Variants of the Basic Calculus of Constructions

M. W. Bunder [1]

School of Mathematics and Applied Statistics, University of Wollongong, N.S.W. 2522 Australia

Jonathan P. Seldin [*,2]

Department of Mathematics and Computer Science, University of Lethbridge, 4401 University Drive, Lethbridge, Alberta, Canada, T1K 3M4

Abstract

In this paper, a number of different versions of the basic calculus of constructions that have appeared in the literature are compared and the exact relationships between them are determined. The biggest differences between versions are those between the original version of Coquand and the version in early papers on the subject by Seldin. None of these results is very deep, but it seems useful to collect them in one place.

Key words: Calculus of constructions, Typed λ-calculus, Pure typed systems
1991 MSC: 03B15, 03B40, 03B70, 68T15

1 Introduction

Since Coquand first introduced the calculus of constructions in [1–5], there have been a number of different versions of the system published. Of the

* Corresponding author.
Email addresses: `martin_bunder@uow.edu.au` (M. W. Bunder), `jonathan.seldin@uleth.ca` (Jonathan P. Seldin).
URLs: `http://www.uow.edu.au/informatics/maths/staff/bunder.html` (M. W. Bunder), `http://www.cs.uleth.ca/~seldin` (Jonathan P. Seldin).
1 Work supported in part by a grant from the Australian Research Council.
2 Work supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada.

versions published by Coquand himself, one appears in [1,4,5], and another appears in [2], and still another appears in [3]. One of the most distinctive versions in the literature is due to Seldin, which differs from the others in some important ways. Seldin had first learned of the calculus of constructions in early 1986, when he was working for Odyssey Research Associates. Richard Platek, who was then president of that company, had hired Seldin to work on a project that involved using some version of typed λ-calculus with the propositions-as-types notion to develop a very general system for formal verification [6], and the first task he assigned Seldin was to choose the best version of typed λ-calculus. As part of the process of choosing, Platek and Seldin visited Carnegie-Mellon University in early 1986 to see both Coquand and Huet, who were visiting there at the time. At this time, only the earliest papers of Coquand and Huet had appeared: [1,4], the latter in preprint form. As a result, it seemed to Seldin that the definition of the system was was subject to modification. Furthermore, a major part of the project on which he was working required formal proofs of consistency (see [6, Introduction]), and Seldin was thus led to reformulate the system in a form in which he could use familiar tools for the proof of consistency: the form of Curry's theory of functionality (his name for type assignment), which he considered a part of illative combinatory logic, and the proof-theoretic tools of Gentzen [7], especially as developed by Curry [8,9], and Prawitz [10]. Seldin also wanted to allow for the possibility of assumptions other than those assigning types to variables; he thought that such assumptions might be useful, for example in dealing with the possibility of subtyping, which Curry had postulated by taking the assumption $\mathsf{I} : \alpha \to \beta$ or $\lambda x \,.\, x : \alpha \to \beta$, where $\alpha \not\equiv \beta$; see [11, Remark 2, p. 23] and [12, pp. 97–99]. Seldin was thus led to systems significantly more general than the original formulation of Coquand and Huet. Originally, Seldin did not think these differences were terribly important, and he continued to use these versions in [6,13,12,14–16]. Recently, however, Seldin has been asked on several occasions about the exact relationship between his versions and the other ones.

After Coquand introduced the calculus of constructions, Berardi [17] and Terlouw [18] introduced the concept of a PTS as a generalization of the GTSs of the Barendregt cube (see [19]). A comparison of the definition of a PTS to the definitions of the original versions of the calculus introduced by Coquand shows that the latter is a special case of the former. More recently, Bunder and Dekkers have been studying variants of PTSs for the purpose of comparing them with systems of illative combinatory logic [20] (see also [21] and [22]), and as a result the exact differences between these different formulations now seem more important. The purpose of this paper is to study these different formulations of the calculus of constructions and to compare them.

Some of these results are new, some have appeared elsewhere, of these some have been proved for PTSs in general. It seems useful to collect in one place those that apply to the calculus of constructions.

These different formulations have some things in common, namely the terms and forms of judgments, the axiom, and the form of the application and product rules. They may differ in the form of the rule(s) for conversions of types, in the form of the abstraction rule, and whether assumptions are sequences and can only be introduced by rules, or are sets and can be arbitrary. Both kinds are natural deduction systems; the latter are more like those that have appeared in the work of Gentzen [7] and Prawitz [10]. Which of these versions one wants will depend on one's purpose. If one has a purpose for which typechecking is important, one will probably prefer one of the P or A versions below with sequences for assumptions, whereas if one wants to obtain consistency proofs or obtain other proof-theoretic results, one of the C or AC versions below with sets of assumptions may be more useful. We hope that the results of this paper will help researchers make this choice.

There are different kinds of extensions of the calculus of constructions that we do not consider here: the extended calculus of constructions [23,24], the calculus of constructions with inductive types [25,26], the calculus of constructions with rewriting [27–33], the calculus of constructions with η-reduction [34,35], and the calculus of constructions as a domain-free pure type system [36]. This is because the different variants we deal with here could be defined for all of these systems, and there are so many of them that considering them all would make the paper much, much longer. Furthermore, in the case of the calculus of constructions with rewriting, new versions are appearing so quickly that it would be difficult to keep up. For this reason, in this paper we are concerned only with the *basic* calculus of constructions.

We would like to thank the anonymous referees for their helpful comments and suggestions.

2 The different variants

All of the formulations are based on the following syntax for *pseudoterms*:

$$M ::= x|c|\mathsf{Prop}|\mathsf{Type}|(\lambda x : M\ .\ M)|(MM)|(\Pi x : M\ .\ M)$$

The reduction relation is β-reduction, where the basic contraction rule is

$$(\beta) \qquad (\lambda x : A\ .\ M)N \rhd [N/x]M,$$

where $[N/x]M$ denotes the substitution of N for all occurrences of x in M with bound variables being changed as necessary to avoid collision. The corresponding conversion relation will be written

$$M =_\beta N.$$

(We do not consider η-reduction or conversion.) The two constants Prop and Type are called *sorts*. (They are called *kinds* in [2] and earlier papers by Seldin.) Unspecified sorts will be denoted here by s, s', etc., so we always have $s, s', \ldots \in \{\mathsf{Prop} : \mathsf{Type}\}$. *Formulas* (called *statements* in PTSs) are of the form $M : A$, where M and A are pseudoterms. All of the versions have the same axiom:

$$(\mathsf{PT}) \qquad \vdash \mathsf{Prop} : \mathsf{Type}.$$

In some formulations, judgments are of the form $\Gamma \vdash E$, where Γ is a *sequence* of assumptions $x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n$, and E is a formula. In systems of this kind, Γ is regarded as *legal* if and only if it is possible to prove $\Gamma \vdash E$ for some formula E.[3] Furthermore, assumptions can only be introduced on the left of $\vdash$ by a rule such as

$$\frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash \mathsf{Prop} : \mathsf{Type},}$$

where $x \notin \mathrm{FV}(\Gamma, A)$, i.e., x does not occur free in Γ or A. Formulations of this kind are similar to the original formulation of Coquand [1], or equivalently, [5], (which is equivalent to the system called TOC2P below), and a formulation of this kind was called TOC2 by Garrel Pottinger [37]. Hence, in this paper, systems of this kind will be called “TOC2-like”. (TOC stands for “Theory of Constructions.”)

In others judgments are of the form $\Delta \vdash E$ where Δ is a *set* of formulas and E is a formula. Here, any premises of the form $M : A$ are possible. In these systems, a set of assumptions which assign types to distinct variables is considered legal if it can be ordered in a sequence in such a way that the assumptions can be discharged by the rules of the system in reverse order. Systems of this kind will be called “TOC0-like”.

In what follows, the names of these systems will be obtained from “TOC0” or “TOC2” by adding letters, “P”, “A”, and “C”. The addition of “P” will refer to a system whose abstraction and conversion rules are essentially those of a PTS;

[3] This is equivalent to $\Gamma \vdash \mathsf{Prop} : \mathsf{Type}$.

the addition of “A” will indicate a system with a modified abstraction rule, such as the systems $\lambda^{\omega}(S)$ of [38], and the addition of “C” will denote a system with a modified rule of conversion between types. If both the abstraction and conversion rules are modified, “AC” will be added to the name. The original TOC0 of Seldin [6,13,12,14–16] is, in this notation, TOC0AC. [4]

As we shall see below, TOC0-like systems and TOC2-like systems with the same letters on the end are equivalent. We shall give below the exact relationships between “P”, “A”, “C”, and “AC” formulations.

3 P-versions: PTS like systems

Let us start with the TOC2-like P version.

Definition 1 *The* system TOC2P *is a system of the above kind with sequences of assumptions. The axiom is* (PT)*. The rules are as follows:*

(Validity) $$\frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash \mathsf{Prop} : \mathsf{Type}}$$ Condition: $x \notin \mathrm{FV}(\Gamma, A)$

(Variable) $$\frac{\Gamma_1, x : A, \Gamma_2 \vdash \mathsf{Prop} : \mathsf{Type}}{\Gamma_1, x : A, \Gamma_2 \vdash x : A}$$

(Product) $$\frac{\Gamma, x : A \vdash B : s}{\Gamma \vdash (\Pi x : A \,.\, B) : s}$$ Condition: $x \notin \mathrm{FV}(\Gamma, A)$

(Abstraction) $$\frac{\Gamma, x : A \vdash M : B \quad \Gamma, x : A \vdash B : s}{\Gamma \vdash (\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)}$$ Condition: $x \notin \mathrm{FV}(\Gamma, A)$

(Application) $$\frac{\Gamma \vdash M : (\Pi x : A \,.\, B) \quad \Gamma \vdash N : A}{\Gamma \vdash MN : [N/x]B}$$

(Conversion) $$\frac{\Gamma \vdash M : A \quad A =_{\beta} B \quad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

[4] Actually, it is a slight generalization; see Remark 43 in §4 below.

