Analiza si întelegerea textelor operatiunilor contabile folosind limbajul Prolog

Lect. Bogdan PATRUT
Facultatea de Stiinte, Universitatea din Bacau
bogdanpatrut@xnet.ro

This paper presents a method of parsing of accounting operations texts, written in natural language. It also presents a method of translation of this kind of texts from natural language into accounting articles (records). The paper presents a PROLOG implementation, which uses lists of strings, on which many transformations have been made, such as the elimination of insignificant words and the (ascending) sorting of the remaining list of words.
Key words: *accounting, natural language processing, artificial intelligence, lists, Prolog, translation, accounting operations*

Problema

Desi contabilitatea este un domeniu în care inteligenta artificiala îsi face simtita prezenta din ce în ce mai mult prin realizarea de sisteme expert, totusi mai este mult pâna când vom putea vorbi despre "expertul contabil artificial", capabil sa ia singur deciziile unui expert contabil uman de astazi. Totusi, asa dupa cum se cunoaste din experienta ultimilor ani, anumite operatii contabile au fost automatizate, în programele de contabilitate comerciale.

Articolul de fata doreste sa trateze un subiect special al contabilitatii, acela de "întelegere" a textului dintr-o lucrare de contabilitate, text aferent unei înregistrari contabile. În literatura de specialitate (limbaje formale si compilatoare, prelucrarea limbajului natural), procedura se numeste "parsare" (engl. "parsing") si consta într-o analiza sintactica, urmata de o "interpretare" si "executare" a textului analizat. Astfel, ne propunem ca, furnizându-i sistemului nostru descrierea unei operatii contabile, sa primim de la el înregistrarea contabila corespunzatoare.

Programul elaborat si prezentat mai jos este pur didactic, el neavând pretentia unui sistem complet functional, dar poate sta la baza unor sisteme mai complexe de acest gen.

De exemplu, am dori ca pentru fraze de genul celor din stânga sa obtinem raspunsuri ca cele din dreapta:

a) se subscrie un capital social în valoare de 3000 lei => 456 = 1011, 3000 lei
b) se depune sub forma de aport în natura un teren în valoare de 2000 lei => 2111 = 456, 2000 lei
c) se înregistreaza capitalul social de 3000 lei ca fiind varsat => 1011 = 1012, 3000 lei

Transformarea frazelor

Cum am putea realiza acest lucru? Mai întâi va avea loc o transformare a frazelor din forma lor initiala de sir de caractere, în liste de siruri de caractere, stiut fiind faptul ca în limbajul PROLOG se pot implementa predicate eficiente pentru lucrul cu liste. Predicatul fa_lista realizeaza transformarea sirului reprezentând textul initial al înregistrarii contabile într-o lista de cuvinte, adica desparte fraza initiala în cuvinte. Apoi, din lista acestor cuvinte se va extrage "cuvântul" care reprezinta suma numerica (folosindu-se predicatul extrage_suma). Acesta va fi determinat usor, fiindca începe cu o cifra.

Pentru ca în continuare lucrurile sa fie simplificate, se elimina cuvintele care nu sunt considerate relevante pentru fraza respectiva. Astfel, nu se vor lua în considerare nici prepozitii, nici alte cuvinte care nu figureaza într-un dictionar dat. Predicatul care simplifica o lista parcurge lista si analizeaza, pe rând, fiecare cuvânt. Acest cuvânt este cautat într-un dictionar (vezi predicatul dictionar si clauzele sale) si, daca

exista, atunci este înlocuit cu varianta sa standard. Celelalte variante sunt considerate sinonime. În caz ca respectivul cuvânt nu exista în dictionar, atunci el nu va mai aparea în lista simplificata.
Dupa eliminarea cuvintelor "nesemnificative", are loc o ordonare alfabetica a cuvintelor ramase, realizata de predicatul sorteaza, ce implementeaza un procedeu de sortare prin inserare. Astfel, propozitii de genul "se varsa capitalul" sau "capital varsat" vor avea acelasi înteles.
În sfârsit, se apeleaza la predicatul trad, care realizeaza traducerea dorita, obtinându-se înregistrarea contabila ceruta.

