This page is intentionally left blank.

Kazimir Majorinc

Hartovi makroi

Povijest Lispa 23.

Razmjena vještina
Hacklab u mami
2. ožujka 2013.

Makroi se javljaju prvo u asembleru. Kasnije COBOL.

Location	*Operation*	*Variable field*
M1	**MAC**	A, B, C, D, E
	OP1	A
B	E	C, R
	OP4	D
	END	

Any macro written

	M1	S1, S2, S3, S4, S5

would generate

	OP1	S1
S2	S5	S3, R
	OP4	S4

Greenwald, A Technique for Handling Macro instructions, **1959.**

Rana pojava makroa u Lispu, **McCarthyjev** metacircularni evaluator.

Primjerice

eval[(F,A);((F,(LAMBDA,(X),X)))] =

= *eval*[((LAMBDA,(X),X),A); ((F,(LAMBDA,(X),X)))]

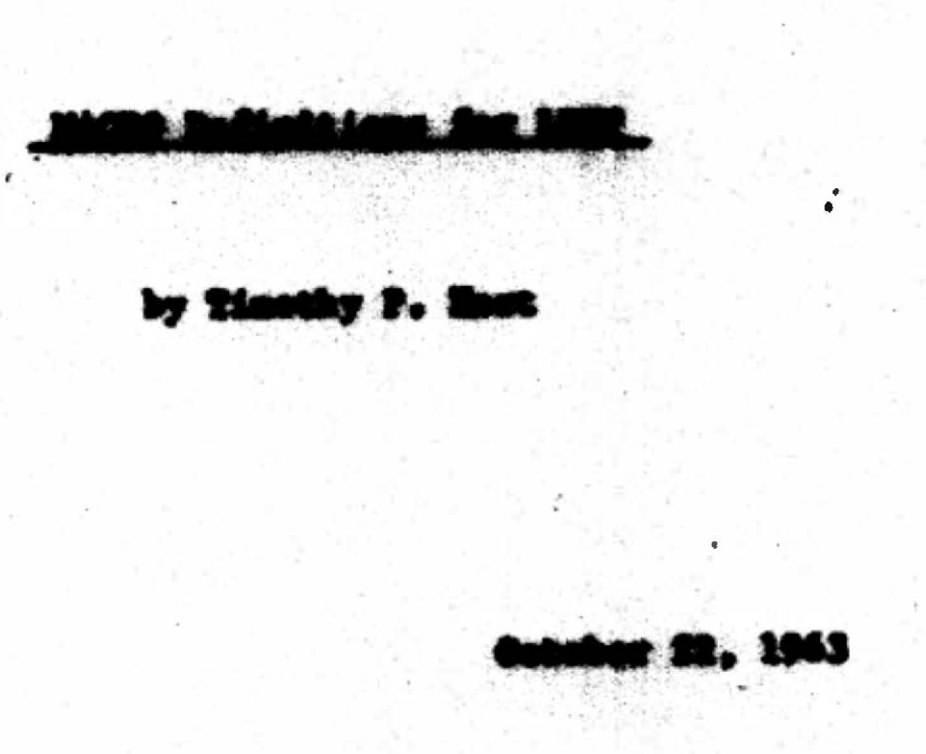
Dio u definiciji funkcije eval:

eval[*e*;*a*]= ... **T** → *eval*[*cons*[*assoc*[*car*[*e*];*a*]; *cdr*[*e*]]
a];

Specijalni, ali još uvijek makroi.

Timothy P. Hart, **MIT-AIM 057**, 22. listopad, **1963**, *Macro Definitions in Lisp*.

Općeniti makroi.



Napisao nekoliko memoa i jedan konferencijski članak. Uspješan poduzetnik. U mirovini.

Po **Hartu**, makroi u LISPu mogu zadovoljiti dvije potrebe za koje su do tada korištene specijalne forme: definiranje operatora sa proizvoljnim brojem argumenata i sprečavanje evaluacije argumenata.

Primjerice, u Lispu 1.5 pseudo-funkcija *cset* kojom se definiraju konstante se često koristi u kombinaciji sa *quote*, u izrazima poput

```
(CSET (QUOTE X) 14)
```
.

Prvi programeri u Lispu su smatrali da bi bilo korisno izbjeći čestu primjenu specijalnog operatora *quote* uvođenjem novog operatora *csetq*, tako da se umjesto gornjeg izraza piše `(CSETQ X 14)`. Dakle, prvi parametar, `X`, ne smije se izračunavati. U Lispu 1.5 postojali su *fexprovi*, operatori koji su se pozivali sa ne-evaluiranim argumentima. Konkretno, postojao je baš i fexpr *csetq*.

Hart predlaže da se za takve svrhe uvedu makroi kao nova vrsta operatora, različita od funkcija i fexprova. Neka je *m* makro, a e_1, ..., e_n neki simbolički izrazi. Poziv makroa

$$(m\ e_1\ \ldots\ e_n)$$

izračunavao be se u dvije faze. U prvoj, koju **Hart** naziva *širenje, ekspanzija* (engl. expansion) poziva se makro sa *neizračunatim* argumentima e_1, ..., e_n. Makro *m* na temelju definicije *m*, "generira" novi simbolički izraz *e*. Primjerice, makro poziv `(CSETQ X 14)` bi se u prvoj fazi raširio u `(CSET (QUOTE X) 14)`. U drugoj fazi, dobiveni izraz *e* se izračunava.

Dvije faze ne moraju se izvršavati jedna za drugom. Po **Hartovom** opisu, pozivi makroa u definicijama funkcija širili bi se odmah čim su funkcije definirane. Za to je potrebno redefinirati funkciju define iz Lispa 1.5.

