RAČUNARSTVO
Sveska 1, Broj 2, 1992

UDK 519.682

SVEREL—PROLOŠKI PROGRAM
(objedinjavanje svih relacija)[1]

SLAVIŠA PREŠIĆ I PAVLE BLAGOJEVIĆ

U ovom članku se izlaže jedan proloski program, i pritom u pisanju pojedinih članaka se podrazumevaju liste. Takvu LISP-ovsku sintaksu koristi, na primer, MICRO-Prolog. Program je pisan upravo u toj verziji prologa.

Osnovna zamisao tog programa je sledeća:

- „Ulaz" je izvestan zadani proloski program `P` (to je ili tekući, radni program ili program u nekoj datoteci).
- „Izlaz", odnosno plod nakon primene SVEREL–programa je nov proloski program `sverel(P)`, koji ima tačno jednu relaciju imena „`sverel`". Ta relacija `sverel` u smislu koji objašnjavamo objedinjuje sve, svejedno kog broja, relacije polaznog programa `P`. Naime, u osnovi imamo ovakvu ekvivalenciju

```
(sverel rel a1 a2 ... an) <--> (rel a1 a2 ... an)
```

tj. Objekti `a1`, `a2`, ... ,`an` su u relaciji `rel` ako i samo ako Objekti `rel`, `a1`, `a2`, ... ,`an` su u relaciji `sverel`.

Primera radi, uočimo sledeći program `P`:

```
((a 1 2)) ((a 3 4)) ((a 5 6)) ((a 2 1))
((a 3 7)) ((a 6 5)) ((a 7 7)) ((b 9 99)) ((b 1 2))
((b _x _y) (a _x _y) (a _y _x))
```

Tada njegovim „sverelovanjem" nastaje ovaj program `sverel(P)`:

```
((sverel a 1 2)) ((sverel a 3 4)) ((sverel a 5 6))
((sverel a 2 1)) ((sverel a 3 7)) ((sverel a 6 5))
```

[1]Rad finansiran od Fonda za nauku Republike Srbije preko Matematičkog instituta, projekat 0401A.

```
((sverel a 7 7)) ((sverel b 9 99)) ((sverel b 1 2))
((sverel b _x _y) (sverel a _x _y) (sverel a _y _x))
```

U vezi sa programom P može se, na primer, postaviti pitanje

```
?((a _x _y) (PP _x _y))
```

i za _x, _y će se dobiti 1,2 tj. prva rešenja. Međutim, nemoguće je na prolog direktno prevesti ovakvo pitanje:

Naći relaciju `rel` *programa* P, *znajući da 1,2 jesu u toj relaciji.*

Naime, pokušaj oblika:

```
?((_rel 1 2) (PP _rel))
```

neće uspeti jer prolog uopšte zahteva da tokom razvoja prološkog algoritma svaka usput pojavljena relacija bude poznata.

Međutim sverelovanjem programa P to pitanje se, uz pomoć `sverel`–relacije može ovako postaviti:

```
?((sverel _rel 1 2) (PP _rel))
```

i kao odgovor će se dobiti a. Da smo, recimo pitali

```
?((sverel _rel 1 2) (PP _rel)FAIL)
```

u odgovoru bismo dobili a i b.

Kao što se već iz tog malog primera vidi, i ako se u prologu u načelu ne mogu postavljati pitanja sa nepoznatim relacijama, upotrebom transformacije sverelovanje pitanja se preobraćaju na nov oblik u kome se mogu uspešno postaviti.

Uočimo još jedan primer—primer faktorijela:

```
((fakt 0 1))
((fakt _n _m) (LESS 0 _n)
              (SUM 1 _nn _n)
              (fakt _nn _mm)
              (TIMES _n _mm _m))
```

gde su LESS, SUM, TIMES tri osnovne (kaže se i sistematske) relacije prologa:

```
(LESS a b) <-> a<b;     (SUM a b c) <-> c=a+b;     (TIMES a b c) <-> c=a·b
```

Pomenimo da relacije SUM i TIMES „su sposobne“ da uz poznavanje dva argumenta izračunaju treći. Recimo, ako se usput pojavi pojavi formula (SUM 1 _nn

6) onda po definiciji relacije **SUM** _nn je izračunljiv i _nn=5. Sverelovanjem tog programa nastaje sledeći program:

```
((sverel fakt 0 1))
((sverel fakt _n _m) (LESS 0 _n)
                     (SUM 1 _nn _n)
                     (sverel fakt _nn _mm)
                     (TIMES _n _mm _m))
```

Primetite, što je bitno, sverelovanje ne „hvata“ nijednu od sistemskih relacija.

