Ezoteryczne Kartki
O programowaniu funkcyjnym
Rachunek λ

Jakub Bachurski

wersja 1.1.1

1 Wstęp

Rachunek lambda to alternatywny sposób stworzenia modelu obliczeń, bardziej opierający się na abstrakcji i prostym zestawie reguł, z których możemy utworzyć skomplikowane zachowania.

Są dwie główne zasady:

1. Istnieją lambdy, i możemy zapisać je w postaci:

   $$\lambda x.\, N$$

   Przy czym x to „parametr" tej lambdy, a N to pewne wyrażenie.

2. Możemy aplikować jedną lambdę do drugiej:

   $$(\lambda x.\, M)\, N$$

   Wtedy wszystkie wystąpienia x w M zastępujemy wyrażeniem N. Czasem nazywamy to zjawisko β-redukcją.

Przyjmujemy, że lambdy które różnią się nazewnictwem parametrów (zmiennych) są takie same. Czyli na przykład $\lambda a.\, N = \lambda b.\, N$. Nosi to nazwę α-konwersji.

Dla przykładu: lambda tożsamościowa to lambda która po aplikacji staje się wyrażeniem, które aplikowaliśmy. Ma ona postać $\lambda x.\, x$. Rozpisując:

$$(\lambda x.\, x)y = y$$

2 Zadania

Od teraz nie ma już nic. Są tylko lambdy. Nie ma ciągów, zbiorów, macierzy, liczb, ani logiki. Wszystko jest lambdą. Odtwórz te mechanizmy matematyczne za pomocą odpowiednich lambd.

2.1 Logika

Najważniejszą częścią logiki jest wartość logiczna: prawda **T** lub fałsz **F**. Lambdy mogą mieć tylko jeden parametr, więc aby wprowadzić dwojaką naturę logiki możemy stworzyć lambdę, która przyjmuje parametr i znowu zwróci lambdę, która przyjmie drugi parametr. W ten sposób możemy zwrócić jedną z dwóch rzeczy: ustalmy zatem, że **T** po aplikacji dwóch parametrów zwróci pierwszy z nich, a **F** zwróci drugi.

W ten sposób stworzyliśmy pomysł lambd wieloparametrowych. Aby uprościć zapis, ustalmy, że $\lambda a.\,(\lambda b.\,N) = \lambda ab.\,N$.

$$\mathbf{T} = \lambda ab.\,a$$

$$\mathbf{F} = \lambda ab.\,b$$

Spróbuj odnaleźć sposób na zapisanie podstawowych mechanizmów logiki:

- Negacja (NOT)
- Koniunkcja (AND)
- Alternatywa (OR)
- Implikacja
- Równoważność
- Warunkowość: lambda zwracająca pierwszy parametr, jeżeli warunek jest prawdziwy, lub drugi w przeciwnym wypadku.

2.2 Arytmetyka

Czas na liczby. Ograniczymy się do liczb naturalnych (wraz z zerem). Ustalmy, że liczba n jako lambda będzie swoistym algorytmem, który n-krotnie wykona aplikację do f poczynając od wartości x. Zatem:

$$0 = \lambda fx.\,x$$

$$1 = \lambda fx.\,f\,x$$

$$2 = \lambda fx.\,f\,(f\,x)$$

- Następnik (inkrementacja)
- Dodawanie
- Sprawdzanie, czy liczba jest zerem
- Mnożenie
- Poprzednik* (dekrementacja)
- Odejmowanie*

2.2.1 Porada

Aby ułatwić przechowywanie informacji w lambdach korzystamy z domknięć. Za pomocą domknięć możemy opracować krotki (w szczególności pary). Domknięcie jest to po prostu korzystanie z faktu, że po aplikacji parametru do lambdy wartość jest „zapamiętana". Aby po aplikacji elementów pary nie zaginęły nam one, może je „otoczyć" aplikacją lambdy będącej trzecim parametrem:

$$\{a, b\} = \lambda ab.\, (\lambda f.f\, a\, b)$$

W ten sposób możemy opisać lambdy do wyciągania pierwszego i drugiego elementu pary:

$$1^{st} = \lambda ab.a$$
$$2^{nd} = \lambda ab.b$$

Wtedy $\{a, b\}\, 1^{st} = a$, $\{a, b\}\, 2^{nd} = b$.