A sequence Γ *of formulas assigning types to variables is* legal *if it is possible to prove* $\Gamma \vdash E$ *for some formula* E*.*

Remark 2 *The condition on (Validity) could be changed to read "*$x \notin \mathrm{FV}(\Gamma)$*" and the conditions on rules (Product) and (Abstraction) could be dropped. If we define dom*(Γ) *to be* $\{x_1, x_2, \ldots, x_n\}$ *when* $\Gamma \equiv x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n$*, then the condition on (Validity) could be changed to read "*$x \notin$ *dom*(Γ)*." This is because [19, Lemma 5.2.8] (the free variable lemma) can be proved for this system; see Lemma 8 below. However, this lemma fails for the systems TOC2C and TOC2AC, which are considered later in the paper. For this reason, we are retaining these conditions explicitly as stated in all the formulations we consider.*

Remark 3 *TOC2P is a restriction of the system called TOC2 in [37,13]. Seldin [6,13,12,14–16] writes* $(\forall x : A)B$ *and others write* $(\Pi x : A)B$ *for* $(\Pi x : A \,.\, B)$*. In PTSs, it is standard to use* $*$ *for* Prop *and* $\Box$ *for* Type*.*

Remark 4 *The system of Coquand [2] is a TOC2P system. The earlier systems of [1,4,5] of Coquand (and Huet) are equivalent to this, but* Type *is not written explicitly,* $\vdash M$ *is written for* $\vdash M : \mathsf{Type}$*, and* $*$ *is used for* Prop*. The fact that* Type *is not expressed explicitly makes the second premise of the rule (Abstraction) automatically true, and hence it is easy to prove their equivalence by induction on the proofs. Coquand is really the first to formulate the calculus of constructions as a TOC2P system.*

The version in [3], is special in several respects:

(1) Type *is not a constant, but a special judgment* type *is introduced for formulas involving* Type*, so that* M type $[\Gamma]$ *is the judgment that replaces* $\Gamma \vdash M : \mathsf{Type}$*.*
(2) There is a type operator T *such that if* $\Gamma \vdash M : \mathsf{Prop}$*, then* $\Gamma \vdash \mathsf{T}(M) : \mathsf{Type}$ *(in the standard notation). Also, there is a separate universal quantifier with the property that* $\mathsf{T}((\forall x : A)B)$ *plays the role of* $(\Pi x : A \,.\, B)$*. Thus,* $(\forall x : A)B$ *might be in* Prop*, while* $(\Pi x : A \,.\, B)$ *can only be in* Type *(in the notation of this paper). (Coquand wrote* $(x : A)B$ *and* $(\forall x : A)B$ *for our* $(\Pi x : A \,.\, B)$ *and* $(\forall x : A \,.\, B)$*).*
(3) There is a judgment Γ valid *that takes the place of* $\Gamma \vdash \mathsf{Prop} : \mathsf{Type}$*.*
(4) Coquand writes $M : A$ $[\Gamma]$ *for our* $\Gamma \vdash M : A$*.*

Remark 5 *Garrel Pottinger, in [37], defined a variant that he called TOC1. It is obtained from TOC2P by omitting the rules (Validity) and (Variable) and* *postulating instead the following two rules:*

(Hypothesis) $$\frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A}$$ Condition: $x \notin \mathrm{FV}(\Gamma, A)$

(Reiteration) $$\frac{\Gamma \vdash E \quad \Gamma, F \vdash G}{\Gamma, F \vdash E}$$ Condition: *E, F, and G are formulas*

This is shown equivalent to TOC2P in [37]. (The rules (Hypothesis) and (Reiteration) are versions of the corresponding rules of Fitch [39]. The condition on the variable x in the rule (Hypothesis) could be omitted and/or modified as indicated in Remark 2.)

Remark 6 *A standard PTS replaces (Validity) and (Variable) by*

(Start) $$\frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A}$$ Condition: $x \notin \mathrm{FV}(\Gamma, A)$

(Weakening) $$\frac{\Gamma \vdash M : B \quad \Gamma \vdash A : s}{\Gamma, x : A \vdash M : B}$$ Condition: $x \notin \mathrm{FV}(\Gamma, A)$

(Start) is the same as (Hypothesis). We prove that (Reiteration) is equivalent to (Weakening) if the other rules of TOC2P are present. The conditions on the variable x in these two rules could be omitted and/or modified as indicated in Remark 2.

Lemma 7 *Any deduction of $\Gamma_1, x : A, \Gamma_2 \vdash M : B$ in TOC2P has a subdeduction of $\Gamma_1 \vdash A : s$ for some s.*

PROOF. By induction on the deduction of $\Gamma_1, x : A, \Gamma_2 \vdash M : B$. $\square$

The following is [19, Lemma 5.2.8].

Lemma 8 *If $\Gamma \equiv x_1 : A_1, x_2 : A_3, \ldots, x_n : A_n$ is legal and $\Gamma \vdash M : A$ in TOC2P for some M and A, then*

(1) The variables $x_1, x_2, \ldots, x_n$ are all distinct,

(2) $\mathrm{FV}(MA) \subset \{x_1, x_2, \ldots, x_n\}$*, and*
(3) $\mathrm{FV}(A_i) \subset \{x_1, x_2, \ldots, x_{i-1}\}$ *for* $1 \leq i \leq n$*.*

PROOF. By induction on the derivation of $\Gamma \vdash M : A$. $\square$

Lemma 9 *In TOC2P, if*

$$\Gamma \vdash M : B \tag{1}$$

and

$$\Gamma \vdash A : s, \tag{2}$$

then

$$\Gamma, x : A \vdash M : B. \tag{3}$$

PROOF. In the derivation of (1), each step in which Γ is formed on the left of $\vdash$ takes the form

$$\frac{\Gamma' \vdash C : s_1}{\Gamma \vdash \mathsf{Prop} : \mathsf{Type}} \text{ (Validity)}$$

where $\Gamma \equiv \Gamma', y : C$ for $y \notin \mathrm{FV}(\Gamma', Cx)$. Each such step and any derivation above it can be replaced by the derivation of (2) and an application of (Validity). The result will be a derivation of

$$\Gamma, x : B \vdash \mathsf{Prop} : \mathsf{Type}.$$

The remaining steps of the derivation of (1) can now be carried out with $\Gamma, x : B$ in place of Γ and some changes of variables free in some steps but not in the conclusion, leading to a derivation of (3). $\square$

Lemma 10 *TOC2P has the same valid judgments when* (Validity) *and* (Variable) *are replaced by* (Start) *and* (Weakening).

PROOF. By Remark 5, it is enough to show that (Weakening) and (Reiteration) are equivalent.

Suppose the basic rule is (Reiteration), and suppose we have deductions of

$$\Gamma \vdash M : B \qquad \text{and} \qquad \Gamma \vdash A : s.$$

By Lemma 9, (Weakening) is valid in TOC2P.

Conversely, suppose we are given

$$\Gamma \vdash E \qquad \text{and} \qquad \Gamma, F \vdash G.$$

By an easy induction on the deduction of $\Gamma, F \vdash G$, F must have the form $x : A$, where $x \notin \mathrm{FV}(\Gamma, A)$. By Lemma 7 $\Gamma \vdash A : s$, so by $\Gamma \vdash E$ and (Weakening), we have $\Gamma, F \vdash E$, which is the conclusion of (Reiteration). □

Remark 11 *The rule (Product) is also not exactly the one in the PTS format. The PTS version is*

$$\textit{(PTSProduct)} \quad \frac{\Gamma, x : A \vdash B : s_2 \quad \Gamma \vdash A : s_1}{\Gamma \vdash (\Pi x : A \,.\, B) : s_3} \quad \begin{array}{l}\text{Condition:}\\ x \notin \mathrm{FV}(\Gamma, A)\end{array}$$

where $(s_1, s_2, s_3) \in R$. The set R for the calculus of constructions is

$$\{(\mathsf{Prop}, \mathsf{Prop}, \mathsf{Prop}), (\mathsf{Prop}, \mathsf{Type}, \mathsf{Type}), (\mathsf{Type}, \mathsf{Prop}, \mathsf{Prop}), (\mathsf{Type}, \mathsf{Type}, \mathsf{Type})\}.$$

By Lemma 7, (PTSProduct) simplifies to (Product) in the calculus of constructions. By Remark 2, the condition on x could be omitted.

Remark 12 *Even with (Validity) and (Variable) replaced by (Start) and (Weakening) and realizing that (Product) is essentially (PTSProduct), TOC2P is not quite a PTS. In a PTS, the rule (Abstraction) would be replaced by*

$$\textit{(PTSAbstraction)} \quad \frac{\Gamma, x : A \vdash M : B \quad \Gamma \vdash (\Pi x : A \,.\, B) : s}{\Gamma \vdash (\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)} \quad \begin{array}{l}\text{Condition:}\\ x \notin \mathrm{FV}(\Gamma, A)\end{array}$$

In some PTSs, these are not equivalent, but in the calculus of constructions they are. By Remark 2, the condition on x could be omitted.

Lemma 13 *TOC2P has the same valid judgments when* (Abstraction) *is replaced by* (PTSAbstraction).

The proof requires the following lemma:

Lemma 14 *If*

$$\Gamma \vdash (\Pi x : A \,.\, B) : C,$$

in TOC2P, then

$$\Gamma, x : A \vdash B : s \qquad \textit{and} \qquad C =_\beta s$$

in TOC2P.