Primera radi, u vezi sa prethodnim programom smemo postaviti i ovakvo pitanje:

```
?((sverel _rel 6 720) (PP _rel))
```

i kao odgovor ćemo dobiti: **fakt**.

U nastavku sledi **sverel**–program. Nekoliko reči o načinu korišćenja. Osnovni način je da se napravi datoteka imena SVEREL.LOG, koja onda po potrebi i želji služi kao „korisnički“ (*utility*) program. Shodno tome kad želimo u okviru prologa učitamo tu datoteku sa:

```
LOAD SVEREL
```

tada ako hoćemo da sverelujemo tekući program, tj. onaj sa kojim već neposredno radimo, postavimo pitanje

```
?((Sver Tekuci))
```

Međutim, ako hoćemo da sverelujemo neki drugi program u fajl imena IME onda postavimo ovakvo pitanje

```
?((Sver Ime))
```

Jedan određen primer: `?((Sver ''BETWEEN.LOG''))`.

Napomena:

Priloženi SVEREL–program kao svoje „službene“ reči koristi:

```
Sver, citaj, sve, sverel, ne_sis
```

što znači da tekući program, na koji hoćemo da upotrebimo taj SVEREL program ne sme među imenima svojih relacija da sadrži neko od tih. Dalje, ime PRIV777.LOG je korišćeno kao ime pomoćne datoteke.

LITERATURA

[1] K. L. Clark i F. G. McCabe *Micro-PROLOG: Programming in Logic*, Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

```
(/* SVEREL PROGRAM )

((Sver Tekuci)
    (/* Ako hocemo tekuci program da sverelujemo)
    (KILL (Sver sve svefor dobro citaj))
    (SAVE "PRIV777.LOG")
    (LOAD "SVEREL.LOG")
    (Sver "PRIV777.LOG"))

(Sver _Ime) (/* Ako hocemo da sverelujemo program u datoteci Ime)
    (OPEN _Ime) (citaj _Ime) (CLOSE _Ime)
    (PP Sverelovanje zavrseno, postavite pitanje))

((citaj _X) (READ _X _Y) (sve _Y _Z) (ADDCL _Z) (citaj _X))
((citaj _X))

    ((svefor (NOT|_A) (NOT |_A1)) (svefor _A _A1))
    ((svefor (!|_A) (!|_A1)) (svefor _A _A1))
    ((svefor (_A _B) (_A _B1))
            (CON _A) (ON _A (ADDCL DELCL CL ?)) (sve _B _B1))
    ((svefor (OR _A _B) (OR _A1 _B1)) (sve _A _A1) (sve _B _B1))
    ((svefor (IF _A _B _C) (IF _A1 _B1 _C1))
            (svefor _A _A1) (sve _B _B1) (sve _C _C1))
    ((svefor (FORALL _A _B) (FORALL _A1 _B1))
            (sve _A _A1) (sve _B _B1))
    ((svefor (ISALL _A _B|_C) (ISALL _A _B|_C1)) (sve _C _C1))
    ((svefor (_A|_B) (sverel|(_A|_B)) (dobro _A))
    ((svefor _A _A))

    (dobro _A)
            (NOT ON _A (Sver citaj sve svefor dobro)) (CON _A) (NOT SYS _A))

    ((sve ((_A _B)) ((_A1 _B1)))
            (CON _A) (ON _A (ADDCL DELCL CL ?)) (sve _B _B1))
    ((sve (_X|_Y) (_P|_Q)) (svefor _X _P) (sve _Y _Q))
    (sve () ()))
```