(Still) Program Extraction from Large Proof Developments

Luís Cruz-Filipe[1,2]

Bas Spitters[1]

[1] University of Nijmegen, Netherlands

[2] Center for Logic and Computation, IST (Lisbon), Portugal

Porquê?

- Biblioteca de matemática construtiva

- Coq tem mecanismo de extracção

- Não funciona. . .

Disclaimer

Por motivos alheios à responsabilidade dos autores, não foi possível executar nenhum dos programas que aqui serão discutidos. Por este motivo, todas as afirmações de tipo

$$\text{o programa } A \text{ é } \left\{ \begin{array}{c} \text{mais} \\ \text{tão} \\ \text{menos} \end{array} \right\} \text{ eficiente } \left\{ \begin{array}{c} \text{que} \\ \text{como} \\ \text{que} \end{array} \right\} \text{ o programa } B$$

devem ser interpretadas de espírito aberto.

Itinerário

1. Introdução
2. Program Extraction: Breve Motivação
3. A Lógica e a Formalização do FTA
4. Resultados Concretos
5. Algumas Conclusões...
6. Future Work

Extracção

- Interpretação BHK: conectivos
- “Realizability” (Kleene): uma visão mais formal
- Isomorfismo de Curry–Howard: provas $\longleftrightarrow$ programas
- Na prática: algoritmo vs. propriedades; tipos como “anotações”

Lógica

- Não há eliminação de termos em $\mathbf{Prop}$ sobre $\mathbf{Set}$ $\rightsquigarrow$ não é possível definir funções por casos

- Lógica em $\mathbf{Set}$

- Programa extraído gigantesco

Uma solução?

Seleccionar proposições *com conteúdo computacional*; tudo o resto vive em $\mathbf{Prop}$.

$\rightsquigarrow$ a maior parte dos termos de prova pode ser posta em $\mathbf{Prop}$

$\rightsquigarrow$ eliminação de uma quantidade significativa de "dead code"

(para mais informações: paper in Procs. TPHOLS 2003)

$$
\begin{array}{rcl}
\neg & : & s \to \mathbf{Prop} \\[2em]
\to & : & s_1 \to s_2 \to s_2 \\[2em]
\vee & : & s_1 \to s_2 \to \mathbf{Set} \\[2em]
\wedge & : & s_1 \to s_2 \to \begin{cases} \mathbf{Prop} & s_1 = s_2 = \mathbf{Prop} \\ \mathbf{Set} & \text{c.c.} \end{cases} \\[2em]
\forall & : & \Pi(A : s_1).(A \to s_2) \to s_2 \\[2em]
\exists & : & \Pi(A : \mathbf{Set}).(A \to s) \to \mathbf{Set}
\end{array}
$$

Resultados

- FTA: é extraído, compila, executa... mas não termina

- Racionais: computações (quase) instantâneas

- Pelo caminho: e, π e $\sqrt{2}$

Calculando e

$$e \stackrel{\text{def}}{=} \sum_{n=0}^{+\infty} \frac{1}{k!}$$

$\leadsto$ cada termo é um racional (sucessão constante)

$\leadsto$ mas há muita coisa a acontecer...

Problemas Óbvios...

- Numeros naturais unários

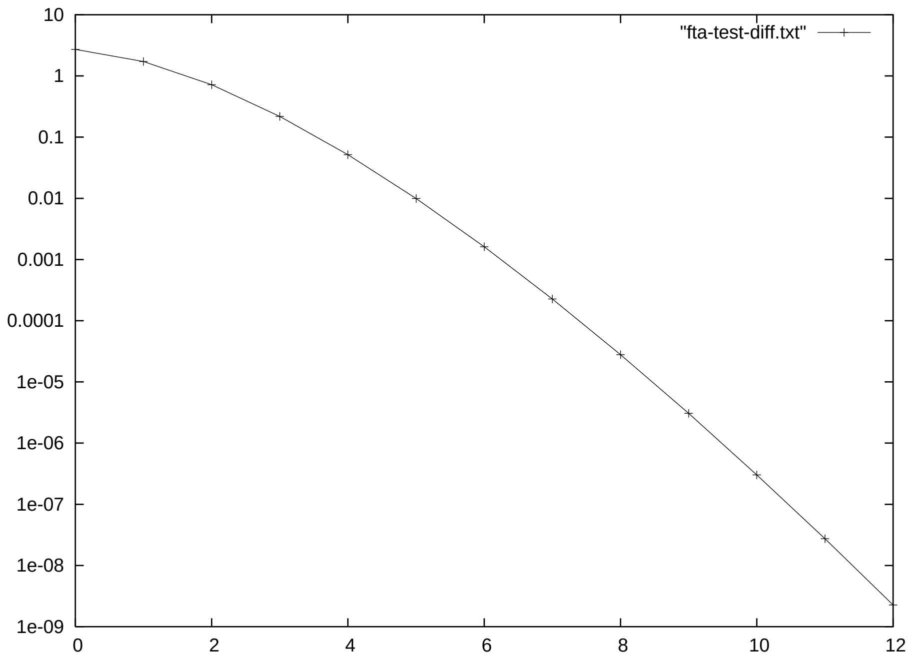
- Uma prova directa de $k! \neq 0$ requer calcular $k!$ em notação unária

