

12-16 Oct 2009

1. Laborator Inteligență Artificială.

Scurta introducere in PROLOG (1)

Abordarea practică a Inteligenței Artificiale prezumă elaborarea de programe în scopul rezolvării unor probleme particulare. Din acest punct de vedere, Prolog-ul este un mediu de lucru în care programatorul **definește și testează un limbaj** care modelează problema particulară dată.

În sine, un program Prolog este cu o eroare mai mică sau mai mare un Agent Inteligent: - o aplicație care acționează adaptat la circumstanțe date, la experiența proprie și la un anumit scop, flexibilă la schimbările de mediu și scop, producând (găsind, alegând) soluții în urma unui calcul finit desfășurat în conformitate cu reguli impuse.

Ca urmare orice program Prolog descrie un **mediu**, caracterizat de un **context** și de **reguli** de deducție, și are un **scop**.

Contextul este precizat prin declararea constantelor (în secțiunea **CONSTANTS**), a tipurilor de date ce vor fi prelucrate (în secțiunea **DOMAINS**) și a unei baze de date (de fapte) interne sau externe (în secțiunea **DATABASE**)

Relațiile între date bine precizate (date bine individualizate, cărora le corespund obiecte particulare din universul real) constituie **fapte**.

Relațiile între date generice (variabile de anume tipuri modelând fiecare o întreagă clasă de obiecte din universul real) constituie **reguli** de deducție. Atât faptele cât și regulile de deducție sunt predicate și se declară în secțiunea **PREDICATES**.

Predicatele: (definesc relații)		<ul style="list-style-type: none"> - Fapte (relaționează date individualizate) - Reguli (relaționează date, fapte, reguli)
-------------------------------------------	--	----------------------------------------------------------------------------------------------------------------------------------------------------

Scopul programului poate fi intern - atunci când este specificat în program în secțiunea **GOAL**, respectiv extern - atunci când este specificat ca interogare la run-time. Structura unui program Prolog este prezentată în exemplul următor:

CONSTANTS

titlu="Family Tree"

DOMAINS

human=symbol

DATABASE

parinte(human, human)

feminin(human)

masculin(human)

PREDICATES

mama(human, human)

GOAL

write(titlu),nl,write("Mama Tatianei este "),mama(X,tatiana),write(X),nl.

CLAUSES

mama(X,Y):-parinte(X,Y),feminin(X).

masculin(ion).

feminin(tatiana).

feminin(cati).

parinte(ion,tatiana).

parinte(cati,tatiana).

Dintre toate secțiunile, obligatorii sunt PREDICATES și CLAUSES.

De remarcat că programarea în Prolog are pe de o parte caracter **declarativ** (descriptiv), programatorul precizând relațiile care au loc între date, și pe de altă parte un caracter **procedural**, programatorul implementând un algoritm care permite obținerea datelor de ieșire din cele de intrare.

De asemenea, Prolog-ul poate fi privit ca mediu de definire a unui limbaj particularizat și extensibil de interogare a unei baze de date, asemenea SQL-ului.

Domeniile predefinite în Prolog nu trebuie specificate în secțiunea DOMAINS. Le amintim aici pe cele uzuale:

- Char ('a'),
- Integer (± 32768),
- Real (cu module cuprinse între $1e-307$ - $1e308$),
- String (maxim 255 de caractere între ghilimele),
- Symbol (orice șir de cifre și litere și “_” începând obligatoriu cu o literă mică).

Declarația domeniilor are forma generală:

tip1, tip2, ..., tipn=tip

unde **tip1, tip2, ..., tipn** sunt numele tipurilor definite la momentul curent de utilizator iar **tip** este numele unui tip predefinit sau definit anterior.

Secțiunea DATABASE conține antetul faptelor, iar secțiunea PREDICATES conține antetul predicatelor:

numepredicat(tip1,tip2,...,tipn)

unde **tip1,tip2,...,tipn** sunt tipuri predefinite sau definite de utilizator.

În Prolog:

- un predicatul poate avea ca argumente: variabile, constante, simboluri;
- numărul de argumente al unui predicat definește *aritatea* acestuia.
- un predicat poate fi definit (asertat) și ca fapt (exemplu: factorial(0,1).);
- un predicat poate fi definit **disjunctiv**, prin alternative separate de “;” (care encodează operatorul logic **sau**);
- un predicat poate fi definit **conjunctiv** prin expresii separate prin “,” (care encodează operatorul logic **și**);
- există predicate standard (predefinite) care nu se declară în PREDICATES și nici nu se implementează în secțiunea CLAUSES: **fail, not, ! (cut), findall**;
- numele unei variabile începe întotdeauna cu **Majusculă** sau “_”;
- o variabilă desemnată prin “_” se numește variabilă anonimă și se folosește atunci când nu interesează returnarea valorii pentru care are loc o anumită proprietate ci doar faptul că există o valoare sau un symbol pentru care proprietatea este

adevărată. Exemple: **parinte(,_)** – folosită pentru a testa dacă, utilizând datele și regulile programului, se poate deduce că există dublete de tip părinte-copil; **parinte(tatiana,_)** – folosită pentru a testa calitatea de părinte a Tatiane; **parinte(ion,tatiana)** – folosită pentru a testa raportul părinte-copil între Ion și Tatiana

