Cálculos de Substituições Explícitas e Aplicações Computacionais

Mauricio Ayala-Rincón
Departamento de Matemática
Universidade de Brasília

Brasília D. F., Brasil

V Encontro de Matemática e Estatística IME-UFG Goiânia — Junho 6, 2001

Plano da Apresentação

1. Lambda Cálculo Tipado

- λ-Cálculo [1932/1933 Church].
- λ-Cálculo Tipado [1940 Church]
- Funções Recursivas são λ-representáveis [Kleene, Rosser 1935]
- Exatamente as funções Turing Computáveis são λ-representáveis [Turing 1937]
- Programação Funcional $\left\{ \begin{array}{c} \text{Representação de} \\ \text{funções computáveis} \\ \text{como } \lambda\text{-termos} \end{array} \right.$

1. Lambda Cálculo Tipado

- Operações do λ-calculus $\left\{\begin{array}{ll} \text{— Aplicação:} & (M \quad N) \\ \text{— Abstração:} & \lambda_x.M \end{array}\right.$
- Regra (de *reescrita*) básica do λ-calculus:

$$(\beta\text{-redução}) \qquad (\lambda_x.M \quad N) \xrightarrow{\beta} M\{N/x\}$$

λ-calculus: teoria *livre de tipos*

$I ::= (\lambda_x.(x \quad x) \quad \lambda_x.(x \quad x)) \quad \Longrightarrow \quad (I \quad I) \xrightarrow{\beta} I$

$(\lambda_x.(x \quad x) \quad \lambda_x.(x \quad x)) \xrightarrow{\beta} (\lambda_x.(x \quad x) \quad \lambda_x.(x \quad x)) \xrightarrow{\beta} \ldots$

— Tipagem importante para verificação parcial de correção

— Disciplina de tipos ajuda a *programar* sem erros

1. Lambda Cálculo Tipado

Teoria de Tipos — λ-calculus tipado $\begin{cases} \text{[Curry 1934] Tipagem implícito} \\ \text{[Chuch 1940] Tipagem explícito} \end{cases}$

$I \ : \ A \to A \qquad\qquad \lambda_{x:A}.x \ : \ A \to A$

No λ-calculus tipado $(I \quad I)$ não faz sentido!

- ML [Milner 1984]. Linguagem tipada implicitamente: tipos têm que ser descobertos
- Algol, Pascal, Delfi, C, Java, etc. Linguagens tipadas explicitamente: Tipos fazem parte dos programas

1. Lambda Cálculo Tipado

Computação de funções numéricas — *Numerais de Church*

$$C_n ::= \lambda_f.\lambda_x.f^n x, \quad n \geq 0$$

Proposição[Rosser] Defina $\begin{cases} A_+ ::= \lambda_{xypq}.xp(ypq); \\ A_* ::= \lambda_{xyz}.x(yz); \\ A_{exp} ::= \lambda_{xy}.yx \end{cases}$

Então $\begin{cases} 1.\ A_+ C_n C_m = C_{n+m}, \forall n, m \geq 0 \\ 2.\ A_* C_n C_m = C_{nm}, \forall n, m \geq 0 \\ 3.\ A_{exp} C_n C_m = C_{n^m}, \forall n \geq 0, m > 0 \end{cases}$

1. Lambda Cálculo Tipado

Exemplo:

$$
\begin{array}{lcl}
(\lambda_{x\,ypq}.xp(ypq))\lambda_{xy}.(x^n\ \ y)\lambda_{xy}.(x^m\ \ y) & \xrightarrow{\beta} & \\
(\lambda_{ypq}.(\lambda_{xy}.(x^n\ \ y))p(ypq))\lambda_{xy}.(x^m\ \ y) & \xrightarrow{\beta} & \\
(\lambda_{ypq}.(\lambda_{y}.(p^n\ \ y))(ypq))\lambda_{xy}.(x^m\ \ y) & \xrightarrow{\beta} & \\
(\lambda_{ypq}.(p^n\ \ (ypq)))\lambda_{xy}.(x^m\ \ y) & \xrightarrow{\beta} & \\
\lambda_{pq}.(p^n\ \ ((\lambda_{xy}.(x^m\ \ y))pq)) & \xrightarrow{\beta} & \\
\lambda_{pq}.(p^n\ \ ((\lambda_{y}.(p^m\ \ y))q)) & \xrightarrow{\beta} & \\
\lambda_{pq}.(p^n\ \ (p^m\ \ q)) & \equiv & C_{n+m}.
\end{array}
$$

