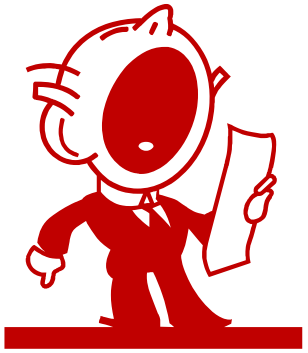
This page is intentionally left blank.

Kazimir Majorinc

Mehaničko izračunavanje izraza (prvi dio)

Povijest Lispa 29.



Razmjena vještina
Hacklab u mami
11. svibnja 2013.

Peter John Landin (1930-2009)

Sredinom **1960**-ih objavio nekoliko vrlo utjecajnih članaka koji se direktno ne tiču Lispa ali su inspirirani Lispom, te su imali velik utjecaj na razvoj funkcionalnog programiranja. Nakon 1960-ih koncentrira se na borbu za prava homoseksualaca.

Mehaničko izračunavanje izraza

Landin ovaj članak opisuje kao „teoriju“ aktivnosti korištenja računala. Pokazuje kako se neke forme izraza u suvremenim programskim jezicima mogu „modelirati“ u Churchovoj lambda-notaciji.

Simbolički izrazi i aplikativne strukture

Mnogo simboličkih izraza je karakterizirano oblikom operator/operand ili „aplikativnom strukturom.“ Primjerice,

$$a/(2b + 3)$$

je izraz čiji operator je /, a dva operanda su a i $2b + 3$. Potonji je izraz koji ima operator + i operande $2b$ i 3. Konačno, izraz $2b$ ima operator × i operande 2 i b.

Ta se struktura može izraziti jasnije u notaciji u kojoj su operandi napisani prije zagrade:

$$/(a, +(\times(2,b), 3)).$$

Neki izrazi zapisani u uobičajenom matematičkom obliku mogu se prevesti u nekoliko različitih „aplikativnih struktura.“ Primjerice,

$$3 + 4 + 5 + 6$$

može se izraziti kao

$$+(+(+(3,4),5),6),$$
$$+(3,+(4,+(5,6))) \text{ ili}$$
$$\Sigma'(3,4,5,6).$$

Koristeći **Churchovu** lambda-notaciju, možemo pripisati aplikativnu strukturu izrazima koji koriste „privatne“, „interne“, „lokalne“ ili „glupe“ varijable. Primjerice, uobičajeno

$$\int_0^x x^2 \mathrm{d}x$$

se može zapisati kao

$$\int(0, x, \lambda x.x^2)$$

Pomoćne definicije

Simbolički izraz

$$(u - 1)(u + 2) \textbf{ where } u = 7 - 3$$

se može zapisati kao

$$\{\lambda\, u\, .\, (u - 1)(u + 2)\}\, [7 - 3]$$

Razlika između operatora i funkcija

Operatori se razlikuju od funkcija. Operator je *podizraz* nekog većeg izraza. O operatoru se može govoriti samo u kontekstu izraza. Funkcija je „vrijednost“ izraza koji se može pojaviti na mjestu operatora. Onako kao što operand kvadratnog korijena mora biti nenegativan broj, tako i vrijednost operatora mora imati smisla.

Uvlačenje kao način grupiranja podizraza

Primjerice,

$$u = 2\ p + 1 \textbf{ and } v = p - 2\ q$$

može značiti svašta, ali

$$u = 2\ p + 1$$
$$\textbf{and } v = p - 2q$$

znači isto što i $(u = 2\ p + 1)$ **and** $(v = p - 2\ q)$.

Funkcije mogu imati više od jednog argumenta

Landin, kao i **McCarthy**, dopunjava **Churchovu** notaciju tako da funkcije mogu imati više od jednog argumenta.

$u\,(u + 1) - v\,(v + 1)$
where $u = 2\,p + q$
and $v = p - 2\,q$

$\{\lambda(u, v).u\,(u + 1) - v(v + 1)\}[2p+q, p-2q]$

Aplikativni izrazi

Aplikativni izrazi su konstruirani od osnovnih komponenata koje su za naše svrhe atomarne; njihova interna struktura nas ne zanima. One sadrže konstante i varijable koje se sastoje od jednog ili više znakova, uključujući znamenke. Ti atomi se nazivaju identifikatori (engl. *identifiers*.) Nema potrebe za preciziniju karakterizaciju identifikatora.

Lambda izraz je izraz koji se sastoji od dva dijela: vezane varijable, pisane između λ i točke, i lambda-tijela, pisanog nakon točke.

