The chicken and the pencil

Ali Assaf

Inria, Ecole Polytechnique

25 March, 2015
Deducteam seminar

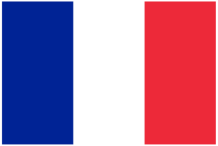
1 A short avian history

2 Enter the pencil

3 A perilous journey

4 And the results are...

Coq



Coqine

First implementation by Boespflug and Burel (2012)

- Fork of Coq
- Based on Cousineau and Dowek's embedding of PTS (2007)
- Inductive types

Coqine

Problems

- Unmaintainable
- Buggy, incomplete
- Inconsistent translation of universes

$$\mathsf{Type} : \mathsf{Type}$$

Coqine 2.0

- Plugin architecture
- Infinite universe hierarchy

$$\mathsf{Type}_i : \mathsf{Type}_{i+1}$$

- Universe cumulativity (Assaf 2014)

$$\frac{M : \mathsf{Type}_i}{M : \mathsf{Type}_{i+1}}$$

Coqine 2.0

Incomplete

- Modules
- Local fixpoints
- Universe polymorphism
- η-conversion
- Co-inductive types

Coqine 2.0

Incomplete

- Modules
- Local fixpoints
- Universe polymorphism
- η-conversion
- Co-inductive types

Needed by standard library

1. A short avian history

2. Enter the pencil

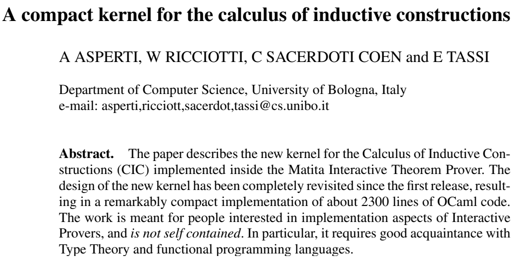
3. A perilous journey

4. And the results are...

Matita

Advantages of using Matita

- Relatively new
- Simpler kernel closer to CIC
- Kernel implementation well-documented



A compact kernel for the calculus of inductive constructions

A ASPERTI, W RICCIOTTI, C SACERDOTI COEN and E TASSI

Department of Computer Science, University of Bologna, Italy
e-mail: asperti,ricciott,sacerdot,tassi@cs.unibo.it

Abstract. The paper describes the new kernel for the Calculus of Inductive Constructions (CIC) implemented inside the Matita Interactive Theorem Prover. The design of the new kernel has been completely revisited since the first release, resulting in a remarkably compact implementation of about 2300 lines of OCaml code. The work is meant for people interested in implementation aspects of Interactive Provers, and *is not self contained*. In particular, it requires good acquaintance with Type Theory and functional programming languages.

Advantages of using Matita

Simplifications

- No modules ✓
- No local fixpoints ✓
- No universe polymorphism ✓

Make your choice

or

?

Krajono

A new translator in the Dedukti family

- “Pencil” in Esperanto
- Fork of Matita
- 2 months of development

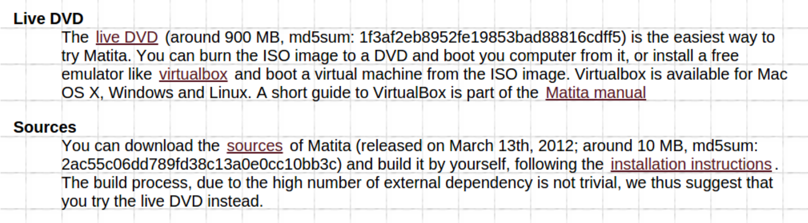
1. A short avian history
2. Enter the pencil
3. A perilous journey
4. And the results are...

Problem 1: Installation

Compiling Matita is so complicated...

Problem 1: Installation

Compiling Matita is so complicated...
... that it is easier to install and run a virtual machine.