2.3 Listy

Na koniec listy, których zdolność zapamiętywania będzie korzystała z domknięć. Ustalmy, że lista $[x, y, z]$ będzie pewną niedokonaną operacją na elementach, poczynając od prawej do lewej. Dokładniej operacja będzie dana f, a początkowa wartość (którą otrzymamy dla pustej listy) to e:

$$[x, y, z] = \lambda fe.\, f\ x\ (f\ y\ (f\ z\ e))$$

Zatem lista pusta to $[] = \lambda fe.e$, a lista jednoelementowa to $[x] = \lambda fe.\, fxe$.

- Dodanie na początek listy (*cons*)
- Długość
- Suma elementów (jeżeli są liczbami)
- Ostatni element*
- Usuwanie ostatniego elementu*
- Dodawanie na koniec
- Odwracanie

3 Inne materiały

- `https://en.wikipedia.org/wiki/Lambda_calculus`
- `https://www.inf.fu-berlin.de/lehre/WS03/alpi/lambda.pdf`
- `https://crypto.stanford.edu/~blynn/lambda/` – interpreter rachunku lambda.

4 Przykładowe rozwiązania

4.1 Logika

Prawda	$\mathbf{T} = \lambda ab.\ a$
Fałsz	$\mathbf{F} = \lambda ab.\ b$
Negacja	$\neg = \lambda a.\ a\ \mathbf{F}\ \mathbf{T}$
Koniunkcja	$\wedge = \lambda ab.\ a\ b\ \mathbf{F}$
Alternatywa	$\vee = \lambda ab.\ a\ \mathbf{T}\ b$
Implikacja	$\Longrightarrow = \lambda ab.\ a\ b\ \mathbf{T}$
Równoważność	$\Longleftrightarrow = \lambda ab.\ a\ b\ (\neg b)$
Warunki	$\mathbf{?} = \lambda ctf.\ c\ t\ f$

4.2 Arytmetyka

Zero	$0 = \lambda fx.\ x$
Jeden	$1 = \lambda fx.\ f\ x$
Następnik	$\uparrow = \lambda nfx.\ f\ (n\ f\ x)$
Dodawanie	$+ = \lambda nmfx.\ n\ f\ (m\ f\ x)$
Test zero	$\Omega = \lambda n.\ n\ (\lambda x.\ \mathbf{F})\ \mathbf{T}$
Mnożenie	$\times = \lambda nmfx.\ n\ f\ (m\ f)\ x$
Poprzednik	$\downarrow = \lambda nfx.\ 1^{st}\ (n\ (\lambda p.\ p\ (\lambda ym.\ \{\mathbf{?}\ m\ y\ (f\ y),\ \mathbf{F}\}))\ \{x,\ \mathbf{T}\})$
Odejmowanie	$- = \lambda nm.\ m \downarrow n$

4.3 Listy

Lista pusta	$[] = \lambda ce.\ e$
Lista	$[x] = \lambda xce.\ (c\ x)\ e$
Doklej	$cons = \lambda xsce.\ c\ x\ (sce)$
Długość	$len = \ \lambda s.\ s\ (\lambda xr.\ \uparrow r)\ 0$
Suma	$sum = \ \lambda s.\ s\ (+)\ 0$
Ostatni	$back = \lambda s.\ 1^{st}\ (s\ (\lambda xr.\ \{\mathbf{?}\ (2^{nd}\ r)\ x\ (1^{st}\ r), \mathbf{F}\})\ \{0, \mathbf{F}\})$
Usuń ostatni	$popback = \lambda sce.\ 1^{st}\ (s\ (\lambda xr.\ \{\mathbf{?}\ (2^{nd}\ r)\ e\ (c\ x\ (1^{st}\ r)), \mathbf{F}\})\ \{e, \mathbf{T}\})$
Doklej koniec	$snoc = \lambda xsce.\ s\ c\ (c\ x\ e)$
Odwróć	$reverse = \lambda s.\ s\ (\lambda xr.\ snoc\ x\ r)\ []$