Remark 15 *This is a special case of Barendregt's Generation Lemma for PTSs; see [19, Lemma 5.2.13].*

PROOF. By induction on the derivation of $\Gamma \vdash (\Pi x : A \,.\, B) : C$, where, in each case, the last inference which is not by rule (Conversion) is considered. □

Proof of Lemma 13 By Lemma 14, we immediately get

$$\Gamma \vdash (\Pi x : A \,.\, B) : s \iff \Gamma, x : A \vdash B : s.$$

□

The remarks and lemmas we have had so far give us

Theorem 16 *TOC2P has the same valid judgments as the PTS λC of Barendregt [19].*

Now for the TOC0-like P version.

Definition 17 *The* system TOC0P *is a system of the above kind with sets of* *assumptions. The rules are as follows, where* $s \in \{\mathsf{Prop}, \mathsf{Type}\}$*:*

(Axiom) $\Delta \vdash \mathsf{Prop} : \mathsf{Type}$

(Assumption) $\Delta \vdash M : A$ Condition: $M : A \in \Delta$

(ss′F) $$\frac{\Delta \vdash A : s \quad \Delta, x : A \vdash B : s'}{\Delta \vdash (\Pi x : A \,.\, B) : s'}$$ Condition: $x \notin \mathrm{FV}(\Delta, A)$

(Πsi) $$\frac{\Delta, x : A \vdash M : B \quad \Delta, x : A \vdash B : s \quad \Delta \vdash A : s'}{\Delta \vdash (\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)}$$

Condition: $x \notin \mathrm{FV}(\Delta, A)$

(Πe) $$\frac{\Delta \vdash M : (\Pi x : A \,.\, B) \quad \Delta \vdash N : A}{\Delta \vdash MN : [N/x]B}$$

(Eq′′) $$\frac{\Delta \vdash M : A \quad A =_\beta B \quad \Delta \vdash B : s}{\Delta \vdash M : B}$$

Remark 18 *If* Δ *is a well-formed environment (Definition 21 below), then so is* $\Delta, x : A$ *provided* $x \notin \mathrm{FV}(\Delta, A)$*. The rule (Weakening) is admissible for this system (and for all TOC0 systems considered in this paper) because extra formulas assigning types to variables can always be added to any judgement in a deduction without affecting the validity of its inferences as long as the conditions on the occurrence of free variables are not violated.*

Remark 19 *It is one of the general conventions of this kind of system that if* $D_1(x)$ *is a deduction whose conclusion is*

$$\Delta, x : A \vdash M : B$$

where $x \notin \mathrm{FV}(\Delta, A)$*, and if* D_2 *is a deduction whose conclusion is*

$$\Delta \vdash N : A,$$

then

$$\begin{array}{c} \mathcal{D}_2 \\ \Delta \ \vdash \ N : A \\ \mathcal{D}_1(N) \\ \Delta \ \vdash \ [N/x]M : [N/x]B \end{array}$$

is the deduction obtained from $\mathcal{D}_1(x)$ by substituting N for x, replacing every occurrence of $\Delta', x : A \ \vdash \ x : A$ which is the conclusion of (Assumption) in $\mathcal{D}_1(x)$ by $\Delta' \ \vdash \ N : A$ and placing $\mathcal{D}_2'$ above that (noting that $\Delta \subset \Delta'$, since the rules of the system allow assumptions to be discharged but not introduced), where $\mathcal{D}_2'$ is obtained from $\mathcal{D}_2$ by the process justifying the rule (Weakening) as described in Remark 18 above.

Remark 20 *In earlier works by Seldin [6,13,12,14–16], this definition would be given in the style of Prawitz [10]. The axiom (*PT*) would be given in the form*

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

The last four rules would be stated as follows:

(ss′ F) $\dfrac{A : s \qquad \begin{array}{c}[x : A] \\ B : s'\end{array}}{(\Pi x : A \,.\, B) : s'}$ Condition: *x does not occur free in A or in any undischarged assumption.*

(Πsi) $\dfrac{\begin{array}{c}[x : A] \\ M : B\end{array} \qquad \begin{array}{c}[x : A] \\ B : s\end{array} \qquad A : s'}{(\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)}$ Condition: *x does not occur free in A or in any undischarged assumption.*

(Πe) $\dfrac{M : (\Pi x : A \,.\, B) \qquad N : A}{MN : [N/x]B}$

(Eq″) $\dfrac{M : A \qquad A =_\beta B \qquad B : s}{M : B}$

If Δ is a set of assumptions, then $\Delta \ \vdash \ M : A$ holds in this formalism if there is a deduction whose last formula is $M : A$ and in which every undischarged assumption occurs in Δ. By this definition, rules (Axiom) and (Assumption) *of Definition 17 follow by the conventions of this method of giving natural deduction rules.*

It is possible to have sets of assumptions in TOC0P that do not correspond to legal sets of assumptions in Definition 17. However, if we want to be able to discharge assumptions, they must all assign types to variables, we need to take them in a certain order, and they need to satisfy certain conditions.

Definition 21 *A set of assumptions* Δ *is a* well-formed environment *with respect to a TOC-system* S *if all of its assumptions assign types to variables and they can be ordered in a sequence*

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n$$

such that the variables $x_1, x_2, \ldots, x_n$ *are all distinct and the following conditions hold for each* i, $1 \leq i \leq n$*:*

(1) x_i *does not occur free in* $A_1, \ldots, A_i$ *(but it may occur free in* $A_{i+1}, \ldots, A_n$*), and*
(2) $x_1 : A_1, x_2 : A_2, \ldots, x_{i-1} : A_{i-1} \vdash A_i : s$ *for some* s *in* S*.*

This sequence will be called a well-formed sequence *with respect to* S*. (Such sequences are called "S-legal" in the rather similar SPTSs of [21].)*

We need some results on well-formedness with respect to TOC2P.

Lemma 22 *If* $\Gamma \vdash E$ *in TOC2P for any formula* E*, and if* Γ' *is any initial segment of* Γ *(possibly* Γ *itself), then each derivation of* $\Gamma \vdash E$ *contains a subderivation of* $\Gamma' \vdash \mathsf{Prop} : \mathsf{Type}$*.*

PROOF. By induction on the proof of $\Gamma \vdash E$. $\Box$

Lemma 23 *If* $\Gamma \vdash \mathsf{Prop} : \mathsf{Type}$ *in TOC2P, then* Γ *is a well-formed sequence with respect to TOC2P.*

PROOF. By induction on the pair $\langle n, m \rangle$, where n is the number of assumptions in Γ and m is the length of the derivation of $\Gamma \vdash \mathsf{Prop} : \mathsf{Type}$.

Basis: Trivial, since Γ is empty.

Induction step: Assume the lemma for any initial subsequence of Γ, and suppose that Γ is $\Gamma', x : A$. Then $\Gamma \vdash A : s$ for some sort s by Lemma 7.

Lemma 24 *If $\Gamma \vdash E$ in TOC2P, then Γ is a well-formed sequence with respect to TOC2P.*

PROOF. Lemmas 3 and 23. $\square$

Lemma 25 *If Γ is a well-formed sequence with respect to TOC2P, then $\Gamma \vdash \mathsf{Prop} : \mathsf{Type}$ in TOC2P.*

PROOF. If Γ is the empty sequence, the result is trivial by axiom (PT). If Γ is not empty, it is $\Gamma', x : A$. By condition 1 of Definition 21, x does not occur free in Γ or in A. By condition 2 of Definition 21, we have in TOC2P

$$\Gamma' \vdash A : s.$$

The lemma follows by rule (Validity). $\square$

Corollary 26 *Γ is a well-formed sequence with respect to TOC2P if and only if $\Gamma \vdash E$ for some formula E (i.e., if and only if Γ is legal with respect to TOC2P).*

PROOF. By Lemmas 23 and 25. $\square$

We can now prove the equivalence of TOC0P and TOC2P. For purposes of this proof, we will write $\Gamma \vdash_2 E$ to indicate that $\Gamma \vdash E$ is provable in TOC2P, and we will write $\Delta \vdash_0 E$ to indicate that $\Delta \vdash E$ is provable in TOC0P. If Γ is a sequence of assumptions, we will write $\{\Gamma\}$ for the set of those assumptions in Γ, and we will write $\Gamma \vdash_0 E$ for $\{\Gamma\} \vdash_0 E$.

Theorem 27 *If*

$$\Gamma \vdash_2 E, \tag{4}$$

then

$$\Gamma \vdash_0 E. \tag{5}$$

PROOF. By induction on the derivation of (4).

Basis: (4) is (PT) in TOC2P. Then Γ is empty, E is $\mathsf{Prop} : \mathsf{Type}$, and (5) holds by rule (Axiom) in TOC0P.

Induction step: The cases are by the last rule in the derivation of (4).

Case (Validity). Trivial by rule (Axiom).

Case (Variable). Trivial by rule (Assumption).

Case (Product). E is $(\Pi x : A \,.\, B) : s$, where x does not occur free in Γ or in A, and the premise is

$$\Gamma, x : A \vdash_2 B : s.$$

By the induction hypothesis,

$$\Gamma, x : A \vdash_0 B : s.$$

Furthermore, by Lemma 7,

$$\Gamma \vdash_2 A : s'.$$

By another application of the induction hypothesis,

$$\Gamma \vdash_0 A : s',$$

and (5) follows by (ss'F).

Case (Abstraction). Similar to Case (Product) using (Πsi).

