DEPENDENT TYPES SIMPLIFIED

TRISTAN BICE

ABSTRACT. We present two logical systems based on dependent types that are comparable to ZFC, both in terms of simplicity and having natural set theoretic interpretations. Our perspective is that of a mathematician trained in classical logic, but nevertheless we hope this paper might go some way to bridging the cultural divide between type theorists coming from computer science.

CONTENTS

INTRODUCTION

Background. Type theory has its origins in Russell and Whitehead's Principia [RW10] from the early 20th century, an ambitious project to set all mathematics of the day on a firm foundation. Broadly speaking it achieved this goal, showing how a significant amount of mathematics could, at least in principal, be expressed in their formalism. Nevertheless, the complexity of its ramified type system meant there was little enthusiasm for using it in practice, especially once simpler foundational systems like Zermelo-Fraenkel set theory become available.

The lambda-calculus underlying modern type theories also had its origin in Church's attempts in [Chu32] to give an alternative foundation for mathematics. While his foundational system was found to be flawed, part of the syntax was proposed in [Chu36] as a formal system for computations. For some time this also had little uptake in the general mathematical community, but for a rather different reason, namely that no precise meaning was given to the formalism. It was only after Scott gave it some proper semantics that it started to gain more acceptance (see [Mis98, §5]). A similar story pertains to intuitionistic logic, for example, which was generally viewed with skepticism until Kripke semantics came along (see [Kri65]).

Thus if history is any guide, to be taken seriously as a foundation for mathematics, a logical system must satisfy two basic criteria – it must be both simple and meaningful. The classic example of such a system is ZFC (Zermelo-Fraenkel set theory with the axiom of choice – see [Kun11]) which has indeed served as a solid foundation for mathematics for over a century now. The language of ZFC is certainly quite simple, just the usual first order logic with a single relation symbol $\in$. The logical symbols have their usual meaning, while $\in$ is naturally interpreted as the membership relation on some universe of sets. The deductive system of ZFC is also just the usual one for first order logic with only two inference rules, modus ponens and universal generalisation. The only slight complexity arises in the axioms, consisting of several logical axioms together with 9 more axioms specific to ZFC.

On the other hand, logical systems based on dependent types usually fail the basic criteria on both counts, being neither simple nor meaningful. Indeed, a typical paper on dependent type theory will start with several pages of ad hoc inference rules with little or no effort devoted to explaining the meaning or motivation behind them. Even those papers that do try to give the formalism some proper semantics do so only after the inference rules have already been fixed. They also often make

some unusual choices, e.g. allowing the interpretations to be affected by assumptions, not just the underlying variable assignment.

It is no surprise then that such systems have not been seen as genuine contenders for foundational systems, despite the best efforts of their proponents. This may be slowly changing with the advent of several proof assistants based on dependent types that have been gaining traction, at least in the computer science community. However, even those mathematicians sympathetic to the idea of automated proof checking tend to balk at the type systems being used as the foundational logic.

Another problematic issue is the habit of type theorists to propose systems that are not even consistent. From the very beginning, early versions of the Principia were inconsistent (see [Lin02]), as were the original systems proposed by Church, Curry and Martin-Löf (see [KR35], [Coq94] and [Hur95]). Right up to the present day, 'bugs' are regularly being found in the kernels of the most popular proof assistants based on dependent types (see [Car24]). In contrast, no inconsistency has been found in ZFC since it was finalised over a century ago. Thus an added benefit of a type system with rigorous set based semantics is that its consistency is put on a par with that of ZFC, the most well studied and well tested logical system in history. Indeed, had the proponents of various inconsistent systems given more thought to their semantics from the outset, they could have quickly seen where the inconsistencies would likely arise and modified their systems accordingly.

That said, despite their chequered past, we believe dependent types do have potential as a foundational system if developed with just a little more care. This is the goal of the present paper – to show that systems of dependent types can be created which are simple enough to be compared with ZFC and which can also be meaningfully interpreted in an elementary but precise way in some universe of sets. Even if these particular systems do not see any practical use, we hope they might at least serve as a bridge of understanding between the two communities in question – mathematicians can gain a quick understanding of dependent types, while computer scientists can see some type systems that would be more appealing to mathematicians.

