Bezpieczny rachunek λ

Bartłomiej Puget

Zespół Katedr i Zakładów Informatyki Matematycznej
Uniwersytetu Jagiellońskiego

5 grudnia 2018

Bezpieczeństwo

Pierwotny pomysłodawca: Werner Damm w 1982.

Wprowadzone przez Knapika, Niwińskiego i Urzyczyna na Conference on Foundations of Software Science and Computation Structures (FoSSaCS) 2002, na seminarium o algorytmach na nieskończonych drzewach generowanych przez gramatyki wyższego rzędu.

Typy w rachunku λ

Definicja

T jest typem wtw.

$$T ::= o \mid T \to T$$

gdzie o nazywamy typem bazowym.

Notacja

Definicja

$$A \to B \to C := A \to (B \to C)$$

Definicja

$$A^n \to B = \begin{cases} B & | \; n = 0 \\ A \to (A^{n-1} \to B) & | \; n > 0 \end{cases}$$

Definicja

$$(A_1, A_2, \ldots, A_n) := A_1 \to A_2 \to \ldots \to A_n$$

gdzie A_n niekoniecznie jest typem bazowym.

Notacja

Definicja

$$\begin{aligned} 0 &\equiv o \\ (k+1) &\equiv k \to o \end{aligned}$$

Przykład

$$\begin{aligned} 0 &\equiv o \\ 1 &\equiv o \to o \\ 2 &\equiv (o \to o) \to o \\ 3 &\equiv ((o \to o) \to o) \to o \\ &\vdots \end{aligned}$$

Rząd typu

Definicja

$ord(T)$ jest rzędem typu T wtw.

$$ord(T) = \begin{cases} 0 & | \ T = o \\ \max\{ord(A) + 1, ord(B)\} & | \ T = A \to B \end{cases}$$

Drzewo typu

Definicja

$tree(T)$ jest drzewem typu T wtw.

$$tree(T) := \begin{cases} o & | \ T = o \\ \begin{array}{c} tree(B) \\ / \\ tree(A) \end{array} & | \ T = A \to B \end{cases}$$

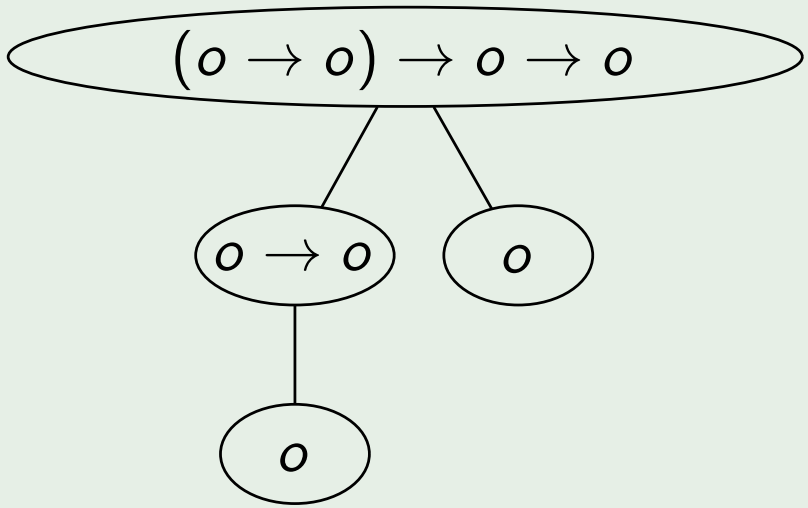
Spostrzeżenie

$ord(T) = height(tree(T))$

Drzewo typu

Przykład

$$tree((o \to o) \to (o \to o)) =$$



Typ homogeniczny

Definicja

Typ $T = A_1 \to A_2 \to \ldots \to A_n \to o$ jest homogeniczny wtw. $ord(A_1) \geq ord(A_2) \geq \ldots \geq ord(A_n)$