Case (Application). E is $MN : [N/x]B$, and the premises are

$$\Gamma \vdash_2 M : (\Pi x : A \,.\, B) \qquad \text{and} \qquad \Gamma \vdash_2 N : A.$$

By the induction hypothesis

$$\Gamma \vdash_0 M : (\Pi x : A \,.\, B) \qquad \text{and} \qquad \Gamma \vdash_0 N : A,$$

and (5) follows by (Πe).

Case (Conversion). Trivial by rule (Eq$'''$). $\square$

Theorem 28 *If Δ is a well-formed environment with respect to TOC0P, and if*

$$\Delta \vdash_0 E, \tag{6}$$

then there is a sequence Γ *such that* $\{\Gamma\}$ *is* Δ *and (4) holds.*

PROOF. By induction on the sum of the length (number of formulas) of the proof of (6) plus the subsidiary proofs that Δ is a well-formed environment. the latter case also proves that $\{\Gamma\}$ is a well-formed environment (with respect to TOC0P). The cases are by the last rule applied in the deduction of (6).

Case (Axiom). E is $\mathsf{Prop} : \mathsf{Type}$. If Δ is empty, (4) is an instance of the axiom (PT). If Δ is not empty, it is $\Delta', x : A$. Since Δ is well-formed with respect to TOC0P, let Γ be the corresponding well-formed sequence with respect to TOC0P. Then Γ will be $\Gamma', x : A$, where Γ' is the sequence corresponding to Δ'. By condition 1 of Definition 21, $x \notin \mathrm{FV}(\Gamma', A)$. By condition 2 of Definition 21, $\Gamma' \vdash_0 A : s$. By induction hypothesis, $\Gamma' \vdash_2 A : s$, and an application of rule (Validity) gives us (4).

Case (Assumption). E is $M : A$, where $M : A \in \Delta$. If Δ is a well-formed environment, then $M : A$ is $x_i : A_i$ for some i ($1 \leq i \leq n$), where $\Delta \equiv \{x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n\}$, x_i does not occur free in $A_1, A_2, \ldots, A_{i-1}$, and

$$x_1 : A_1, x_2 : A_2, \ldots, x_{i-1} : A_{i-1} \vdash_0 A_i : s.$$

By the induction hypothesis,

$$x_1 : A_1, x_2 : A_2, \ldots, x_{i-1} : A_{i-1} \vdash_2 A_i : s,$$

which is (4) with $\Gamma \equiv x_1 : A_1, x_2 : A_2, \ldots, x_{i-1} : A_{i-1}$.

Case (ss'F). E is $(\Pi x : A \,.\, B) : s'$, where $x \notin \mathrm{FV}(\Delta, A)$. The premises are

$$\Delta \vdash_0 A : s \qquad \text{and} \qquad \Delta, x : A \vdash_0 B : s'.$$

It follows that $\Delta, x : A$ is well-formed with respect to TOC0P, and the corresponding sequence is $\Gamma, x : A$, where $\{\Gamma\}$ is Δ, and so by the induction hypothesis,

$$\Gamma, x : A \vdash_2 B : s'.$$

then (4) follows by (Product).

Case (Πe). E is $MN : [N/x]B$, and the premises are

$$\Delta \vdash_0 M : (\Pi x : A \,.\, B) \qquad \text{and} \qquad \Delta \vdash_0 N : A.$$

By the induction hypothesis, there is a well-formed sequence Γ such that $\{\Gamma\}$ is Δ and

$$\Gamma \vdash_2 M : (\Pi x : A \,.\, B) \qquad \text{and} \qquad \Gamma \vdash_2 N : A.$$

Then (4) follows by (Application).

Case (Πsi). E is $(\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)$, and the premises are

$$\Delta, x : A \vdash_0 M : B, \qquad \Delta, x : A \vdash_0 B : s, \qquad \Delta \vdash_0 A : s',$$

where $x \notin \mathrm{FV}(\Delta, A)$. By the induction hypothesis, there is a well-formed sequence Γ such that $\{\Gamma\}$ is Δ, and, since it also follows that $\Delta, x : A$ is well-formed, $\{\Gamma, x : A\}$ is $\Delta, x : A$, and $x \notin \mathrm{FV}(\Gamma, A)$, and, in addition,

$$\Gamma, x : A \vdash_2 M : B \qquad \text{and} \qquad \Gamma, x : A \vdash_2 B : s.$$

Then (4) follows by (Abstraction).

Case (Eq$'''$). Trivial by (Conversion). $\square$

Corollary 29 *A sequence* Γ *is well-formed with respect to TOC0P if and only if it is well-formed with respect to TOC2P.*

4 A-versions: relaxing the abstraction rule

The A-versions of these systems are obtained from the P-versions by omitting the second premise from the abstraction rule (rule (Abstraction) in the TOC2-like version and rule (Πsi) in the TOC0-like version). Paula Severi [38] has studied variants of PTSs with this change; she calls them "PTSs without the Π-condition." If $\lambda(S)$ is a regular PTS with specification S, then Severi calls the corresponding PTS without the Π-condition $\lambda^{\omega}(S)$. Similar variants are considered by van Benthem Jutting, McKinna, and Pollack [40], in particular with respect to the conditions under which they are equivalent to the corresponding ordinary PTSs.

Definition 30 *The* system TOC2A *is obtained from the system TOC2P by replacing rule (Abstraction) of Definition 1 by the following rule:*

$$\textit{(AAbstraction)} \quad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash (\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)} \quad \begin{array}{l}\text{Condition:}\\ x \notin \mathrm{FV}(\Gamma, A)\end{array}$$

If S is the specification for the calculus of constructions, then this is the system $\lambda^\omega(S)$ of [38], and the system $\lambda(S)$ is TOC2P.

As in Remark 2, we can omit the condition on x. We are stating the condition in this formulation for the reasons given in Remark 2.

The change from (Abstraction) to (AAbstraction) in this definition does not affect Remark 5, Remark 6, Lemma 10, Remark 11, Lemma 14, Lemma 3, Lemma 23, Lemma 24, Lemma 25, and Corollary 26 or their proofs, which apply to TOC2A as well as TOC2P.

Definition 31 *The* system TOC0A *is obtained from the system TOC0P by replacing rule (*Πsi*) of Definition 17 by the following rule:*

$$(A\Pi si)\quad \frac{\Delta, x : A \vdash M : B \qquad \Delta \vdash A : s}{\Delta \vdash (\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)} \quad \begin{array}{l}\text{Condition:}\\ x \notin \mathrm{FV}(\Delta, A)\end{array}$$

Remark 32 *In the style of Prawitz [10] used by Seldin in earlier papers, this rule would be written as follows:*

$$(A\Pi si)\quad \frac{\begin{array}{l}[x : A]\\ M : B \qquad\qquad\qquad A : s\end{array}}{(\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)}$$

Condition: *x does not occur free in A or in any undischarged assumption.*

Remark 33 *Note that rule (*$A\Pi si$*) is similar in a sense to the deduction theorem for restricted generality given by Bunder in [41, Theorem 6, p. 26],*[5] *which has the form*

$$\Delta, Xx \vdash Yx \quad\Rightarrow\quad \Delta, \mathsf{L}X \vdash \Xi XY,$$

where $x \notin$ $\mathrm{FV}(\Delta, X, Y)$*. The sense in which it is similar is that the only*

[5] Bunder acknowledges in a footnote an appearance of essentially the same theorem from different assumptions that appeared in Seldin [42, Theorem 4C1, p. 111], which actually appeared before [41]. However, Seldin had seen Bunder present his version of this theorem in a seminar in Amsterdam in early November, 1967, whereas Seldin did not begin his own work on this result until May, 1968. Furthermore, Seldin's system was later proved inconsistent [43].

restriction on the form of the rule

$$\Delta, Xx \vdash Yx \;\Rightarrow\; \Delta \vdash \Xi XY,$$

which is known to be inconsistent, applies to the antecedent. The corresponding rule for Curry's theory of functionality *is*

$$\Delta, Xx \vdash Y(Zx) \;\Rightarrow\; \Delta, \mathsf{L}X \vdash \mathsf{F}XYZ,$$

where, again, $x \notin \mathrm{FV}(\Delta, X, Y, Z)$. The theory of functionality was Curry's version of type assignment, he considered it closely related to his theory of restricted generality, and in modern notation this would be written

$$\Delta, x : X \vdash Zx : Y \;\Rightarrow\; \Delta, X : \mathsf{L}X \vdash Z : X \to Y.$$

In a Curry's generalized functionality,[6] *where $\mathsf{G}XYZ$ means $Z : (\Pi x : X . Y)$, the corresponding rule would be*

$$\Delta, Xx \vdash Yx(Zx) \;\Rightarrow\; \Delta, \mathsf{L}X \vdash \mathsf{G}XYZ,$$

in Curry's notation and, in modern notation,

$$\Delta, x : X \vdash Zx : Yx \;\Rightarrow\; \Delta, x : \mathsf{L} \vdash Z : (\Pi x : X . Y).$$

Church [45, Theorem I, p. 358] had a version of the deduction theorem for restricted generality with a restriction only on the antecedent:

$$\Delta, Xx \vdash Yx \;\Rightarrow\; \Delta, \Sigma X \vdash \Xi XY,$$

where ΣX means that there is a U such that XU. Otherwise, it appears that all restricted forms of the deduction theorem for restricted generality that appeared in print before that of [41] restricted both X and Y.

