiama il goal. Se ce un fatto nella knowledge base file uguale al predicato(alla richiesta) nel listener, allora il listener risponde con ‘true.’ Se non lo trova allora il listener risponde con ‘false.’ Prolog’s pattern matching si chiama unificazione. Nel caso quando il database contiene solo dei fatti, l’unificazione a successo se compiono tre 3 condizioni.

1. Il predicato usato come goal (richiesta) e quello del db e uguale
2. Gli predicate hanno lo stesso numero di argomenti.
3. Tutti gli argumenti sono uguali.

Esempio

- Il predicato `parent( tom, bob).` Altra forma informativa parent/2
- **Parent** relazione tra gli oggetti (in questo caso Tom e il parente di Bob)
- **tom, bob** gli argumenti (gli oggetti)

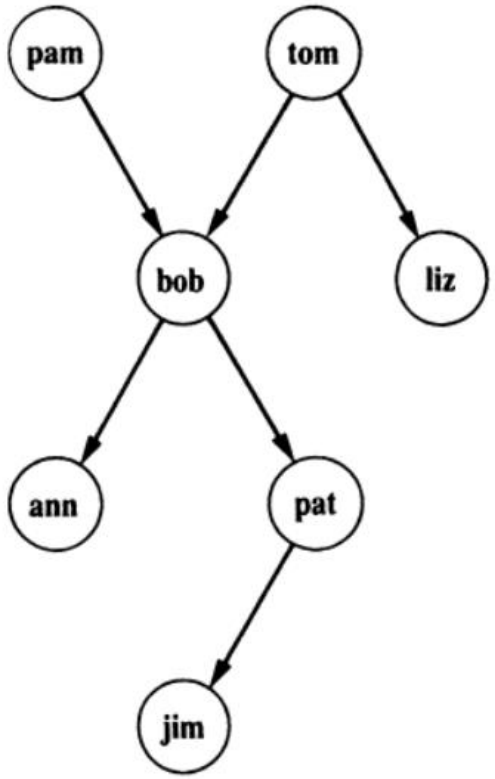


Figure 1.1 A family tree.

La knowledge base

```
parent( pam, bob).
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).
```

E Bob il genitore di Pat?

```
?- parent( bob, pat).
```

Mostrate tutti I genitori di liz?

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

variabel

La knowledge base

```
parent( pam, bob).
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).
```

Che risultato verra dalla richiesta?

```
[debug]  ?- parent(X,bob).
```

Mostrate tutti i figli di Bob?

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

```
[debug]  ?- parent(bob,X).
X = ann ;
X = pat.
```

```
[debug]  ?- parent(X,bob).
X = pam ;
X = torn ;
false.
```

Mostrate tutte le combinazioni dove X e il genitore di Y?
X e Y in questo caso sono variabli (cominciano con un charatere maiuscolo)

```
[debug]  ?- parent(X,Y).
X = pam,
Y = bob ;
X = torn,
Y = bob ;
X = torn,
Y = liz ;
X = bob,
Y = ann ;
X = bob,
Y = pat ;
X = pat,
Y = jim.
```

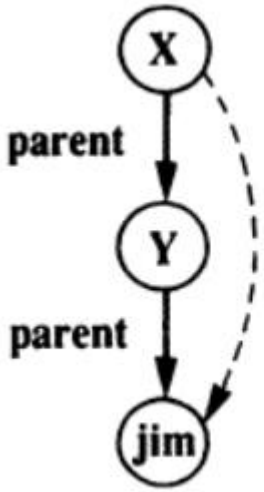
← Richiesta

← Risposta

La knowledge base

```
parent( pam, bob).
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).
```

Chi e il nonno di Jim?



grandparent

Chi e il genitore di Y?→X

Chi e il genitore di Jim?→Y

Risultato: [debug] ?- parent(Y,jim),parent(X,Y).

parent(Y, jim) and parent(X, Y)

La knowledge base

```
parent( pam, bob).
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).
```

Chi sono I nipoti di tom?

Pratica

1. Dimostrate se Ann e Pat hanno lo stesso genitore?

   a)chi e il genitore di ann, X
   b) questo genitore X e anche di Pat

2. Che risposta otteremo dopo l'execuzione dei predicati suggeriti?

(a) **?- parent(jim, X).**

(b) **?- parent(X, jim).**

(c) **?- parent(pam, X), parent(X, pat).**

(d) **?- parent(pam, X), parent(X, Y), parent(Y, jim).**

Pratica

3.Formulate in Prolog le seguenti richieste?