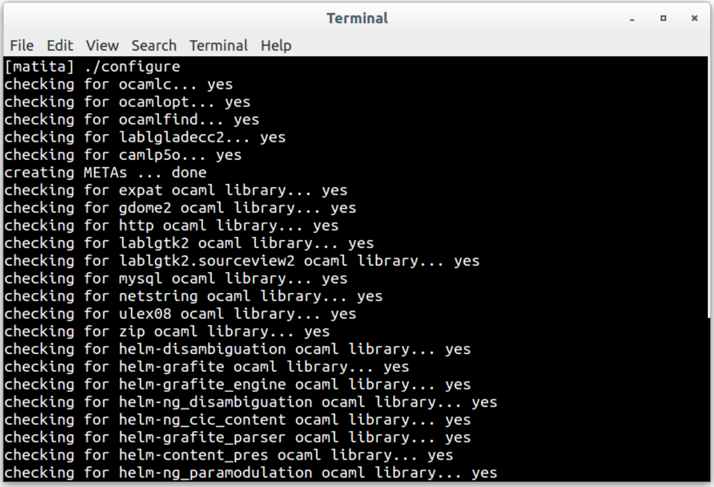
Live DVD

The live DVD (around 900 MB, md5sum: 1f3af2eb8952fe19853bad88816cdff5) is the easiest way to try Matita. You can burn the ISO image to a DVD and boot you computer from it, or install a free emulator like virtualbox and boot a virtual machine from the ISO image. Virtualbox is available for Mac OS X, Windows and Linux. A short guide to VirtualBox is part of the Matita manual

Sources

You can download the sources of Matita (released on March 13th, 2012; around 10 MB, md5sum: 2ac55c06dd789fd38c13a0e0cc10bb3c) and build it by yourself, following the installation instructions. The build process, due to the high number of external dependency is not trivial, we thus suggest that you try the live DVD instead.

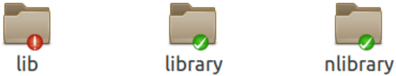
Dependencies



```
[matita] ./configure
checking for ocamlc... yes
checking for ocamlopt... yes
checking for ocamlfind... yes
checking for lablgladecc2... yes
checking for camlp5o... yes
creating METAs ... done
checking for expat ocaml library... yes
checking for gdome2 ocaml library... yes
checking for http ocaml library... yes
checking for lablgtk2 ocaml library... yes
checking for lablgtk2.sourceview2 ocaml library... yes
checking for mysql ocaml library... yes
checking for netstring ocaml library... yes
checking for ulex08 ocaml library... yes
checking for zip ocaml library... yes
checking for helm-disambiguation ocaml library... yes
checking for helm-grafite ocaml library... yes
checking for helm-grafite_engine ocaml library... yes
checking for helm-ng_disambiguation ocaml library... yes
checking for helm-ng_cic_content ocaml library... yes
checking for helm-grafite_parser ocaml library... yes
checking for helm-content_pres ocaml library... yes
checking for helm-ng_paramodulation ocaml library... yes
```

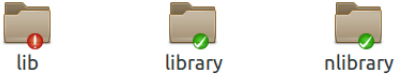
Problem 2: Finding a benchmark

- Matita is shipped with 3 different "libraries"...



Problem 2: Finding a benchmark

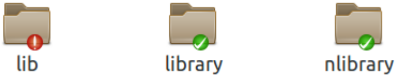
- Matita is shipped with 3 different “libraries”...



- ... only 1 of which compiles...

Problem 2: Finding a benchmark

- Matita is shipped with 3 different “libraries”...



- ... only 1 of which compiles...
- ... barely.

The benchmark

`lib/arithmetics`

- Part of the “standard” library
- Compiles completely
- Non-trivial, used by the devs

The benchmark

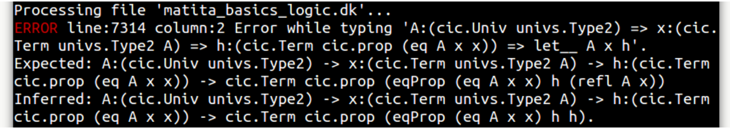
`lib/arithmetics`

- Part of the “standard” library
- Compiles completely
- Non-trivial, used by the devs

Other potential benchmark: CerCo project

Problem 3: type-checking error

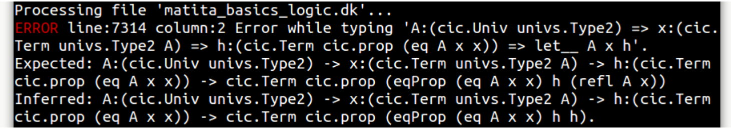
Towards the end of `logic.ma`:



```
Processing file 'matita_basics_logic.dk'...
ERROR line:7314 column:2 Error while typing 'A:(cic.Univ univs.Type2) => x:(cic.
Term univs.Type2 A) => h:(cic.Term cic.prop (eq A x x)) => let__ A x h'.
Expected: A:(cic.Univ univs.Type2) -> x:(cic.Term univs.Type2 A) -> h:(cic.Term
cic.prop (eq A x x)) -> cic.Term cic.prop (eqProp (eq A x x) h (refl A x))
Inferred: A:(cic.Univ univs.Type2) -> x:(cic.Term univs.Type2 A) -> h:(cic.Term
cic.prop (eq A x x)) -> cic.Term cic.prop (eqProp (eq A x x) h h).
```