Remark 34 *The system TOC2A is not an APTS in the sense of Bunder and Dekkers [21] because the APTS systems have an additional restriction on the abstraction rule. For the calculus of constructions, the restriction says that $B \not\equiv \mathsf{Type}$. With this restriction, TOC2A is equivalent to TOC2P.*

We can now prove the equivalence of TOC2A and TOC0A. For this proof, $\Gamma \vdash_2 E$ will mean that $\Gamma \vdash E$ is provable in TOC2A, and $\Gamma \vdash_0 E$ will mean

[6] See [44].

that $\Gamma \vdash E$ is provable in TOC0A. The other conventions from the proofs of Theorems 27 and 28 will remain unchanged.

Theorem 35 *If*

$$\Gamma \vdash_2 E \tag{7}$$

then

$$\Gamma \vdash_0 E. \tag{8}$$

PROOF. The proof is the same as that of Theorem 27 except for the case for (Abstraction) in the induction step, which must be replaced as follows: E is $(\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)$, and the premise is

$$\Gamma, x : A \vdash_2 M : B.$$

Furthermore, by Lemma 7, it follows that there is a subdeduction of (7)

$$\Gamma \vdash_2 A : s.$$

By the induction hypothesis to both of these, we have

$$\Gamma, x : A \vdash_0 M : B \qquad \text{and} \qquad \Gamma \vdash_0 A : s,$$

and (8) follows by rule (AΠsi). $\square$

Theorem 36 *If Δ is a well-formed environment with respect to TOC0A, and if*

$$\Delta \vdash_0 E \tag{9}$$

then there is a sequence Γ such that $\{\Gamma\}$ is Δ and (7) holds.

PROOF. The proof is the same as that of Theorem 28 except that the case for (Πsi) must be replaced with the following case for (AΠsi): E is $(\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)$ and the premises are

$$\Delta, x : A \vdash_0 M : B \qquad \text{and} \qquad \Delta \vdash_0 A : s,$$

where $x \notin \mathrm{FV}(\Delta, A)$. By the induction hypothesis, there is a well-formed sequence Γ such that $\{\Gamma\}$ is Δ, and, since it also follows that $\Delta, x : A$ is well-formed, $\{\Gamma, x : A\}$ is $\Delta, x : A$, and $x \notin \mathrm{FV}(\Gamma, A)$, and, in addition,

$$\Gamma, x : A \vdash_2 M : B \qquad \text{and} \qquad \Gamma \vdash_2 A : s.$$

Then (7) follows by (AAbstraction). $\square$

Corollary 37 *A sequence Γ is well-formed with respect to TOC0A if and only if it is well-formed with respect to TOC2A.*

It remains to state the relationship between the P-versions and the A-versions. Part of this is easy as noted by Severi [38] in the second lst paragraph of §2.

Theorem 38 *Every judgment valid in TOC2P (respectively TOC0P) is valid in TOC2A (respectively TOC0A).*

PROOF. By an easy induction on the proof of the judgment in TOC2P (respectively TOC0P). $\square$

To state the converse to this relationship, we need to recall some results from [12, §2].

Recall, first, that by [12, Definition 2] terms which reduce to the form

$$(\Pi x_1 : A_1 \,.\, \Pi x_2 : A_2 \,.\, \ldots \,.\, \Pi x_n : A_n \,.\, \mathsf{Prop})$$

are called *contexts*, whereas terms which reduce to the form

$$(\Pi x_1 : A_1 \,.\, \Pi x_2 : A_2 \,.\, \ldots \,.\, \Pi x_n : A_n \,.\, \mathsf{Type})$$

are called *supercontexts*. Terms in the form

$$(\Pi x_1 : A_1 \,.\, \Pi x_2 : A_2 \,.\, \ldots \,.\, \Pi x_n : A_n \,.\, \mathsf{Type})$$

are called *standard supercontexts* or *supercontexts in standard form*. The following result is similar to [12, Theorem 8]:[7]

Theorem 39 *If $\Gamma \vdash M : A$ in TOC2P, then exactly one of the following*

[7] See also [6, Theorem 4.11] and [13, p. 433f].

holds:

(1) $\Gamma \vdash A : s$, *or*
(2) A *is* Type.

PROOF. Since $\Gamma \not\vdash \mathsf{Type} : s$, we need only prove $\Gamma \vdash A : s$ or A is Type, which we prove by induction on the deduction of $\Gamma \vdash M : A$ in TOC2P. The basis is trivial, since $M : A$ is $\mathsf{Prop} : \mathsf{Type}$.

For the induction step, there are cases by the last rule of the deduction. If the rule is (Validity), the result is clear since $M : A$ is $\mathsf{Prop} : \mathsf{Type}$. If the rule is (Variable), the result follows from the fact that Γ is well-formed and A is a type in Γ, and by (Weakening), which holds by Lemma 10. If the rule is (Product), the result is clear since A is s, which is either Prop or Type. If the rule is (Abstraction) or (Conversion), the right premise (and, in the case of (Abstraction), an application of rule (Product)), gives us the result.

This leaves the case for (Application). Here, M is PN, A is $[N/x]C$, and the premises are

$$\Gamma \vdash P : (\Pi x : B \,.\, C) \qquad \text{and} \qquad \Gamma \vdash N : B.$$

By the induction hypothesis and the first premise, $\Gamma \vdash (\Pi x : B \,.\, C) : s$ (since it cannot be Type), and by Lemma 14, $\Gamma, x : B \vdash C : s$. Hence, by [19, Lemma 5.2.11], $\Gamma \vdash A : s$. $\Box$

For the systems TOC2A and TOC0A, this result must be generalized. For the rule (AAbstraction) does not exclude a conclusion with a supercontext on the right of the colon, and if supercontexts can occur there, then in the case for (Application), we have to allow for the possibility that $(\Pi x : B \,.\, C)$ is a supercontext, in which case $[N/x]C$ will also be a supercontext. With these modifications of the above proof, and noting that Barendregt's substitution lemma [19, Lemma 5.2.11] can be proved for TOC2A, we get the following result:

Theorem 40 *If* $\Gamma \vdash M : A$ *holds in TOC2A, then exactly one of the following holds:*

(1) $\Gamma \vdash A : s$, *or*
(2) A *is a standard supercontext.*

Severi [38] proved this as Theorem 4.9. Note that this theorem holds for TOC0A if Δ is a well-formed environment.

Remark 41 *TOC2A (respectively TOC0A) is actually stronger than TOC2P (respectively TOC0P), since in the former we can prove*

$$(\lambda x : \mathsf{Prop} \,.\, \mathsf{Prop}) : (\Pi x : \mathsf{Prop} \,.\, \mathsf{Type}),$$

but we cannot prove this in the latter (by the generation lemma, [19, Lemma 5.2.13]). Severi [38] noted this with a similar example in the last paragraph of §2.

Remark 42 *The difference between the P-versions and the A-versions is that in the latter, supercontexts that differ from* Type *may occur as the type of a term, whereas these cannot occur in the former.*

Remark 43 *[12, Theorem 8] says that if* $\Gamma \vdash M : A$ *where* Γ *is a well-formed environment, then exactly one of the following holds:*

(1) $\Gamma \vdash A : \mathsf{Prop}$,
(2) $\Gamma \vdash A : T$*, where* T *is a supercontext, or*
(3) A *is a supercontext.*

That theorem applies to a system, called TOC0 in [12], of which both TOC2P and TOC2A (and, equivalently, both TOC0P and TOC0A) are subsystems. In that version of TOC0, assumptions are allowed of the form $x : T$*, where* T *is a supercontext. (Note that [12, Definition 3] differs from Definition 21 above precisely in allowing among well-formed sets to include assumptions of the form* $x : T$ *where* T *is a supercontext.) This is not allowed in well-formed environments in any of the systems considered in this paper. (We are dealing with supercontexts here instead of standard supercontexts because this version of TOC0 is an AC system. See §6.)*

Remark 44 *It might appear that we can get a PTS equivalent to TOC2A by adding a sort* Type_2*, adding the axiom* $\mathsf{Type} : \mathsf{Type}_2$*, and adding rules* $(\mathsf{Type}, \mathsf{Type}_2)$ *and* $(\mathsf{Type}_2, \mathsf{Type}_2)$*. It is true that in this PTS, supercontexts other than* Type *can occur as types of terms. However, this PTS adds too much, since it allows assumptions of the form* $x : \mathsf{Type}$ *or* $x : T$ *for a supercontext* T *in well-formed environments.*

5 C-versions: relaxing the conversion rule

The C-versions of these systems are obtained from the P-versions primarily by omitting the third premise of the conversion rules (rule (Conversion) in TOC2P and rule (Eq‴) in TOC0P). This allows any term that converts to a type to be a type. Since we want to preserve Theorem 39, we also need to add a rule that any term convertible to a term in a sort is also in that sort.

Definition 45 *The* system TOC2C *is obtained from TOC2P by replacing the rule (Conversion) of Definition 1 by the rule*

$$\textit{(CConversion)} \qquad \frac{\Gamma \vdash M : A \quad A =_\beta B}{\Gamma \vdash M : B}$$

and by adding the rule

$$\textit{(sort-Conversion)} \qquad \frac{\Gamma \vdash A : s \quad A =_\beta B}{\Gamma \vdash B : s}$$

Remark 46 *As pointed out in Remark 2, the conditions on variables in rules (Validity), (Product), and (Abstraction) are needed here. This is because occurrences of free variables are not invariant of conversion.*

The change from (Conversion) to (CConversion) and the addition of (sort-Conversion) in this definition do not affect Remark 5, Remark 6, Lemma 10, Remark 11, Remark 12, Lemma 13, Lemma 14, Lemma 3, Lemma 23, Lemma 24, Lemma 25, and Corollary 26, which apply to TOC2C as well as TOC2P.