Podstawienie capture-permitting

Definicja

$M\{N/x\}$ jest podstawieniem capture-permitting N pod x w M wtw.

$$\begin{array}{rcll} x\{N/x\} & \equiv & N & \\ y\{N/x\} & \equiv & y & \mid x \not\equiv y \\ (M_1 M_2)\{N/x\} & \equiv & (M_1\{N/x\})(M_2\{N/x\}) & \\ (\lambda x.M)\{N/x\} & \equiv & \lambda x.M & \\ (\lambda y.M)\{N/x\} & \equiv & \lambda y.M\{N/x\} & \mid x \not\equiv y \end{array}$$

Przykład

$$\begin{array}{rcl} \lambda y.x\{y/x\} & \equiv & \lambda y.y \\ \lambda z.x\{y/x\} & \equiv & \lambda z.y \end{array}$$

Podstawienie

Definicja

$M[N/x]$ jest podstawieniem N pod x w M wtw.

$$\begin{array}{rcll} x[t/x] & \equiv & t & \\ y[t/x] & \equiv & y & \mid x \not\equiv y \\ (M_1 M_2)[t/x] & \equiv & (M_1[t/x])(M_2[t/x]) & \\ (\lambda x.M)[t/x] & \equiv & \lambda x.M & \\ (\lambda y.M)[t/x] & \equiv & \lambda y.M[z/y][t/x] & \mid x \not\equiv y \wedge z : \text{nowa zmienna} \end{array}$$

Przykład

$$\begin{array}{rcl} \lambda y.x[y/x] & \equiv & \lambda z'.y \\ \lambda z.x[y/x] & \equiv & \lambda z'.y \end{array}$$

Bezpieczna gramatyka

Definicja

Gramatyka wyższego rzędu G o homogenicznych typach niterminali jest niebezpieczna wtw. istnieje produkcja $Fz_1z_2 \ldots z_n \rightarrow e$, gdzie e zawiera podterm, który:

- pojawia się w roli argumentu;
- zawiera parametr typu o niższym rzędzie, niż on sam.

Bezpieczna gramatyka

Definicja (równoważna)

Zakładając homogeniczność typów:

- term t rzędu k jest niebezpieczny wtw. zawiera parametr rzędu $< k$;
- wystąpienie niebezpiecznego termu t w termie t' jest bezpieczne wtw. t pojawia się w kontekście $\ldots(ts)\ldots$;
- gramatyka jest bezpieczna wtw. żaden niebezpieczny term nie pojawia się w sposób niebezpieczny po prawej stronie żadnej produkcji.

Bezpieczna gramatyka

Przykład

$$\begin{aligned} f &: (o,o,o) \\ g,h &: (o,o) \\ a,b &: o \end{aligned}$$

$$\begin{aligned} S^o &\to Fgab \\ F^{((o,o),o,o,o)}\phi^{(o,o)}x^o y^o &\to f(F(F\phi x)y(hy))(f(\phi x)y) \end{aligned}$$

$$\begin{aligned} S^o &\to G(ga)b \\ G^{(o,o,o)}z^o y^o &\to f(G(Gzy)(hy))(fzy) \end{aligned}$$

Bezpieczna gramatyka

Przykład

$$\begin{aligned} f &: (o,o,o) \\ g,h &: (o,o) \\ a,b &: o \end{aligned}$$

$$\begin{aligned} S^o &\to Fgab \\ F^{((o,o),o,o,o)}\phi^{(o,o)}x^o y^o &\to f(F(\underline{F\phi x})y(hy))(f(\phi x)y) \end{aligned}$$

$$\begin{aligned} S^o &\to G(ga)b \\ G^{(o,o,o)}z^o y^o &\to f(G(Gzy)(hy))(fzy) \end{aligned}$$

Bezpieczny rachunek λ

Definicja

System w rachunku λ jest bezpieczny wtw. zmienne wolne pojawiające się w termie mają rząd niemniejszy, niż sam term.

