Type-Based Termination in Coq

Gilles Barthe Benjamin Grégoire Fernando Pastawski
Jorge Luis Sacchini

February 2, 2010

Outline

Defining recursive functions

In systems like Coq, termination is ensured by syntactic criterion. Recursive functions must have a type of the form $I \to T$, where I is an inductive type.

$$\frac{\Gamma(f : I \to T) \vdash M : I \to T \qquad \mathcal{G}(f, M)}{\Gamma \vdash (\text{fix } f : I \to T := M) : I \to T}$$

- The predicate $\mathcal{G}(f, M)$ checks that all recursive calls of f in M are guarded by destructors;
- Reduction is restricted to constructor forms:

$$(\text{fix } f : I \to T := M)\, C \to M\,[f := (\text{fix } f : I \to T := M)]\; C$$

C must be headed by a constructor

Guard predicate

- Syntax sensitive
- Difficult to understand (e.g. div/minus, map)
- Too weak (e.g. do not allow functions like quicksort)
- Difficult to implement

Outline

Sized Types

- Most type systems for termination are based on the notion of *sized types*: user-defined datatypes are decorated with size information.
  - Ex: Nat^s (natural numbers smaller than s)
- User-defined datatypes are represented by fixpoints of some monotone operator

$$Nat ::= O : Nat \mid S : Nat \rightarrow Nat$$

- Sized types are approximations of these operators
  - $Nat^\infty = \{0, 1, 2, \ldots\}$
  - $Nat^s = \{0, 1, \ldots, s-1\}$
  - $Nat^s \leq Nat^{s+1} \leq \ldots \leq Nat^\infty$

Sized Types

With sized types, recursive functions are defined on approximations of an inductive type:

$$\frac{\Gamma(f : I^{\iota} \to T) \vdash M : I^{\iota+1} \to T}{\Gamma \vdash (\text{fix } f : I \to T := M) : I^{\infty} \to T}$$

- Recursive call are only allowed on terms of smaller size
- The reduction rule is not changed:

$$(\text{fix } f : I \to T := M)\, C \to M\, [f := (\text{fix } f : I \to T := M)]\ C$$

if C is in constructor form.

Outline

CIC$\widehat{\ }$: Syntax and Typing Rules

Inductive types

- Inductive types are decorated with a size (or stage) expression:

$$I^s$$

- Stages:

$$s ::= \iota \mid \widehat{s} \mid \infty$$

- Subtyping: less-or-equal relation on stages

$$\frac{}{s \sqsubseteq \widehat{s}} \qquad \frac{}{s \sqsubseteq \infty} \qquad (\infty \sqsubseteq \widehat{\infty} \sqsubseteq \infty)$$

defines the subtyping rule:

$$\frac{s \sqsubseteq r}{I^s \leq I^r}$$

$\widehat{\text{CIC}}$: Syntax and Typing Rules

Inductive types

$$\text{Inductive } \mathit{Nat} := O : \mathit{Nat} \mid S : \mathit{Nat} \to \mathit{Nat}$$

$$\frac{}{\Gamma \vdash O : \mathit{Nat}^{\widehat{s}}} \qquad \frac{\Gamma \vdash M : \mathit{Nat}^{s}}{\Gamma \vdash S\, M : \mathit{Nat}^{\widehat{s}}}$$

- Constructors are always fully applied
- Subtype relation defined by

$$\frac{s \sqsubseteq r}{\mathit{Nat}^{s} \leq \mathit{Nat}^{r}}$$

Ex: $\mathit{Nat}^{\iota} \leq \mathit{Nat}^{\widehat{\iota}} \leq \mathit{Nat}^{\infty}$, but $\mathit{Nat}^{\iota} \not\leq \mathit{Nat}^{\kappa}$.

