Ordered Models of CIC

Kenji Maillard

Inria Nantes, team Gallinette
j.w.w.

Meven Lennon-Bertrand
Nicolas Tabareau
Éric Tanter
Théo Laurent

GdT LHC
Friday the 5th of February, 2021
What is CIC?
CIC: Calculus of (Co)Inductive Constructions
A rich logical system & an expressive programming language

- Inductive and coinductive types with pattern-matching,
- Functions \((a : A) \rightarrow B\), (well-founded) fixpoints,
- Dependent types

\[ \vdash A \text{ type} \quad x : A \vdash B \text{ type} \quad n : \mathbb{N} \vdash \text{vect } A n \text{ type} \]
CIC: Calculus of (Co)Inductive Constructions
A rich logical system & an expressive programming language

- Inductive and coinductive types with pattern-matching,
- functions \( (a : A) \rightarrow B \), (well-founded) fixpoints,
- **Dependent types** internalized with Universes \( \mathbb{U}_i \):

\[
\begin{align*}
\vdash A : \mathbb{U}_i \\
\vdash \text{vect } A : (A : \mathbb{U}_i)(n : \mathbb{N}) \rightarrow \mathbb{U}_i \\
\vdash \mathbb{U}_i : \mathbb{U}_{i+1}
\end{align*}
\]
CIC: Calculus of (Co)Inductive Constructions
A rich logical system & an expressive programming language

- Inductive and coinductive types with pattern-matching,
- functions \((a : A) \rightarrow B\), (well-founded) fixpoints,
- **Dependent types** internalized with Universes \(\mathbb{U}_i\)

\[
\begin{align*}
\Gamma & \vdash A : \mathbb{U}_i \\
\Gamma & \vdash A \text{ type} \\
\Gamma & \vdash \text{vect } A : (A : \mathbb{U}_i)(n : \mathbb{N}) \rightarrow \mathbb{U}_i \\
\Gamma & \vdash \mathbb{U}_i : \mathbb{U}_{i+1}
\end{align*}
\]

Idealized metatheory of various proofs assistants:
Computations

Conversion

\[
\begin{align*}
\vdash 1 + 2 & \equiv 3 : \mathbb{N} \\
\vdash \text{vect } A (1 + 2) & \equiv \text{vect } A 3 \\
\vdash t : A & \vdash A \equiv B \\
\vdash t : B & 
\end{align*}
\]

Trade-offs between \textit{decidability} and \textit{expressivity}

Weak TT \hspace{1cm} \text{CIC} \hspace{1cm} \text{Extensional TT}

Trivial conversion \hspace{2cm} \text{βδιζη} \hspace{2cm} \text{Provable equality}

Checking proofs \hspace{1cm} \text{vs} \hspace{1cm} \text{Writing proofs}
Models of CIC For Fun And Profit
Add new proof principles:
  - Uniqueness of identity proofs (UIP)
  - Function extensionality (funext)
  - Quotients
  - Univalence principle
  - Markov principle
  - Parametricity

Account for existing programming features:
  - Exceptions
  - Access to a global environment
  - Subtyping
  - Dynamic type
Models of CIC in CIC:

- Defined inductively on the syntax of terms/types

\[
\llbracket - \rrbracket : \text{Type} \rightarrow \text{Type} \quad \llbracket - \rrbracket : \text{Term} \rightarrow \text{Term}
\]

- Preserving conversion (no coherence hell)

\[
\Gamma \vdash A \equiv B \quad \implies \quad [\Gamma] \vdash [A] \equiv [B]
\]

Main goal/theorem:

\[
\Gamma \vdash t : A \quad \implies \quad [\Gamma] \vdash [t] : [A]
\]
Syntactic Models

Models of CIC in CIC:

▶ Defined inductively on the syntax of terms/types

\[ [-] : Type \rightarrow Type \quad [-] : Term \rightarrow Term \]

▶ Preserving conversion (no coherence hell)

\[ \Gamma \vdash A \equiv B \quad \implies \quad [\Gamma] \vdash [A] \equiv [B] \]

Why syntactic models?