Problem 3: type-checking error

Towards the end of `logic.ma`:



```
Processing file 'matita_basics_logic.dk'...
ERROR line:7314 column:2 Error while typing 'A:(cic.Univ univs.Type2) => x:(cic.
Term univs.Type2 A) => h:(cic.Term cic.prop (eq A x x)) => let__ A x h'.
Expected: A:(cic.Univ univs.Type2) -> x:(cic.Term univs.Type2 A) -> h:(cic.Term
cic.prop (eq A x x)) -> cic.Term cic.prop (eqProp (eq A x x) h (refl A x))
Inferred: A:(cic.Univ univs.Type2) -> x:(cic.Term univs.Type2 A) -> h:(cic.Term
cic.prop (eq A x x)) -> cic.Term cic.prop (eqProp (eq A x x) h h).
```

The type

$$\text{eqProp}\ \ (\text{eq}\ A\ x\ x)\ \ h\ h$$

is not equivalent to

$$\text{eqProp}\ \ (\text{eq}\ A\ x\ x)\ \ h\ (\text{refl}\ \ A\ x)$$

A thorn in the side

Prop irrelevance

$$\frac{M_i \equiv N_i \vee M_i : A_i : \text{Prop} \quad \forall i \in 1 \cdots n}{c\, M_1 \cdots M_n \equiv \; c\, N_1 \cdots N_n}$$

A thorn in the side

Prop irrelevance

$$\frac{M_i \equiv N_i \vee M_i : A_i : \mathrm{Prop} \quad \forall i \in 1 \cdots n}{c\, M_1 \cdots M_n \equiv \ c\, N_1 \cdots N_n}$$

Non-conservative extension of CIC

- Axiom K (UIP)

A thorn in the side

Prop irrelevance

$$\frac{M_i \equiv N_i \vee M_i : A_i : \mathrm{Prop} \quad \forall i \in 1 \cdots n}{c\, M_1 \cdots M_n \equiv \; c\, N_1 \cdots N_n}$$

Non-conservative extension of CIC

- Axiom K (UIP)
- Subsets (e.g. vectors)

$$\{x : A \mid P(x)\} = \Sigma x : A.P(x)$$

Prop irrelevance

- Cannot typecheck terms that use this feature.

Prop irrelevance

- Cannot typecheck terms that use this feature.

Remark

That's a good thing!

Prop irrelevance

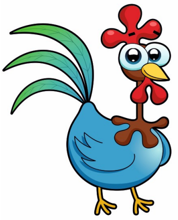
- Cannot typecheck terms that use this feature.

Remark

That's a good thing!

- Solution: comment the proofs.

Make your choice



or

?

In the end

Success!

- Translation of Matita to Dedukti
- Minimal modification of the Matita kernel
- Minimal modification of the Matita library

1. A short avian history

2. Enter the pencil

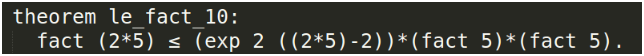
3. A perilous journey

4. And the results are...

	Compressed size (B)		Compiled size (B)		Time (s)	
Name	Matita	Dedukti	Matita	Dedukti	Matita	Dedukti
arithmetics/bigops.ma	4736	209137	357886	3093447	13.1	1.1
arithmetics/binomial.ma	2330	42064	107207	321836	8.9	0.2
arithmetics/bounded_quantifiers.ma	843	3786	16833	34302	0.7	0.0
arithmetics/chinese_reminder.ma	1479	32436	68050	297282	70.9	0.2
arithmetics/congruence.ma	1298	12921	49880	114767	1.3	0.1
arithmetics/div_and_mod.ma	3812	33212	142484	407073	38.4	0.1
arithmetics/exp.ma	1658	6462	35500	63873	10.2	0.0
arithmetics/factorial.ma	1604	63591	120377	527390	11.4	1285.0
arithmetics/fermat_little_theorem.ma	2133	14134	40948	101381	21.1	0.2
arithmetics/gcd.ma	3694	39121	124201	391963	59.0	0.2
arithmetics/iteration.ma	686	1085	9402	9599	0.7	0.0
arithmetics/log.ma	2174	10770	52920	91783	5.7	0.1
arithmetics/lstar.ma	1547	8255	37167	112547	0.8	0.0
arithmetics/minimization.ma	3128	28415	103919	339775	12.6	0.1
arithmetics/nat.ma	6412	40364	246845	562027	58.3	0.2
arithmetics/ord.ma	3846	31914	109807	293024	6.8	0.2
arithmetics/permutation.ma	3030	17787	72502	216218	1.4	0.1
arithmetics/pidgeon_hole.ma	1373	7559	26013	55472	6.2	0.1
arithmetics/primes.ma	4162	24558	121816	249941	26.5	15.9
arithmetics/sigma_pi.ma	1952	19957	77985	220514	16.6	0.1
arithmetics/sqrt.ma	2154	22084	64365	149516	2.1	0.1
...	...	...	...	...	...	...
Total	89819	1068830	3195300	12242374	437.7	1412.3
Factor		11.9		3.8		3.2