Przykład (termy Kiersteada)

$$\begin{aligned} M_1 &= \lambda f^{((o,o),o)}.f(\lambda x^o.f(\lambda y^o.y)) \\ M_2 &= \lambda f^{((o,o),o)}.f(\lambda x^o.f(\lambda y^o.x)) \end{aligned}$$

Bezpieczny rachunek λ

Definicja

System w rachunku λ jest bezpieczny wtw. zmienne wolne pojawiające się w termie mają rząd niemniejszy, niż sam term.

Przykład (termy Kiersteada)

$$\begin{aligned} M_1 &= \lambda f^{((o,o),o)}.f(\lambda x^o.f(\lambda y^o.y)) \\ M_2 &= \lambda f^{((o,o),o)}.f(\lambda x^o.f(\underline{\lambda y^o.x})) \end{aligned}$$

Bezpieczny rachunek λ

Definicja

$$(\text{var})\ \frac{}{x : A \vdash_s x : A} \qquad (\text{const})\ \frac{}{\vdash_s f : A}\ f \in \Xi \qquad (\text{wk})\ \frac{\Gamma \vdash_s M : A}{\Delta \vdash_s M : A}\ \Gamma \subset \Delta$$

$$(\text{app}_{as})\ \frac{\Gamma \vdash_{asa} M : A \to B \quad \Gamma \vdash_s N : A}{\Gamma \vdash_{asa} M\,N : B} \qquad (\delta)\ \frac{\Gamma \vdash_s M : A}{\Gamma \vdash_{asa} M : A}$$

$$(\text{app})\ \frac{\Gamma \vdash_{asa} M : A \to B \quad \Gamma \vdash_s N : A}{\Gamma \vdash_s M\,N : B}\ ord\,B \leq ord\,\Gamma$$

$$(\text{abs})\ \frac{\Gamma, x_1 : A_1, \ldots, x_n : A_n \vdash_{asa} M : B}{\Gamma \vdash_s \lambda x_1^{A_1} \ldots x_n^{A_n}.M : (A_1, \ldots, A_n, B)}\ ord(A_1, \ldots, A_n, B) \leq ord\,\Gamma$$

Bezpieczny rachunek λ

Przykład

$$M_1 = \lambda f^2.f(\lambda x^0.f(\lambda y^0.y))$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{f:2 \vdash_{\mathbf{s}} f:2}(\text{var})}{f:2 \vdash_{\mathbf{asa}} f:2}(\delta)}{f:2, x:0 \vdash_{\mathbf{s}} f:2}(\text{wk})}{f:2, x:0 \vdash_{\mathbf{asa}} f:2}(\delta) \quad \dfrac{\dfrac{\dfrac{\dfrac{}{y:0 \vdash_{\mathbf{s}} y:0}(\text{var})}{y:0 \vdash_{\mathbf{asa}} y:0}(\delta)}{y:0 \vdash_{\mathbf{s}} \lambda y^{\mathbf{0}}.y:1}(\text{abs})}{f:2, x:0 \vdash_{\mathbf{s}} \lambda y^{\mathbf{0}}.y:1}(\text{wk})}{f:2, x:0 \vdash_{\mathbf{s}} f(\lambda y^{\mathbf{0}}.y):0}(\text{app}) \quad \dfrac{f:2, x:0 \vdash_{\mathbf{s}} f(\lambda y^{\mathbf{0}}.y):0}{f:2, x:0 \vdash_{\mathbf{asa}} f(\lambda y^{\mathbf{0}}.y):0}(\delta)}{f:2 \vdash_{\mathbf{s}} \lambda x^{\mathbf{0}}.f(\lambda y^{\mathbf{0}}.y):1}(\text{abs})}{f:2 \vdash_{\mathbf{s}} f(\lambda x^{\mathbf{0}}.f(\lambda y^{\mathbf{0}}:y)):0}(\text{app}) \quad \dfrac{f:2 \vdash_{\mathbf{s}} f(\lambda x^{\mathbf{0}}.f(\lambda y^{\mathbf{0}}:y)):0}{f:2 \vdash_{\mathbf{asa}} f(\lambda x^{\mathbf{0}}.f(\lambda y^{\mathbf{0}}:y)):0}(\delta)}{\vdash_{\mathbf{s}} \lambda f^{\mathbf{2}}.f(\lambda x^{\mathbf{0}}.f(\lambda y^{\mathbf{0}}.y)):3}(\text{abs})$$

