



THIS PAGE IS
INTENTIONALLY
LEFT BLANK.

Kazimir Majorinc

AMBICIOZNI REFORMATOR LIONELLO LOMBARDI

Povijest Lispa 26.



Razmjena
vještina

Hacklab u mami
13. travnja 2013.

Tijekom 1960-ih **Lionello A. Lombardi** je pokušao razviti „*novu filozofiju*“ „*inkrementalnog računala*.“ Po njemu, postoji klasa informacijskih sustava karakterizirana stalnim tijekom informacija iz velike, evolucionarne baze podataka i evolucijom algoritama koju je nemoguće unaprijed predvidjeti. Takvi sustavi ne trebaju čekati sve podatke potrebne za izračunavanje nego trebaju izvršiti onoliko izračunavanja koliko mogu sa podacima kojima raspolažu.

„Inkrementalna računala“ se, vjeruje **Lombardi**, bitno razlikuju od **Turingovih** i **von Neumannovih**, „*konvencionalnih računala*“ koja koriste potpuno određene, dovršene algoritme. U svakodnevnom

životu, smatra **Lombardi**, inkrementalna računala bi bila mnogo korisnija od konvencionalnih.

Bertram Raphael¹ je djelomično implementirao dio funkcija „inkrementalnog računala“ u Lispu.

¹ **Lombardi & Raphael**, *LISP as the language for an incremental computer*, u **Berkeley** and **Bobrow**, *Programming language LISP - its operations and applications*, 1964., p. 204-19.

Funkcije se mogu pozivati sa *izostavljenim argumentima* (engl. omitted arguments.) Poredak ostalih, dostavljenih argumenata mora ostati nepromijenjen, i to se postiže korištenjem posebnog simbola, **NIL*** za izostavljeni argument. Dostavljeni argumenti se izračunavaju, vrijednost se supstituira na odgovarajuća mjesta u definiciji, i odgovarajuća varijabla se briše iz liste vezanih varijabli funkcije. Primjerice,

$$\mathit{cons}[\mathbf{NIL}^*; \mathbf{B}] = \lambda[[x]; \mathit{cons}[x; \mathbf{B}]].$$

Inkrementalno računalo omogućuje i izračunavanje izraza sa *neodređenim argumentima* (engl. undetermined arguments.)

Primjerice, funkcija

$$list3 = \lambda[[x;y;z]; cons[x; cons[y; cons[z; NIL]]]$$

može se primjeniti na druge funkcije, primjerice

$$list3[D; \lambda[[u]; cons[E; u]]; \lambda[[u]; car[u]]] .$$

Vrijednost posljednjeg izraza se dobija uvrštavanjem argumenata u definiciju funkcije.

$$\text{cons}[\mathbf{D};$$
$$\quad \text{cons}[\lambda[[u];\text{cons}[\mathbf{E};u]];$$
$$\quad \quad \text{cons}[\lambda[[u];\text{car}[u]];$$
$$\quad \quad \quad \mathbf{NIL} \quad \quad \quad]]] =$$

te preformuliranjem dobijene funkcije u obliku lambda-izraza.

$$= \lambda[[r;s];$$
$$\quad [\text{cons}[\mathbf{D};$$
$$\quad \quad \text{cons}[\text{cons}[\mathbf{E};r];$$
$$\quad \quad \quad \text{cons}[\text{car}[s];$$
$$\quad \quad \quad \quad \mathbf{NIL} \quad \quad \quad]]]].$$

Lombardi je adresirao i temu koja se uskoro pokazala glavnim problemom Lispa - konflikte varijabli. Da bi se izbjegli konflikti varijabli svaka vezana varijabla, onog trena kad je unešena u sistem, mijenja se novim atomskim simbolom generiranim uz pomoć *gensym*. Kako je ondašnji *gensym* generirao simbole u obliku *g00001*, *g00002* itd, to bi gornji lambda-izraz bio

```
= λ[[g00001; g00002];  
    [cons[ D;  
          cons[ cons[E; g00001];  
              cons[car[g00002];  
                  NIL ]]]].
```


Inkrementalni kompjuter je implementiran LISP funkcijom *evalquote1* koja se ponaša kao *evalquote*, samo što prethodno provjeri je li potrebno neko „inkrementalno procesiranje.“ Ako da, *evalquote1* provodi „parcijalnu evaluaciju.“ Neka pojednostavljena programa se mogu izvršiti automatski. Primjerice,

$$f[\text{NIL}^*; \dots; \text{NIL}^*] = f.$$

Lombardijevi opisi „inkrementalnog računala“ su entuzijastični a iznešene ideje zanimljive, i moglo bi se očekivati, bar u nekom kontekstu korisne. Same ideje su, međutim, toliko labavo povezane da čitaoc ostaje u nedoumici čine li one zaista neki općenitiji, novi koncept koji omogućuje ono što „tradicionalna računala“ ne omogućuju, ili ih je bolje promatrati nezavisno.