A Proof of Strong Normalization For the Theory of Constructions Using a Kripke-Like Interpretation[1]

Thierry Coquand *Jean Gallier*[2]

INRIA
Domaine de Voluceau, Rocquencourt
B.P. 105
78153 Le Chesnay Cedex, France

Department of Computer and Information Science
University of Pennsylvania
200 South 33rd St.
Philadelphia, PA 19104, USA

Abstract. We give a proof that all terms that type-check in the theory of constructions are strongly normalizing (under β-reduction). The main novelty of this proof is that it uses a "Kripke-like" interpretation of the types and kinds, and that it does not use infinite contexts. We explore some consequences of strong normalization, consistency and decidability of type-checking. We also show that our proof yields another proof of strong normalization for LF (under β-reduction), using the reducibility method.

1 Introduction

We give a proof that all terms that type-check in the theory of constructions are strongly normalizing (under β-reduction). The main novelty of this proof is that it uses a "Kripke-like" interpretation of the types and kinds, and that it does not use infinite contexts. The idea used for avoiding infinite contexts comes from Coquand's thesis [Coq85]. Our proof yields as a corollary another proof of strong normalization (under β-reduction) of well-formed terms of LF. In fact, it is easy to see that this proof does not use the candidates of reducibility at all. We are unaware of similar proofs (using reducibility "à la Tait") for LF.

[1] The results in this paper were first presented at the First Annual Workshop on Logical Frameworks, Esprit Basic Research Action, Antibes, May 7-11, 1990.

[2] Partially supported by ONR Grant NOOO14-88-K-0593.

Our experience with proofs of strong normalization is that besides their intrinsic difficulty, their clarity and ease of understanding are greatly affected by the choice of notation, and the order in which the concepts are introduced. For example, it is logical to define "the values" before defining $[\![\Gamma \triangleright A]\!]\rho\Delta$, the interpretation of types, since this latter definition requires the former (the term "values" is used by Coquand [Coq87], but in Girard's terminology and ours, these are the sets $\mathcal{C}_{A,\Delta}$ of "candidates of reducibility"). However, we believe that it more intuitive and easier for understanding such proofs, if $[\![\Gamma \triangleright A]\!]\rho\Delta$ is defined *before* the families of candidates (In Coquand [Coq87], $[\![\Gamma \triangleright A]\!]\rho\Delta$ is called the *interpretation of a term*, and it is denoted as $Eval\rho\ M$). It is possible to do so by first giving a rough and intuitive idea of what families of candidates are. Another difficulty is to package in a convenient way the various ingredients making up a candidate assignment (the substitution component, the candidate assignment component, etc). This is one place where the idea of viewing a context Δ as a world, as in Kripke semantics of intuitionistic logic, seems helpful.

A key remark for our presentation is the following: proofs of normalization that follow the reducibility method are intuitionistic. Hence, it should be possible to carry them in any intuitionistic model, hence in any Kripke model. There is furthermore one natural Kripke "term model" where we take for the Kripke worlds, the valid contexts of the type system. It should be noted that this paper represents "ongoing research", and that this is a preliminary version of a paper in which we intend to explore more thoroughly the nature of Kripke models for the theory of constructions.

Among the sources of inspiration for this research, Moggi and Mitchell's work on Kripke models for the simply-typed λ-calculus [MM87] should be mentioned. We also became recently aware of work by Aarne Ranta [Ran90], in which the notion of "contexts as (finite approximation of) worlds" is used. One of the motivations for this work is to give an intuitionistic treatment of the notion of "possible worlds". Ranta's notion of Kripke structure is more general than ours, in that he does consider any *interpretation* between contexts, and not only projection (here seen dually as inclusion). It may be interesting to see if one can formulate a normalization proof in this framework.

One should be careful in referring to "the" theory of constructions, since different versions of this theory have been formulated and these versions are not all equivalent. Thus, in order to avoid ambiguities, we formulate in the next section a version of the theory of constructions equivalent (but not identical in syntax) to the version presented by Coquand and Huet [CH88]. We refer to this version of the theory of constructions as CC.

2 Syntax of the Theory of Constructions CC

We find it pedagogically convenient to first describe a theory of constructions whose syntax has three levels (*kinds*, *type families*, and *terms*). The special kind $\star$ is the logical kind of propositions. In other words, types (propositions) are exactly those type families whose kind is $\star$. In the simple theory of type, Church used the notation o for $\star$. Another notation used in place of $\star$ is *Prop* (or even *Type*). Furthermore, some authors use *Type* instead of *kind*, but we find this practice somewhat confusing, since in the Curry-Howard formula–as–type analogy, propositions correspond to types.

We begin by defining *raw terms*.

Definition 2.1 We use the nonterminal K to range over kinds, A to range over type families, and M to range over terms. We also use two kinds of variables, ranging over kinds and type families. *Raw terms* are defined by the following grammar.

$K \longrightarrow \star \mid (\Pi t\colon K)K \mid (\Pi x\colon A)K$

$A \longrightarrow t \mid (\forall t\colon K)A \mid (\forall x\colon A)A \mid (AA) \mid (AM) \mid (\Lambda t\colon K.\ A) \mid (\lambda x\colon A.\ A)$

$M \longrightarrow x \mid (MM) \mid (MA) \mid (\Lambda t\colon K.\ M) \mid (\lambda x\colon A.\ M).$

A *context* is an ordered sequence of pairs $\Delta = \langle\langle x_1, A_1\rangle, \ldots, \langle x_m, A_m\rangle\rangle$, where x_i is a variable and A_i is a kind or a type family, and $x_i \neq x_j$ whenever $i \neq j$. A context Δ is usually written as $x_1\colon A_1, \ldots, x_m\colon A_m$. If $\Delta = x_1\colon A_1, \ldots, x_m\colon A_m$, we let

$$dom(\Delta) = \{x_1, \ldots, x_m\}$$

and

$$\Delta(x_i) = A_i.$$

There are four categories of *judgments*:

Definition 2.2 *Judgments* are expressions of the form:

$\Delta \rhd$ (Δ is a valid context),

$\Delta \rhd K\colon kind$ (K is a valid kind in context Δ),

$\Delta \rhd A\colon K$ (A kind-checks with kind K in context Δ),

$\Delta \rhd M\colon A$ (M type-checks with type A in context Δ).

We define β-reduction and β-conversion in the usual manner on raw terms. This means that redexes will be of the form $(\Lambda t\colon K.\ A)B$, $(\lambda x\colon A.\ B)M$, $(\Lambda t\colon K.\ M)B$, and $(\lambda x\colon A.\ M)N$. We emphasize that we *do not* consider η-conversion in this paper. There appears to be some difficulties with the Church-Rosser theorem if $\beta\eta$-conversion is defined on raw terms, and it seems that one needs to define judgments of the form $\Delta \rhd M \overset{*}{\longleftrightarrow}_{CC} M'$ (equality judgments), which is quite cumbersome.

3 Typing Rules for CC

We could list the typing rules assuming the above syntax, but it is possible to state them more concisely if certain conventions are adopted.

- Firstly, we will not distinguish between type variables and term variables.
- Secondly, we will use the symbol κ to denote either *kind* or $\star$.
- Thirdly, we will denote both judgments $\Delta \triangleright K\colon kind$ and $\Delta \triangleright \sigma\colon \star$ as $\Delta \triangleright A\colon \kappa$, and similarly, we will denote both judgments $\Delta \triangleright \sigma\colon K$ and $\Delta \triangleright M\colon \sigma$ as $\Delta \triangleright M\colon A$.
- Finally, we identify $\forall$ and Π, and Λ and λ.

Note that now, there is only one kind of raw terms given by the following grammar:

$M \longrightarrow x \mid \star \mid (\Pi x\colon M)M \mid (MM) \mid (\lambda x\colon M.\, M)$.

With the above conventions, we only have one rule for each kind of rule.

Definition 3.1 In the rules below, $\kappa, \kappa_1, \kappa_2 \in \{\star, kind\}$.