„Mnogo izraza“ se može formirati:

1. formirajući lambda izraz (tzv. funkcionalna apstrakcija koja se sastoji od vezanih varijabli i lambda tijela)

$$\{\lambda\, u\, .\, (u - 1)(u + 2)\}$$

2. formirajući operator/operand izraz (tzv. funkcionalna aplikacija)

$$/(a, +(\times(2,b), 3)).$$

3. formirajući listu izraza

$$(a,b,c)$$

(Pokazat će se da je 3. poseban slučaj 2.)

Aplikativni izrazi su, dakle, identifikatori, lambda-izrazi i operator/operand kombinacije.

Matematički izrazi se mogu prevesti u aplikativne izraze.

Prijevod u kojem intuitivno značenje matematičkog izraza odgovara vrijednosti aplikativnih izraza naziva se „semantički prihvatljivim.“

Definicije složenih informacijskih struktura

„Složene informacijske strukture“ se definiraju tako da se specificira koliko komponenti svaki član klase „složenih informacijskih struktura“ ili „konstruiranih objekata“ ima, i kakva vrsta informacije je prihvatljiva na kojem mjestu.

Definicija strukture specificira identifikatore koji se mogu koristiti za različite operacije nad konstruiranim objektima. Dijele se na

1. predikate
2. selektore
3. konstruktore

(Uobičajena terminologija u OOP. Što je sa mutatorima?)

Definicije funkcija

Dvije ekvivalentne definicije:

$$f(y) = y(y + 1)$$
$$\mathrm{f} = \lambda\, y.\, y\, (y + 1)$$

Izbjegavanje suvišnih zagrada

$$\{\lambda\, a\, .\, \lambda\, x\, .\, ax\, (a + x)\}[7 - 3]\ 3$$
$$fa + f3 + Dfb.$$

Problem: kako definirati tri važna konstrukta - liste, uvjetne izraze i rekurzivne definicije.

Liste

Lista je ili null-lista ili ima glavu(h) i rep(t), koji je lista.

Binarne i trinarne funkcije su funkcije koje su primjenjene na listu duljine 2 ili 3. Ako lista L ima dva elementa, onda je

$$\{\lambda\ (x, y, z).x + y + z\}[\text{constructlist}(3, L)].$$

prihvatljivo.

Uvjetni izrazi

if *uvjet* **then** *a* **else** *b*

if *uvjet* $(\lambda().a, \lambda().b)()$

Rekurzivne definicije

Rekurzivne ili cirkularne ili samopozivajuće definicije su definicije u kojima se definirani objekt nalazi i sa lijeve i sa desne strane jednakosti. Primjerice, jednadžba

$$L = (a, L, (b, c))$$

ima za rješenje „beskonačnu listu“ L

$$(a,(a,(a,(\ldots),(b,c)),(b,c)),(b,c)).$$

Operator Y

Koristeći lambda notaciju, možemo svaku cirularnu definiciju izraziti tako da se ima samo jedno cirkularno pojavljivanje.

$$x = \{\lambda\, x\, .\, (a, x, (b,c))\}\ L$$

Dakle, svaka cirkularna definicija može se prikazati u obliku

$$x = F\ x$$

Tj, F je funkcija i x je fiksna točka funkcije F. Definiramo sa Y funkciju koja danoj funkciji F pridružuje fiksnu točku.

$$x = Y\ F$$

Y se može koristiti i za rješavanje sustava rekurzivno definiranih funkcija

$$f(x)=F[f,g,x]$$
$$g(x)=G[f,g,x]$$

$$(f,g)=(\lambda(x).F[f,g,x],\lambda(x).G[f,g,x])$$

$$(f,g)=Y\ (\lambda(f,g).(\lambda(x)F[f,g,x],\ \lambda(x)G[f,g,x]))$$

Razlika između strukture i zapisane reprezentacije

U **Landinovom** sustavu postoji nekoliko načina na koji se može napisati isti aplikativni izraz. Svi ti načini su ekvivalentni, jer se na pitanja

Q1: Je li izraz identifikator? Ako da, koji?

Q2: Je li izraz lambda-izraz? Ako da, koje su vezane varijable, što je tijelo lambda izraza?

Q3: Je li izraz operator/operand kombinacija? Ako je, što je operator? Što su operandi?

odgovara na isti način.

Primjer je „where notacija“, gdje se

f(x) **where** x=3 **and** f=sin

piše kao

(λ(x,f).f(x))[x,sin]

Kako je korištenje „where- notacije“ „semantički prihvatljiva“ alternativa, **Landin** ju naziva „sintaktički šećer.“