1. Lambda Cálculo Tipado

Teorema As funções recursivas são λ-definíveis

Funções iniciais: $\left\{ \begin{array}{l} U_p^i ::= \lambda_{x_1 \ldots x_p}.x_i \\ S ::= \lambda_{xyz}.y(xyz)(\equiv A_+ C_1) \\ Z ::= \lambda_x.C_0 \end{array} \right.$

$\left. \begin{array}{l} \textit{Composição} \\ \textit{Recurrência Primitiva} \\ \textit{Minimização} \end{array} \right\}$ funções λ-definíveis são fechadas

1. Lambda Cálculo Tipado

- Regra (de *reescrita*) adicional do λ-calculus:

$$(\eta\text{-redução}) \qquad \lambda_x.(M \;\; x) \xrightarrow{\eta} M, \text{ se } x \text{ não ocorre livre em } M$$

Teorema O λ-calculus satisfaz a propriedade de Church-Rosser:
$M =_{\beta\eta} N$ implica $\exists L$ tal que $M \xrightarrow{\beta\eta^*} L$ e $N \xrightarrow{\beta\eta^*} L$

1. Lambda Cálculo Tipado

Tipos.

$$(\lambda_{x:A}.M \;\; N) \xrightarrow{\beta} M\{N/x\} \qquad \text{e} \qquad \lambda_{x:A}.(M \;\; x) \xrightarrow{\eta} M, \text{ se } x \notin \mathcal{FV}(M)$$

Regras de tipagem $\longrightarrow$ disciplina de tipos

$$\frac{x \notin \Gamma}{x{:}A, \Gamma \vdash x : A}\ (\text{Start}) \qquad \frac{x \notin \Gamma \qquad \Gamma \vdash M : B}{x{:}A, \Gamma \vdash M : B}\ (\text{Weak})$$

$$\frac{x{:}A, \Gamma \vdash M : B}{\Gamma \vdash \lambda_{x:A}.M : A \to B}\ (\text{Abs}) \qquad \frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M \;\; N) : B}\ (\text{Appl})$$

1. Lambda Cálculo Tipado

Exemplo

$$(1) \qquad \frac{}{y{:}A{\to}B, x{:}A \vdash y : A{\to}B}\ \text{(Start)}$$

$$(2) \qquad \frac{}{y{:}A{\to}B, x{:}A \vdash x : A}\ \text{(Start)}$$

$$(3) \qquad \frac{(1) \qquad (2)}{y{:}A{\to}B, x{:}A \vdash (y\ \ x) : B}\ \text{(Appl)}$$

$$(4) \qquad \frac{(3)}{x{:}A \vdash \lambda_{y:A\to B}.(y\ \ x) : (A{\to}B){\to}B}\ \text{(Abs)}$$

$$(5) \qquad \frac{(4)}{\vdash \lambda_{x:A}.\lambda_{y:A\to B}.(y\ \ x) : A{\to}((A{\to}B){\to}B)}\ \text{(Abs)}$$

1. Lambda Cálculo Tipado

Hisomorfismo de Curry-Howard:

Teoria de Tipos		**Lógica Intuicionista minimal**
Tipos	$\longleftrightarrow$	Proposições
$\rightarrow$		$\Rightarrow$
Regras de tipagem		Regras dedutivas

Regras dedutivas da lógica intuicionista minimal:

$$\frac{}{\Omega, A \vdash_I A}\ (\text{Axiom}) \qquad \frac{\Omega, A \vdash_I B}{\Omega \vdash_I A \Rightarrow B}\ (\text{Intro}) \qquad \frac{\Omega \vdash_I A \Rightarrow B \qquad \Omega \vdash_I A}{\Omega \vdash_I B}\ (\text{Elim})$$