Prawie-bezpieczeństwo

Definicja

Term t jest prawie bezpieczny wtw. t jest prawie bezpieczną aplikacją lub termem w formie $\lambda x_1^{A_1} x_2^{A_2} \dots x_n^{A_n}.M$ gdzie $n \geq 1$ oraz M jest prawie bezpieczną aplikacją.

Spostrzeżenie

Termy prawie bezpieczne nie są koniecznie bezpieczne, ale mogą się takie stać. Czy to przez abstrakcję zmiennych, czy przez aplikację.

Bezpieczny rachunek λ

Lemat (o nieprzechwytywaniu zmiennych)

Jeśli $\Gamma, x : B \vdash_s M : A$ *oraz* $\Gamma \vdash_s N : B$, *nie zajdzie przechwytywanie zmiennych wykonując capture-permitting podstawienie* $M[N/x]$.

β-redukcja

Przykład

$$\begin{aligned} w, z &: o \\ f &: (o, o, o) \end{aligned}$$

$$(\lambda x^o y^o . fxy) zw \rightarrow_\beta (\underline{\lambda y^o . fzy}) w$$

β-redukcja

Przykład

$$\begin{aligned} w, z &: o \\ f &: (o, o, o) \end{aligned}$$

$$(\lambda x^o y^o . fxy)zw \rightarrow_\beta (\underline{\lambda y^o . fzy})w \rightarrow_\beta fzw$$

β_s-redukcja

Definicja

Redeks jest bezpieczny wtw. jest prawie bezpieczną aplikacją w formie:

$$(\lambda x_1^{A_1} x_2^{A_2} \dots x_n^{A_n}.M) N_1 N_2 \dots N_l$$

gdzie $l, n \geq 1$ i M jest prawie bezpieczną aplikacją.

Definicja

$$\begin{aligned} \beta_s \quad &= \quad \{ (\lambda x_1^{A_1} \dots x_n^{A_n}.M) N_1 \dots N_l \mapsto \lambda x_{l+1}^{A_{l+1}} \dots x_n^{A_n}.M \left[\overline{N}/x_1 \dots x_l\right] \mid n > l \} \\ &\cup \quad \{ (\lambda x_1^{A_1} \dots x_n^{A_n}.M) N_1 \dots N_l \mapsto M \left[N_1 \dots N_n/\overline{x}\right] N_{n+1} \dots N_l \mid n \leq l \} \ . \end{aligned}$$

β_s-redukcja

Spostrzeżenie

Dla $l < n$, bezpieczny redeks ma natępujące drzewo wywodu:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dots}{\Gamma', \overline{x} : \overline{A} \vdash_{\mathbf{s}} M : (A_{n+1}, \dots, A_l, B)}\,(\mathrm{abs})}{\Gamma' \vdash_{\mathbf{s}} \lambda x_1^{A_1} \dots x_n^{A_n}.M : (A_1, \dots, A_l, B)}\,(\mathrm{wk})}{\Gamma \vdash_{\mathbf{s}} \lambda x_1^{A_1} \dots x_n^{A_n}.M : (A_1, \dots, A_l, B)}\,(\delta) \qquad \dfrac{\dots}{\Gamma \vdash_{\mathbf{s}} N_1 : A_1}}{\Gamma \vdash_{\mathbf{as}} \lambda x_1^{A_1} \dots x_n^{A_n}.M : (A_1, \dots, A_l, B)}\,(\mathrm{app_{as}})}{\Gamma \vdash_{\mathbf{as}} (\lambda x_1^{A_1} \dots x_n^{A_n}.M) N_1 : (A_2, \dots A_l, B)}$$