Context Formation:

$$\emptyset \triangleright \qquad \text{empty context}$$

$$\frac{\Delta \triangleright}{\Delta \triangleright \star\colon kind}$$

$$\frac{\Delta \triangleright A\colon \kappa}{\Delta, x\colon A \triangleright} \qquad x \notin dom(\Delta)$$

Axiomatic Judgments:

$$\frac{\Delta \triangleright}{\Delta \triangleright x\colon A} \qquad x\colon A \in \Delta$$

Product Formation and Quantification:

$$\frac{\Delta \triangleright A_1\colon \kappa_1 \quad \Delta, x\colon A_1 \triangleright A_2\colon \kappa_2}{\Delta \triangleright (\Pi x\colon A_1)A_2\colon \kappa_2}$$

Abstraction:

$$\frac{\Delta \triangleright A_1\colon \kappa_1 \quad \Delta, x\colon A_1 \triangleright A_2\colon \kappa_2 \quad \Delta, x\colon A_1 \triangleright M\colon A_2}{\Delta \triangleright (\lambda x\colon A_1.\, M)\colon (\Pi x\colon A_1)A_2}$$

Application:

$$\frac{\Delta \triangleright M\colon (\Pi x\colon A_1)A_2 \quad \Delta \triangleright N\colon A_1}{\Delta \triangleright MN\colon A_2[N/x]}$$

Kind and Type Conversion:

$$\frac{\Delta \triangleright M\colon A_1 \quad \Delta \triangleright A_2\colon \kappa \quad A_1 \overset{*}{\longleftrightarrow}_{CC} A_2}{\Delta \triangleright M\colon A_2}$$

It turns out that the above typing rules can be simplified, because some of the premises are redundant. Of course, this has to be justified carefully, but this has been verified by Coquand and Huet [CH88], and others. For the reader's convenience, we recall some of the main basic properties of CC.

4 Some Basic Properties of CC

We shall use the notation $\Delta \triangleright E$ as an abbreviation for all forms of judgments. Given contexts Γ and Δ, the notation $\Gamma \leq \Delta$ means that Γ is an initial subsequence of Δ.

First, we note that under β-conversion alone, the Church-Rosser theorem holds even for raw terms.

Theorem 4.1 *(Martin Löf)*
The Church-Rosser property holds for raw terms of CC (even the economical version).

Proof. Such a proof using the so called "Tait/Martin Löf's method" was given by Martin Löf [ML72]. □

It should be noted that theorem 4.1 is quite handy. It appears that if β-conversion is defined on raw terms, which is definitely more convenient than using equality judgments, many important properties of CC make use of the Church-Rosser property.

The propositions listed below consist of the translation in English and in our terminology of properties 1-7 in Chapter 1 of Coquand's thesis [Coq85]. In some cases, these proofs require some amplification. First, we need the following definitions, which are translations in our terminology of Coquand's definitions.

Definition 4.2 K is a *kind* iff K is of the form $\star$ or $(\Pi x_1\colon A_1)\ldots(\Pi x_m\colon A_m)\star$, and $\Delta \triangleright K\colon kind$ for some context Δ;

A is a *type family* iff $\Delta \triangleright A\colon K$ for some context Δ and some kind K;

A is a *type* iff $\Delta \triangleright A\colon \star$ for some context Δ;

M is a *proof* (or *proof term*) iff $\Delta \triangleright M\colon A$ where A is not a kind.

When we want to stress that a context Δ is well-formed, that is, when $\Delta \triangleright$ is derivable, we say that Δ is a *valid context*, and similarly for kinds, type families, types, and proofs.

Lemma 4.3 *If $\Delta \triangleright E$, then $\Delta' \triangleright$ for every $\Delta' \leq \Delta$, and more generally, every derivation of $\Delta \triangleright E$ contains a derivation of $\Delta' \triangleright$ as a subderivation.*

Lemma 4.4 *If $\Delta \triangleright A\colon K$ and $\Delta \triangleright A\colon K'$ where both K and K' are kinds, then $K \overset{*}{\longleftrightarrow}_{CC} K'$. Similarly, if $\Delta \triangleright M\colon A$ and $\Delta \triangleright M\colon A'$, where both A and A' are not kinds, then $A \overset{*}{\longleftrightarrow}_{CC} A'$.*

The proof of the above lemma actually seems to require the Church-Rosser property and the following proposition.

Proposition 4.5 *Assume that $\Delta \triangleright (\Pi x\colon A)K\colon kind$ and $\Delta \triangleright (\Pi x\colon A')K'\colon kind$. If $(\Pi x\colon A)K \overset{*}{\longleftrightarrow}_{CC} (\Pi x\colon A')K'$, then $A \overset{*}{\longleftrightarrow}_{CC} A'$ and $K \overset{*}{\longleftrightarrow}_{CC} K'$.*

Proposition 4.6 *Both $\Delta \triangleright M\colon A$ and $\Delta \triangleright M\colon kind$ are not derivable at the same time.*

Definition 4.7 Given any two contexts Δ, Δ', we say that $\Delta \subseteq \Delta'$ iff for every x, if $x \in dom(\Delta)$ then $x \in dom(\Delta')$ and $\Delta(x) = \Delta'(x)$.

Lemma 4.8 *Assume that $\Delta \triangleright$, $\Delta' \triangleright$, and $\Delta \subseteq \Delta'$. If $\Delta \triangleright E$, then $\Delta' \triangleright E$. In particular, if $\Delta \leq \Delta'$, $\Delta' \triangleright$, and $\Delta \triangleright E$, then $\Delta' \triangleright E$.*

Lemma 4.9 *If $\Delta \triangleright M\colon A$ and $\Delta, x\colon A, \Delta' \triangleright E$, then $\Delta, \Delta'[M/x] \triangleright E[M/x]$.*

Lemma 4.10 *If $\Delta \triangleright M\colon A$ and A is not a kind, then $\Delta \triangleright A\colon \star$. If $\Delta \triangleright M\colon A$ and A is a kind, then $\Delta \triangleright A\colon kind$.*

Lemma 4.11 *If $\Delta, x\colon A, \Delta' \triangleright E$, $A \overset{*}{\longleftrightarrow}_{CC} A'$, and either $\Delta \triangleright A'\colon \star$ or $\Delta \triangleright A'\colon kind$, then $\Delta, x\colon A', \Delta' \triangleright E$.*

Lemma 4.12 *If $\Delta \triangleright M\colon A$ and $M \overset{*}{\longrightarrow}_{CC} N$, then $\Delta \triangleright N\colon A$.*

Lemma 4.13 *The judgments $\Delta \triangleright M\colon A$ where A is a type and $\Delta \triangleright M\colon K$ where K is a kind cannot hold simultaneously.*

The proof of the above lemma seems to require the Church-Rosser property.

In view of proposition 4.6, kinds are disjoint from type families and proofs. In view of lemma 4.13, proofs and type families are disjoint.

5 Strong Normalization in CC

The proof of strong normalization for well-typed terms of CC is obtained by generalizing the proof given by Girard for the system F_ω [Gir72], as presented in Gallier [Gal90]. However, there are some significant technical complications. In F_ω, we have an ascending hierarchy, kinds, type families, and terms, where kinds do not depend on type families or terms, and type families do not depend on terms. However, in CC, kinds, type families, and terms, are defined in a single big simultaneous inductive construction. The main difficulty is to ensure that the interpretations $[\![\Gamma \rhd A]\!]\rho\Delta$ are nondegenerate (i.e., nonempty sets).

The first step is to define the concept of a candidate assignment, which packages together a substitution and a valuation assigning candidates to variables.

Definition 5.1 A *substitution* is a function $\varphi\colon \mathcal{V} \to \mathit{Terms}$ such that $\varphi(x) \neq x$ for only finitely many x, and for every $\varphi(x)$, there is some context Δ and either some type A such that $\Delta \rhd \varphi(x)\colon A$ (and $\Delta \rhd A\colon \star$), or some kind K such that $\Delta \rhd \varphi(x)\colon K$ (and $\Delta \rhd K\colon \mathit{kind}$). The domain $D(\varphi)$ of φ is the set $D(\varphi) = \{x \mid \varphi(x) \neq x\}$.

Every substitution φ has a unique homomorphic extension $\widehat{\varphi}\colon \mathit{Terms} \to \mathit{Terms}$. Given a term M (term, type family, or kind), the result of applying φ to M is $\widehat{\varphi}(M)$, and it is denoted as $\varphi(M)$ or $M[\varphi]$.

Some form of Kripke structure is lurking around. Contexts are going to play the role of worlds. Consequently, most concepts will be defined "in world Δ". The notion of inclusion of worlds is the relation $\subseteq$ defined in definition 4.7. Substitutions will also play the role of valuations assigning values to variables. This motivates the following definition.

Definition 5.2 Given two valid contexts Γ, Δ, where Γ is used to type/kind check, and Δ acts as a world, given a substitution φ, we say that $\Gamma[\varphi]$ *type-checks in* Δ iff $\Delta \rhd x[\varphi]\colon \Gamma(x)[\varphi]$ for every $x \in \mathit{dom}(\Gamma)$.

At first glance, one may be concerned that this condition is circular. However, this is not so. Indeed, if $\Gamma = x_1\colon A_1, \ldots, x_m\colon A_m$ is a valid context, it can be easily shown that $FV(A_i) \subseteq \{x_1, \ldots, x_{i-1}\}$ for all i, $1 \leq i \leq m$, and that $\Gamma[\varphi]$ type-checks in Δ means that $\Delta \rhd x_i[\varphi]\colon A_i[x_1[\varphi]/x_1, \ldots, x_{i-1}[\varphi]/x_{i-1}]$ for all i, $1 \leq i \leq m$, which is possible.