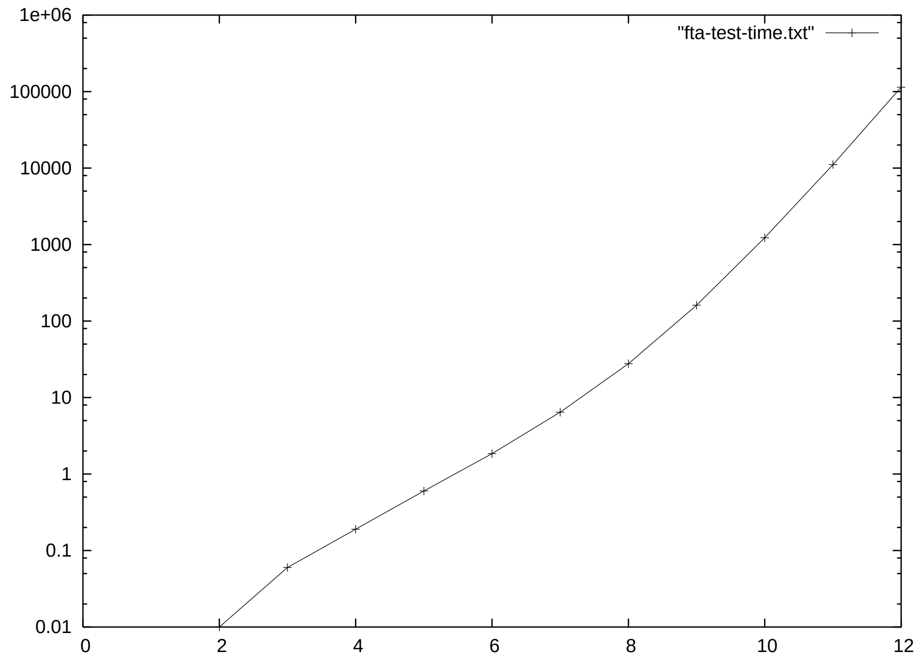
. . . & Soluções

- Injecção de $\mathbb{Z}^+$ em $\mathbb{R}$

- Provar $k! \neq 0$ por indução em k

Alguns dados estatísticos. . .





Melhor ainda (Obrigado, Pierre!)

Optimizar os resultados trabalhando no modelo:

- Definição mais eficiente de factorial;
- Provas mais simples, termos de prova mais curtos

$\leadsto$ 100ª aproximação em 77 segundos (com 157 algarismos de precisão)

O próximo passo: $\sqrt{2}$

Várias formulações construtivas do Teorema de Bolzano. . .

- para funções totais;
- para funções parciais;
- para funções monótonas;
- para funções localmente não constantes (!);
- para polinómios.

. . . e diferentes programas extraídos:

- nova versão de $\sqrt{2}$ com primeira aproximação computável em apenas 6 segundos (em vez 52 horas);

- complexidade exponential;

- lema fundamental (versão para funções crescentes)

$$a < b \Rightarrow f(a) < f(b), \text{ onde } f \text{ é a função a ser iterada}$$

Conclusões

- Quanto mais abstracta a formalização, menos eficiente o programa extraído

- Obter um programa *que funciona* está longe de ser trivial

- Pequenas e bem planeadas modificações na formalização podem fazer uma enorme diferença no programa extraído

- Futuras modificações em Coq podem trazer grandes alterações. . .

Future Work

- $\sqrt{2}$ "Computável"

- Melhorando o mecanismo de extracção: pruning, módulos (?)

- Eventualmente: o FTA

Convite

Vocaal Ensemble PANiek

- 21 de Outubro de 2003, 21h30, Igreja de Benfica
- 23 de Outubro de 2003, 18h00, FCUL
- 24 de Outubro de 2003, 19h00, CCB (Bar-Terraço)