Definition 47 *The* system TOC0C *is obtained from the system TOC0P by replacing the rule (Eq‴) of Definition 17 by the rule*

$$\textit{(Eq}'') \qquad \frac{\Delta \vdash M : A \quad A =_\beta B}{\Delta \vdash M : B}$$

and adding the rule

$$\textit{(Eq}'s) \qquad \frac{\Delta \vdash A : s \quad A =_\beta B}{\Delta \vdash B : s}$$

Remark 48 *In TOC0C, it is easy to prove that*

$$\Delta \ \vdash \ M : A \ \& \ \Delta =_\beta \Delta' \ \Rightarrow \ \Delta' \ \vdash \ M : A.$$

It follows from this and the alterations allowed by the remarks and lemmas that TOC0C is equivalent to an SPTS in the sense of Bunder and Dekkers [21].

Remark 49 *The rule*

$$\frac{\Gamma \ \vdash \ M : A \quad M =_\beta N}{\Gamma \ \vdash \ N : A}$$

is not admissible in TOC2C (or in any of the systems we consider in this paper). The Subject-Reduction Theorem*, which says that*

$$\frac{\Gamma \ \vdash \ M : A \quad M \rhd_\beta N}{\Gamma \ \vdash \ N : A}$$

is admissible, does hold for all these systems, but the Subject-Expansion Theorem*, which asserts the admissibility of*

$$\frac{\Gamma \ \vdash \ M : A \quad N \rhd_\beta M}{\Gamma \ \vdash \ N : A,}$$

is not admissible without severe and complicated restrictions. The reasons are the same ones given for simple type assignment (basic functionality) in [46, §9C3]: 1) if a subterm N is cancelled in a contraction, as in the contraction of $(\lambda x : A \,.\, M)N$ (where $x \notin \mathrm{FV}(M)$) to M, the fact that M is assigned a type does not guarantee that N is assigned one, and 2) if a subterm N is duplicated in a contraction from $(\lambda x : A \,.\, M)N$ to $[N/x]M$, the fact that $[N/x]M$ is assigned a type does not guarantee that N is assigned the same type in all occurrences, and if it is not then it will not, in general, be possible to assign a type to $(\lambda x : A \,.\, M)$.

This rule can be added to TOC2C and TOC0C, where it would take the place of its special case, (sort-Conversion). Seldin made this change to the original TOC0 that is mentioned in Remark 43, calling the resulting system TOCE, and proved that if $\Gamma \ \vdash \ M : A$ in TOCE, then there is a term M' such that $M =_\beta M'$ and $\Gamma \ \vdash \ M' : A$ in that original TOC0; see [14, Theorem 1]. Although we have not checked the details, we believe that the same result holds for TOC0AC, TOC2AC, TOC2C, and TOC0C.

Theorems 27 and 28 hold word-for-word if the references to TOC2P and TOC0P are replaced respectively by TOC2C and TOC0C. The proofs are obtained by replacing the case for the conversion rule in each theorem and adding the trivial case for the new conversion rule in each. This proves

Theorem 50 *If*

$$\Gamma \vdash_2 E, \tag{10}$$

then

$$\Gamma \vdash_0 E. \tag{11}$$

Theorem 51 *If Δ is a well-formed environment with respect to TOC0C, and if*

$$\Delta \vdash_0 E, \tag{12}$$

then there is a sequence Γ such that $\{\Gamma\}$ is Δ and (10) holds.

Now for the relation between TOC2C and TOC2P. If Γ is

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n,$$

we say that $\Gamma' =_\beta \Gamma$ if Γ' is

$$x_1 : A'_1, x_2 : A'_2, \ldots, x_n : A'_n$$

and $A'_i =_\beta A_i$ for $i = 1, 2, \ldots n$.

Theorem 52 *If*

$$\Gamma \vdash M : A \tag{13}$$

holds in TOC2C, then there are terms A' and M' and a sequence Γ', such that $\Gamma' =_\beta \Gamma$, $A' =_\beta A$ and $M' =_\beta M$, where if A is a sort then A' is the same sort and if M is a variable then M' is the same variable, and

$$\Gamma' \vdash M' : A' \tag{14}$$

in TOC2P.

PROOF. By induction on the proof of (13).

Basis: Trivial, since $\Gamma' \equiv \Gamma$ is empty, $A' \equiv \mathsf{Type} \equiv A$, and $M' \equiv \mathsf{Prop} \equiv M$.

Induction step: The cases are by the last inference in the derivation of (13).

Case (Validity). Γ is $\Gamma_1, x : B$; $M : A$ is $\mathsf{Prop} : \mathsf{Type}$; $x \notin \mathrm{FV}(\Gamma_1, B)$ and the premise is

$$\Gamma_1 \vdash B : s.$$

By the hypothesis of induction, there are Γ_1' and B' such that $\Gamma_1' =_\beta \Gamma_1$ and $B' =_\beta B$ and

$$\Gamma_1' \vdash B' : s$$

in TOC2P. If we put $\Gamma' \equiv \Gamma_1', x : B'$; $M' \equiv M$; and $A' \equiv A$, then $x \notin \mathrm{FV}(\Gamma_1', B')$ by [19, Lemma 5.2.8], and (14) follows by (Validity).

Case (Variable). Γ is $\Gamma_1, x : A, \Gamma_2$; M is x; and the premise is

$$\Gamma_1, x : A, \Gamma_2 \vdash \mathsf{Prop} : \mathsf{Type}.$$

By the induction hypothesis, there are Γ_1', A', and Γ_2' such that $\Gamma_1' =_\beta \Gamma_1$, $A' =_\beta A$, and $\Gamma_2' =_\beta \Gamma_2$, and

$$\Gamma_1', x : A', \Gamma_2' \vdash \mathsf{Prop} : \mathsf{Type}$$

holds in TOC2P. If we put $M' \equiv x$ and $\Gamma' \equiv \Gamma_1', x : A', \Gamma_2'$, then (14) follows by (Variable).

Case (Product). Then M is $(\Pi x : B.C)$, A is s, $x \notin \mathrm{FV}(\Gamma, B)$, and the premise is

$$\Gamma, x : B \vdash C : s.$$

By the induction hypothesis, there are Γ', B', and C' such that $\Gamma' =_\beta \Gamma$, $B' =_\beta B$, and $C' =_\beta C$ and

$$\Gamma', x : B' \vdash C' : s$$

holds in TOC2P. If we put $A' \equiv (\Pi x : B' \,.\, C')$, then (14) follows by (Product).

Case (Abstraction). M is $(\lambda x : B \,.\, N)$, A is $(\Pi x : B \,.\, C)$, and the premises are

$$\Gamma, x : B \vdash N : C \qquad \text{and} \qquad \Gamma, x : B \vdash C : s.$$

By the induction hypothesis, there are Γ'', B'', N', C'', Γ''', B''', and C''' such that $\Gamma =_\beta \Gamma'' =_\beta \Gamma'''$, $B =_\beta B'' =_\beta B'''$, $N =_\beta N'$, and $C =_\beta C'' =_\beta C'''$ such that

$$\Gamma'', x : B'' \vdash N' : C'' \qquad \text{and} \qquad \Gamma''', x : B''' \vdash C''' : s$$

hold in TOC2P. By the Church-Rosser Theorem, there are Γ', B', C', such that $\Gamma'' \triangleright \Gamma'$, $\Gamma''' \triangleright \Gamma'$, $B'' \triangleright B'$, $B''' \triangleright B'$, $C'' \triangleright C'$, and $C''' \triangleright C'$. By a combination of the proof of [19, Lemma 5.2.15] and [19, Corollary 5.2.16], we have

$$\Gamma', x : B' \vdash N' : C' \qquad \text{and} \qquad \Gamma', x : B' \vdash C' : s$$

in TOC2P, and setting $M' \equiv \lambda x : B' \,.\, N'$ and $A' \equiv \Pi x : B' \,.\, C'$, we get (14) by (Abstraction).

Case (Application). M is PN, A is $[N/x]C$, and the premises are

$$\Gamma \vdash P : (\Pi x : B \,.\, C) \qquad \text{and} \qquad \Gamma \vdash N : C.$$

By the induction hypothesis, the Church-Rosser Theorem, [19, Lemma 5.2.15], and the proof of [19, Corollary 5.2.16], there are (as in the case for (Abstraction) above) Γ', B', C', P', and N' such that $\Gamma =_\beta \Gamma'$, $B =_\beta B'$, $C =_\beta C'$, $P =_\beta P'$, $N =_\beta N'$, and

$$\Gamma' \vdash P' : (\Pi x : B' \,.\, C') \qquad \text{and} \qquad \Gamma' \vdash N' : C'.$$

holds in TOC2P. If we put $M' \equiv (P'N')$ and $A' \equiv [N'/x]C'$, we get (14) by (Application).

Case (CConversion). The premises are

$$\Gamma \vdash M : B \qquad \text{and} \qquad A =_\beta B.$$

By the the first of these and the induction hypothesis, there are Γ', M', and B' such that $\Gamma =_\beta \Gamma'$, $M =_\beta M'$, and $B =_\beta B'$ and

$$\Gamma' \vdash M' : B'$$

holds in TOC2P. Since $A =_\beta B'$, we can put $A' \equiv B'$, and this is (14).