1. Lambda Cálculo Tipado

Exemplo

$$(1) \qquad \frac{}{A, A \Rightarrow B \vdash_I A \Rightarrow B}\ (\text{Axiom})$$

$$(2) \qquad \frac{}{A, A \Rightarrow B \vdash_I A}\ (\text{Axiom})$$

$$(3) \qquad \frac{(1) \qquad (2)}{A, A \Rightarrow B \vdash_I B}\ (\text{Elim})$$

$$(4) \qquad \frac{(3)}{A \vdash_I (A \Rightarrow B) \Rightarrow B}\ (\text{Intro})$$

$$(5) \qquad \frac{(4)}{\vdash_I A \Rightarrow ((A \Rightarrow B) \Rightarrow B)}\ (\text{Intro})$$

2. Lambda Cálculo sem nomes — Notação de *de Bruijn*

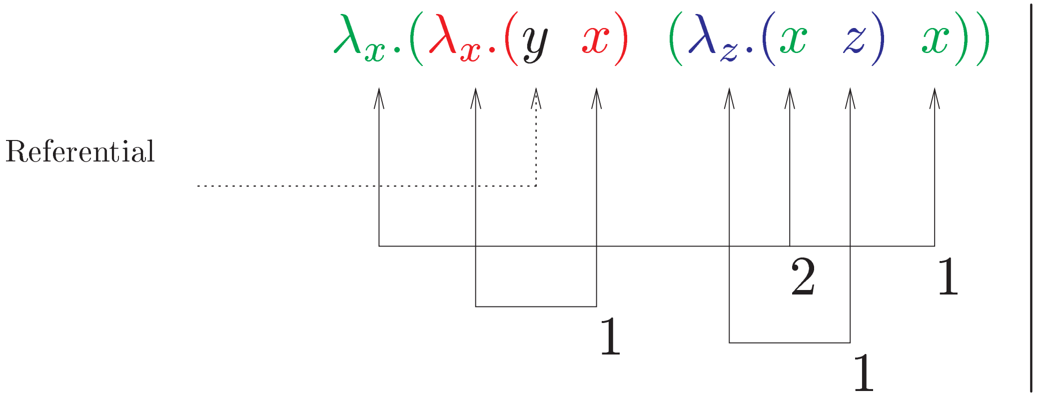
- "*Implicitude*" da substituição é o "Talão de Achilles" do λ-calculus:
- β-redução dada via renomeamento de variáveis implícito/informal

$$(\beta\text{-redução}) \quad (\lambda_x.M \;\; N) \xrightarrow{\beta} M\{N/x\}$$

- $\boxed{\text{Substituição implícita não fornece nenhum mecanismo formal para a análise de propriedades computacionais essenciais}}$ como $\left\{ \begin{array}{l} \text{— complexidade tempo e} \\ \text{— espaço} \end{array} \right.$

2. Lambda Cálculo sem nomes — Notação de *de Bruijn*

- Termos em notação de de Bruijn, $\Lambda_{dB}(\mathcal{X})$: $a ::= \mathbb{N} \mid \mathcal{X} \mid (a \;\; a) \mid \lambda.a$, onde $\mathcal{X}$ meta-variáveis e $\mathbb{N}$ conjunto de indíces de de Bruijn.



$$\rightarrow_\beta \;\; \lambda_x.(y \;\; (\lambda_z.(x \;\; z) \;\; x))$$

$$\rightarrow_\beta \;\; \lambda_x.(y \;\; (x \;\; x))$$

P. ex., para o referencial $x, y, z, \ldots$: $\boxed{\lambda.(\lambda.(\underline{4} \;\; \underline{1}) \;\; (\lambda.(\underline{2} \;\; \underline{1}) \;\; \underline{1}))}$

2. Lambda Cálculo sem nomes — Notação de *de Bruijn*

$$\lambda.(\lambda.(\underline{4}\ \ \underline{1})\ \ (\lambda.(\underline{2}\ \ \underline{1})\ \ \underline{1})) \xrightarrow{\beta} \lambda.(\underline{3}\ \ (\lambda.(\underline{2}\ \ \underline{1})\ \ \underline{1})) \xrightarrow{\beta} \lambda.(\underline{3}\ \ (\underline{1}\ \ \underline{1}))$$