Related Work. We would be remiss if we did not also mention simple type theories, which can indeed be sufficiently simple and meaningful that they could rightly be viewed as satisfying our criteria. Why they have not received widespread recognition as potential foundational systems we can only speculate. Suffice to say that the interest seems to have shifted towards dependent type theories, perhaps because of some nice features they possess, e.g. a uniform treatment of terms and types

and the compatibility of universal quantification with the 'propositions as types' philosophy.

We have also neglected to mention the significant body of work devoted to categorical semantics and the more sophisticated type systems designed to handle them, like homotopy type theory. As a foundation for higher category theory, some kind of homotopy type theory may well prove to be the best framework, as will no doubt be borne out by continued research in this area. However, a foundational system for mathematics as a whole should first and foremost have some natural set based semantics, simple enough to be understood by any mathematician, not just those with expertise in category theory. Of course any semantics beyond sets, category theoretic or otherwise, is definitely welcome, it is just not the primary goal of such a foundational system.

Preliminaries

We follow the usual mathematical practice of working in some naïve universe of sets, making use of standard set theoretic notation and terminology throughout. This can all be formalised in ZFC but this is not the main point – much of what we say is simple enough that it would be valid in any reasonable formal theory of sets. If particular axioms of ZFC (as detailed in [Kun11, §I.3], for example) or large cardinals become relevant, we will explicitly point this out.

As usual, the *power set* of a set S will be denoted by

$$\mathcal{P}(S) = \{R \mid R \subseteq S\}.$$

Ordered pairs will be denoted by angle brackets, e.g. $\langle r, s \rangle$. The *product* of any sets R and S is thus given by

$$R \times S = \{\langle r, s \rangle \mid r \in R \text{ and } s \in S\}.$$

We often use the standard infix notation to denote an ordered pair $\langle r, d \rangle$ being a member of a set S, i.e.

$$r \, S \, d \qquad \Leftrightarrow \qquad \langle r, d \rangle \in S.$$

The *domain* and *range* of a set S are then given by

(Domain) $\operatorname{dom}(S) = \{d \mid \exists r \, (r \, S \, d)\}.$

(Range) $\operatorname{ran}(S) = \{r \mid \exists d \, (r \, S \, d)\}.$

The *inverse* of a set S is given by

(Inverse) $S^{-1} = \{\langle d, r \rangle \mid r \, S \, d\}.$

The *image* of a set S under a set R is given by

(Image) $R[S] = \{r \mid \exists s \in S \, (r \, R \, s)\}.$

In particular, the image $R[\{s\}]$ of a singleton set $\{s\}$ will be abbreviated as $R\{s\}$. The *application* of a set R to a set s is given by

(Application) $$R(s) = \bigcup R\{s\} = \bigcup \{r \mid \langle r, s \rangle \in R\}.$$

In particular, if $s \notin \mathrm{dom}(R)$ then $R(s) = R\{s\} = \emptyset$. The *composition* of a set R with a set S is given by

(Composition) $$R \circ S = \{\langle r, s \rangle \mid R^{-1}\{r\} \cap S\{s\} \neq \emptyset\}.$$

Equivalently, for any r and s,

$$r \; R \circ S \; s \qquad \Leftrightarrow \qquad \exists t \, (r \; R \; t \; S \; s).$$

The *transitive closure* of R is defined to be the smallest set R^{t} containing R such that $R^{\mathsf{t}} \circ R^{\mathsf{t}} \subseteq R^{\mathsf{t}}$, i.e.

(Transitive Closure) $$R^{\mathsf{t}} = R \cup (R \circ R) \cup (R \circ R \circ R) \cup \ldots.$$

A *binary relation* R is a set of ordered pairs. Equivalently, R is a binary relation if and only if $R \subseteq \mathrm{ran}(R) \times \mathrm{dom}(R)$. A *function* F is a binary relation where, for each $x \in \mathrm{dom}(F)$, there is a unique $y \in \mathrm{ran}(F)$ with $y \; F \; x$. Equivalently, F is a function if and only if $F = \{\langle F(x), x \rangle \mid x \in \mathrm{dom}(F)\}$.

Remark. When F is a function, we immediately see that

$$y = F(x) \qquad \Leftrightarrow \qquad y \; F \; x.$$

Indeed, this is usually taken as the definition of application, i.e. $F(x)$ is defined to be the unique y satisfying $y \; F \; x$, but only when F is a function and x lies in its domain. However, we find it convenient to define applications for arbitrary sets as in [Acz99] in order that our interpretations below can be defined on arbitrary terms.