$\widehat{\mathrm{CIC}}$: Syntax and Typing Rules

Inductive types

$$
\begin{aligned}
\text{Inductive } \mathit{Ord} := {} & O : \mathit{Ord} \\
\mid {} & S : \mathit{Ord} \rightarrow \mathit{Ord} \\
\mid {} & \lim : (\mathit{Nat} \rightarrow \mathit{Ord}) \rightarrow \mathit{Ord}
\end{aligned}
$$

$$
\frac{}{\Gamma \vdash O : \mathit{Ord}^{\widehat{s}}} \qquad \frac{\Gamma \vdash M : \mathit{Ord}^{s}}{\Gamma \vdash S\,M : \mathit{Ord}^{\widehat{s}}}
$$

$$
\frac{\Gamma \vdash F : \mathit{Nat}^{\infty} \rightarrow \mathit{Ord}^{s}}{\Gamma \vdash \lim\ F : \mathit{Ord}^{\widehat{s}}}
$$

- Previously defined inductive types are tagged with ∞

$\mathrm{CIC}\hat{}$: Syntax and Typing Rules

Implicit size polymorphism

- Sizes are not first-class terms
- We have a form of implicit size polymorphism: sizes are not applied nor explicitly quantified.
- This allows to keep the same reduction mechanism
- But to keep Subject Reduction, terms in *type positions* have no size information. Ex:

$$\vdash \lambda x : Nat.x : Nat^s \to Nat^s, \qquad \text{for any } s$$

- Type positions includes: types of abstraction, case, fixpoint, and parameters of constructors.

$\mathrm{CIC}\widehat{\ }$: Syntax and Typing Rules

Inductive types with parameters

- Parameters can have a polarity.

$$
\begin{aligned}
&\text{Inductive } \mathit{Tree}(A+ : \text{Type})(B- : \text{Type}) := \\
&\quad \mid \mathit{leaf} : A \to \mathit{Tree}\ A\ B \\
&\quad \mid \mathit{node} : (B \to \mathit{Tree}\ A\ B) \to \mathit{Tree}\ A\ B
\end{aligned}
$$

- Subtyping rule: $\mathit{Tree}^{\iota}\ \mathit{Nat}^{\kappa}\ \mathit{Nat}^{\infty} \leq \mathit{Tree}^{\widehat{\iota}}\ \mathit{Nat}^{\infty}\ \mathit{Nat}^{\iota}$

CIC$\widehat{\ }$: Syntax and Typing Rules

Inductive types with parameters

- Parameters can have a polarity.

$$
\begin{aligned}
&\text{Inductive } \mathit{Tree}(A+ : \text{Type})(B- : \text{Type}) := \\
&\quad \mid \mathit{leaf} : A \to \mathit{Tree}\ A\ B \\
&\quad \mid \mathit{node} : (B \to \mathit{Tree}\ A\ B) \to \mathit{Tree}\ A\ B
\end{aligned}
$$

- Subtyping rule: $\mathit{Tree}^{\iota}\ \mathit{Nat}^{\kappa}\ \mathit{Nat}^{\infty} \leq \mathit{Tree}^{\widehat{\iota}}\ \mathit{Nat}^{\infty}\ \mathit{Nat}^{\iota}$
- Size information on parameters of constructors is erased

$$
\frac{\Gamma \vdash A : \text{Type} \qquad \Gamma \vdash B : \text{Type} \qquad \Gamma \vdash M : A}{\Gamma \vdash \mathit{leaf}\ |A|\ |B|\ M : \mathit{Tree}^{\widehat{s}}\ A\ B}
$$

$$
\frac{\Gamma \vdash A : \text{Type} \qquad \Gamma \vdash B : \text{Type} \qquad \Gamma \vdash M : B \to \mathit{Tree}^{s}\ A\ B}{\Gamma \vdash \mathit{node}\ |A|\ |B|\ M : \mathit{Tree}^{\widehat{s}}\ A\ B}
$$

$\mathrm{CIC}\widehat{}$: Syntax and Typing Rules

Fixpoint rule

Fixpoint: fix $f : T^{\star} := M$

$$\frac{T = \Pi x : Nat^{\iota}.U \qquad \iota \text{ pos } U \qquad \iota \text{ does not appear in } \Gamma, M \qquad \Gamma(f : T^{\iota}) \vdash M : T^{\widehat{\iota}}}{\Gamma \vdash (\text{fix } f : T^{\star} := M) : T^{s}}$$

$\mathrm{CIC}\widehat{}$: Syntax and Typing Rules

Fixpoint rule

Fixpoint: fix $f : T^\star := M$

$$\frac{T = \Pi x : Nat^\iota . U \qquad \iota \text{ pos } U \qquad \iota \text{ does not appear in } \Gamma, M \qquad \Gamma(f : T^\iota) \vdash M : T^{\widehat{\iota}}}{\Gamma \vdash (\text{fix } f : T^\star := M) : T^s}$$

- $T^\star$ is a *position type*: size annotations are either empty, or $\star$ to indicate recursive positions.