- variabilele generice neanonime cărora nu li s-a precizat încă o valoare se numesc **variabile libere**. Cele cu valoare alocată sunt numite **variabile legate**.
- **unei variabile legate nu i se poate schimba valoarea**. Dacă utilizatorul inițializează variabila X cu 2 ($X=2$), atunci până la sfârșitul programului valoarea lui X va rămâne 2 și orice încercare de a “reinițializa” variabila ($X=3$, de exemplu) va fi tratată ca test al egalității lui X cu “noua” valoare (este variabila X egală cu 3?) și va returna **False**.
- rândurile comentate încep cu ”%”, blocurile comentate sunt delimitate la început și la sfârșit prin”/*” și “*/”.
- nu contează ordinea declarării și implementării predicatelor ci doar ordinea clauzelor care implementează un predicat.

Secțiunea CLAUSES implementează **fapte** (în exemplul de mai sus: masculin(ion).) și **reguli** de deducție. Regula de deducție mama(X,Y) din programul de mai sus înseamnă: pentru orice variabile X și Y este adevărată afirmația:

X este mama lui Y	Dacă	X este părinte pentru Y	și	X este de gen feminin.
Mama(X,Y)	:-	parinte(X,Y)	,	feminin(X).

Scopul intern (în caz că există) este (obligatoriu) o conjuncție de expresii terminată cu punct și poate fi văzută ca fiind principalul predicat al programului. Scopul extern este precizat în timpul rulării programului și poate fi orice conjuncție de predicate definite în program sau/și predefinite în Prolog.

Aplicație practică:

1. Să se dezvolte programul anterior prin mărirea numărului de simboluri și prin definirea de noi predicate, conform contextului reprezentat grafic în figura 1, în care: persoanele (Ion, Cati, Vasile, Ana, Adrian, Tatiana, Nicu, Anda, Alex, Gelu, Maria) vor fi reprezentate ca simboluri iar faptele definite inițial vor corespunde relațiilor soț-soție, părinte-fiu/fiică și genului masculin/feminin al persoanelor numite. Se va urmări ca programul să deducă relații noi din cele existente.

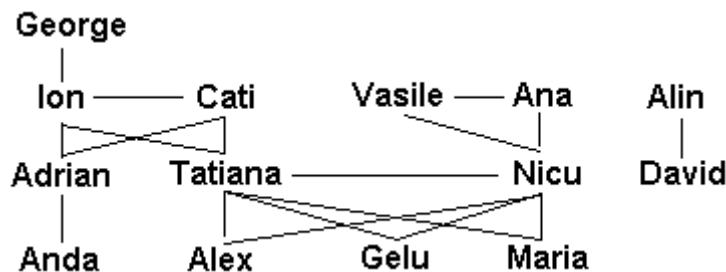


Fig.1 Graful unor relații familiale

2. Să se definească predicatele: tată, mamă, frate, soră, soră sau frate, bunic sau bunică, bunic, bunică, unchi, mătușă,
3. Să se definească predicatele: predecesor, succesori;

4. Formulați și implementați alte interogări posibile (minim una, eventual cu un grad mai ridicat de dificultate);
5. Să se definească predicatul:
 - **un cel mai îndepărtat predecesor,**
 - **un cel mai îndepărtat succesori,**
 - **înrudire genetică,**
 - **înrudire non-genetică** (“prin alianță”);
6. Ce puteți spune acum despre “tehnica de redactare” a predicatelor Prolog?
7. Ce găsiți că este de remarcant cu privire la predicatul **predecesor** și **sucesori**?
8. Cum vă explicați apariția soluțiilor multiple care apar ca rezultate ale unor anume interogări?
9. Folosind predicatul construit, exersați interogări care conțin și/numai variabile anonime. Explicați (“traduceți” în limbaj natural) atât interogările cât și rezultatul acestora.

Barem pentru evaluarea activității de laborator:

- 1 punct pentru fiecare cerință tratată satisfăcător
- 1 punct suplimentar pentru prima cerință;

Bibliografie:

<http://fmi.spiruharet.ro/bodorin/aicl.html#comlectures>