Also note that for us a function with domain D and range R is subset of $R \times D$ not $D \times R$, as often seen in the literature. We adopt this convention so that the notion of composition above agrees with the standard notions for both relations and functions.

The set of all functions from a set X to a set Y will be denoted by

$$Y^X = \{F \subseteq Y \times X : F \text{ is a function with } \mathrm{dom}(F) = X\}.$$

More generally, given F with $\mathrm{dom}(F) = X$, the set of functions f on X such that $f(x) \in F(x)$, for all $x \in X$, will be denoted by

$$\prod F = \prod_{x \in X} F(x) = \{f \in \big(\bigcup_{x \in X} F(x) \big)^X \mid f(x) \in F(x), \text{ for all } x \in X\}.$$

This is called the *dependent product* defined by F.

A *string* S in some alphabet A is an element of $\mathsf{A}^{<\omega} = \bigcup_{n \in \omega} \mathsf{A}^n$, i.e. a function from some finite ordinal $n = \{0, \ldots, n-1\}$ to elements of A. We call n the *length* of S. We concatenate strings as usual so if R and S are strings of length m and n respectively then RS denotes the string of length $m + n$ defined by

$$RS(k) = \begin{cases} R(k) & \text{if } k < m \\ S(k - m) & \text{if } k \geq m. \end{cases}$$

Likewise, given sets of strings $\mathsf{R}, \mathsf{S} \subseteq \mathsf{A}^{<\omega}$, we let

$$\mathsf{RS} = \{RS \mid R \in \mathsf{R} \text{ and } S \in \mathsf{S}\}.$$

We also identify symbols $a \in \mathsf{A}$ with the corresponding strings of length 1, i.e. where $a(0) = a$. Thus if $a, b, c \in \mathsf{A}$ then abc denotes the string of length 3 such that $abc(0) = a$, $abc(1) = b$ and $abc(2) = c$.

We will introduce our type systems in the same way classical logical systems are usually introduced. First we specify the language, i.e. the alphabet A and the strings $\mathsf{C}, \mathsf{V}, \mathsf{T} \subseteq \mathsf{A}^{<\omega}$ that we take as the *constants*, *variables* and *terms* of our language. These terms are meant to represent sets, as we make precise by defining *interpretations* $[\![\cdot]\!]$ as certain maps on terms determined by some assignment of sets to the variables and constants. We then define *statements* S formed from terms. These represent actual mathematical statements about our sets which either hold or fail in a particular interpretation. This in turn is made precise by showing how interpretations $[\![\cdot]\!]$ determine a subset of statements Σ *satisfied* by $[\![\cdot]\!]$. In this case we call $[\![\cdot]\!]$ a *model* for Σ.

This naturally leads to a semantic *consequence relation* $\vDash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$, where $\Gamma \vDash X$ indicates that every model for all the statements in Γ is a model for the statement X. We then investigate syntactic properties of $\vDash$ which in the end we take as inference rules defining a syntactic *inference relation* $\vdash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$, i.e. $\vdash$ is the smallest relation obeying all the inference rules. As we derive the inference rules from properties of $\vDash$ from the outset, the inference relation will automatically be *sound* with respect to the consequence relation, i.e. $\vdash \subseteq \vDash$. Finally we examine properties of $\vdash$ and discuss the axioms that would be needed to use $\vdash$ as a foundation for mathematics.

Part 1. **A Bare Bones Dependent Type System**

1. TERMS

The alphabet of our first system has 6 symbols which we denote by

$$\mathsf{A} = \{\mathsf{c}, \mathsf{v}, {}', \rho, \beta, \lambda\}.$$

The strings we are interested in are defined in Backus-Naur form by

(Constants) $\qquad$ C ::= c | C′

(Variables) $\qquad$ V ::= v | V′

(Terms) $\qquad$ T ::= C | V | ρT | βTT | λVTT

More explicitly, the set of constant and variable strings are given by

$$\mathsf{C} = \{\mathsf{c}, \mathsf{c}', \mathsf{c}'', \ldots\} \quad \text{and}$$
$$\mathsf{V} = \{\mathsf{v}, \mathsf{v}', \mathsf{v}'', \ldots\},$$

while the terms are the smallest set T of strings such that

$$\mathsf{C} \cup \mathsf{V} \cup \rho\mathsf{T} \cup \beta\mathsf{T}\mathsf{T} \cup \lambda\mathsf{V}\mathsf{T}\mathsf{T} \subseteq \mathsf{T}.$$