A small theorem

In file `arithmetics/factorial.ma`:



```
theorem le_fact_10:
  fact (2*5) ≤ (exp 2 ((2*5)-2))*(fact 5)*(fact 5).
```

$$(2 \times 5)! \leq 2^{2\times 5 - 2} \times 5! \times 5!$$

"Even the smallest theorem
can change the course of a benchmark." – Galadriel

	Compressed size (B)		Compiled size (B)		Time (s)	
Name	Matita	Dedukti	Matita	Dedukti	Matita	Dedukti
arithmetics/bigops.ma	4736	209137	357886	3093447	13.1	1.1
arithmetics/binomial.ma	2330	42064	107207	321836	8.9	0.2
arithmetics/bounded_quantifiers.ma	843	3786	16833	34302	0.7	0.0
arithmetics/chinese_reminder.ma	1479	32436	68050	297282	70.9	0.2
arithmetics/congruence.ma	1298	12921	49880	114767	1.3	0.1
arithmetics/div_and_mod.ma	3812	33212	142484	407073	38.4	0.1
arithmetics/exp.ma	1658	6462	35500	63873	10.2	0.0
arithmetics/factorial.ma	1604	63591	120377	527390	11.4	0.5
arithmetics/fermat_little_theorem.ma	2133	14134	40948	101381	21.1	0.2
arithmetics/gcd.ma	3694	39121	124201	391963	59.0	0.2
arithmetics/iteration.ma	686	1085	9402	9599	0.7	0.0
arithmetics/log.ma	2174	10770	52920	91783	5.7	0.1
arithmetics/lstar.ma	1547	8255	37167	112547	0.8	0.0
arithmetics/minimization.ma	3128	28415	103919	339775	12.6	0.1
arithmetics/nat.ma	6412	40364	246845	562027	58.3	0.2
arithmetics/ord.ma	3846	31914	109807	293024	6.8	0.2
arithmetics/permutation.ma	3030	17787	72502	216218	1.4	0.1
arithmetics/pidgeon_hole.ma	1373	7559	26013	55472	6.2	0.1
arithmetics/primes.ma	4162	24558	121816	249941	26.5	15.9
arithmetics/sigma_pi.ma	1952	19957	77985	220514	16.6	0.1
arithmetics/sqrt.ma	2154	22084	64365	149516	2.1	0.1
...	...	...	...	...	...	...
Total	89819	1068830	3195300	12242374	437.7	127.8
Factor		11.9		3.8		0.3

Conclusion

- There are no typecheckers for CIC out there

Conclusion

- There are no typecheckers for CIC out there
- (except Coqine and Krajono)

Conclusion

- There are no typecheckers for CIC out there
- (except Coqine and Krajono)

https://www.rocq.inria.fr/deducteam/Krajono/

Thank you

Bibliography

- D. Cousineau and G. Dowek
  Embedding pure type systems in the $\lambda\Pi$-calculus modulo
  TLCA 2007

- M. Boespflug and G. Burel
  CoqInE: Translating the calculus of inductive constructions into the$\lambda\Pi$-calculus modulo
  PxTP 2012

- A. Assaf
  A calculus of constructions with explicit subtyping
  Types 2014

- A. Asperti, W. Riccotti, C. Sacerdoti Coen and E. Tassi
  A compact kernel for the calculus of inductive constructions
  Sadhana Vol. 34, Issue 1, February 2009