Exemplu
Pentru exemplificare, sa consideram fraza: "se subscrie un capital social în valoare de 3000 lei".
Dupa aplicarea primei transformari (fa_lista), se obtine lista ["se", "subscrie", "capitalul", "social", "in", "valoare", "de", "3000", "lei"]. Dupa aceasta, predicatul extrage_suma va obtine valoarea de 3000 (lei), care va fi afisata (la finalul executiei programului) alaturi de înregistrarea contabila. Aplicând predicatul simplifica asupra listei de cuvinte obtinute mai înainte, ramânem doar cu lista ["subscrie", "capital"], pentru ca s-a ales varianta standard nearticulata "capital" în locul lui "capitalul", iar celelalte cuvinte nu apar în dictionarul nostru. Apoi, lista aceasta se ordoneaza alfabetic, rezultând ["capital", "subscrie"], care, conform traducerii date de predicatul trad, conduce la înregistrarea contabila 456=1011.
Fireste, programul nostru este pur demonstrativ. Un program complex ar avea inclusa sau ar apela la o baza de date care sa cuprinda o multitudine de variante de "traducere", precum si alt gen de prelucrari ale frazei. Aceste tehnici fac obiectul disciplinei numite Prelucrarea limbajului natural. Consideram, însa, ca abordarea acestui subiect pentru realizarea unor sisteme informatice inteligente este o problema de interes pentru toti programatorii economisti.

Implementarea Turbo-Prolog

```
domains
      cont=integer
      lista=string*

predicates
      dictionar(string,lista)
      membru(string,lista)
      simplifica(lista,lista)
      fa_lista(string,lista)
      trad(lista,cont,cont)
      transf(string,cont,cont,integer)
      repeat
      run
      sorteaza(lista,lista)
      insereaza(string,lista,lista)
      extrage_suma(lista,integer)
clauses
      dictionar(terenuri,[terenuri,teren,suprafata,suprafete]).
      dictionar(constructii,[constructii,constructir,cladire,cladiri]).
      dictionar(subscrie,[subscrie,subscris]).
      dictionar(aport,[aport,contributie,adus]).
      dictionar(varsa,[varsa,varsat]).
      dictionar(capital,[capital, capitalul]).
      dictionar(rezerve,[rezerve]).
      dictionar(incorporeaza,[incorporeaza]).
      dictionar(rascumparare,[rascumparare]).
      dictionar(actiuni,[actiuni]).
```

```
trad([capital,subscrie],456,1011).
trad([capital,varsa],1011,1012).
trad([aport,terenuri],2111,456).
trad([aport,constructii],2121,456).
trad([incorporeaza,rezerve],1061,1012).
trad([actiuni,rascumparare],502,5121).

membru(C,[C|_]) if !.
membru(C,[_|Rest]) if membru(C,Rest).

fa_lista("",[]) if !.
fa_lista(S,L) if fronttoken(S,Cuv,S1), fa_lista(S1,L1),L=[Cuv|L1].

extrage_suma([],0) if !.
extrage_suma([X|_],Suma) if str_int(X,Suma), !.
extrage_suma([_|Rest],Suma) if extrage_suma(Rest,Suma).

simplifica([],[]) if !.
simplifica([Cap|Rest],L) if
      dictionar(CapStandard,Sinonime),
      membru(Cap,Sinonime),
      simplifica(Rest,L1),L=[CapStandard|L1],!.
simplifica([_|Rest],L)
      if simplifica(Rest,L).

transf(Propozitie,CD,CC,Suma) if
      fa_lista(Propozitie,L), extrage_suma(L,Suma),
      simplifica(L,L1), sorteaza(L1,L2), !, trad(L2,CD,CC).

sorteaza([X|Rest],S) if sorteaza(Rest,R),insereaza(X,R,S).
sorteaza([],[]).

insereaza(X,[Y|Rest],[Y|Rest1]) if X>Y, insereaza(X,Rest,Rest1).
insereaza(X,Rest,[X|Rest]).

repeat.
repeat if repeat.

run if repeat,
      write(">>"), readln(Propozitie),
      transf(Propozitie,ContDebitat,ContCreditat,Suma),
      write(ContDebitat," = ",ContCreditat,", ",Suma," lei"),
      nl, fail.
```

Bibliografie

1) Alecsandru P. Tacu, Romul Vancea, Stefan Holban, Aurel Burciu - "Inteligenta artificiala. Teorie si aplicatii în economie". Editura Economica, Bucuresti, 1998

2) Cristian Masalagiu, Liliana Ibanescu, Stefan Andrei - "Practica programarii în Turbo Prolog", Editura Universitatii "Al. I. Cuza", Iasi, 1998

3) Vasile Patrut, Aristita Rotila - "Contabilitatea întreprinderii", Editura Alma Mater, Bacau, 2002

4) Judith Meszaros - "Turbo Prolog 2.0. Ghid de utilizare", Editura Albastra, Cluj-Napoca, 1996