The ρ, β and λ terms here are meant to represent dependent products, applications and abstractions respectively, as will soon be clear from the interpretations we define below. Terms like these are standard in dependent type systems, although our formal syntax differs a little from that commonly seen in the literature. For example, our β-terms would usually just be written by juxtapostion without the β, i.e. βRS would just be RS. We are adding the β so our formal language requires no parentheses, a side-benefit being that β-reduction then does indeed amount to reducing β-terms. Also λ-terms are often written with extra colons, dots and arrows, but for simplicity's sake we prefer to avoid unnecessary punctuation, at least in the formal language. Likewise, products are usually written like λ-terms just with their own binder π replacing λ. We prefer a simpler language with a single binder λ, but we can still view π as an abbreviation (so-called 'syntactic sugar') for $\rho\lambda$. We will discuss more abbreviations below.

Before moving on, let us stratify the terms based on the number of operations needed to form them. Specifically, let $\mathsf{T}^0 = \mathsf{C} \cup \mathsf{V}$ and

$$\mathsf{T}^{n+1} = \mathsf{T}^0 \cup \rho\mathsf{T}^n \cup \beta\mathsf{T}^n\mathsf{T}^n \cup \lambda\mathsf{V}\mathsf{T}^n\mathsf{T}^n,$$

for all $n \in \omega$. So T^0 consists of the *atomic* terms and $\mathsf{T} = \bigcup_{n \in \omega} \mathsf{T}^n$. As we are taking strings in V and C as atomic, we do not want to view v′ as a substring of v″, for example. Accordingly we define the substring relation $\sqsubseteq$ on T as follows. First, for all $Q, R, S, T \in \mathsf{A}^{<\omega}$, we define

$$S \sqsubseteq^Q_T R \qquad \Leftrightarrow \qquad R = QST \text{ and } T(0) \neq {}'.$$

Then we define

$$S \sqsubseteq R \qquad \Leftrightarrow \qquad S \sqsubseteq^Q_T R, \text{ for some } Q, T \in \mathsf{A}^{<\omega}.$$

2. Interpretations

Now that we have defined our terms, a type theorist might expect us to immediately define statements and list the inference rules that can be applied to them. However, we would first like to make our intended interpretations of the terms clear. Only then does it make sense to introduce statements and inference rules based on these interpretations.

To interpret terms, we first assign sets to variables and constants. Accordingly, we call a function $[\![\cdot]\!]^0$ on T^0 an *assignment*. Given an assignment $[\![\cdot]\!]^0$, we can modify it to agree with a function ψ defined on a subset of T^0. We denote this modification by $[\![\cdot]\!]^0_\psi$, i.e.

$$[\![t]\!]^0_\psi = \begin{cases} \psi(t) & \text{if } t \in \mathrm{dom}(f) \\ [\![t]\!]^0 & \text{otherwise.} \end{cases}$$

In particular, given any $x \in \mathsf{V}$ and s, we get another assignment $[\![\cdot]\!]^0_{\{\langle s,x\rangle\}}$ which we abbreviate to $[\![\cdot]\!]^0_{\langle s,x\rangle}$ so that

$$[\![t]\!]^0_{\langle s,x\rangle} = \begin{cases} s & \text{if } t = x \\ [\![t]\!]^0 & \text{if } t \in \mathsf{T}^0 \setminus \{x\}. \end{cases}$$

Definition 2.1. We extend any assignment $[\![\cdot]\!]^0$ on T^0 to functions $[\![\cdot]\!]^n = ([\![\cdot]\!]^0)^n$ on T^n, for each $n \in \omega$, so that

$$[\![\rho S]\!]^{n+1} = \prod [\![S]\!]^n,$$
$$[\![\beta RS]\!]^{n+1} = [\![R]\!]^n([\![S]\!]^n) \quad \text{and}$$
$$[\![\lambda x R S]\!]^{n+1} = \{\langle [\![S]\!]^n_{\langle r,x\rangle}, r\rangle \mid r \in [\![R]\!]^n\},$$

for $R, S \in \mathsf{T}^n$ and $x \in \mathsf{V}$, where $[\![\cdot]\!]^n_{\langle r,x\rangle} = ([\![\cdot]\!]^0_{\langle r,x\rangle})^n$. The *interpretation* coming from $[\![\cdot]\!]^0$ is the function $[\![\cdot]\!] = [\![\cdot]\!]^\omega = ([\![\cdot]\!]^0)^\omega$ on T defined by

$$[\![\cdot]\!] = \bigcup_{n \in \omega} [\![\cdot]\!]^n.$$

Restricting an interpretation $[\![\cdot]\!]$ to T^0 is immediately seen to recover the underlying assignment $[\![\cdot]\!]^0$, from which we then define

$$[\![\cdot]\!]_\psi = \bigcup_{n \in \omega} [\![\cdot]\!]^n_\psi = \bigcup_{n \in \omega} ([\![S]\!]^0_\psi)^n,$$