1. Chi sono I parenti di Pat?
2. Mostrate se Liza ha un figlio?
3. Chi sono I nonni di Pat?

Definizione delle relazioni basata sulle regole

Prima abbiamo scelto predicati **parent** e possiamo aggiungere anche **genere** per esempio

female(pam).
male(torn).
male(bob).
female(liz).

Se pensiamo di usare un altro predicato come **mother(pam,bob)** allora sarebbe meglio usare una regola che usa le predicate esistenti.

For all X and Y,
X is the mother of Y if
X is a parent of Y, and X is female.

Logica

mother(X, Y) :- parent(X, Y), female(X).

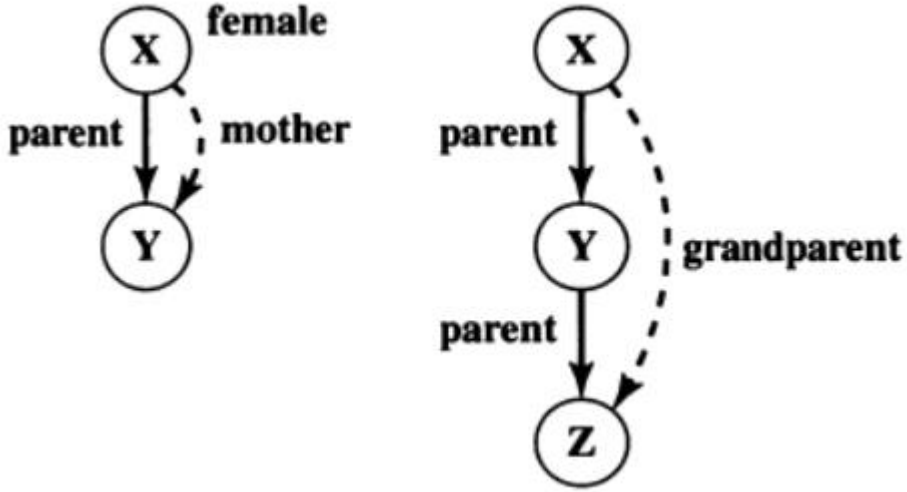
Richiesta su prolog

The Prolog symbol ':-' is read as 'if'. This clause can also be read as:

For all X and Y,
if X is a parent of Y and X is female then
X is the mother of Y.

Il fatto (facts) come `parent( tom, liz).` ha solo uno dei due risultati **true** oppure **false**

La regola (rules) come `mother( X, Y) :- parent( X, Y), female( X).`




```
grandparent( X, Z) :-
    parent( X, Y),
    parent( Y, Z).
```

Figure 1.3

Definizioni importanti

I program Prolog programs si possono extendere se aggiungiamo clause nuove.

Le clause di Prolog sono 3 tipi: facts, rules e questions.

- **Facts declare things that are always, unconditionally, true.**
- **Rules declare things that are true depending on a given condition.**
- **By means of questions the user can ask the program what things are true.**

La clausa di Prolog consiste dal head e il body. Il body e una lista di goal separate dale virgole. Virgole tra I goal vogliono dire una conjuction (concatenazione) tra I goal.

Definizioni importanti

- Il fatto e una clausea che ha soltanto il head e no il body.
- La richiesta ha soltanto il body.
- Le regole consistono dal head e il (non-empty) body.
- Durante la computazione, il variablo si po sostituire da un altro oggetto. In questo caso noi diciamo che il variablo e instantiated

Definizioni importanti

Variables are assumed to be universally quantified and are read as 'for all'.

Alternative readings are, however, possible for variables that appear only in the body. For example: hasachild(X) :- parent(X, Y).

(a) For all X and Y,

if X is a parent of Y then

X has a child.

(b) For all X,

X has a child if

there is some Y such that X is a parent of Y.

Questo due sono equivalenti.

Pratica

Traducette il testo in Prolog rules:

(a) Everybody who has a child is happy (introducete una relazione con un'argumento happy). **happy(X):-hasChild(X)**

(b) For all X, if X has a child who has a sister then X has two children (introduce new relation hastwochildren). **hastwochildren(X) :- parent(X,Y), sister(Y,_).**

Definite la relazione **grandchild** usando la relazione **parent**. Sara uguale ala relazione **grandparent** (Figure 1.3). **grandchild(X,Y) :- parent(Z,X), parent(Y,Z)**

Definite la relazione **aunt(X, Y)** in termini delle relazioni **parent** e **sister**.