$\mathrm{CIC}\widehat{\ }$: Syntax and Typing Rules

Fixpoint rule

Fixpoint: fix $f : T^\star := M$

$$\frac{T = \Pi x : Nat^\iota . U \qquad \iota \text{ pos } U \qquad \iota \text{ does not appear in } \Gamma, M \qquad \Gamma(f : T^\iota) \vdash M : T^{\widehat{\iota}}}{\Gamma \vdash (\text{fix } f : T^\star := M) : T^s}$$

- $T^\star$ is a *position type*: size annotations are either empty, or $\star$ to indicate recursive positions. Ex:
  - $\vdash (\text{fix } f : Nat^\star \to Nat := \lambda x : Nat.O) : Nat^\iota \to Nat^{\widehat{\kappa}}$

$\mathrm{CIC}\widehat{\ }$: Syntax and Typing Rules

Fixpoint rule

Fixpoint: fix $f : T^\star := M$

$$\frac{T = \Pi x : Nat^\iota.U \qquad \iota \text{ pos } U \qquad \iota \text{ does not appear in } \Gamma, M \qquad \Gamma(f : T^\iota) \vdash M : T^{\widehat{\iota}}}{\Gamma \vdash (\text{fix } f : T^\star := M) : T^s}$$

- $T^\star$ is a *position type*: size annotations are either empty, or $\star$ to indicate recursive positions. Ex:
  - $\vdash (\text{fix } f : Nat^\star \to Nat := \lambda x : Nat.O) : Nat^\iota \to Nat^{\widehat{\kappa}}$
  - $\vdash (\text{fix } f : Nat^\star \to Nat^\star := \lambda x : Nat.O) : Nat^\iota \to Nat^\iota$

$\mathrm{CIC}\widehat{\ }$: Syntax and Typing Rules

Fixpoint rule

Fixpoint: fix $f : T^\star := M$

$$\frac{T = \Pi x : Nat^\iota . U \qquad \iota \text{ pos } U \qquad \iota \text{ does not appear in } \Gamma, M \qquad \Gamma(f : T^\iota) \vdash M : T^{\widehat{\iota}}}{\Gamma \vdash (\text{fix } f : T^\star := M) : T^s}$$

- $T^\star$ is a *position type*: size annotations are either empty, or $\star$ to indicate recursive positions. Ex:
  - $\vdash (\text{fix } f : Nat^\star \to Nat := \lambda x : Nat.O) : Nat^\iota \to Nat^{\widehat{\kappa}}$
  - $\vdash (\text{fix } f : Nat^\star \to Nat^\star := \lambda x : Nat.O) : Nat^\iota \to Nat^\iota$
- They are useful to have compact general types:
  - $\vdash (\text{fix } f : Nat \to Nat := \lambda x : Nat.O) :?$

$\widehat{\text{CIC}}$: Syntax and Typing Rules

Fixpoint rule

- Types for fixpoint must be of the form

$$\Pi\Delta.(x : I^{\iota}\ \vec{a}).U$$

  with ι pos U, and ι does not appear in Δ
- Ex:
  - $Nat^{\iota} \rightarrow Nat^{\iota}$
  - $Nat^{\iota} \rightarrow Nat^{\infty} \rightarrow Nat^{\iota}$ (div, minus)
  - $List^{\iota}\ A \rightarrow List^{\iota}\ A$ (filter, map)
- Not allowed for fixpoint
  - $Nat^{\iota} \rightarrow Nat^{\iota} \rightarrow Nat^{\iota}$ (max)
  - $(Nat^{\infty} \rightarrow Nat^{\iota}) \rightarrow Nat^{\infty}$ (leads to non-termination) [Abel]

