Dedukti - A universal type-checker

Guillaume Genestier
Under the supervision of Frédéric Blanqui et Olivier Hermant

Wednesday, June 13th 2018
A problem has been detected and Windows has been shut down to prevent damage to your computer.

**PAGE_FAULT_IN_NONPAGED_AREA**

If this is the first time you've seen this error screen, restart your computer. If this screen appears again, follow these steps:

Check to make sure any new hardware or software is properly installed. If this is a new installation, ask your hardware or software manufacturer for any Windows updates you might need.

If problems continue, disable or remove any newly installed hardware or software. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select Safe Mode.

Technical information:

*** STOP: 0x00000050 (0x8872A990, 0x00000001, 0x804F35D7, 0x00000000)
*** ati3diag.dll - Address ED80AC55 base at ED88F000, Date Stamp 3dcb24d0
Formal methods are required

**Medicine**

**Energy**

**Transportation**
A full galaxy of provers

- CubicalTT
- Coq
- Dedukti
- Isabelle/HOL
- Matita
- PVS
- Agda
- Automath
- Beluga
- Twelf
- Abella
- ProofCert

Dedukti - A universal type-checker

Guillaume Genestier
A full galaxy of provers

- PVS
- Agda
- Isabelle/HOL
- CubicalTT
- Automath
- Dedukti
- Coq
- Abella
- Matita
- Twelf
- Beluga
- ProofCert

Dedukti - A universal type-checker
Guillaume Genestier
Dedukti is well-suited for interoperability
Dedukti is a multipurpose type-checker based on the λΠ-calculus modulo theory.

Example of rewrite rule

Nat : Type.
0 : Nat.
S : Nat → Nat.

def plus : Nat → Nat → Nat.
[n] plus 0 n --> n
[m,n] plus (S m) n --> S (plus m n)
[m,n] plus m (S n) --> S (plus m n).

Example of dependent type

List : Nat → Type.
nil : List 0.
Cons : (n:Nat) → Nat → List n → List (S n)
How to use *Dedukti* to perform translation?

![Diagram showing the relationship between Matita, HOL, and Dedukti](diagram.png)
Abstraction:

\[ \Gamma \vdash A : \text{Type} \quad \Gamma, x : A \vdash B : \text{Type} \quad \Gamma, x : A \vdash t : B \]

\[ \Gamma \vdash \lambda(x : A).t : (x : A) \to B \]

Application:

\[ \Gamma \vdash t : (x : A) \to B \quad \Gamma \vdash u : A \]

\[ \Gamma \vdash t \, u : B[u/x] \]

Conversion:

\[ \Gamma \vdash t : A \quad \Gamma \vdash B : \text{Type} \]

\[ \Gamma \vdash t : B \]

\[ A \leftrightarrow^* B \]
How to decide convertibility?

The strategy of *Dedukti*:

\[ t_1 \rightarrow^* v_1 \overset{?}{=} v_2 \rightarrow^* t_2 \]
Expected properties of rewriting

- **Termination**: There is no infinite path.

  \[ f(x) \rightsquigarrow g(x) \]
  \[ g(x) \rightsquigarrow f(x) \]

- **Confluence**: Two reducts of the same term have a common reduct.

\[ t \rightarrow t' \]
\[ t' \rightarrow t'' \]
\[ t'' \rightarrow t''' \]
Dedukti - A universal type-checker

Guillaume Genestier
Under the supervision of Frédéric Blanqui et Olivier Hermant

Wednesday, June 13th 2018