Case (sort-Conversion). Here $A \equiv s$, $M \equiv B$, and the premises are

$$\Gamma \vdash C : s \qquad \text{and} \qquad C =_\beta B.$$

By the first of these and the induction hypothesis, there are $\Gamma' =_\beta \Gamma$ and $C' =_\beta C$ such that

$$\Gamma' \vdash C' : s$$

holds in TOC2P. Since $B =_\beta C'$, this is (14). □

Remark 53 *An easier proof would use [21, Theorem 5.4] of Bunder and Dekkers. We have previously shown that TOC0P is equivalent to TOC2P, and this theorem shows that TOC2P is equivalent modulo conversion to TOC0C. Furthermore, TOC2C is equivalent to TOC0C.*

Because the Subject-Reduction Theorem holds for TOC2P, we have

Corollary 54 *In Theorem 52, M', A', and the types of Γ' can be assumed to be in normal form.*

Remark 55 *It seems obvious that with the change from the P and A systems to the C systems the proof of strong normalization breaks down. However, it is possible to prove normalization for these systems, and strong normalization fails only for the types of bound variables. The proof is based on deduction reductions. In TOC0C, the reduction step takes*

$$\cfrac{\cfrac{\cfrac{\begin{array}{c}D_1(x)\\ \Delta, x : A \vdash M : B\end{array} \quad \begin{array}{c}D_2(x)\\ \Delta, x : A \vdash B : s\end{array} \quad \begin{array}{c}D_3\\ \Delta \vdash A : s'\end{array}}{\Delta \vdash (\lambda x : A \,.\, M) : (\Pi x : A \,.\, B)}\ (\Pi si)}{\Delta \vdash (\lambda x : A \,.\, M) : (\Pi x : C \,.\, D)}\ (\mathrm{Eq}'') \qquad \begin{array}{c}D_4\\ \Delta \vdash N : C\end{array}}{\begin{array}{c}\Delta \vdash (\lambda x : A \,.\, M)N : [N/x]D\\ D_5\end{array}}\ (\Pi e)$$

to

$$\cfrac{\begin{array}{c}\cfrac{\begin{array}{c}D_4'\\ \Delta' \vdash N : C\end{array}}{\Delta' \vdash N : A}\ (\mathrm{Eq}'')\\ D_1(N)\\ \Delta \vdash [N/x]M : [N/x]B\end{array}}{\begin{array}{c}\Delta \vdash [N/x]M : [N/x]D\\ D_5'\end{array}}\ (\mathrm{Eq}'')$$

where $A =_\beta C$, $B =_\beta D$, $x \notin \mathrm{FV}(\Delta, A)$, D_5' is obtained from D_5 by replacing appropriate occurrences of $(\lambda x : A \,.\, M)N$ by $[N/x]M$, $\Delta \subset \Delta'$, and D_4' is obtained from D_4 as described in Remark 18. In the style of the earlier works by Seldin (see Remark 20), this would be written as a reduction of

$$\dfrac{\dfrac{\dfrac{\begin{matrix}1\\ [x:A]\\ D_1(x)\\ M:B\end{matrix} \qquad \begin{matrix}2\\ [x:A]\\ D_2(x)\\ B:s\end{matrix} \qquad \begin{matrix} D_3\\ A:s'\end{matrix}}{(\lambda x:A\,.\,M):(\Pi x:A\,.\,B)}\,(\Pi s\mathrm{i}-1-2)}{(\lambda x:A\,.\,M):(\Pi x:C\,.\,D)}\,(\mathrm{Eq}'') \qquad \begin{matrix}D_4\\ N:C\end{matrix}}{\begin{matrix}(\lambda x:A\,.\,M)N:[N/x]D\\ D_5\end{matrix}}\,(\Pi\mathrm{e})$$

to

$$\dfrac{\begin{matrix}\dfrac{\begin{matrix}D_4\\ N:C\end{matrix}}{N:A}\,(\mathrm{Eq}'')\\ D_1(N)\\ [N/x]M:[N/x]B\end{matrix}}{\begin{matrix}[N/x]M:[N/x]D\\ D_5'\end{matrix}}\,(\mathrm{Eq}'')$$

where D_5' is obtained from D_5 by replacing $(\lambda x : A \,.\, M)N$ by $[N/x]M$. In TOC2C, the reduction step takes

$$\dfrac{\dfrac{\dfrac{\Gamma, x:A \vdash M:B \quad \Gamma, x:A \vdash B:s}{\Gamma \vdash (\lambda x:A\,.\,M):(\Pi x:A\,.\,B)}\,(\mathrm{Abstraction})}{\Gamma \vdash (\lambda x:A\,.\,M):(\Pi x:C\,.\,D)}\,(\mathrm{CConversion}) \qquad \Gamma \vdash N:C}{\Gamma \vdash (\lambda x:A\,.\,M)N:[N/x]D}\,(\mathrm{Application})$$

to

$$\dfrac{\dfrac{\Gamma, x:A \vdash M:B \quad \dfrac{\Gamma \vdash N:C}{\Gamma \vdash N:A}\,(\mathrm{CConversion})}{\Gamma \vdash [N/x]M:[N/x]B}\,(\mathrm{Substitution\ \ Lemma})}{\Gamma \vdash [N/x]M:[N/x]D.}\,(\mathrm{CConversion})$$

Strong normalization for these deduction reductions is proved in [12, Theorem 11] for an extension of TOC0AC, where it is assumed that Δ is a well-formed environment. The normalization theroem for terms is proved in [12, Theorem 12], and the proof that the only subterms that are not strongly normalizing are those that occur as the types of bound variables or those whose types are either

Prop *or* Type *is given in [12, Corollary 12.1]. Although all of these results are proved for an extension of TOC0AC, they hold for TOC0C because it is a subsystem of TOC0AC and for TOC2C by Theorems 50 and 51 above.*

Remark 56 *It may appear that typechecking is lost in the C versions. However, this is not the case. By Corollary 54, typechecking can be applied to TOC2C by reducing the term involved to its normal form (which it has by Remark 55) and typechecking in TOC2P. Since TOC0C is a subsystem of TOC2AC, a similar conclusion follows by [12, Corollary 12.3].*

6 AC-versions: relaxing both the abstraction and conversion rules

Definition 57 *The* system TOC2AC *is obtained in one of three ways (all of which are equivalent):*

(1) from TOC2P by replacing (Abstraction) by (AAbstraction), replacing (Conversion) by (CConversion), and by adding (sort-Conversion).
(2) from TOC2A by replacing (Conversion) by (CConversion), and by adding (sort-Conversion).
(3) from TOC2C by replacing (Abstraction) by (AAbstraction).

The changes to not affect Remark 5, Remark 6, Lemma 10, Remark 11, Lemma 3, Lemma 23, Lemma 24, Lemma 25, and Corollary 26 or their proofs, which apply to TOC2AC as well as TOC2P.

Definition 58 *the* system TOC0AC *is obtained in one of three ways (all of which are equivalent):*

(1) from TOC0P by replacing (Πsi) by (AΠsi), replacing (Eq'') by (Eq'), and by adding ($Eq's$).
(2) from TOC0A by replacing (Eq'') by (Eq'), and by adding ($Eq's$).
(3) from TOC0C by replacing (Πsi) by (AΠsi).

To prove the equivalence of TOC2AC and TOC0AC, take alternative 2 in each of Definitions 57 and 58. Then Theorems 35 and 36 hold word-for-word if the references to TOC2A and TOC0A are replaced respectively by TOC2AC and TOC0AC. The proofs are obtained by replacing the case for the conversion rule in each theorem and adding the trivial case for the new conversion rule in each. This proves

Theorem 59 *If*

$$\Gamma \vdash_2 E, \tag{15}$$

then

$$\Gamma \vdash_0 E. \tag{16}$$

Theorem 60 *If Δ is a well-formed environment with respect to TOC0AC, and if*

$$\Delta \vdash_0 E, \tag{17}$$

then there is a sequence Γ such that $\{\Gamma\}$ is Δ and (4) holds.

For the relationship between the AC systems and the other systems, let us begin with the definition by taking alternative 3 in each of Definitions 57 and 58. By Theorem 52, Theorem 39 holds for TOC2C in the following modified form: if $\Gamma \vdash M : A$ in TOC2C, then either $\Gamma \vdash A : s$ in TOC2C or else $A =_\beta \mathsf{Type}$. For the same reason, the proof of Theorem 40 will also carry over to TOC2AC: if $\Gamma \vdash M : A$ in TOC2AC, then either $\Gamma \vdash A : s$ in TOC2AC or else A is a supercontext (i.e., A converts to a standard supercontext). This means that we can sum up the relations between the AC systems and the others as follows:

Theorem 61 *(1) If $\Gamma \vdash M : A$ in TOC2AC, and if A is not a supercontext, then there are $\Gamma' =_\beta \Gamma$, $M' =_\beta M$, and $A' =_\beta A$ such that $\Gamma' \vdash M' : A'$ in TOC2P.*

(2) If $\Gamma \vdash M : A$ in TOC2AC, and if A is not a supercontext distinct from Type, then $\Gamma \vdash M : A$ in TOC2C.

(3) If $\Gamma \vdash M : A$ in TOC2AC, then there are $\Gamma' =_\beta \Gamma$, $M' =_\beta M$, and $A' =_\beta A$ such that $\Gamma' \vdash M' : A'$ in TOC2A.

Remark 62 *The additional restriction $B \neq_\beta \mathsf{Type}$ on the abstraction rule, which amounts to the same as the type of A not being a supercontext distinct from Type in Theorem 61 1 and 2, converts TOC0AC into an SAPTS in the sense of Bunder and Dekkers [21]. This SAPTS, with the calculus of constructions specification, is equivalent to the corresponding PTS and so to TOC0P.*

Remark 63 *In his earliest work on the calculus of constructions, Seldin [6] included a rule*

$(\equiv'_\alpha)$	$\dfrac{M : A}{N : A}$	Condition: *N is obtained from M by changes of bound variables.*

At the time, it was thought that this was the only way to obtain deductions of the form

$$\Gamma \vdash (\lambda x : A \,.\, M) : (\Pi y : A \,.\, B).$$

However, such deductions can be obtained using the rule of (Conversion) in one of its forms, so this rule is unnecessary.