Nadalje, potrebno je odrediti način na koji korisnik može sam definirati makroe. **Hart** predlaže da se makroi definiraju kao funkcije sa jednim argumentom i to je, čini se, prvi puta da su makroi tako definirani u nekom programskom jeziku. Primjerice, makro *csetq* moglo bi se definirati kao

```
MACRO (((CSETQ (LAMBDA (FORM)
                       (LIST (QUOTE CSET)
                             (LIST (QUOTE QUOTE)
                                   (CADR FORM))
                             (CADDR FORM))))))
```
[1] .

Prilikom poziva, parametru funkcije, u ovom primjeru `FORM`, se pridružuje, kao vrijednost cijeli makro poziv, u ovom primjeru `(CSET X 14)`.

---

[1] Ova se definicija makroa *setq* razlikuje od originalne **Hartove** u kojoj ima grešaka.

Drugi **Hartov** primjer je makro *stash*, koji dodaje vrijednost na početak liste. Makro poziv `(STASH` e_1 e_2`)` treba se *raširiti* u `(SETQ` e_2 `(CONS` e_1 e_2`))`, gdje su e_1 i e_2 bilo koji izrazi. Jedna moguća definicija makroa *stash* je:

```
MACRO (((STASH (LAMBDA (FORM)
                       (LIST (QUOTE SETQ)
                             (CADDR FORM)
                             (LIST (QUOTE CONS)
                                   (CADR FORM)
                                   (CADDR FORM)))))))).
```

Treći **Hartov** primjer, makro *enter*, koji kreira "točkasti par" i upisuje ga u *alistu*, ilustrira korištenje makroa pri definiranje novih makroa. Makro poziv

(ENTER e_1 e_2 e_3**)**

treba se raširiti u **(STASH (CONS** e_1 e_2**)** e_3**)**. Jedna moguća definicija makroa enter je dana meta-izrazom

```
                    MACRO
enter[form]    =    list[ STASH;
                          list[ CONS;
                                cadr[form];
                                caddr[form] ];
                          cadddr[form]        ]
```

Četvrti **Hartov** primjer odnosi se na makro *select*, koji je već definiran i implementiran u LISPu 1.5 kao fexpr. Izraz

select[*q*;(*q*$_1$ *e*$_1$);(*q*$_2$ *e*$_2$); ...; (*q*$_n$ *e*$_n$); *e*]

se izračunava tako da se prvo izračuna *q*, a onda se redom izračunavaju *q*$_1$, ..., *q*$_n$ dok se ne pronađe *q*$_i$ čija je vrijednost jednaka vrijednosti *q*. Tada se izračuna vrijednost odgovarajućeg *e*$_i$, i to je ujedno vrijednost cijelog izraza. Ako niti jedan od *q*$_i$ nema vrijednost jednaku vrijednosti *q*, onda se izračuna vrijednost posljednjeg argumenta, *e*, i to je ujedno vrijednost cijelog izraza.

Hartov macro transformira gornji izraz u

```
((LAMBDA(g)(COND ((EQ g q1) e1)
                 ...
                 ((EQ g qn) en) (T e)))
 q)
```

pri čemu je *g* simbol koji se ne pojavljuje niti u jednom od izraza *q*, *q*$_1$, ..., *g*$_n$, *e*$_1$, ..., *e*$_n$, *e*, čak niti indirektno.

Primjerice, ako izraz *q* je **(EVAL P)**, onda niti vrijednost **P** u trenutku izračunavanja ne smije biti S-izraz u kojem se pojavljuje *g*. To se postiže korištenjem LISP 1.5 funkcije *gensym* koja vraća potpuno novi, prethodno nekorišteni simbol.

Sam makro je presložen da bi ga ovdje iznijeli.

Uvođenje makroa zahtjeva promjene u Lisp sistemu. Prvo, treba definirati operator *macro*,

$$macro[l] = deflist[l;\mathbf{MACRO}]$$

gdje je *deflist* funkcija iz Lisp 1.5 koja prima dva argumenta. Prvi je lista parova $((u_1\ v_1)(u_2\ v_2)...(u_n\ v_n))$ gdje su *ui* simboli a *vi* lambda-izrazi. Drugi je „indikator.“ Na temelju toga, *deflist* kreira simbole i upisuje lambda-izraze i indikator u odgovarajuću „listu svojstava.“

Drugo, treba redefinirati funkciju *define*, tako da se odmah pri definiranju korišteni pozivi makroa raširе.

$$define[l]=deflist[mdef[l];\mathbf{EXPR}]$$

Verzija makroa *mdef* iz Q-32 compilera za Lisp 1.5 koji je distribuiran iz MIT tijekom 1964 godine (ili ranije).

```
(MDEF (LAMBDA (L)
              (COND ((ATOM L) L)

                    ((EQ (CAR L) (QUOTE QUOTE)) L)

                    ((MEMBER (CAR L)
                             (QUOTE (LAMBDA LABEL PROG)))
                     (CONS (CAR L)
                           (CONS (CADR L)
                                 (MDEF (CDDR L)) )) )

                    ((GET (CAR L) (QUOTE MACRO))
                     (MDEF ((GET (CAR L) (QUOTE MACRO)) L)))

                    (T (MAPLIST L
                                (FUNCTION (LAMBDA (J)
                                                  (MDEF (CAR J)))))))))
```

Na kraju, zašto je **Hart** uveo makroe? Ne piše u memou, samo piše da može zamijeniti fexpr-ove, pogotovo što se fexpr-ovi ne mogu definirati u postojećem kompajleru, iako je ta mogućnost specificirana u Lisp 1.5 priručniku. Dakle, MIT zajednica je imala problema sa implementacijom fexpr-ova.