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash_{\mathbf{as}} (\lambda x_1^{A_1} \dots x_n^{A_n}.M) N_1 \dots N_{l-1} : (A_l, B)}\,(\mathrm{app_{as}}) \qquad \dfrac{\dots}{\Gamma \vdash_{\mathbf{s}} N_l : A_l}}{\Gamma \vdash_{\mathbf{s}} (\lambda x_1^{A_1} \dots x_n^{A_n}.M) N_1 \dots N_l : B}\,(\mathrm{app})$$

β_s-redukcja

Lemat (Podstawianie zachowuje bezpieczeństwo)

Niech $\Gamma \vdash_s N : A$. *Wtedy:*

1. $\Gamma, x : A \vdash_s M : B \implies \Gamma \vdash_s M[N/x] : B$
2. $\Gamma, x : A \vdash_{asa} M : B \implies \Gamma \vdash_{asa} M[N/x] : B$

β_s-redukcja

Lemat (β_s-redukcja zachowuje bezpieczeństwo)

Niech $M_1 \beta_s M_2$. Wtedy:

- *M_2 jest prawie bezpieczne;*
- *jeśli M_1 jest bezpieczne, to M_2 również.*

β_s-redukcja

Definicja

Bezpieczna β-redukcja ($\rightarrow_{\beta_s}$) jest najmniejszą relacją, że jeśli $M_1 \beta_s M_2$ oraz $C[M]$ jest bezpiecznym termem w kontekście $C[-]$, to $C[M_1] \rightarrow_{\beta_s} C[M_2]$.

Lemat (β_s-redukcja zachowuje bezpieczeństwo)

Jeśli $\Gamma \vdash_s M_1 : A$ *oraz* $M_1 \rightarrow_{\beta_s} M_2$, *to* $\Gamma \vdash_s M_2 : A$

η-długa forma normalna

Definicja

η-długa forma normalna termu $M : (A_1, \ldots, A_n, o)$, $\lceil M \rceil$ jest zdefiniowana następująco:

$$\begin{aligned}
\lceil x \rceil &\equiv \lambda.x \\
\lceil \lambda x^\tau.N \rceil &\equiv \lambda x^\tau.\lceil N \rceil \\
\lceil xN_1 \ldots N_m \rceil &\equiv \lambda \overline{\varphi}^{\overline{A}}.x\lceil N_1 \rceil \ldots \lceil N_m \rceil \lceil \varphi_1 \rceil \ldots \lceil \varphi_n \rceil \\
\lceil (\lambda x^\tau.N)N_1 \ldots N_p \rceil &\equiv \lambda \overline{\varphi}^{\overline{A}}.(\lambda x^\tau.\lceil N \rceil)\lceil N_1 \rceil \ldots \lceil N_m \rceil \lceil \varphi_1 \rceil \ldots \lceil \varphi_n \rceil
\end{aligned}$$

gdzie $m \geq 0$, $p \geq 1$, $\overline{\varphi} = \varphi_1 \ldots \varphi_n$, gdzie wszystkie φ_i są nowymi zmiennymi.

Przykład

$$\begin{aligned}
\lceil x^o \rceil &\equiv \lambda.x \\
\lceil \lambda f^{(o,o,o)}.fx \rceil &\equiv \lambda f^{(o,o,o)}\varphi_1^o.fx\varphi_1 \\
\lceil (\lambda f^{(o,o,o)}x^o y^o.fxy)sa \rceil &\equiv \lambda \varphi_1^o(\lambda f^{(o,o,o)}x^o y^o.fxy)sa\varphi_1
\end{aligned}$$

η-długa forma normalna

Lemat

Term jest bezpieczny wtw. jego η-długa forma normalna jest bezpieczna.