We now assume that for every world Δ and type family A that kind-checks in Δ, we have a set $\mathcal{C}_{A,\Delta}$ of nonempty sets called *candidates* to be defined soon. All we need to know is that, when A is a type, every $C \in \mathcal{C}_{A,\Delta}$ is a set of terms $\Delta' \rhd M$ such that $\Delta' \rhd M\colon A$ for some $\Delta' \supseteq \Delta$, and when A kind-checks with kind $(\Pi x\colon B)K$, every element of $\mathcal{C}_{A,\Delta}$ is a certain function. We also have a set $\mathcal{C}_{\star,\Delta}$ consisting of nonempty sets of types $\Delta' \rhd A$ such that $\Delta' \rhd A\colon \star$ for some $\Delta' \supseteq \Delta$, and a set $\mathcal{C}_{\mathit{kind},\Delta}$ consisting of nonempty sets of kinds $\Delta' \rhd K$ such that $\Delta' \rhd K\colon \mathit{kind}$ for some $\Delta' \supseteq \Delta$.

Definition 5.3 A *candidate assignment* is any function ρ from $\mathcal{V} \cup \{\star, \mathit{kind}\}$ to $\mathit{Terms} \cup (\mathit{Terms} \times \bigcup \mathcal{C})$, such that the following properties hold:

(1) If we define the function $\rho_s : \mathcal{V} \to \mathit{Terms}$ such that,

$$\rho_s(x) = \begin{cases} A & \text{if } \rho(x) = \langle A, C \rangle, \\ A & \text{if } \rho(x) = A, \end{cases}$$

then ρ_s is a substitution (which means that $\rho_s(x) \neq x$ only for finitely many $x \in \mathcal{V}$), and,

(2) If $\rho(x) = \langle A, C \rangle$, then A is a type-family that kind-checks in some context Δ and $C \in \mathcal{C}_{A,\Delta}$, else if $\rho(x) = A$ then A is a term (proof) that type-checks in some context Δ;

(3) $\rho(\star) = \langle \star, C \rangle$ where $C \in \mathcal{C}_{\star,\Delta}$ for some context Δ, and $\rho(\mathit{kind}) = \langle \mathit{kind}, C \rangle$ where $C \in \mathcal{C}_{\mathit{kind},\Delta}$ for some context Δ.

The function ρ_s is called the *substitution part of* ρ. We also denote ρ_s as $[\rho]$. When the range of ρ is $\mathit{Terms} \times \bigcup \mathcal{C}$, the function ρ also defines the function ρ_c such that $x \mapsto C$, $\star \mapsto C$, and $\mathit{kind} \mapsto C$. The function ρ_c is called the *candidate part of* ρ. By abuse of notation, both ρ_s and ρ_c are often denoted as ρ, when the context makes it clear which is referred to. The notation $\rho[x := \langle A, C \rangle]$ is used to denote the modified candidate assignment which agrees with ρ for every variable different from x, and maps x to $\langle A, C \rangle$ (possibly overwriting the previous value $\rho(x)$), and similarly for $\rho[x := N]$.

Definition 5.4 A candidate assignment ρ *satisfies* Γ *at* Δ iff

(1) $\Gamma[\rho_s]$ type-checks in Δ.

(2) Whenever $\rho(x) = \langle A, C \rangle$ or $\rho(x) = A$, then A kind/type-checks in Δ. Furthermore, $C \in \mathcal{C}_{A,\Delta}$ when $\rho(x) = \langle A, C \rangle$, $C \in \mathcal{C}_{\star,\Delta}$ when $\rho(\star) = \langle \star, C \rangle$, and $C \in \mathcal{C}_{\mathit{kind},\Delta}$ when $\rho(\mathit{kind}) = \langle \mathit{kind}, C \rangle$.

It is easy to verify that if $\Delta \subseteq \Delta'$ and ρ satisfies Γ at Δ, then ρ satisfies Γ at Δ'. We can now define $[\![\Gamma \triangleright A]\!]\rho\Delta$, where Γ is a context, either A is a type family that kind-ckecks in Γ, or A is a kind valid in Γ, or $A = \mathit{kind}$, ρ is a candidate assignment, and Δ is a context viewed as a world. The expression $[\![\Gamma \triangleright A]\!]\rho\Delta$ is the *interpretation of* $\Gamma \triangleright A$ with respect to the candidate assignment ρ, in the world Δ. The definition is by induction on the complexity of $\Gamma \triangleright A$ (if $\Gamma = x_1 : A_1, \ldots, x_m : A_m$, then the complexity of $\Gamma \triangleright A$ is the sum of the sizes of $A_1, \ldots, A_m, A$). It only makes sense when ρ satisfies Γ at Δ.

Definition 5.5 In the clauses below, K stands for a kind, σ for a type, A, B for type families, D for a kind or a type, M for a type family or a term (proof), and N for a term (proof).

$$\begin{aligned}
[\![\Gamma \triangleright kind]\!]\rho\Delta &= \rho_c(kind),\\
[\![\Gamma \triangleright \star]\!]\rho\Delta &= \rho_c(\star),\\
[\![\Gamma \triangleright x]\!]\rho\Delta &= \rho_c(x),\\
[\![\Gamma \triangleright AB]\!]\rho\Delta &= [\![\Gamma \triangleright A]\!]\rho\Delta(\Delta \triangleright B[\rho_s],\ [\![\Gamma \triangleright B]\!]\rho\Delta),\\
[\![\Gamma \triangleright AN]\!]\rho\Delta &= [\![\Gamma \triangleright A]\!]\rho\Delta(\Delta \triangleright N[\rho_s]),\\
[\![\Gamma \triangleright (\Pi x{:}\, K)D]\!]\rho\Delta &= \{\Delta' \triangleright M \mid \Delta' \triangleright M{:}\, ((\Pi x{:}\, K)D)[\rho_s],\ \Delta' \supseteq \Delta,\ \text{and } \forall\Delta''\forall A\forall C\\
&\qquad (\text{if } \Delta'' \supseteq \Delta', \Delta'' \triangleright A \in [\![\Gamma \triangleright K]\!]\rho\Delta'',\ \text{and } C \in \mathcal{C}_{A,\Delta''},\ \text{then}\\
&\qquad \Delta'' \triangleright (MA) \in [\![\Gamma, x{:}\, K \triangleright D]\!]\rho[x{:}= \langle A, C\rangle]\Delta'')\},\\
[\![\Gamma \triangleright (\Pi x{:}\, \sigma)D]\!]\rho\Delta &= \{\Delta' \triangleright M \mid \Delta' \triangleright M{:}\, ((\Pi x{:}\, \sigma)D)[\rho_s],\ \Delta' \supseteq \Delta,\ \text{and } \forall\Delta''\forall N\\
&\qquad (\text{if } \Delta'' \supseteq \Delta' \text{ and } \Delta'' \triangleright N \in [\![\Gamma \triangleright \sigma]\!]\rho\Delta'',\ \text{then}\\
&\qquad \Delta'' \triangleright (MN) \in [\![\Gamma, x{:}\, \sigma \triangleright D]\!]\rho[x{:}= N]\Delta'')\},\\
[\![\Gamma \triangleright \lambda x{:}\, K.\ B]\!]\rho\Delta &= \lambda(\Delta' \triangleright A)\lambda C.\ [\![\Gamma, x{:}\, K \triangleright B]\!]\rho[x{:}= \langle A, C\rangle]\Delta',\\
&\qquad \text{a function with domain}\\
&\qquad \{\langle \Delta' \triangleright A, C\rangle \mid \Delta' \triangleright A{:}\, K[\rho_s],\ \Delta' \supseteq \Delta,\ C \in \mathcal{C}_{A,\Delta'}\},\\
[\![\Gamma \triangleright \lambda x{:}\, \sigma.\ B]\!]\rho\Delta &= \lambda(\Delta' \triangleright N).\ [\![\Gamma, x{:}\, \sigma \triangleright B]\!]\rho[x{:}= N]\Delta',\\
&\qquad \text{a function with domain}\\
&\qquad \{\Delta' \triangleright N \mid \Delta' \triangleright N{:}\, \sigma[\rho_s],\ \Delta' \supseteq \Delta\}.
\end{aligned}$$

We emphasize again the fact that in $[\![\Gamma \triangleright x]\!]\rho\Delta$, we have $\Gamma \triangleright x{:}\, K$ for some kind K, i.e., x is a type variable.

Definition 5.6 Given any judgment $\Gamma \triangleright M{:}\, A$ (where A can even be *kind*), given any candidate assignment ρ, and any context Δ viewed as a world, we write

$$\Delta \Vdash \Gamma[\rho]$$

iff

(1a) ρ satisfies Γ at Δ, and

(2a) $\Delta \triangleright x[\rho] \in [\![\Gamma \triangleright \Gamma(x)]\!]\rho\Delta$ for every $x \in dom(\Gamma)$.