We then see that the above defining properties of $[\![\cdot]\!]^n$ on T^n remain valid for the whole interpretation $[\![\cdot]\!]$ on T, i.e.

$$[\![\rho S]\!] = \prod [\![S]\!],$$
$$[\![\beta RS]\!] = [\![R]\!]([\![S]\!]) \quad \text{and}$$
$$[\![\lambda x RS]\!] = \{\langle [\![S]\!]_{\langle r,x \rangle}, r \rangle \mid r \in [\![R]\!]\}.$$

Our interpretations are thus consistent with those defined in [Acz99]. This contrasts with other notions of interpretation, like those given in [MW03], which further depend on a given set of assumptions Γ.

Note that the above means that a λ-term $\lambda x RS$ is interpreted as a function with domain $[\![R]\!]$ such that, for all $r \in [\![R]\!]$,

$$[\![\lambda x RS]\!](r) = [\![S]\!]_{\langle r,x \rangle}.$$

In particular, if we have another term $T \in \mathsf{T}$ with $[\![T]\!] \in [\![R]\!]$ then

$$[\![\beta \lambda x RST]\!] = [\![\lambda x RS]\!]([\![T]\!]) = [\![S]\!]_{\langle [\![T]\!],x \rangle}.$$

Based on our definition of interpretations, $[\![S]\!]_{\langle [\![T]\!],x \rangle}$ should be the same as the interpretation of another term $S_{[T/x]}$ obtained by substituting T for all 'free' occurrences of x in S, as long as we are careful to rename bound variables in S so the binder λ does not unintentionally capture any free variables in T. Let us now make this more precise.

3. Free Variables

Intuitively, a variable x should be free in a term T if changing an interpretation at x could conceivably change the interpretation of T. Accordingly, we define the *free variables* $\mathsf{F}(T) \subseteq \mathsf{V}$ of any term T so that, for all $a \in \mathsf{C}$, $x \in \mathsf{V}$ and $R, S \in \mathsf{T}$,

$$\mathsf{F}(a) = \emptyset.$$
$$\mathsf{F}(x) = \{x\}.$$
$$\mathsf{F}(\rho R) = \mathsf{F}(R).$$
$$\mathsf{F}(\beta RS) = \mathsf{F}(R) \cup \mathsf{F}(S).$$
$$\mathsf{F}(\lambda x RS) = \mathsf{F}(R) \cup (\mathsf{F}(S) \setminus \{x\}).$$

Proposition 3.1. *For any interpretation $[\![\cdot]\!]$, $S \in \mathsf{T}$, $x \in \mathsf{V}$ and s,*

$$x \notin \mathsf{F}(S) \qquad \Rightarrow \qquad [\![S]\!] = [\![S]\!]_{\langle s,x \rangle}.$$

Proof. First note that if $t \in \mathsf{T}^0$ and $x \in \mathsf{V} \setminus \mathsf{F}(t)$ then $x \neq t$ and hence $[\![t]\!]_{\langle s,x \rangle} = [\![t]\!]$. Now assume the result holds for any $R, S \in \mathsf{T}^n$. If $x \in \mathsf{V} \setminus \mathsf{F}(R)$ then $[\![\rho R]\!]_{\langle s,x \rangle} = \prod [\![R]\!]_{\langle s,x \rangle} = \prod [\![R]\!] = [\![\rho R]\!]$. Likewise, if $x \in \mathsf{V} \setminus (\mathsf{F}(R) \cup \mathsf{F}(S))$ then $[\![\beta RS]\!]_{\langle s,x \rangle} = [\![R]\!]_{\langle s,x \rangle}([\![S]\!]_{\langle s,x \rangle}) = [\![\beta RS]\!]$.