Se vi aiuta create una diagrama usando lo stile di 1.3 per la relazione **aunt** . **aunt(X,Y) :- parent(Z,Y), sister(X,Z).**

Recursione delle regole (rules)

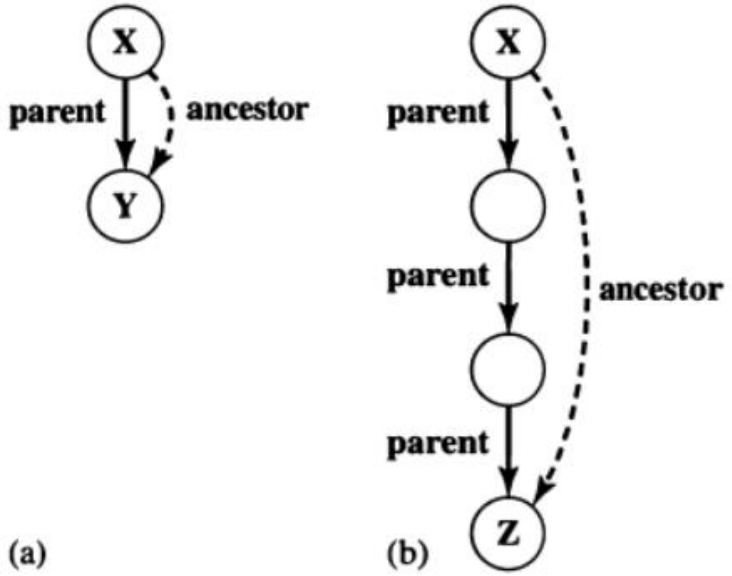
Per esempio vogliamo mostrare gli antenati di X. Con la relazione parent possiame vederi I parenti di X che sono anche antenati, ma se questo continua con I nonni I bisnonni etc allora non e piu una semplice situazione (ciclo infinito)

```
ancestor( X, Z) :-
   parent( X, Z).

ancestor( X, Z) :-
   parent( X, Y),
   parent( Y, Z).

ancestor( X, Z) :-
   parent( X, Y1),
   parent( Y1, Y2),
   parent( Y2, Z).

ancestor( X, Z) :-
   parent( X, Y1),
   parent( Y1, Y2),
   parent( Y2, Y3),
   parent( Y3, Z).
. . .
```

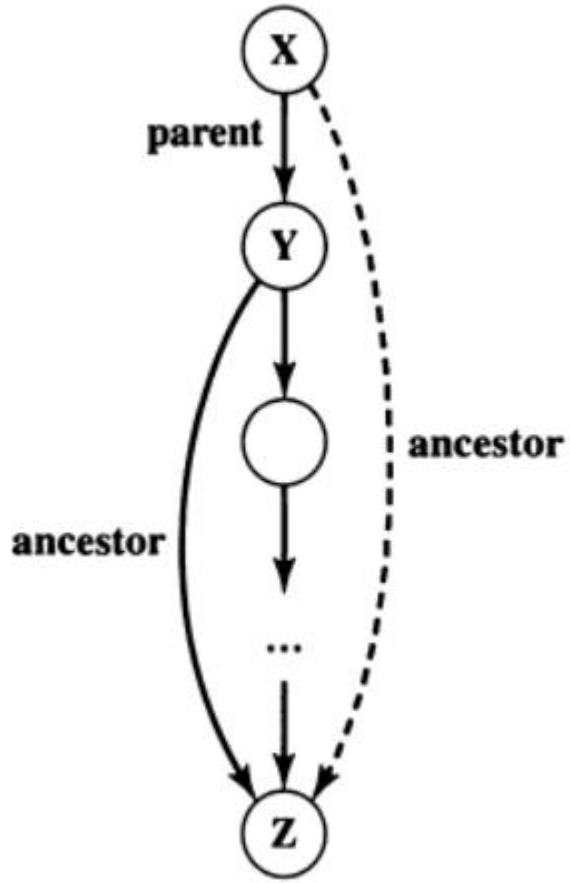


Relazione antenati (ancestor)

La descrizione piu elegante e correta e questa :

- For all X and Z,
- X is an ancestor of Z if
- there is a Y such that
- (1) X is a parent of Y and
- (2) Y is an ancestor of Z.

ancestor(X, Z) :-
parent(X, Y),
ancestor(Y, Z).