$\widehat{\text{CIC}}$: Syntax and Typing Rules

Fixpoint rule

Fixpoint: fix $f : T^\star := M$

$$\frac{T = \Pi\Delta.(x : I^\iota\, \vec{a}).U \qquad \#\Delta = n - 1 \qquad \iota \text{ pos } U \qquad \iota \text{ does not appear in } \Gamma, \Delta, \vec{a}, M \qquad \Gamma(f : T) \vdash M : T\,[\iota := \widehat{\iota}]}{\Gamma \vdash (\mathrm{fix}_n\ f : |T|^\iota := M) : T\,[\iota := s]}$$

- μ-reduction:

$$(\mathrm{fix}_n\ f : T^\star := M)\,\vec{u}\,C \to_\mu M\,[f := (\mathrm{fix}_n\ f : T^\star := M)]\ \vec{u}\,C$$

if $\#\vec{u} = n - 1$ and C is headed by a constructor

Example: subtraction

$$
\begin{array}{l}
(\mathrm{minus} : \mathit{Nat}^{\iota} \to \mathit{Nat}^{\infty} \to \mathit{Nat}^{\iota}) \vdash \\
\quad \lambda x : \mathit{Nat}^{\widehat{\iota}}.\lambda y : \mathit{Nat}^{\infty}. \\
\quad \text{case } x \text{ of} \\
\qquad \mid\ O \Rightarrow O \qquad\qquad : \mathit{Nat}^{\widehat{\iota}} \\
\qquad \mid\ S\, x_1^{\mathit{Nat}^{\iota}} \Rightarrow \text{case } y \text{ of} \\
\qquad\qquad\qquad \mid\ O \Rightarrow x \qquad\qquad : \mathit{Nat}^{\widehat{\iota}} \\
\qquad\qquad\qquad \mid\ S\, y_1^{\mathit{Nat}^{\infty}} \Rightarrow \mathrm{minus}\ x_1\ y_1 \quad : \mathit{Nat}^{\iota} \\
\qquad : \mathit{Nat}^{\widehat{\iota}} \to \mathit{Nat}^{\infty} \to \mathit{Nat}^{\widehat{\iota}}
\end{array}
$$

$$
\overline{\vdash \text{fix } \mathrm{minus} : \mathit{Nat}^{\star} \to \mathit{Nat} \to \mathit{Nat}^{\star} := \ldots : \mathit{Nat}^{s} \to \mathit{Nat}^{\infty} \to \mathit{Nat}^{s}}
$$

Example: subtraction

$$
\begin{array}{l}
(\mathrm{minus} : Nat^{\iota} \to Nat^{\infty} \to Nat^{\iota}) \vdash \\
\quad \lambda x : Nat^{\widehat{\iota}}.\lambda y : Nat^{\infty}. \\
\quad \text{case } x \text{ of} \\
\qquad \mid\ O \Rightarrow O\ (*\ x\ *) \quad : Nat^{\widehat{\iota}} \\
\qquad \mid\ S\, x_1^{Nat^{\iota}} \Rightarrow \text{case } y \text{ of} \\
\qquad\qquad\qquad \mid\ O \Rightarrow x\ (*\ S\, x_1\ *) \quad : Nat^{\widehat{\iota}} \\
\qquad\qquad\qquad \mid\ S\, y_1^{Nat^{\infty}} \Rightarrow \mathrm{minus}\ x_1\ y_1 \quad : Nat^{\iota} \\
\qquad : Nat^{\widehat{\iota}} \to Nat^{\infty} \to Nat^{\widehat{\iota}}
\end{array}
$$

$$\vdash \text{fix } \mathrm{minus} : Nat^{\star} \to Nat \to Nat^{\star} := \ldots : Nat^{s} \to Nat^{\infty} \to Nat^{s}$$

Example: division

$$div\ m\ n = \left\lceil \frac{m}{n+1} \right\rceil$$

$$
\begin{array}{l}
(\mathrm{div} : Nat^{\iota} \rightarrow Nat^{\infty} \rightarrow Nat^{\iota}) \vdash \\
\quad \lambda x : Nat^{\widehat{\iota}}.\lambda y : Nat^{\infty}. \\
\quad \text{case } x \text{ of} \\
\qquad \mid\ O \Rightarrow O \quad : Nat^{\widehat{\iota}} \\
\qquad \mid\ S\, x_1^{Nat^{\iota}} \Rightarrow S(\mathrm{div}\ (\mathrm{minus}\ x_1\ y)^{Nat^{\iota}}\ y) \quad : Nat^{\widehat{\iota}} \\
\qquad : Nat^{\widehat{\iota}} \rightarrow Nat^{\infty} \rightarrow Nat^{\widehat{\iota}}
\end{array}
$$

$$\overline{\vdash \text{fix div} : Nat^{\star} \rightarrow Nat \rightarrow Nat^{\star} := \ldots : Nat^{s} \rightarrow Nat^{\infty} \rightarrow Nat^{s}}$$