Now let $\llbracket\cdot\rrbracket_{\langle r,x\rangle\langle s,y\rangle} = (\llbracket\cdot\rrbracket_{\langle r,x\rangle})_{\langle s,y\rangle}$ denote $\llbracket\cdot\rrbracket$ changed first at x to r and then at y to s. If $x \neq y$, we can reverse the order, i.e.

$$x \neq y \qquad \Rightarrow \qquad \llbracket\cdot\rrbracket_{\langle r,x\rangle\langle s,y\rangle} = \llbracket\cdot\rrbracket_{\langle s,y\rangle\langle r,x\rangle}.$$

But if $x = y$ then the second change cancels the first, i.e.

$$\llbracket\cdot\rrbracket_{\langle r,x\rangle\langle s,x\rangle} = \llbracket\cdot\rrbracket_{\langle s,x\rangle}.$$

Thus if $x \in \mathsf{V} \setminus (\mathsf{F}(R) \cup (\mathsf{F}(S) \setminus \{y\}))$ then

$$\begin{aligned}
\llbracket\lambda y R S\rrbracket_{\langle s,x\rangle} &= \{\langle\llbracket S\rrbracket_{\langle s,x\rangle\langle r,y\rangle}, r\rangle : r \in \llbracket R\rrbracket_{\langle s,x\rangle}\} \\
&= \{\langle\llbracket S\rrbracket_{\langle r,y\rangle}, r\rangle : r \in \llbracket R\rrbracket\}. \\
&= \llbracket\lambda y R S\rrbracket.
\end{aligned}$$

Indeed, if $x = y$ then $\llbracket S\rrbracket_{\langle s,x\rangle\langle r,y\rangle} = \llbracket S\rrbracket_{\langle s,y\rangle\langle r,y\rangle} = \llbracket S\rrbracket_{\langle r,y\rangle}$, while if $x \neq y$ then $x \notin \mathsf{F}(S)$ so $\llbracket S\rrbracket_{\langle s,x\rangle\langle r,y\rangle} = \llbracket S\rrbracket_{\langle r,y\rangle\langle s,x\rangle} = \llbracket S\rrbracket_{\langle r,y\rangle}$, by the inductive hypothesis. This proves the result for all terms in T^{n+1}, as required. $\quad\square$

Likewise, we define the *bound variables* $\mathsf{B}(T)$ of any term T so that, for all $t \in \mathsf{T}^0$, $x \in \mathsf{V}$ and $R, S \in \mathsf{T}$,

$$\mathsf{B}(t) = \emptyset.$$
$$\mathsf{B}(\rho R) = \mathsf{B}(R).$$
$$\mathsf{B}(\beta R S) = \mathsf{B}(R) \cup \mathsf{B}(S).$$
$$\mathsf{B}(\lambda x R S) = \mathsf{B}(R) \cup \mathsf{B}(S) \cup \{x\}.$$

Equivalently, $x \in \mathsf{B}(T)$ if it appears next to the binder λ in T, i.e.

$$\mathsf{B}(T) = \{x \in \mathsf{V} \mid \lambda x \sqsubseteq T\}.$$

The set of all variables appearing in T will be denoted by

$$\mathsf{V}(T) = \mathsf{F}(T) \cup \mathsf{B}(T) = \{x \in \mathsf{V} \mid x \sqsubseteq T\}.$$

4. Substitution

To define substitution on T^0, take any $x \in \mathsf{V}$ and $T \in \mathsf{T}$ and let

$$t_{[T/x]} := \begin{cases} T & \text{if } t = x \\ t & \text{if } t \in \mathsf{T}^0 \setminus \{x\}. \end{cases}$$

Then extend from T^n to T^{n+1} by defining, for all $R, S \in \mathsf{T}^n$ and $y \in \mathsf{V}$,

$$(\rho R)_{[T/x]} := \rho R_{[T/x]},$$
$$(\beta R S)_{[T/x]} := \beta R_{[T/x]} S_{[T/x]} \quad \text{and}$$
$$(\lambda y R S)_{[T/x]} := \begin{cases} \lambda y R_{[T/x]} S & \text{if } y = x \text{ or } x \notin \mathsf{F}(S) \\ \lambda z R_{[T/x]} S_{[z/y][T/x]} & \text{otherwise, where } z \notin \mathsf{F}(\lambda y T S). \end{cases}$$