We will also write

$$\Delta \Vdash (\Gamma \triangleright M{:}\, A)[\rho]$$

iff

(1b) ρ satisfies Γ at Δ, and

(2b) $\Delta \rhd M[\rho] \in [\![\Gamma \rhd A]\!]\rho\Delta$.

Then, the main theorem reads as follows:

Whenever $\Gamma \rhd M\colon A$ and $\Delta \Vdash \Gamma[\rho]$, then $\Delta \Vdash (\Gamma \rhd M\colon A)[\rho]$.

This looks like a Kripke-style type soundness result.

Actually, it is not obvious that the inductive definition of $[\![\Gamma \rhd A]\!]\rho\Delta$ defines nonempty sets and total functions, and this depends on some properties of the sets $\mathcal{C}_{A,\Delta}$. One of the crucial facts is that for every valid context Δ and type or kind A, there is some term or type family $\Delta' \rhd M$ with $\Delta' \supseteq \Delta$ such that $\Delta' \rhd M\colon A$. Indeed $\Delta' = \Delta, x\colon A$ where $x \notin \mathit{dom}(\Delta)$ does the job, since $\Delta, x\colon A \rhd x\colon A$ is derivable.

We can now define the sets $\mathcal{C}_{A,\Delta}$. For this this, we need a complexity measure for types and kinds.

Definition 5.7 Let A be any valid type, and K any valid kind. We define $c(A)$ and $c(K)$ inductively as follows:

$$c(A) = 0,$$

$$c(K) = \begin{cases} 1 & \text{if } K = \star, \\ \mathit{max}(c(B), c(D)) + 1 & \text{if } K = (\Pi x\colon B)D. \end{cases}$$

It is easily verified that if $K \overset{*}{\longleftrightarrow}_{CC} K'$, then $c(K) = c(K')$. The main property of this complexity measure is that it is invariant under substitution.

Lemma 5.8 *For every type family or term M, for every kind K, $c(K[M/x]) = c(K)$.*

Proof. We proceed by induction on the structure of K. If $K = \star$, the lemma holds since $\star[M/x] = \star$. If $K = (\Pi x\colon B)D$, there are two cases. If B is also a kind, by the induction hypothesis, $c(B[M/x]) = c(B)$, $c(D[M/x]) = c(D)$, and the lemma holds since $K[M/x] = (\Pi x\colon B[M/x])D[M/x]$. If B is a type, then $B[M/x]$ is also a type, and since $c(B[M/x]) = 0$ and by the induction hypothesis $c(D[M/x]) = c(D)$, the lemma holds. $\square$

We also let $c(\mathit{kind}) = 0$. The sets $\mathcal{C}_{A,\Delta}$ are defined by induction on $c(K)$, where $\Delta \rhd A\colon K$. Since $c(K)$ only depends on the equivalence class of K modulo β-conversion, this definition is proper. The definition of the sets $\mathcal{C}_{A,\Delta}$ given next is a bit more general than really required for proving strong normalization. The reason for giving it in this form is that it can be used to extend our proof to other properties besides strong normalization. This definition also contains all the closure conditions that will come up during the proof of the main result.

Definition 5.9 The family $\mathcal{C}$ of sets $\mathcal{C}_{A,\Delta}$ where A is a kind or a type family valid in the context Δ, is defined by the properties listed below. It is called the *family of saturated sets*.

1. $\mathcal{C}_{kind,\Delta}$ is the set of sets C, such that, each C is a nonempty set of strongly normalizing kinds $\Delta' \triangleright K$, with $\Delta' \supseteq \Delta$, and the following properties hold:

(a) $\Delta' \triangleright \star \in C$ for all $\Delta' \supseteq \Delta$.

(b) For every kind $\Delta' \triangleright (\Pi x{:}\, K)D$, with $\Delta' \supseteq \Delta$ and K a kind, if $\Delta' \triangleright K \in C$ and $\Delta', x{:}\, K \triangleright D \in C$, then $\Delta' \triangleright (\Pi x{:}\, K)D \in C$.

(c) For every kind $\Delta' \triangleright (\Pi x{:}\, \sigma)D$, with $\Delta' \supseteq \Delta$ and σ a type, for every $C' \in \mathcal{C}_{\star,\Delta}$, if $\Delta' \triangleright \sigma \in C'$ and $\Delta', x{:}\, \sigma \triangleright D \in C$, then $\Delta' \triangleright (\Pi x{:}\, \sigma)D \in C$.

(d) Whenever $\Delta' \triangleright K \in C$ and $\Delta' \subseteq \Delta''$, then $\Delta'' \triangleright K \in C$.

2. $\mathcal{C}_{\star,\Delta}$ is the set of sets C, such that, each C is a nonempty set of strongly normalizing types $\Delta' \triangleright A$, with $\Delta' \supseteq \Delta$, and the following properties hold:

(S0) For every type $\Delta' \triangleright (\Pi x{:}\, K)A$, with $\Delta' \supseteq \Delta$ and K a kind, for every $C' \in \mathcal{C}_{kind,\Delta}$, if $\Delta' \triangleright K \in C'$ and $\Delta', x{:}\, K \triangleright A \in C$, then $\Delta' \triangleright (\Pi x{:}\, K)A \in C$, and for every type $\Delta' \triangleright (\Pi x{:}\, \sigma)A$, with $\Delta' \supseteq \Delta$ and σ a type, if $\Delta' \triangleright \sigma \in C$ and $\Delta', x{:}\, \sigma \triangleright A \in C$, then $\Delta' \triangleright (\Pi x{:}\, \sigma)A \in C$.

(S1) For every variable x, if $\Delta' \triangleright xN_1 \ldots N_m{:}\, \star$ for some $\Delta' \supseteq \Delta$ and $N_1, \ldots, N_m$ are SN, then $\Delta' \triangleright xN_1 \ldots N_m \in C$.

(S2) Whenever $\Delta' \triangleright M[N/x]N_1 \ldots N_m{:}\, \star$ and $\Delta' \triangleright N{:}\, B$ is SN for some $\Delta' \supseteq \Delta$, if $\Delta' \triangleright M[N/x]N_1 \ldots N_m \in C$, then $\Delta' \triangleright (\lambda x{:}\, B.\, M)NN_1 \ldots N_m \in C$.

(S3) Whenever $\Delta' \triangleright A \in C$ and $\Delta' \subseteq \Delta''$, then $\Delta'' \triangleright A \in C$.

3. When A is a type (and $\Delta \triangleright A{:}\, \star$), $\mathcal{C}_{A,\Delta}$ is the set of sets C, such that, each C is a nonempty set of strongly normalizing terms $\Delta' \triangleright M$ such that $\Delta' \triangleright M{:}\, A$ for some $\Delta' \supseteq \Delta$, and the following properties hold:

(S1) For every variable x, if $\Delta' \triangleright xN_1 \ldots N_m{:}\, A$ for some $\Delta' \supseteq \Delta$ and $N_1, \ldots, N_m$ are SN, then $\Delta' \triangleright xN_1 \ldots N_m \in C$.

(S2) Whenever $\Delta' \triangleright M[N/x]N_1 \ldots N_m{:}\, A$ and $\Delta' \triangleright N{:}\, B$ is SN for some $\Delta' \supseteq \Delta$, if $\Delta' \triangleright M[N/x]N_1 \ldots N_m \in C$, then $\Delta' \triangleright (\lambda x{:}\, B.\, M)NN_1 \ldots N_m \in C$.

(S3) Whenever $\Delta' \triangleright M \in C$ and $\Delta' \subseteq \Delta''$, then $\Delta'' \triangleright M \in C$.

4. When A is a type family such that $\Delta \triangleright A{:}\, (\Pi x{:}\, B)D$ (and $\Delta \triangleright (\Pi x{:}\, B)D{:}\, kind$), $\mathcal{C}_{A,\Delta}$ is the set of functions with the following properties:

(a) If B is a kind, then

- $f \in \mathcal{C}_{A,\Delta}$ is a function with domain

$$\{\langle \Delta' \rhd M, C \rangle) \mid \Delta' \rhd M \colon B,\ \Delta' \supseteq \Delta,\ C \in \mathcal{C}_{M,\Delta'}\}$$

such that $f(\Delta' \rhd M, C) \in \mathcal{C}_{AM,\Delta'}$, and

- $f(\Delta' \rhd M_1, C) = f(\Delta' \rhd M_2, C)$ whenever $M_1 \overset{*}{\longleftrightarrow}_{CC} M_2$.