β-redução

$$(\lambda.M\ \ N) \xrightarrow{\beta} M\{N/\underline{1}\}$$

η-redução

$$\lambda.(M\ \ \underline{1}) \xrightarrow{\eta} N, \text{ se } N^{+} = M$$

2. Lambda Cálculo sem nomes — Notação de *de Bruijn*

Regras de tipagem:

$$\frac{1 \leq i \leq n}{A_1.A_2.\ldots A_n \vdash \underline{i} : A_i}\ \text{(Var)}$$

$$\frac{A.\Gamma \vdash M : B}{\Gamma \vdash \lambda_A.M : A{\rightarrow}B}\ \text{(Abs)}$$

$$\frac{\Gamma \vdash M : A{\rightarrow}B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M\ \ N) : B}\ \text{(Appl)}$$

3. Cálculos de Substituições Explícitas

Variantes do λ-calculus que *explicitam* a substituição

- Exemplos relevantes:
  - $\lambda\sigma$ [Abadi,Cardelli,Curien, Lévy 1991];
  - λs_e [Kamareddine, Ríos 1995];
  - *calculus de suspensão* [Nadathur, Wilson 1990's] λ-Prolog! (D. Miller).
- Ferramenta teórica adequada para o estudo e implementação da linguagens declarativas, assistentes de prova de ordem superior.

3. Cálculos de Substituições Explícitas

Propriedades desejáveis:

- Terminalidade do calculus de substituição (SN)
- Confluência (CR)
- Simulação da β-redução: Se $M \xrightarrow{\beta} N$, então $M \xrightarrow{*} N$ no s.e.
- Preservação da terminação (PSN): Se M termina no λ-calculus, então termina no s.e.

Questões não triviais!

3. Cálculos de Substituições Explícitas

$$
\begin{array}{lcl}
(\lambda M \;\; N) & \xrightarrow{(\text{Beta})} & M[N \cdot id] \\
(M \;\; N)[S] & \xrightarrow{(\text{App})} & (M[S] \;\; N[S]) \\
(\lambda M)[S] & \xrightarrow{(\text{Abs})} & \lambda M[\underline{1} \cdot (S \circ \uparrow)] \\
M[S][T] & \xrightarrow{(\text{Clos})} & M[S \circ T] \\
\underline{1}[M \cdot S] & \xrightarrow{(\text{VarCons})} & M \\
M[id] & \xrightarrow{(\text{Id})} & M \\
(S_1 \circ S_2) \circ T & \xrightarrow{(\text{Assoc})} & S_1 \circ (S_2 \circ T) \\
(M \cdot S) \circ T & \xrightarrow{(\text{Map})} & M[T] \cdot (S \circ T) \\
id \circ S & \xrightarrow{(\text{IdL})} & S \\
S \circ id & \xrightarrow{(\text{IdR})} & S \\
\uparrow \circ (M \cdot S) & \xrightarrow{(\text{ShiftCons})} & S \\
\underline{1} \cdot \uparrow & \xrightarrow{(\text{VarShift})} & id \\
\underline{1}[S] \cdot (\uparrow \circ S) & \xrightarrow{(\text{SCons})} & S \\
\lambda(M \;\; \underline{1}) & \xrightarrow{(\text{Eta})} & N \quad \text{if} \quad M =_\sigma N[\uparrow]
\end{array}
$$