Numerale Churcha

Definicja

Każda liczba $n \in \mathbb{N}$ może być zakodowana jako:

$$\overline{n}^{I} := \lambda s^{(o,o)} z^{o}.s^{n} z$$

gdzie $I = ((o, o), o, o)$

Definicja

p-arna funkcja $f : \mathbb{N}^p \to \mathbb{N}$ dla $p \geq 0$ jest reprezentowana przez term $F : (I^p, I)$ wtw. $\forall m_i \in \mathbb{N}$:

$$F\overline{m_1} \ldots \overline{m_p} =_\beta \overline{f(m_1, \ldots, m_p)}$$

Numerale Churcha

Twierdzenie (Schwichtenberg, 1976)

Funkcje numeryczne wyrażalne za pomocą numerali Churcha w prosto-typowalnym rachunku λ to dokładnie wielomiany wielu zmiennych rozszerzone o funkcję warunkową.

Twierdzenie

Funkcje numeryczne wyrażalne za pomocą numerali Churcha w bezpiecznym rachunku λ to dokładnie wielomiany wielu zmiennych.

Przykład

Term $\lambda FGH\alpha x.F(\underline{\lambda y.G\alpha x})(H\alpha x)$, zaproponowany przez Schwichtenberga, nie jest bezpieczny.

Funkcje na słowach

Definicja (alfabet binarny)

$\Sigma = \{a, b\}$

Definicja (numeral)

$\mathbf{k} := a \dots a \mid |\mathbf{k}| = k$

Definicja (c)

$c(n, k) : (\Sigma^*)^n \to \Sigma^* := \mathbf{k}$

Definicja (katenacja)

$app(x, y) : (\Sigma^*)^2 \to \Sigma^* :=$ katenacja x oraz y.

Funkcje na słowach

Definicja (podmiana)

$sub(x, y, z) : (\Sigma^*)^3 \to \Sigma^* :=$ słowo powstałe przez podmianę w słowie x wszystkich wystąpień a przez y oraz wystąpień b przez z. Formalnie:

$$\begin{aligned} sub(\epsilon, y, z) &= \epsilon \\ sub(ax, y, z) &= app(y, sub(x, y, z)) \\ sub(bx, y, z) &= app(z, sub(x, y, z)) \end{aligned}$$

Przykład

$$\begin{aligned} sub(abab, a, b) &= abab \\ sub(abab, b, a) &= baba \\ sub(abab, aba, a) &= abaaabaa \end{aligned}$$

Funkcje na słowach

Definicja (ucięcie prefiksu)

$cut_a(x) : \Sigma^* \to \Sigma^* :=$ maksymalny prefiks x zawierający same a.
Formalnie:

$$\begin{aligned} cut_a(\epsilon) &= \epsilon \\ cut_a(ax) &= app(a, cut_a(x)) \\ cut_a(bx) &= \epsilon \end{aligned}$$

Przykład

$$\begin{aligned} cut_a(\epsilon) &= \epsilon \\ cut_a(baaaaaa) &= \epsilon \\ cut_a(aaabaabbbbab) &= aaa \end{aligned}$$

Funkcje na słowach

Definicja (rzutowanie)

$\pi_k(x_1, \ldots, x_k, \ldots x_n) : (\Sigma^*)^n \to \Sigma^* = x_k$, gdzie $1 \leq k \leq n$

Definicja (funkcja stała)

$cst_w(x) : \Sigma^* \to \Sigma^* = w$

Funkcje na słowach

Definicja (niepustość)

$$\overline{sq}(x) : \Sigma^* \to \Sigma^* = \begin{cases} \mathbf{0} & | \; x = \epsilon \\ \mathbf{1} & | \; x \neq \epsilon \end{cases} = cut_a(app(sub(x, b, b), a)$$