(b) If B is a type, then

- $f \in \mathcal{C}_{A,\Delta}$ is a function with domain $\{\Delta' \rhd N \mid \Delta' \rhd N \colon B,\ \Delta' \supseteq \Delta\}$ such that $f(\Delta' \rhd N) \in \mathcal{C}_{AN,\Delta'}$, and
- $f(\Delta' \rhd N_1) = f(\Delta' \rhd N_2)$ whenever $N_1 \overset{*}{\longleftrightarrow}_{CC} N_2$.

Note that this definition is proper, because we can prove that the sets $\mathcal{C}_{M,\Delta'}$, $\mathcal{C}_{AM,\Delta'}$, and $\mathcal{C}_{AN,\Delta'}$, needed in (4) are well defined, where $\Delta \rhd A \colon (\Pi x \colon B)D$, $\Delta' \rhd M \colon B$, and $\Delta' \rhd N \colon B$ with $\Delta' \supseteq \Delta$. This is correct, since $\Delta' \rhd AM \colon D[M/x]$, $\Delta' \rhd AN \colon D[N/x]$, $c(B) < c((\Pi x \colon B)D)$, and by lemma 5.8, $c(D[M/x]) = c(D) < c((\Pi x \colon B)D)$, and $c(D[N/x]) = c(D) < c((\Pi x \colon B)D)$. One can also easily prove that if $A \overset{*}{\longleftrightarrow}_{CC} A'$, then $\mathcal{C}_{A,\Delta} = \mathcal{C}_{A',\Delta}$.

Given a type family A such that $\Delta \rhd A \colon K$, we can prove by induction on $c(K)$ that each $\mathcal{C}_{A,\Delta}$ is nonempty.

Lemma 5.10 *Whenever A kind-checks in Δ, $\mathcal{C}_{A,\Delta}$ is nonempty.*

Proof. We define an element $can_{A,\Delta}$ of $\mathcal{C}_{A,\Delta}$ where $\Delta \rhd A \colon K$ such that $A \overset{*}{\longleftrightarrow}_{CC} A'$ implies that $can_{A,\Delta} = can_{A',\Delta}$, by induction on $c(K)$. We call $can_{A,\Delta}$ the *canonical member* of $\mathcal{C}_{A,\Delta}$.

When $A = \mathit{kind}$, note that the set $can_{\mathit{kind},\Delta}$ of strongly normalizing kinds of the form $\Delta' \rhd K$ for some $\Delta' \supseteq \Delta$ is nonempty, since $\Delta' \rhd \star \colon \mathit{kind}$ for every Δ', and it is obvious that (a), (b), (c), and (d), are also satisfied.

When $A = \star$, note that the set $can_{\star,\Delta}$ of strongly normalizing types of the form $\Delta' \rhd N$ for some $\Delta' \supseteq \Delta$ is nonempty, since $\Delta, x \colon \star \rhd x \colon \star$ for $x \notin \mathit{dom}(\Delta)$. Properties (S0), (S1), (S2), and (S3), are also easily verified.

When A is a type, note that the set $can_{A,\Delta}$ of strongly normalizing terms of the form $\Delta' \rhd N$ such that $\Delta' \rhd N \colon A$ for some $\Delta' \supseteq \Delta$ is nonempty, since $\Delta, x \colon A \rhd x \colon A$ for $x \notin \mathit{dom}(\Delta)$. Properties (S1), (S2), and (S3), are also easily verified. That $A \overset{*}{\longleftrightarrow}_{CC} A'$ implies $can_{A,\Delta} = can_{A',\Delta}$ follows from the fact that $\Delta' \rhd N \colon A$ and $A \overset{*}{\longleftrightarrow}_{CC} A'$ implies that $\Delta' \rhd N \colon A'$.

When $\Delta \triangleright A\colon (\Pi x\colon B)D$, we define the function $can_{A,\Delta}$ as follows. By the induction hypothesis, for every M such that $\Delta' \triangleright M\colon B$ for some $\Delta' \supseteq \Delta$, $can_{M,\Delta'}$ is defined. We define $can_{A,\Delta}$ such that $can_{A,\Delta}(\Delta' \triangleright M, C) = can_{AM,\Delta'}$, and $can_{A,\Delta}(\Delta' \triangleright M) = can_{AM,\Delta'}$ if B is a type. If $A \overset{*}{\longleftrightarrow}_{CC} A'$, then $AM \overset{*}{\longleftrightarrow}_{CC} A'M$, and this implies $can_{AM,\Delta'} = can_{A'M,\Delta'}$ by the induction hypothesis. $\Box$

Remark: It will be observed later that for proving strong normalization, we can simply define $\mathcal{C}_{kind,\Delta}$ and $\mathcal{C}_{\star,\Delta}$ as the singleton families $\mathcal{C}_{kind,\Delta} = \{can_{kind,\Delta}\}$ and $\mathcal{C}_{\star,\Delta} = \{can_{\star,\Delta}\}$.

In order to show that the closure properties of the family $\mathcal{C}$ insure that the sets $[\![\Gamma \triangleright A]\!]\rho\Delta$ are also in $\mathcal{C}$, we need the following technical lemma.

Lemma 5.11 *If $\mathcal{C}$ is the family of saturated sets, for any two ρ and ρ' satisfying Γ at Δ, if $x[\rho] \overset{*}{\longleftrightarrow}_{CC} x[\rho']$ for every $x \in dom(\Gamma)$, then $[\![\Gamma \triangleright A]\!]\rho\Delta = [\![\Gamma \triangleright A]\!]\rho'\Delta$.*

Proof. A fairly simple induction on the size of A. $\Box$

Now, we can prove that $\mathcal{C}$ contains the sets $[\![\Gamma \triangleright A]\!]\rho\Delta$.

Lemma 5.12 *If $\mathcal{C}$ is the family of saturated sets, whenever ρ satisfies Γ at Δ, then $[\![\Gamma \triangleright A]\!]\rho\Delta \in \mathcal{C}_{A[\rho],\Delta}$.*

Proof. One proceeds by induction on the size of A, also adding to the induction hypothesis the fact proved in lemma 5.11 that for any two ρ and ρ' satisfying Γ at Δ, if $x[\rho] \overset{*}{\longleftrightarrow}_{CC} x[\rho']$ for every $x \in dom(\Gamma)$, then $[\![\Gamma \triangleright A]\!]\rho\Delta = [\![\Gamma \triangleright A]\!]\rho'\Delta$. $\Box$

Given two valid contexts $\Gamma = x_1\colon A_1, \ldots, x_m\colon A_m$ and $\Gamma' = x_1\colon A'_1, \ldots, x_m\colon A'_m$, we say that $\Gamma \overset{*}{\longleftrightarrow}_{CC} \Gamma'$ iff $A_i \overset{*}{\longleftrightarrow}_{CC} A'_i$ for all i, $1 \leq i \leq m$.

Lemma 5.13 *If $\mathcal{C}$ is the family of saturated sets, whenever ρ satisfies Γ and Γ' at Δ and $\Gamma \overset{*}{\longleftrightarrow}_{CC} \Gamma'$, then $[\![\Gamma \triangleright A]\!]\rho\Delta = [\![\Gamma' \triangleright A]\!]\rho\Delta$.*

Proof. A fairly simple induction on the size of A. $\Box$

We also have the following technical property known as "substitution property". This is perhaps the lemma whose proof is the most technical.

Lemma 5.14 *If $\mathcal{C}$ is the family of saturated sets, and ρ satisfies Γ at Δ, if $\Gamma, x\colon K \triangleright A\colon B$ for some B, and $\Gamma \triangleright D\colon K$ where K is a kind, then*

$$[\![\Gamma \triangleright A[D/x]]\!]\rho\Delta = [\![\Gamma, x\colon K \triangleright A]\!]\rho[x\colon= \langle D[\rho], [\![\Gamma \triangleright D]\!]\rho\Delta\rangle]\Delta,$$

and if $\Gamma, x\colon \sigma \triangleright A\colon B$ for some B, and $\Gamma \triangleright M\colon \sigma$ where σ is a type, then

$$[\![\Gamma \triangleright A[M/x]]\!]\rho\Delta = [\![\Gamma, x\colon \sigma \triangleright A]\!]\rho[x\colon= M[\rho]]\Delta.$$

Proof. In order to prove this lemma, it is necessary to prove the following stronger property:

Assuming that ρ satisfies $\Gamma, x\colon K, \Gamma'$ at Δ, if $\Gamma, x\colon K, \Gamma' \triangleright A\colon B$ for some B, and $\Gamma \triangleright D\colon K$ where K is a kind, then

$$[\![\Gamma, \Gamma'[D/x] \triangleright A[D/x]]\!]\rho\Delta = [\![\Gamma, x\colon K, \Gamma' \triangleright A]\!]\rho[x\colon= \langle D[\rho], [\![\Gamma \triangleright D]\!]\rho\Delta\rangle]\Delta,$$

and if ρ satisfies $\Gamma, x\colon \sigma, \Gamma'$ at Δ, $\Gamma, x\colon \sigma, \Gamma' \triangleright A\colon B$ for some B, and $\Gamma \triangleright M\colon \sigma$ where σ is a type, then

$$[\![\Gamma, \Gamma'[M/x] \triangleright A[M/x]]\!]\rho\Delta = [\![\Gamma, x\colon \sigma, \Gamma' \triangleright A]\!]\rho[x\colon= M[\rho]]\Delta.$$

The proof of this property is by induction on the size of A, and it uses lemma 5.11 and lemma 5.13. □

Using the previous lemma, we can show the following important lemma.

