Programare Logică – listă subiecte de examen

Claudia MUREŞAN, c.muresan@yahoo.com, cmuresan@fmi.unibuc.ro
Universitatea din Bucureşti, Facultatea de Matematică şi Informatică

2019–2020, Semestrul I

Exerciţiul 1. Considerăm un limbaj de ordinul I conţinând un simbol de operaţie ternară f, unul de operaţie binară g, unul de operaţie unară h şi un simbol de constantă c. Fie X, Y şi Z variabile distincte.

Să se deseneze arborii de expresii asociaţi următorilor doi termeni, apoi, prin aplicarea algoritmului de unificare, să se determine dacă aceşti termeni au unificator şi, în caz afirmativ, să se determine un cel mai general unificator pentru aceştia:

$$f(g(h(X), g(X, h(c))), f(X, h(X), g(X, X)), h(h(h(c)))) \text{ şi } f(g(h(c), g(Z, h(Z))), f(c, h(Z), g(Z, Z)), h(h(h(Z)))).$$

Exerciţiul 2. Având următoarea bază de cunoştinţe în Prolog, scrisă respectând sintaxa Prolog:
$joc(sah, 2).$ $joc(reversi, 2).$ $joc(solitaire, 1).$
$joc(cartiJoc, OricatiJucatori).$
$prefera(ana, Joc)$:- $joc(Joc, 1).$
$prefera(victor, Joc)$:- $joc(Joc, 2).$
să se scrie arborele de derivare prin rezoluţie SLD pentru următoarea interogare:
?- $prefera(Cine, reversi), prefera(ana, Ce).$

Exerciţiul 3. Să se scrie în Prolog un predicat binar $elimlist(ListaListe, ListaListeFaraElementeListe)$, definit ca mai jos, precum şi toate predicatele auxiliare necesare pentru definirea acestuia:

elimlist să fie satisfăcut ddacă ambele argumente ale sale sunt liste de liste, iar al doilea argument al său se obţine din primul său argument prin eliminarea elementelor care, la rândul lor, conţin elemente de tip listă;

şi, într–o interogare în Prolog, *elimlist* să funcţioneze sub forma: dacă primeşte o listă arbitrară de liste LL în primul argument, să obţină în al doilea argument lista elementelor lui L care nu conţin, la rândul lor, liste; de exemplu:

la interogările următoare:	Prologul să răspundă:
?- $elimlist([\,], Lista)$.	$Lista = [\,]$;
?- $elimlist([[\,], [1, 2, V]], Lista)$.	$Lista = [[\,], [1, 2, V]]$;
?- $elimlist([[\,], [1, 2], [[\,]], [[\,], X], [a, b, c, A], [x, [1, 2], c]], Lista)$.	$Lista = [[\,], [1, 2], [a, b, c, A]]$.

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

elimmmar să fie satisfăcut ddacă primul argument şi 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 obţine din primul prin eliminarea subtermenilor cu operatorul dominant de aritate strict mai mare decât al doilea argument (cei mai mari posibili, adică indiferent de arităţile operatorilor dominanţi ai subtermenilor acestora), astfel că, dacă toate argumentele unui operator sunt eliminate, atunci acel operator devine constantă;

şi, într–o interogare în Prolog, *elimmmar* să funcţioneze sub forma: dacă primeşte un termen arbitrar T în primul argument şi 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 obţinut din T prin eliminarea subtermenilor cu operatorul dominant de aritate mai strict mai mare decât A; de exemplu:

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