Definicja (pustość)

$$sq(x) : \Sigma^* \to \Sigma^* = \overline{sq}(\overline{sq}(x))$$

Definicja (występowanie)

$$occ_l(x) : \Sigma^* \to \Sigma^* = \begin{cases} \mathbf{1} & | \; l \in x \\ \mathbf{0} & | \; l \notin x \end{cases} = sq(sub(x, l, \epsilon))$$

Funkcje na słowach

Definicja (typ słowa binarnego)

$$\mathsf{B} = (o \to o) \to (o \to o) \to o \to o$$

Przykład

$$\begin{aligned} \underline{\epsilon} &= \lambda u^{o\to o} v^{o\to o} x^{o}.x \\ \underline{a} &\equiv \lambda u^{o\to o} v^{o\to o} x^{o}.ux \\ \underline{abaa} &\equiv \lambda u^{o\to o} v^{o\to o} x^{o}.u(v(u(ux))) \end{aligned}$$

Funkcje na słowach

Twierdzenie (Zaionc, 1987)

Zbiór słów na słowach definiowalnych w rachunku λ to minimalny zbiór zawierający

1. *funkcje stałe cst_w*
2. *rzutowania π_k*
3. *katenację app*
4. *ucięcie prefiksu cut_a*

domknięty na złożenia.

Funkcje na słowach

Przykład

$$
\begin{aligned}
\pi_k &\equiv \lambda x_1 \dots x_n.x_i \\
cst_w &\equiv \lambda x_1 \dots x_n.\underline{w} \\
app &\equiv \lambda cduvx.cuv(duvx) \\
sub &\equiv \lambda xdeuvx.c(\lambda y.duvy)(\lambda y.euvy)x \\
cut_a &\equiv \lambda cuvx.cu(\lambda y.x)x \\
cut_b &\equiv \lambda cuvx.c(\lambda y.x)vx \\
sq &\equiv \lambda cuvx.c(\lambda y.ux)(\lambda y.ux)x \\
\overline{sq} &\equiv \lambda cuvx.c(\lambda y.x)(\lambda y.x)(ux) \\
occ_a &\equiv \lambda cuvx.c(\lambda y.ux)(\lambda y.y)x \\
occ_b &\equiv \lambda cuvx.c(\lambda y.y)(\lambda y.ux)x
\end{aligned}
$$

Funkcje na słowach

Twierdzenie

Funkcje na słowach definiowalne w bezpiecznym rachunku λ to mimimalny zadany przez:

1. *funkcje stałe cst_w*
2. *rzutowania π_k*
3. *katenację app*
4. *podmianę sub*

domknięty na złożenia.

Języki na słowach

Twierdzenie (Damm, Goerdt)

Języki generowane przez bezpieczne gramatyki rzędu n odpowiadają kolejnym klasom automatów z zagnieżdżonym stosem, tj. językom regularnym, bezkontekstowym, indeksowanym, . . .

Drzewa

Twierdzenie (Knapik z ekipą)

Twierdzenia monadycznej logiki drugiego rzędu (MSO) na drzewach generowanych przez bezpieczne deterministyczne gramatyki są rozstrzygalne.

Grafy

Twierdzenie (Caucal)

Twierdzenia MSO są rozstrzygalne dla grafów generowanych przez bezpieczne gramatyki dowolnego skończonego rzędu.

Twierdzenie (Hague)

Twierdzenia MSO dla grafów generowanych przez niebezpieczne gramatyki jest nierozstrzygalne, a twierdzenia μ-rachunku są n-EXPTIME zupełne.

Bibliografia

 William Blum, *The Safe Lambda Calculus*, praca doktorska, Oxford University Computing Laboratory, Michaelmas 2008

 William Blum and C.-H. Luke Ong, *The Safe Lambda Calculus*, Logical Methods in Computer Science, Vol. 5 (1:3) 2009, str. 1–38