Lemma 5.15 *If $\mathcal{C}$ is the family of saturated sets, whenever ρ satisfies Γ at Δ and $A \overset{*}{\longleftrightarrow}_{CC} A'$, then $[\![\Gamma \triangleright A]\!]\rho\Delta = [\![\Gamma \triangleright A']\!]\rho\Delta$.*

Proof. The proof is by induction on the sum of the sizes of A and A', and it uses lemma 5.11, lemma 5.13, and lemma 5.14. □

Finally, we can prove the main theorem. Recall from definition 5.6 that

$$\Delta \Vdash \Gamma[\rho]$$

means

(1) ρ satisfies Γ at Δ, and

(2) $\Delta \triangleright x[\rho] \in [\![\Gamma \triangleright \Gamma(x)]\!]\rho\Delta$ for every $x \in dom(\Gamma)$.

It is easy to verify that if $\Delta \subseteq \Delta'$ and $\Delta \Vdash \Gamma[\rho]$, then $\Delta' \Vdash \Gamma[\rho]$.

Theorem 5.16 *If $\mathcal{C}$ is the family of saturated sets, whenever $\Gamma \triangleright M\colon A$ and $\Delta \Vdash \Gamma[\rho]$, then $\Delta \triangleright M[\rho] \in [\![\Gamma \triangleright A]\!]\rho\Delta$.*

Proof. The proof is by induction on a deduction proving that A type/kind-checks in Γ. Lemma 5.15 is crucial in taking care of the case where the last inference is the type or kind equality rule. □

As mentioned earlier, if we define

$$\Delta \Vdash (\Gamma \triangleright M\colon A)[\rho]$$

iff

(1) ρ satisfies Γ at Δ, and

(2) $\Delta \triangleright M[\rho] \in [\![\Gamma \triangleright A]\!]\rho\Delta$,

then, the main theorem reads as follows:

$$\text{Whenever } \Gamma \triangleright M\colon A \text{ and } \Delta \Vdash \Gamma[\rho], \text{ then } \Delta \Vdash (\Gamma \triangleright M\colon A)[\rho].$$

This looks like a Kripke-style type soundness result.

By letting ρ_s be the identity substitution and ρ_c assign the canonical element $can_{\Gamma(x),\Gamma}$ to each $x \in dom(\Gamma)$, $can_{\star,\Gamma}$ to $\star$, and $can_{kind,\Gamma}$ to $kind$, we obtain the fact that all valid terms of the theory of construction are SN.

Theorem 5.17 *Whenever $\Gamma \triangleright M\colon A$, the term M is SN. This applies to kinds, types, and terms (proofs).*

An interesting consequence of theorem 5.17 is an elementary proof of the consistency of CC. There are other elementary methods for showing that CC is consistent, for example, the "proof-irrelevance semantics", which consists in interpreting types as Zermelo-Fraenkel sets, and $\star$ as the set $\{0, 1\}$ (for details, see Coquand [Coq90]). What is more interesting, is that theorem 5.17 can be used to show in an elementary fashion that certain contexts are consistent, as shown in Coquand [Coq90].

Definition 5.18 We say that a context Δ is *consistent* iff there is some valid type σ (with $\Delta \triangleright \sigma\colon\star$) such that $\Delta \triangleright M\colon\sigma$ is **not** provable for any (proof) term M. We also say that a type σ is *inhabited* in the context Δ iff there is some (proof) term M such that $\Delta \triangleright M\colon\sigma$ is derivable.

Saying that CC is consistent means that the empty context is consistent, which is equivalent to the fact that some valid closed type is *not* inhabited. An elegant combinatorial proof of the consistency of CC using theorem 5.17 is given below.

Lemma 5.19 *The theory CC is consistent. Furthermore, the valid type $(\Pi x\colon\star)x$ is not inhabited.*

Proof. First, observe that the judgment $x\colon\star \triangleright x\colon\star$ is derivable, and so $\triangleright\ (\Pi x\colon\star)x\colon\star$ is derivable. We make use of the following crucial fact: If M is a valid proof in some context Δ and M is a normal form w.r.t. β-reduction, then M is of the shape

$$\lambda x_1\colon A_1.\ \ldots.\ \lambda x_m\colon A_m.\ yN_1 \ldots N_n,$$

where y is a variable possibly among $x_1, \ldots, x_m$, and $N_1, \ldots, N_n$ are normal forms $(m, n \geq 0)$, but not necessarily of the same shape as M, since some N_i's could be products.

The above fact is easily shown by induction on the size of M. The case where $M = M_1 M_2$ is the only interesting one. Because M is normal, M_1 cannot be an abstraction. However, it must be a proof, and by the induction hypothesis, it must be either a variable or an application of the form $xN_1 \ldots N_n$.

Now, assume that there is a valid closed proof M such that $\rhd M\colon (\Pi x\colon \star)x$ is derivable. By theorem 5.17 and by lemma 4.12, we can assume that M is in normal form. But then, it is easily seen that it must be the case that we have $M = \lambda x\colon \star.\ yN_1 \ldots N_n$ and that we have a derivation $x\colon \star \rhd yN_1 \ldots N_n\colon x$. However, it is a simple property of CC that for every judgment $\Delta \rhd E$, $FV(E) \subseteq dom(\Delta)$. This implies that $y = x$. However, x is now both a proof and a type, which is impossible by lemma 4.13. □

In Coquand [Coq90], it is shown using theorem 5.17 that a nontrivial context *Inf* is consistent. The proof is elementary, except for the use of theorem 5.17. As we shall see below, there cannot be any elementary direct proof of the consistency of the context *Inf* (say in first-order Peano arithmetic, or even in classical higher-order arithmetic). Letting *Void* $= (\Pi x\colon \star)x$ (the "absurd" type),

$$\begin{aligned} \mathit{Inf} = {} & A\colon \star,\ f\colon A \to A,\ R\colon A \to A \to \star, \\ & h_1\colon (\forall x\colon A)(Rxx \to \mathit{Void}), \\ & h_2\colon (\forall x, y, z\colon A)(Rxy \to Ryz \to Rxz), \\ & h_3\colon (\forall x\colon A)Rx(fx). \end{aligned}$$

The context *Inf* can be viewed as a kind of axiom of infinity. In turn, it can be shown that the consistency of this context implies the consistency of classical higher-order arithmetic. The proof is elementary, except for the use of theorem 5.17. Thus, by Gödel's second incompleteness theorem, we obtain that strong normalization in CC (theorem 5.17) is **not** provable in classical higher-order arithmetic.

Theorem 5.17 and the Church-Rosser property also imply the decidability of type-checking in CC. In fact, a stronger result holds. The main lines of a proof of the above result were given by the first author in a communication to the "Types forum". This proof is quite similar to a proof by Martin Löf [ML72].

Lemma 5.20 *Given any context $\Delta = x_1\colon A_1, \ldots, x_n\colon A_n$ and any expression M, it is decidable whether $\Delta \rhd$, and if so, whether $\Delta \rhd M\colon kind$ or $\Delta \rhd M\colon A$ for some A (which is given by the algorithm).*

Proof sketch. There are two kinds of problems: testing whether $\Delta \rhd$ or $\Delta \rhd \star\colon kind$, and testing whether M kind/type-checks in the context Δ. We associate a complexity measure to these two problems as follows. Let $c(\langle x_1 \colon A_1, \ldots, x_n \colon A_n\rangle) = 1+$ the sum of the sizes of each A_i (and the same value for $c(\langle x_1 \colon A_1, \ldots, x_n \colon A_n\rangle, \star))$, and $c(\langle x_1 \colon A_1, \ldots, x_n \colon A_n\rangle, M) =$ the size of $M+$ the sum of the sizes of each A_i. We proceed by induction on complexity measures. There are several cases.

1. The problem is $\Delta \rhd ?$ or $\Delta \rhd \star \colon \mathit{kind}?$ and $\Delta = \emptyset$. The answer is yes.

2. The problem is $x_1 \colon A_1, \ldots, x_n \colon A_n \rhd$ or $x_1 \colon A_1, \ldots, x_n \colon A_n \rhd \star \colon \mathit{kind}$ and $n > 0$. Check whether A_n is well formed in $x_1 \colon A_1, \ldots, x_{n-1} \colon A_{n-1}$. If the algorithm returns B, check that either the normal form of B is $\star$, or that B is a kind.