Remark 64 *On the subject of normalization, see Remark 55, the results of which apply to the AC versions as well as the C versions.*

Remark 65 *Typechecking holds for the AC systems. See Remark 56.*

7 Conclusion

In this paper, we have compared six different versions of the calculus of constructions. As we pointed out in the introduction, which version one will want to use will depend on one's purposes. If one wants to implement a version in which fast type-checking is important, one will probably prefer either TOC2P or TOC2A.[8] On the other hand, in about 1987, Garrel Pottinger remarked to Seldin that proving the strong normalization theorem was easier with TOC0AC than with a P or A version.[9] Furthermore, since the AC versions are the strongest, consistency results proved for them will carry over to the other versions. Thus, a version like TOC0AC may be more useful for some proof-theoretic purposes.

References

[1] T. Coquand, Une théorie des constructions, Ph.D. thesis, University of Paris VII (1985).

[2] T. Coquand, Metamathematical investigations of a calculus of constructions, february 9, 1987. Privately circulated (1987).

[3] T. Coquand, Metamathematical investigations of a calculus of constructions, in: P. Odifreddi (Ed.), Logic and Computer Science, Academic Press, 1990, pp. 91–122.

[8] Severi [38] defined a semi-algorithm for type-checking in her $\lambda^{\omega}(S)$, which, in the case of the calculus of constructions, is equivalent to our TOC2A.

[9] Pottinger was, of course, referring to Seldin's proof of strong normalization for *deductions*.

[4] T. Coquand, G. Huet, Concepts mathématiques et informatiques formalisés dans le calcul des constructions, in: C. Berline, E. Bouscaren, M. Dickmann, J.-L. Krivine, D. Lascar, M. Parigot, E. Pelz, G. Sabbagh (Eds.), Logic Colloquium '85: Proceedings of the Colloquium Held in Orsay, July 7-13, 1985, Vol. 122 of Studies in Logic and the Foundations of Mathematics, North-Holland, Amsterdam and New York, 1986, pp. 123–146.

[5] T. Coquand, G. Huet, The calculus of constructions, Information and Computation 76 (1988) 95–120.

[6] J. P. Seldin, MATHESIS: The mathematical foundation for ULYSSES, Interim Report RADC-TR-87-223, Rome Air Development Center, Air force Systems Command, Griffiss Air Force Base, New York, actually published in 1988. (November 1987).

[7] G. Gentzen, Untersuchungen über das logische Schliessen, Mathematische Zeitschrift 39 (1934) 176–210, 405–431, translated in Sabo (ed.), *The Collected Papers of Gerhard Gentzen* as "Investigations into Logical Deduction," pages 68–131, North-Holland, Amsterdam and London, 1969.

[8] H. B. Curry, A Theory of Formal Deducibility, Notre Dame Mathematical Lectures Number 6, University of Notre Dame, Notre Dame, Indiana, 1950, second edition, 1957.

[9] H. B. Curry, Foundations of Mathematical Logic, McGraw-Hill Book Company, Inc., New York, San Francisco, Toronto, and London, 1963, reprinted by Dover, 1977 and 1984.

[10] D. Prawitz, Natural Deduction, Almqvist & Wiksell, Stockholm, Göteborg, and Uppsala, 1965.

[11] J. P. Seldin, A sequent calculus for type assignment, Journal of Symbolic Logic 42 (1977) 11–28.

[12] J. P. Seldin, On the proof theory of Coquand's calculus of constructions, Annals of Pure and Applied Logic 83 (1997) 23–101.

[13] J. P. Seldin, Coquand's calculus of constructions: a mathematical foundation for a proof development system, Formal Aspects of Computing 4 (1992) 425–441.

[14] J. P. Seldin, A Gentzen-style sequent calculus of constructions with expansion rules, Theoretical Computer Science 243 (2000) 199–215.

[15] J. P. Seldin, On lists and other abstract data types in the calculus of constructions, Mathematical Structures in Computer Science 10 (2000) 261–276, special issue in honor of J. Lambek.

[16] J. P. Seldin, Extensional set equality in the calculus of constructions, Journal of Logic and Computation 11 (3) (2001) 483–493, presented at Festival Workshop in Foundations and Computations held at Heriot-Watt University, Edinburgh, 16-18 July, 2000.

[17] S. Berardi, Towards a mathematical analysis of the Coquand-Huet calculus of constructions and the other systems of the Barendregt cube, Tech. rep., Department of Computer Science, Carnegie-Mellon University and Dipartimento di Matematica, Universita di Torino (1988).

[18] J. Terlouw, Een nadere bewijstheoretische analyse van GTS's, Tech. rep., Department of Computer Science, Catholic University, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands (1989).

[19] H. Barendregt, Lambda calculi with types, in: S. Abramsky, D. Gabbay, T. Maibaum (Eds.), Handbook of Logic in Computer Science, Vol. 2, Oxford University Press, 1992, pp. 117–309.

[20] M. Bunder, W. Dekkers, Equivalences between pure type systems and systems of illative combinatory logic, submitted to the *Notre Dame Journal of Formal Logic*.

[21] M. Bunder, W. Dekkers, Pure type systems with more liberal rules, Journal of Symbolic Logic 66 (2001) 1561–1580.

[22] M. Bunder, W. Dekkers, H. Geuvers, Equivalences between illative combinatory logics and pure type systems, in: Study of Nonclassical Logic based on Sequent Calculi and Kripke Semantics, RIMS Kokyuroku, Kyoto, 1997, pp. 119–135.

[23] Z. Luo, ECC, an extended calculus of constructions, in: Proceedings of the Fourth Annual Symposium on Logic in Computer Science, June 1989, Asilomar, California, U.S.A., 1989.

[24] Z. Luo, An extended calculus of constructions, Ph.D. thesis, University of Edinburgh (1990).

[25] T. Altenkirch, Constructions, inductive types and strong normalization, Ph.D. thesis, University of Edinburgh (1993).

[26] B. Werner, Une théorie des constructions inductives, Ph.D. thesis, Université Paris 7 (1994).

[27] F. Barbanera, Adding algebraic rewriting to the calculus of constructions: Strong normalization preserved, in: Conditional and Typed Rewriting Systems (Montreal, PQ, 1990), Vol. 516 of Lecture Notes in Computer Science, Springer, Berlin, 1991, pp. 260–271.

[28] F. Barbanera, Combining term rewriting and type assignment systems, International Journal of Foundations of Computer Science 1 (1990) 165–184.

[29] F. Barbanera, M. Fernández, Combining first and higher order rewrite systems with type assignment systems, in: Typed Lambda Calculus and Applications (Utrecht, 1993), Vol. 664 of Lecture Notes in Computer Science, Springer, 1993, pp. 60–74.

[30] F. Barbanera, M. Fernández, Modularity of termination and confluence in combinations of rewrite systems with λ_ω, in: Automata, languages and programming (Lund, 1993), Vol. 820 of Lecture Notes in Computer Science, Springer, 1993, pp. 657–668.

[31] F. Blanqui, Definitions by rewriting in the calculus of constructions, in: Proceedings of the 16th IEEE Symposium on Logic in Computer Science, Boston, MA, 2001, IEEE, 2001, pp. 9–18.

[32] F. Blanqui, Rewriting modulo in deduction modulo, in: Rewriting Techniques and Applications (Valencia, Spain, 2003), Vol. 2706 of Lecture Notes in Computer Science, Springer, 2003, pp. 395–409.

[33] D. Walukiewicz-Chrzaszcz, Termination of rewriting in the calculus of constructions, Journal of Functional Programming 13 (2) (2003) 339–414.

[34] H. Geuvers, the church-rosser property for $\beta\eta$-reduction in typed λ-calculi, in: Proceedings of the Seventh Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society, IEEE Computer Society Press, 1992, pp. 453–460.

[35] H. Geuvers, B. Werner, On the church-rosser property for expressive type systems and its consequences for their metatheoretic study, in: Proceedings of the Ninth Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society, IEEE Computer Society Press, 1994, pp. 320–329.

[36] G. Barthe, M. H. Sørensen, Domain-free pure type systems, Journal of Functional Programming 10 (5) (2000) 412–452.

[37] G. Pottinger, Strong normalization for terms of the theory of constructions, Tech. Rep. TR 11–7, Odyssey Research Associates (December 1987).

[38] P. Severi, Type inference for pure type systems, Information and Computation 143 (1) (1998) 1–23.

[39] F. B. Fitch, Symbolic Logic, The Ronald Press Company, New York, 1952.

[40] L. S. van Benthem Jutting, J. McKinna, R. Pollack, Checking algorithms for pure type systems, in: Types for Proofs and Programs: International Workshop TYPES'93, Nijmegen, May 1993, Selected Papers, Vol. 806 of Lecture Notes in Computer Science, Springer, Berlin, 1994, pp. 19–61.

[41] M. Bunder, Set theory based on combinatory logic, Ph.D. thesis, University of Amsterdam (1969).

[42] J. P. Seldin, Studies in illative combinatory logic, Ph.D. thesis, University of Amsterdam (1968).

[43] M. W. Bunder, Some inconsistencies in illative combinatory logic, Zeitschrift für mathematische Logik und Grundlagen der Mathematik 20 (1974) 199–201.

[44] J. P. Seldin, Progress report on generalized functionality, Annals of Mathematical Logic 17 (1979) 29–59.

[45] A. Church, A set of postulates for the foundation of logic, Annals of Mathematics 33 (1932) 346–366.

[46] H. B. Curry, R. Feys, Combinatory Logic, Vol. 1, North-Holland Publishing Company, Amsterdam, 1958, reprinted 1968 and 1974.