▶ Useful to prototype extensions of CIC
▶ Proposes extensions more amenable to implementations
▶ Help designing reductions/conversion rules
Examples from the literature

Reflexive graphs model: external parametricity  [Atkey et al.]
Types equipped with a reflexive relation

Setoid model: UIP, funext  [Altenkirch et al.]
Types equipped with an irrelevant equivalence relation

Exceptional model: Exceptions  [Pédrot et al.]
Pointed types

Reader model: Reading and setting a global cell  [Boulier et al.]
Presheaves on a set of states
Crucial steps

1. Give the structure of types, type families and terms
Crucial steps

1. Give the structure of types, type families and terms
2. Translate type constructors ($\mathbb{N}, \Pi$) & universes $[U_i] : [U_{i+1}]$
Syntactic Models: A Recipe

Crucial steps

1. Give the structure of types, type families and terms
2. Translate type constructors \((\mathbb{N}, \Pi)\) & universes \([\mathbb{U}_i] : [\mathbb{U}_{i+1}]\)
3. Check that conversion is preserved \((\beta\delta\iota\varsigma\eta \ldots)\)
Crucial steps

1. Give the structure of types, type families and terms
2. Translate type constructors ($\mathbb{N}, \Pi$) & universes $[\mathbb{U}_i] : [\mathbb{U}_{i+1}]$
3. Check that conversion is preserved ($\beta\delta\iota\zeta\eta \ldots$)
4. Extend the source CIC to a richer theory $\mathcal{T}$
   adding new constants and conversion rules
Ordered models of CIC
Step 1: Equip the translation of a type $A$ with a relation

$$\leq^A : A \to A \to Type$$

- reflexive: $(a : A) \to a \leq^A a$
- transitive: $(a_0 a_1 a_2 : A) \to a_0 \leq^A a_1 \to a_1 \leq^A a_2 \to a_0 \leq^A a_2$
- irrelevant: $(a_0 a_1 : A)(h h' : a_0 \leq^A a_1) \to h = h'$
- antisymmetric: $(a_0 a_1 : A) \to a_0 \leq^A a_1 \to a_1 \leq^A a_0 \to a_0 = a_1$

Middle point between the reflexive graph and setoid models.
Type families?

Translation of a type family \( x : A \vdash B \) type

\[
B : A \to \text{Preorder}
\]

\[
B^\leq_{(a_0 \ a_1 : A)} : a_0 \leq^A a_1 \to B a_0 \leadsto B a_1
\]

indexed variants of reflexive, transitive...

Multiple choices for \( (\leadsto) \):

- Relations respecting the order
- Monotone maps
- Galois connections
- Embedding-projection pairs \( X \lhd Y \)

\[
\uparrow : X \to Y \\
\downarrow : Y \to X
\]

such that

\[
\begin{align*}
\uparrow x \leq^Y y & \iff x \leq^X \downarrow y \\
\downarrow \uparrow x &= x
\end{align*}
\]
Interpretation of type constructors

Natural numbers

\[ \vdash 0 : \mathbb{N} \quad \vdash S : \mathbb{N} \rightarrow \mathbb{N} \]\n
\[ \vdash 0 \leq \mathbb{N} 0 \]

\[ \vdash \text{pf} : p \leq \mathbb{N} q \]

\[ \vdash \text{CongrS pf} : S p \leq \mathbb{N} S q \]

Order relation \( \leq \mathbb{N} \) induced by parametricity [Bernardy-Lasson]

Dependent products

\[ (a : A) \xrightarrow{\text{mon}} B := \{ f : (a : A) \rightarrow B \mid (a_01 : a_0 \leq^A a_1) \rightarrow B \leq a_{01} (f\ a_0) (f\ a_1) \} \]

\[ f \leq g := (a : A) \rightarrow f\ a \leq^{B\ a} g\ a \]
Models for Gradual Types
Mixing orders and exceptions

Required ingredients for a Gradual model:

- Types $X$ endowed with a *precision* preorder $\sqsubseteq^X$
- Universal placeholders $?_X$ such that $\forall x : X, x \sqsubseteq^X ?_X$
- Errors $\text{raise}^X_X$ such that $\forall x : X, \text{raise}^X_X \sqsubseteq^X x$
- Whenever $X \sqsubseteq^U Y$, a pair of an upcast $\uparrow : X \to Y$ and a downcast $\downarrow : Y \to X$ forming an ep-pair $(\uparrow, \downarrow) : X \triangleleft Y$
Mixing orders and exceptions

Required ingredients for a Gradual model:

- Types $X$ endowed with a *precision* preorder $\sqsubseteq^X$
- Universal placeholders $?_X$ such that $\forall x : X, x \sqsubseteq^X ?_X$
- Errors $\text{raise}_X$ such that $\forall x : X, \text{raise}_X \sqsubseteq^X x$
- Whenever $X \sqsubseteq^U Y$, a pair of an upcast $\uparrow : X \to Y$ and a
downcast $\downarrow : Y \to X$ forming an ep-pair $(\uparrow, \downarrow) : X \triangleleft Y$

Natural numbers

\[
\begin{align*}
\vdash 0 : \mathbb{N} & \quad \vdash S : \mathbb{N} \to \mathbb{N} & \quad \vdash ?_\mathbb{N} : \mathbb{N} & \quad \vdash \text{raise}_\mathbb{N} : \mathbb{N} \\
0 \sqsubseteq^\mathbb{N} 0 & \quad p \sqsubseteq^\mathbb{N} q & \quad \text{raise}_\mathbb{N} \sqsubseteq^\mathbb{N} p & \quad 0, ?_\mathbb{N} \sqsubseteq ?_\mathbb{N} & \quad p \sqsubseteq ?_\mathbb{N} \\
S & \quad p \sqsubseteq S \quad q & \quad \text{raise}_\mathbb{N} \sqsubseteq^\mathbb{N} p & \quad S & \quad p \sqsubseteq S \quad q
\end{align*}
\]
An Inductive-recursive hierarchy of Universes

Key ideas

1. Universes and their precision order must be defined mutually

\[ \vdash A : \mathbb{U}_i \quad \vdash B : A \xrightarrow{\text{mon}} \mathbb{U}_i \]

\[ \vdash (a : A) \xrightarrow{\text{mon}} B a : \mathbb{U}_i \]
Key ideas

1. Universes and their precision order must be defined mutually

\[
\vdash A : \mathbb{U}_i \quad \vdash B : A \xrightarrow{\text{mon}} \mathbb{U}_i \\
\vdash (a : A) \xrightarrow{\text{mon}} B a : \mathbb{U}_i
\]

2. \(X \sqsubseteq^\mathbb{U} Y\) irrelevant requires \textit{intensional} data on types
An Inductive-recursive hierarchy of Universes

Key ideas

1. Universes and their precision order must be defined mutually

\[
\begin{align*}
\vdash A : \mathbb{U}_i & \quad \vdash B : A \xrightarrow{\text{mon}} \mathbb{U}_i \\
\vdash \pi A B : \mathbb{U}_i & \quad \El (\pi A B) := (a : A) \xrightarrow{\text{mon}} B a
\end{align*}
\]

2. \( X \sqsubseteq^\mathbb{U} Y \) irrelevant requires \textit{intensional} data on types

\begin{itemize}
\item Inductive universe of codes \( \mathbb{U}_i \) and
\item Recursive decoding function \( \El : \mathbb{U}_i \to \text{Type} \)
\end{itemize}
An Inductive-recursive hierarchy of Universes

Key ideas

1. Universes and their precision order must be defined mutually

$$
\vdash A : \mathbb{U}_i \quad \vdash B : A \xrightarrow{\text{mon}} \mathbb{U}_i \\
\vdash \pi \ A \ B : \mathbb{U}_i \\
\text{El} (\pi \ A \ B) := (a : A) \xrightarrow{\text{mon}} B \ a
$$

2. $X \sqsubseteq^\mathbb{U} Y$ irrelevant requires intensional data on types

   Inductive universe of codes $\mathbb{U}_i$ and
   Recursive decoding function $\text{El} : \mathbb{U}_i \rightarrow \text{Type}$

3. Precision on codes decodes to embedding-projection pairs

   $$\text{El}^{\text{rel}} : \quad X \sqsubseteq Y \quad \rightarrow \quad X \vartriangleleft Y$$