Table 1: Regras de reescrita do $\lambda\sigma$-cálculo

$$
\begin{array}{lcl}
(\lambda M \;\; N) & \xrightarrow{(\sigma\text{-generation})} & M\, \sigma^1\, N \\
(\lambda M)\, \sigma^i N & \xrightarrow{(\sigma\text{-}\lambda\text{-transition})} & \lambda(M\, \sigma^{i+1}\, N) \\
(M_1 \;\; M_2)\, \sigma^i N & \xrightarrow{(\sigma\text{-app-transition})} & ((M_1\, \sigma^i N) \;\; (M_2\, \sigma^i N)) \\
\underline{n}\, \sigma^i N & \xrightarrow{(\sigma\text{-destruction})} & \begin{cases} \underline{n-1} & \text{if } \; n > i \\ \varphi_0^i\, N & \text{if } \; n = i \\ \underline{n} & \text{if } \; n < i \end{cases} \\
\varphi_k^i(\lambda M) & \xrightarrow{(\varphi\text{-}\lambda\text{-transition})} & \lambda(\varphi_{k+1}^i\, M) \\
\varphi_k^i(M_1 \;\; M_2) & \xrightarrow{(\varphi\text{-app-transition})} & ((\varphi_k^i\, M_1) \;\; (\varphi_k^i\, M_2)) \\
\varphi_k^i\, \underline{n} & \xrightarrow{(\varphi\text{-destruction})} & \begin{cases} \underline{n+i-1} & \text{if } \; n > k \\ \underline{n} & \text{if } \; n \leq k \end{cases} \\
(M_1\, \sigma^i M_2)\, \sigma^j\, N & \xrightarrow{(\sigma\text{-}\sigma\text{-transition})} & (M_1\, \sigma^{j+1}\, N)\; \sigma^i\, (M_2\, \sigma^{j-i+1}\, N) \quad \text{if } \; i \leq j \\
(\varphi_k^i\, M)\, \sigma^j\, N & \xrightarrow{(\sigma\text{-}\varphi\text{-transition 1})} & \varphi_k^{i-1}\, M \quad \text{if } \; k < j < k+i \\
(\varphi_k^i\, M)\, \sigma^j\, N & \xrightarrow{(\sigma\text{-}\varphi\text{-transition 2})} & \varphi_k^i(M\, \sigma^{j-i+1}\, N) \quad \text{if } \; k+i \leq j \\
\varphi_k^i(M\, \sigma^j\, N) & \xrightarrow{(\varphi\text{-}\sigma\text{-transition})} & (\varphi_{k+1}^i\, M)\, \sigma^j\, (\varphi_{k+1-j}^i\, N) \quad \text{if } \; j \leq k+1 \\
\varphi_k^i\, (\varphi_l^j\, M) & \xrightarrow{(\varphi\text{-}\varphi\text{-transition 1})} & \varphi_l^j\, (\varphi_{k+1-j}^i\, M) \quad \text{if } \; l+j \leq k \\
\varphi_k^i\, (\varphi_l^j\, M) & \xrightarrow{(\varphi\text{-}\varphi\text{-transition 2})} & \varphi_l^{j+i-1}\, M \quad \text{if } \; l \leq k < l+j \\
\lambda(M \;\; \underline{1}) & \xrightarrow{(\text{Eta})} & N \quad \text{if } \; M =_{s_e} \varphi_0^2 N
\end{array}
$$

Table 2: Regras de reescrita do λs_e-cálculo

3. Cálculos de Substituições Explícitas

$\lambda\sigma$ Não CR (CR em termos básicos); σ-calculus CR e SN; Simula β-redução; Não PSN [Melliès 1995]; Finito; Dois tipos de entidades: "substituições" e "termos"

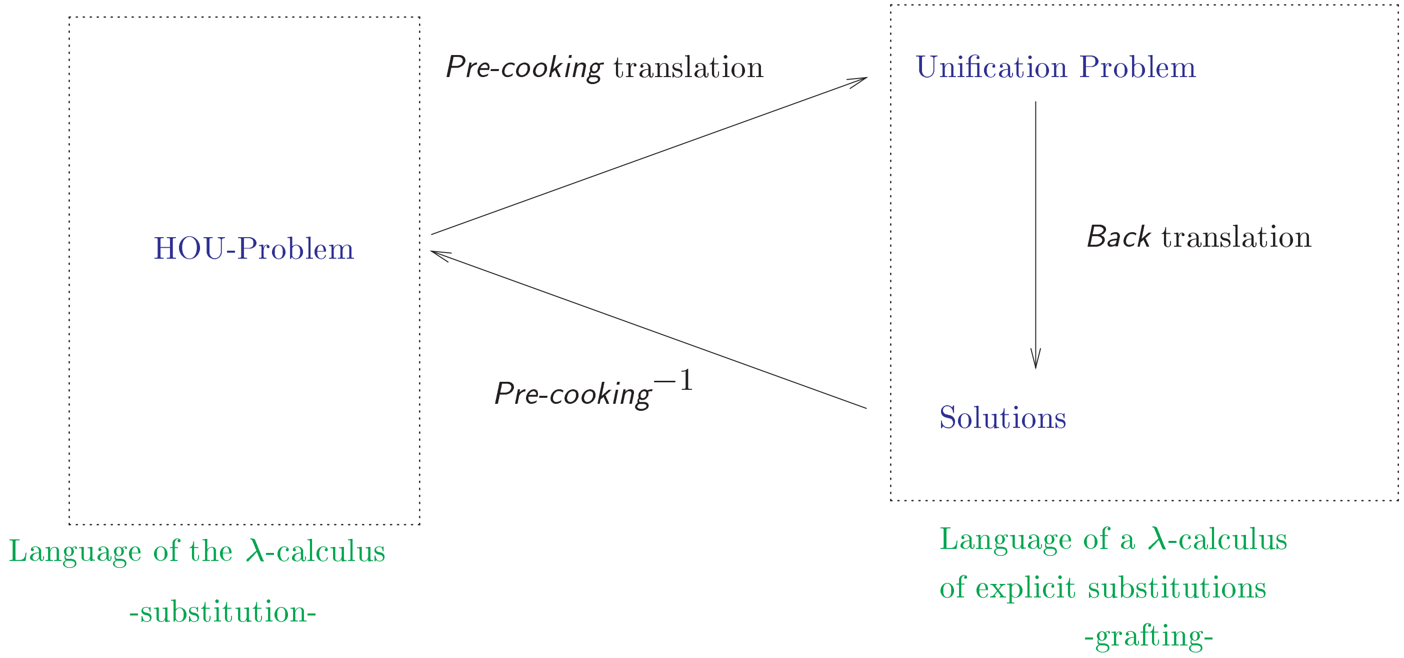
λs_e CR; s_e SN; Simula β-redução; ; Não PSN [Guilliaume 1999]; Infinito; Um só tipo de entidades: "termos"