Data Objects

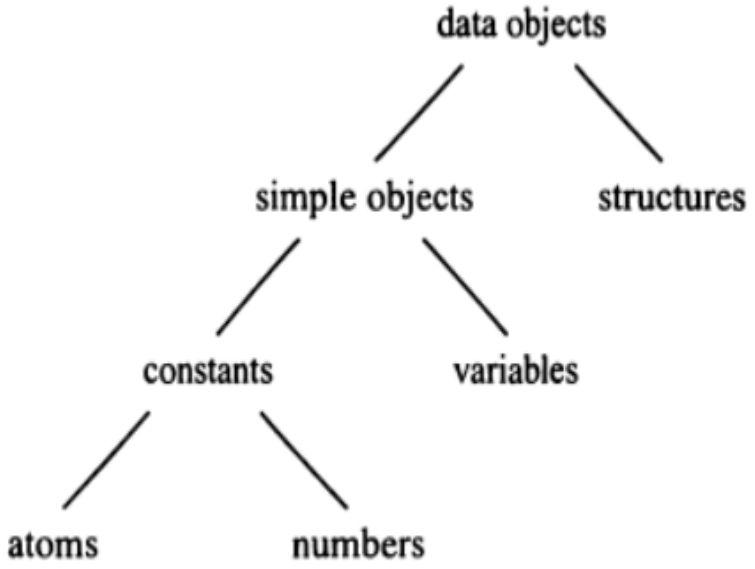


Figure 2.1 Data objects in Prolog.

Gli atomi e I numeri

- upper-case letters A, B, . . . , Z
- lower-case letters a, b, . . . , z
- digits 0, 1, 2, . . . , 9
- special characters such as + − * / < > = : . & _ ~

Gli atomi

I atomi si possono costruire in 3 modi:

1. Delle stringe con letter, numeri e charateri specali come underscore '_', e che cominciano con il charatere miuscolo per esempio

anna, x25, x_25 , x_25AB , x_ , x y ,alpha_beta_procedure , missjfones eccetera.

1. Stringe con charateri speciali

<- - ->
= = = = = = =>
+
. . .
. : .
: : =

Ci sono dei casi come ':-'. Che e def

1. Stringa di charateri messi

'Tom'
'South_America'
'Sarah Jones'

I numeri

- I numeri usati in Prolog includono integers e numeri reali . La sintasse dei integer e semplice per esempio: 1 1313 0 -97 (integer e numeri positivi/negative)

Non tutti I numeri integer si possono rapresentare in un computer , a tal quale la range e limitata dall'interval permesso dal implementazione di un Prolog particolare.

- La sintasse dei numeri reali e mostrata dal esempio:

3.14 -0.0035 100.2 7.15E-9

I numeri

L’ultimo esempio usa la notazione dell’exponete e e equivalente a 0.00000000715.

Dobbiamo essere attenti durante l’uso dei numeri reali qundo vogliamo usare la funzione del round (arithmetic) perche possiamo effetuare dei errori come quella presentata dalla espressione

10000000000 + 0.0000001 - 10000000000

E il risultato puo essere 0 invece del correto resultato 0.0000001.

I variabli

I variabli sono dei string di lettere , numeri e charateri speciali come underscore ‘_’.

Sempre cominciano con una lettere maiuscola oppure con un underscore.

Queste due richieste scritte sul listener avrano lo stesso risultato?

- has_a_child(X):- parent(X, Y).
- has_a_child(X):- parent(X, _).

X
Result
Object2
Participant_list
ShoppingList
_A
_x23
_23

Practica

Qualle delle richieste fatte sul Prolog ha la sintasse correta? Che tipi di oggetti sono questi (atom, number, variable)?

(a) Diana

(b) diana

(c) 'Diana'

(d) _diana

(e) 45

Esercizi (predicati con un argumento)

1. Create delle predicati con un argumento (per la knowledge base) che identifica il gender delle persone di una famiglia
2. Usate delle richieste (query) che:
    1. Confirma che uno delle persone della famiglia e maschio oppure e femmina
    2. Mostra tutti i maschi della famiglia
    3. Mostra tutte le femmine della famiglia

Knowledge base

```
room( kitchen).
room( office).
room( hall).
room(' dining room’).
room( cellar).

door( office, hall).
door( kitchen, office).
door( hall, 'dining room’).
door( kitchen, cellar).
door(' dining room', kitchen).

location( desk, office).
location( apple, kitchen).
location( flashlight, desk).
location(' washing machine', cellar).
 location( broccoli, kitchen).
ocation( crackers, kitchen).
location( computer, office).
```

1.Trovate la risposte che seguirano le richieste (query) sul listener

?- location(apple, kitchen).
?- location(kitchen, apple).

?- door(office, hall).
?- door(hall, office).

?- location(X, kitchen).
?- location(computer, X).

2.Create una richiesta che mostra tutte le room?

Domande?