

Programare Logică – LISTĂ SUBIECTE DE EXAMEN

Claudia MUREȘAN, c.muresan@yahoo.com, cmuresan@fmi.unibuc.ro

UNIVERSITATEA DIN BUCUREȘTI, FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

2019–2020, Semestrul I

Exercițiul 1. Considerăm un limbaj de ordinul I conținând un simbol de operație ternară f , unul de operație binară g , unul de operație unară h și un simbol de constantă c . Fie X, Y și Z variabile distincte.

Să se deseneze arborii de expresii asociați următorilor doi termeni, apoi, prin aplicarea algoritmului de unificare, să se determine dacă acești termeni au unificator și, în caz afirmativ, să se determine un cel mai general unificator pentru aceștia:

$f(g(h(X), g(X, Y)), f(X, h(X), g(X, X)), h(h(Y)))$ și $f(g(h(c), g(h(c), h(c))), f(c, h(Z), g(h(c), h(c))), h(h(Z)))$.

Exercițiul 2. Având următoarea bază de cunoștințe în Prolog, scrisă respectând sintaxa Prolog:

joc(sah, 2). joc(solitaire, 1).

joc(cartiJoc, OricatiJucatori).

prefera(ana, Joc) :- joc(Joc, 1).

prefera(victor, Joc) :- joc(Joc, 2).

să se scrie arborele de derivare prin rezoluție SLD pentru următoarea interogare:

?- *prefera(Cine, Ce).*

și să se determine soluțiile date de Prolog acestei interogări folosind acest arbore de derivare.

Exercițiul 3. Să se scrie în Prolog un predicat binar

listmaxfelemlist(ListaListe, CMLungaListaFaraElementeListe),

definit ca mai jos, precum și toate predicatele auxiliare necesare pentru definirea acestuia:

listmaxfelemlist să fie satisfăcut primul său argument este o listă LL de liste având elemente care nu au elemente de tip listă, iar al doilea argument al său este una dintre listele din LL care nu are elemente de tip listă la rândul ei și este de lungime maximă raportat la această proprietate;

și, într-o interogare în Prolog, *listmaxfelemlist* să funcționeze sub forma: dacă primește, în primul argument, o listă arbitrară de liste LL , atunci să întoarcă false dacă LL nu conține liste care să nu conțină liste, iar, în caz contrar, să obțină în al doilea argument o listă L din LL de lungime maximă astfel încât L să nu conțină liste; de exemplu:

la interogările următoare:	Prologul să răspundă:
?- <i>listmaxfelemlist([], Lista).</i>	<i>false;</i>
?- <i>listmaxfelemlist([[], [1, 2, [V]]], Lista).</i>	<i>false;</i>
?- <i>listmaxfelemlist([[], [1, 2, [V]]], Lista).</i>	<i>Lista = [];</i>
?- <i>listmaxfelemlist([[], [1, 2], [], [], X], [a, V], [x, [1, 2], c], Lista).</i>	<i>Lista = [1, 2] sau Lista = [a, V];</i>
?- <i>listmaxfelemlist([[], [1, 2], [], [], X], [a, b, c, A], [x, [1, 2], c], Lista).</i>	<i>Lista = [a, b, c, A].</i>

Exercițiul 4. Să se scrie în Prolog un predicat binar *elimmmar(Termen, Aritate, TermenModificat)* definit ca mai jos, precum și toate predicatele auxiliare necesare pentru definirea acestuia:

elimmmar să fie satisfăcut ddacă primul argument și cel de-al treilea argument al său sunt termeni Prolog, al doilea argument al său este un număr natural mai mic sau egal cu aritatea operatorului dominant al primului său argument, iar al treilea argument se obține din primul prin eliminarea subtermenilor cu operatorul dominant de aritate strict mai mare decât al doilea argument (cei mai mari posibili, adică

indiferent de aritățile operatorilor dominanți ai subtermenilor acestora), astfel că, dacă toate argumentele unui operator sunt eliminate, atunci acel operator devine constantă;

și, într-o interogare în Prolog, *elimmmar* să funcționeze sub forma: dacă primește un termen arbitrar T în primul argument și un număr natural arbitrar A în al doilea argument, să întoarcă false dacă aritatea operatorului dominant al lui T este strict mai mare decât A , iar, în caz contrar, să construiască, în al treilea argument, termenul obținut din T prin eliminarea subtermenilor cu operatorul dominant de aritate mai strict mai mare decât A ; de exemplu:

la interogările următoare:	Prologul să răspundă:
?- <i>elimmmar</i> (ct , 0, <i>Termen</i>).	$Termen = ct$;
?- <i>elimmmar</i> ($f(ct)$, 0, <i>Termen</i>).	$false$;
?- <i>elimmmar</i> ($f(a, b)$, 1, <i>Termen</i>).	$false$;
?- <i>elimmmar</i> ($h(h(f(a, b)))$, 1, <i>Termen</i>).	$Termen = h(h)$;
?- <i>elimmmar</i> ($f(a, f(g(x, y, Z), h(U, g(a, b, g(V, t, u))))$), 2, <i>Termen</i>).	$Termen = f(a, f(h(U)))$.
?- <i>elimmmar</i> ($f(a, f(f(g(a, b, c), g(A, B, C)), h(U, g(a, b, g(V, t, u))))$), 2, <i>Termen</i>).	$Termen = f(a, f(f, h(U)))$.