3. M is a variable x. Check whether A_n is well formed in $x_1 \colon A_1, \ldots, x_{n-1} \colon A_{n-1}$. If the algorithm returns B, check that the normal form of B is $\star$ or that B is a kind, and whether x is one of the x_i.

4. M is of the form $(\Pi x \colon A)B$. Check whether A is well formed in $x_1 \colon A_1, \ldots, x_n \colon A_n$ and whether B is well formed in $x_1 \colon A_1, \ldots, x_n \colon A_n, x \colon A$.

5. M is of the form $\lambda x \colon A.\, N$. Check whether A is well formed in $x_1 \colon A_1, \ldots, x_n \colon A_n$ and whether N is well formed in $x_1 \colon A_1, \ldots, x_n \colon A_n, x \colon A$. If the answer to the second problem is yes and the algorithm returns P, then $x_1 \colon A_1, \ldots, x_n \colon A_n \rhd M \colon (\Pi x \colon A)P$.

6. M is of the form $M_1 M_2$. This case requires the fact that every term has a unique normal form. First, we check whether both M_1 and M_2 are well-formed in $x_1 \colon A_1, \ldots, x_n \colon A_n$. If so, we check whether the normal form of the type/kind of M_1 is of the form $(\Pi x \colon A)P$ and the normal form of the type/kind of M_2 is P. □

A closer look at definition 5.5, especially the definitions of $[\![\Gamma \rhd (\Pi x \colon K)D]\!]\rho\Delta$ and $[\![\Gamma \rhd (\Pi x \colon \sigma)D]\!]\rho\Delta$, suggests the definition of certain dependent products. Let Δ be a context and $(\Pi x \colon K)D$ be a kind or a type such that $\Delta \rhd (\Pi x \colon K)D \colon \kappa$, $\kappa \in \{\star, \mathit{kind}\}$, with K a kind.

Definition 5.21 Let $\mathcal{A} = (\mathcal{A}_{\Delta'})_{\Delta' \supseteq \Delta}$ be any Δ'-indexed family of candidates such that $\mathcal{A}_{\Delta'} \in \mathcal{C}_{K,\Delta'}$, and let F be any function with domain $\{\langle \Delta' \rhd A, C\rangle \mid \Delta' \rhd A \colon K,\ \Delta' \supseteq \Delta, \text{ and } C \in \mathcal{C}_{A,\Delta'}\}$, and such that $F(\Delta' \rhd A, C) \in \mathcal{C}_{D[A/x],\Delta'}$. The dependent product $\prod(\mathcal{A}, F; (\Pi x \colon K)D)$ is defined as follows:

$$\begin{aligned}\prod(\mathcal{A}, F; (\Pi x \colon K)D) = \{\Delta' \rhd M \mid{} & \Delta' \rhd M \colon (\Pi x \colon K)D,\ \Delta' \supseteq \Delta, \text{ and} \\ & \forall \Delta'' \supseteq \Delta',\ \forall \Delta'' \rhd A \in \mathcal{A}_{\Delta''},\ \forall C \in \mathcal{C}_{A,\Delta''}, \\ & \Delta'' \rhd (MA) \in F(\Delta'' \rhd A, C)\}.\end{aligned}$$

Let Δ be a context and $(\Pi x \colon \sigma)D$ be a kind or a type such that $\Delta \rhd (\Pi x \colon \sigma)D \colon \kappa$, $\kappa \in \{\star, \mathit{kind}\}$, with σ a type.

Definition 5.22 Let $\mathcal{A} = (\mathcal{A}_{\Delta'})_{\Delta' \supseteq \Delta}$ be any Δ'-indexed family of candidates such that $\mathcal{A}_{\Delta'} \in \mathcal{C}_{\sigma,\Delta'}$, and let F be any function with domain $\{\Delta' \rhd N \mid \Delta' \rhd N\colon \sigma,\ \Delta' \supseteq \Delta\}$, and such that $F(\Delta' \rhd N) \in \mathcal{C}_{D[N/x],\Delta'}$. The dependent product $\prod(\mathcal{A}, F; (\Pi x\colon \sigma)D)$ is defined as follows:

$$\begin{aligned}\prod(\mathcal{A}, F; (\Pi x\colon \sigma)D) = \{\Delta' \rhd M \mid {} & \Delta' \rhd M\colon (\Pi x\colon \sigma)D,\ \Delta' \supseteq \Delta,\ \text{and}\\ & \forall \Delta'' \supseteq \Delta',\ \forall \Delta'' \rhd N \in \mathcal{A}_{\Delta''},\\ & \Delta'' \rhd (MN) \in F(\Delta'' \rhd N)\}.\end{aligned}$$

Then, we can express $[\![\Gamma \rhd (\Pi x\colon K)D]\!]\rho\Delta$ and $[\![\Gamma \rhd (\Pi x\colon \sigma)D]\!]\rho\Delta$ as dependent products:

$$[\![\Gamma \rhd (\Pi x\colon K)D]\!]\rho\Delta = \prod(([\![\Gamma \rhd K]\!]\rho\Delta')_{\Delta' \supseteq \Delta},\ F;\ ((\Pi x\colon K)D)[\rho]),$$

where F is the function such that

$$\langle \Delta' \rhd A, C\rangle \mapsto [\![\Gamma, x\colon K \rhd D]\!]\rho[x\colon= \langle A, C\rangle]\Delta',$$

with $\Delta' \rhd A\colon K[\rho]$ and $C \in \mathcal{C}_{A,\Delta'}$, and

$$[\![\Gamma \rhd (\Pi x\colon \sigma)D]\!]\rho\Delta = \prod(([\![\Gamma \rhd \sigma]\!]\rho\Delta')_{\Delta' \supseteq \Delta},\ F;\ ((\Pi x\colon \sigma)D)[\rho]),$$

where F is the function such that

$$\Delta' \rhd N \mapsto [\![\Gamma, x\colon \sigma \rhd D]\!]\rho[x\colon= N]\Delta',$$

with $\Delta' \rhd N\colon \sigma[\rho]$.

The definition of $\prod(\mathcal{A}, F; (\Pi x\colon \sigma)D)$ is inspired by the definition of the dependent product $\prod(A, F)$ given by Coquand and Huet on page 107 of their paper [CH88]. The difference is that Coquand and Huet give a definition of $\prod(A, F)$ for *untyped* λ-terms. They have no definition analogous to our dependent product $\prod(\mathcal{A}, F; (\Pi x\colon K)D)$ where K is a kind. Also, Coquand and Huet's main theorem on page 109 of their paper [CH88], can be considered as a version of our theorem 5.16 for "stripped terms" (that is, valid terms of CC from which type information has been erased). However, theorem 5.16 is a stronger result, since it yields theorem 5.17 as a corollary, whereas Coquand and Huet's theorem only shows that the *type erasure* $Erase(M)$ of any valid term of CC is SN. As far as we know, there does not seem to be any way to infer from the fact that $Erase(M)$ is SN that M itself must be SN. This is in contrast with the situation in $\lambda^{\forall}$ (and system F_{ω}).

We now examine the special case of LF, and note that strong normalization holds as a corollary, but does not make any use of families of candidates. Only the canonical $can_{A,\Delta}$ are needed.

6 Strong Normalization in LF

Since LF can be viewed as a fragment of CC obtained by disallowing products and abstractions over type variables, it follows immediately from theorem 5.17 that all valid terms of LF are strongly normalizing (under β-reduction). However, it turns out that the powerful artillery of the $\mathcal{C}_{A,\Delta}$ is unnecessary to prove this result. In LF, we can only have products of the form $(\Pi x{:}\,\sigma)D$, and abstractions of the form $\lambda x{:}\,\sigma.\, B$, when σ is a *type* (but *not a kind*). Thus, we have a simpler definition of $[\![\Gamma \triangleright A]\!]\rho\Delta$. Again, A is either a type family or a kind valid in Γ, and the definition only makes sense when ρ satisfies Γ at Δ.