   $\leadsto$ induces casts $\uparrow, \downarrow$ between types
$\omega$-cpo and the Scott model

$\mu : \mathcal{U}$  
(by def)
ω-cpo and the Scott model

\[ \forall U : U \]
\[ \forall U \rightarrow \forall U : U \quad \text{(by def)} \]
\[ (U \text{ closed under } \rightarrow) \]
\( \omega \text{-cpo and the Scott model} \)

\[
\begin{align*}
\text{by def} & \quad \therefore U : U \\
\therefore U & \rightarrow \therefore U : U \\
\therefore U & \rightarrow \therefore U \sqsubseteq \therefore U \\
\text{by decoding} & \quad \therefore U \text{ hosts a model of pure } \lambda \text{-calculus}
\end{align*}
\]

Let's go back to Scott's domain theory.

Add an \( \omega \text{-cpo} \) structure on a type \( A \):

\[
\sup A : (\omega \rightarrow A) \rightarrow A
\]

\( \therefore U \text{ closed under } \rightarrow \)

\( \therefore U \text{ maximal for } \sqsubseteq \)
\( \omega \)-cpo and the Scott model

\[
\begin{align*}
?_U &: U \\
?_U &\rightarrow ?_U : U \\
?_U &\rightarrow ?_U \sqsubseteq ?_U \\
?_U &\rightarrow ?_U \bowtie ?_U
\end{align*}
\]

(by def)

(\( U \) closed under \( \rightarrow \))

(\( ? \) maximal for \( \sqsubseteq \))

(by decoding)

\( ?_U \) hosts a model of pure \( \lambda \)-calculus
ω-cpo and the Scott model

\[ ?_U : U \]
\[ ?_U \rightarrow ?_U : U \]  \hspace{1cm} \text{(by def)}
\[ ?_U \rightarrow ?_U \sqsubseteq ?_U \]  \hspace{1cm} \text{(? maximal for \( \sqsubseteq \))}
\[ ?_U \rightarrow ?_U \smallfrown ?_U \]  \hspace{1cm} \text{(by decoding)}

\( ?_U \) hosts a model of pure \( \lambda \)-calculus

Let’s go back to Scott’s domain theory

Add an \( \omega \)-cpo structure on a type \( A \):

\[ \text{sup}^A : (\omega \xrightarrow{\text{mon}} A) \rightarrow A \]
Dynamic type $?_U$ as a sequential colimit

\[
\begin{align*}
\perp & \xrightarrow{\triangleleft} F \perp \xrightarrow{\triangleleft} \ldots \xrightarrow{\triangleleft} F^n \perp \xrightarrow{\rightarrow} \ldots \\
\xrightarrow{\rightarrow} \text{colim}_{n \in \omega} F^n \perp = ?_U
\end{align*}
\]

where

\[
FX \equiv \mathbb{N} + X \to X + \ldots
\]
Dynamic type $\mathcal{U}$ as a sequential colimit

\[
\begin{align*}
\bot & \xrightarrow{\triangleleft} F \bot \xrightarrow{\triangleleft} \ldots \xrightarrow{\triangleleft} F^n \bot \xrightarrow{\rightarrow} \ldots \\
& \Rightarrow \quad \text{colim}_{n \in \omega} F^n \bot = \mathcal{U}
\end{align*}
\]

where

\[
F \times \cong \mathbb{N} + X \rightarrow X + \ldots
\]

What’s a typical element of $\mathcal{U}$

- a tag corresponding to a summand of $F$, e.g. $\rightarrow$
- and an element of the corresponding type, e.g. $\mathcal{U} \rightarrow \mathcal{U}$

$A \subseteq \mathcal{U}$ decomposes elements along the structure of $A$!
Recap

- CIC is a subtle equilibrium
- ... and I passed over many important details
  (impredicativity, indexed types, induction-recursion)
- Syntactic models can help prototyping extensions
- Even simple objects (orders) give rise to a whole spectrum

Further directions

- Study these models systematically
- As well as how they relate!
- Design full-fledge type theories (hard!)