Example: quicksort

$$\text{filter} \equiv \ldots : \Pi A.(A \to \mathsf{bool}) \to \mathit{List}^s\, A \to \mathit{List}^s\, A$$
$$\text{append} \equiv \ldots : \Pi A.\mathit{List}^s\, A \to \mathit{List}^r\, A \to \mathit{List}^\infty\, A$$

$$\begin{aligned}
\text{quicksort} \equiv (&\lambda A.\mathsf{fix}\,\text{quicksort} : \mathit{List}^\star\, A \to \mathit{List}\, A := \\
&\lambda x : \mathit{List}^{\widehat{\iota}}\, A.\mathsf{case}\ x\ \mathsf{of} \\
&\mid \mathsf{nil} \Rightarrow \mathsf{nil} \\
&\mid \mathsf{cons}\, x\, xs^{\mathit{List}^\iota\, A} \Rightarrow \text{append}(\text{quicksort}\ (\text{filter}(< x)xs)^{\mathit{List}^\iota\, A}) \\
&\qquad\qquad (\mathsf{cons}\, x\, (\text{quicksort}\ (\text{filter}(\geq x)xs)^{\mathit{List}^\iota\, A}) \\
&) : \quad \Pi A.\mathit{List}^s\, A \to \mathit{List}^\infty\, A
\end{aligned}$$

Properties

$\text{CIC}\hat{}$ satisfies several desired metatheoretical properties:

- Confluent reduction
- Substitution
- Subject Reduction
- Decidability of Type Checking (assuming Strong Normalization)

Type Inference

There exists an algorithm that given an unannotated context Γ and an unannotated term M it returns either:

- an error if there is no well-typed annotation of M in Γ; or
- a fully annotated context Γ^+, a fully annotated term M^+ and a type of the form $C \Rightarrow T$ where C is a set of constraints such that:

Soundness for every stage substitution ρ satisfying C, we have $\Gamma\rho \vdash M\rho : T\rho$;

Completeness for every stage substitution ρ' and annotation M' of M such that $\Gamma\rho' \vdash M' : T'$, there exists ρ satisfying C such that $\Gamma\rho = \Gamma\rho'$ and $M\rho = M'$ and $T\rho \leq T'$.

Outline

Conclusions

- Type-based termination is more practical and simpler than syntactic criterions
- Size inference
- Not much more complicated for the user

Future Work

- Metatheoretical properties: consistency and strong normalization
  - Limited results: CC with universes and sized natural numbers, but a restricted type system with respect to $\mathrm{CIC}\hat{}$.
- Global definitions
- Coinductive types
- Mutually recursive functions

Future Work

- Explicit stage polymorphism:

$$\frac{\Gamma \vdash M : T \qquad \iota \text{ does not appear in } M}{\Gamma \vdash M : \forall \iota . T}$$

  Ex: $\vdash \lambda x : \mathit{Nat}.x : \forall \iota . \mathit{Nat}^{\iota} \rightarrow \mathit{Nat}^{\iota}$

- More operations on sizes: $+$, $\max$, ...
  - $\mathrm{append} : \mathit{List}^{\iota}\, A \rightarrow \mathit{List}^{\kappa}\, A \rightarrow \mathit{List}^{\iota+\kappa}\, A$

Future Work

- Explicit stage polymorphism:

$$\frac{\Gamma \vdash M : T \qquad \iota \text{ does not appear in } M}{\Gamma \vdash M : \forall \iota . T}$$

  Ex: $\vdash \lambda x : Nat.x : \forall \iota . Nat^{\iota} \rightarrow Nat^{\iota}$

- More operations on sizes: $+$, $\max$, ...
  - $\mathrm{append} : List^{\iota}\ A \rightarrow List^{\kappa}\ A \rightarrow List^{\iota+\kappa}\ A$

Thank you! Questions?