Definition 6.1 In the clauses below, K stands for a kind, σ for a type, A, B for type families, D for a kind or a type, M for a type family or a term (proof), and N for a term (proof).

$$
\begin{aligned}
[\![\Gamma \triangleright kind]\!]\rho\Delta &= \rho_c(kind),\\
[\![\Gamma \triangleright \star]\!]\rho\Delta &= \rho_c(\star),\\
[\![\Gamma \triangleright x]\!]\rho\Delta &= \rho_c(x),\\
[\![\Gamma \triangleright AB]\!]\rho\Delta &= [\![\Gamma \triangleright A]\!]\rho\Delta(\Delta \triangleright B[\rho],\ [\![\Gamma \triangleright B]\!]\rho\Delta),\\
[\![\Gamma \triangleright AN]\!]\rho\Delta &= [\![\Gamma \triangleright A]\!]\rho\Delta(\Delta \triangleright N[\rho]),\\
[\![\Gamma \triangleright (\Pi x{:}\,\sigma)D]\!]\rho\Delta &= \{\Delta' \triangleright M \mid \Delta' \triangleright M{:}\,((\Pi x{:}\,\sigma)D)[\rho_s],\ \Delta' \supseteq \Delta,\ \text{and } \forall\Delta''\forall N\\
&\qquad (\text{if } \Delta'' \supseteq \Delta' \text{ and } \Delta'' \triangleright N \in [\![\Gamma \triangleright \sigma]\!]\rho\Delta'',\ \text{then}\\
&\qquad \Delta'' \triangleright (MN) \in [\![\Gamma, x{:}\,\sigma \triangleright D]\!]\rho[x{:}= N]\Delta'')\},\\
[\![\Gamma \triangleright \lambda x{:}\,\sigma.\, B]\!]\rho\Delta &= \lambda(\Delta' \triangleright N).\, [\![\Gamma, x{:}\,\sigma \triangleright B]\!]\rho[x{:}= N]\Delta',\\
&\qquad \text{a function with domain}\\
&\qquad \{\Delta' \triangleright N \mid \Delta' \triangleright N{:}\,\sigma[\rho],\ \Delta' \supseteq \Delta\}.
\end{aligned}
$$

Remarkably, the candidates, that is, the sets $C \in \mathcal{C}_{A,\Delta}$, *do not* appear anywhere in these definitions. The only place where they play a role is in $[\![\Gamma \triangleright x]\!]\rho\Delta$ and $[\![\Gamma \triangleright \star]\!]\rho\Delta$. However, this role is very passive. In fact, all we need to establish strong normalization is to assign the canonical sets and functions $can_{A,\Delta}$. More precisely, $\rho_c(kind)$ is the set $can_{\star,\Delta}$ of SN kinds, $\rho_c(\star)$ is the set $can_{\star,\Delta}$ of SN types, and $\rho_c(x) = can_{\Gamma(x)[\rho],\Delta}$. Only the substitution component ρ_s of ρ needs to be arbitrary for the proof to go through, the other component ρ_c remaining constant (and determined by the canonical elements). Thus, the proof of strong normalization for LF uses little more than is needed for the proof of strong normalization in the simply-typed λ-calculus, namely the existence of the canonical sets and functions, which itself depends on the existence of the measure $c(K)$, where K a kind. This is not surprising in view of another proof by Harper, Honsell, and Plotkin [HHP89], in which a mapping from LF into the simply-typed λ-calculus is used. It should be noted that their proof applies to β and η reduction, but we do not know presently how to extend our approach to η-reduction.

7 Other Proofs

This section lists other proofs of normalization or strong normalization that we are aware of, in chronological order. We apologize if we are unaware of other proofs not mentioned here. To us, the history of this proof seems sufficiently interesting to be told, especially in a preliminary report, even if it is incomplete. It has been reported that some of these proofs contain errors. We are indeed aware of some errors, and we will briefly mention what they are. We apologize for any (unintentional) omissions or misinterpretations.

1. Coquand, January 1985 [Coq85]. This is Thierry Coquand's thesis. A proof of normalization is given, as well as some indications on how to extend it to strong normalization. There is a problem with the definition of the sets $\mathcal{C}_{A,\Delta}$ when A is a type family of kind $(\Pi x\colon B)D$. The members of $\mathcal{C}_{A,\Delta}$ are indeed functions, but only of one argument, the candidate argument. A similar problem arises in the definition of $[\![\Gamma \triangleright \lambda x\colon K.\ B]\!]\rho\Delta$, where the argument $\Delta' \triangleright A$ is omitted. As a consequence, $[\![\Gamma \triangleright A]\!]\rho\Delta$ is not always well-defined.

2. Jutting, December 1986 [vBJ86]. This is a note attempting to correct Coquand's proof of normalization given in his thesis. The introduction mentions discussions with Coquand, leading to this note. As we see it, the definition of $[\![\Gamma \triangleright A]\!]\rho\Delta$ is indeed repaired correctly. However, Δ is dropped from the $\mathcal{C}_{A,\Delta}$, which becomes a family of sets of *closed* terms. To insure that each $C \in \mathcal{C}_A$ is nonempty (A a closed type family or closed kind), Jutting adds a countably infinite set of constants. Unfortunately, this causes a problem. Indeed, the language has now been enriched, new types can be formed, and some new closed types may not be inhabited.

3. Coquand, 1987 [Coq87]. This is a note in which Coquand fixes the problem with the addition of new constants, and gives a proof of strong normalization for the first time. The proof uses infinite contexts, and basically Henkin's technique for adding new witnesses, so that all closed types are inhabited.

4. Pottinger, February 1987 [Pot87]. This paper refers to Coquand 1987, and gives a proof of strong normalization apparently inspired by Coquand's proof. Infinite contexts are also used, as well as an idea due to Seldin. Although we need to examine it more closely, the proof seems correct, but rather difficult to follow.

5. Seldin, November 1987 [Sel87]. This is a report, "Mathesis: the Mathematical Foundations of Ulysses", in which a proof of strong normalization for a *variant* of the theory of constructions is given. We have not yet had the time to examine this proof carefully, but it appears that it also uses infinite contexts. It appears to be more along the line of Martin Löf's proof of normalization for F_ω, defined as a Prawitz-style natural deduction system.

6. Zhaohui Luo, 1989 [Luo90]. There is apparently a proof of strong normalization for an extension of CC with universes, given in Luo's thesis. We do not have this document yet.

7. Geuvers and Nederhof, June 1989 [GN89]. The authors present what they call a modular proof of strong normalization, by reducing strong normalization in CC to strong normalization in Girard's F_ω. This is accomplished by defining a mapping from CC to F_ω, such that reduction of terms is preserved. Strong normalization for the terms of F_ω is itself reduced to strong normalization for the erased (raw) terms of F_ω, which is proved directly.

8. Berardi, 1989 [Ber89]. Berardi gives a proof (apparently due to Terlouw) in an appendix of his thesis.

Acknowledgment: We wish to thank Val Breazu-Tannen and Sunil Shende for many helpful comments.

References

[Ber89] S. Berardi. *?* PhD thesis, Universita di Torino, 1989.

[CH88] Thierry Coquand and G. Huet. The calculus of constructions. *Information and Computation*, 76, 2/3:95–120, March 1988.

[Coq85] Thierry Coquand. *Une Théorie Des Constructions*. PhD thesis, Université Paris VII, January 1985. Thèse de 3^{eme} Cycle.

[Coq87] Thierry Coquand. Metamathematical investigations of a calculus of constructions. Technical report, INRIA, Domaine de Voluceau, Rocquencourt, 1987. Privately circulated manuscript.

[Coq90] Thierry Coquand. Metamathematical investigations of a calculus of constructions. In P. Odifreddi, editor, *Logic And Computer Science*, pages 91–122. Academic Press, London, New York, May 1990.

[Gal90] J. Gallier. On Girard's "candidats de reductibilité". In P. Odifreddi, editor, *Logic And Computer Science*, pages 123–203. Academic Press, London, New York, May 1990.

[Gir72] Jean Yves Girard. *Interprétation fonctionelle et élimination des coupures de l'arithmétique d'ordre supérieur*. PhD thesis, Université Paris VII, Paris, 1972. Thèse de Doctorat d'Etat.

[GN89] H. Geuvers and M.-J. Neherhof. A modular proof of strong normalization for the calculus of constructions. *Journal of Functional Programming*, page pp. 38, June 1989. Submitted for publication.

[HHP89] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *J. ACM*, 1989. Submitted for publication.

[Luo90] Z. Luo. *ECC, an extended calculus of constructions (check title?)*. PhD thesis, University of Edinburgh, Edinburgh, Scotland, 1990? Forthcoming thesis.

[ML72] P. Martin Löf. An intuitionistic theory of types. Technical report, University of Stokholm, Stockholm, Sweden, 1972. Privately circulated manuscript.

[MM87] J. C. Mitchell and E. Moggi. Kripke-style models for typed lambda calculus. In *Second Symposium on Logic in Computer Science*, pages 303–314, Ihaca, New York, June 22-25 1987. IEEE.

[Pot87] G. Pottinger. Strong normalization for terms of the theory of constructions. Technical report, Odyssey Research Associates, Ithaca, New York, February 1987.

[Ran90] A. Ranta. Constructing possible worlds. *Theoria*, 1990. To appear.

[Sel87] J. Seldin. Mathesis: The mathematical foundations of ulysses. Technical Report RADC-TR-87-223, Odyssey Research Associates, Ithaca, New York, November 1987. Interim Report.

[vBJ86] L. S. van Benthem Jutting. Normalization in coquand's system. Technical report, ?, December 1986. Private Communication.