$\lambda\sigma$ e λs_e incomparáveis [Kamareddine, Ríos 1999]

4. Aplicações

- Implementação de linguagens declarativas: λ-Prolog (calculus de suspensão) [Nadathur, Wilson, (Miller) 1990's]
- Idéias inseridas em assistentes de prova (de ordem superior)
  ELAN (ρ-calculus [Borovanský,Kirchner 1995])
  Maude [Messeguer]; etc.
- Idéias originadas no próprio sistema de *de Bruijn*, AUTOMATH, [de Bruijn 1965-1975]
- Unificação de ordem superior (HOU)

4. Aplicações — HOU



- Introduzido por [G. Dowek, T. Hardin e C. Kirchner 2000] usando o $\lambda\sigma$-cálculo.
- Subsume o método de HOU de [G. Huet's 1975].
- Extendido para o λs_e-cálculo [M. Ayala, F. Kamareddine 2000]

5. Conclusões

- Cálculos de substituições explícitas (tipados) são importantes na análise e implementação de assistentes de prova e linguagens (tipadas) declarativas.
- "Atomizam" as operações básicas subjacentes na substituição implícita, usualmente implementada via técnicas informais, como renomeamento de variáveis.
- Relevantes em mecanismos onde a "substituição" é uma opreação essencial. P.ex. no processo de unificação. Usado na extensão do princípio de resolução de Robinson a teorias de ordem superior.

5. Conclusões — Trabalho em Andamento e Futuro

- Comparação entre os cálculos: λs_e versus calculus de suspensão.
- Implementação de técnicas baseadas nestes cálculos. Como λs_e-HOU e $\lambda\sigma$-HOU.
- Desenvolvimento de melhor notação para os cálculos existentes.
- Estudo de novos mecanismos para definir tempo/espaço computacional no contexto do λ-calculus. Idéia original de [Abadi,Cardelli,Curien,Lévy 1991].

Palestra e artigos disponíveis em:
$\Rightarrow$ `www.mat.unb.br/~ayala/`

Orientação de projetos de mestrado relacionados no MAT/UnB:
$\Rightarrow$ `www.mat.unb.br/~ayala/offprojects.html`

5. Conclusões — Eventos Relacionados

- *Workshop on Explicit Substitutions: Theory and Applications to Programs and Proofs* - WESTTAP (1998 Tsukuba, Japan; 1999 Trento, Italy; 2000 Norwich, England; 2001 Utrecht, The Netherlands).
- *Eighth Workshop on Logic, Language, Information and Computation* - WoLLIC 2001, 31 de julho - 03 de agosto 2001, Brasília
  $\Rightarrow$ `www.cin.ufpe.br/~wollic/wollic2001`
- Participação:
  Fairouz Kamareddine (Tutorial: História da teoria de tipos)
  Gilles Dowek (Tutorial: Mixturando deduções e computações)