As far as interpretations are concerned, it does not matter which variable outside of $\mathsf{F}(\lambda yTS) = \mathsf{F}(T) \cup (\mathsf{F}(S) \setminus \{y\})$ we take for z above. However, for some basic syntactic properties, it will be important to avoid unnecessary changes of bound variables. Accordingly, let us take $z = y$ above whenever $y \notin \mathsf{F}(T)$ (otherwise take z to be the shortest variable outside of $\mathsf{V}(TS)$, for example). This ensures that

$$x \neq y \notin \mathsf{F}(T) \quad \Rightarrow \quad (\lambda yRS)_{[T/x]} = \lambda yR_{[T/x]}S_{[T/x]}.$$

This completes the recursive definition of substitution on T.

Our original motivation for defining free variables was in terms of changing interpretations. However, freeness is more accurately characterised by changing under substitution, specifically

$$\mathsf{F}(S) = \{x \in \mathsf{V} \mid \forall T \in \mathsf{T} \setminus \{x\} \ (S_{[T/x]} \neq S)\}.$$

This is immediate from the following result.

Proposition 4.1. *For all $x \in \mathsf{V}$ and $S, T \in \mathsf{T}$,*

$$T \neq x \in \mathsf{F}(S) \qquad \Leftrightarrow \qquad S_{[T/x]} \neq S.$$

Proof. First we claim that

$$x \notin \mathsf{F}(S) \quad \Rightarrow \quad S_{[T/x]} = S.$$

To see this, note that if $t \in \mathsf{T}^0$ then $x \notin \mathsf{F}(t) \subseteq \{t\}$ means $t_{[T/x]} = t$. Now assume the claim holds for all $S \in \mathsf{T}^n$ and take any $R, S \in \mathsf{T}^n$. If $x \notin \mathsf{F}(\rho R) = \mathsf{F}(R)$ then $(\rho R)_{[T/x]} = \rho R_{[T/x]} = \rho R$. Likewise, if $x \notin \mathsf{F}(\beta RS) = \mathsf{F}(S) \cup \mathsf{F}(R)$ then $(\beta RS)_{[T/x]} = \beta R_{[T/x]}S_{[T/x]} = \beta RS$. If $x \notin \mathsf{F}(\lambda yRS)$ then either $y = x \notin \mathsf{F}(R)$ or $y \neq x \notin \mathsf{F}(R) \cup \mathsf{F}(S)$ and so in both cases $(\lambda yRS)_{[T/x]} = \lambda yR_{[T/x]}S = \lambda yRS$. This proves the claim on T^{n+1} and hence on T, by induction.

An even simpler inductive argument proves $S_{[x/x]} = S$, for all $S \in \mathsf{T}$, the only slightly non-trivial thing to note is that $y \neq x$ implies $y \notin \mathsf{F}(x)$ and hence $(\lambda yRS)_{[x/x]} = \lambda yR_{[x/x]}S_{[x/x]} = \lambda yRS$. This completes the proof of the $\Leftarrow$ part of the result.

For the $\Rightarrow$ part first note $t \in \mathsf{T}^0$ and $T \neq x \in \mathsf{F}(t) = \{t\}$ implies $t_{[T/x]} = x_{[T/x]} = T \neq t$, proving $\Rightarrow$ on T^0. Now assume $\Rightarrow$ holds for all $S \in \mathsf{T}^n$ and take any $R, S \in \mathsf{T}^n$. Then $T \neq x \in \mathsf{F}(\rho R) = \mathsf{F}(R)$ implies $(\rho R)_{[T/x]} = \rho R_{[T/x]} \neq \rho R$. Also $T \neq x \in \mathsf{F}(\beta RS) = \mathsf{F}(R) \cup \mathsf{F}(S)$ implies $R \neq R_{[T/x]}$ or $S \neq S_{[T/x]}$ and hence $(\beta RS)_{[T/x]} = \beta R_{[T/x]}S_{[T/x]} \neq \beta RS$. Now if $T \neq x \in \mathsf{F}(\lambda yRS)$ then $y = x \in \mathsf{F}(R)$ or $y \neq x \in \mathsf{F}(R) \cup \mathsf{F}(S)$. In the former case $(\lambda yRS)_{[T/x]} = \lambda yR_{[T/x]}S \neq \lambda yRS$ while in the latter case either $R_{[T/x]} \neq R$ or $S_{[T/x]} \neq S$ and hence $S_{[z/y][T/x]} \neq S$, whenever $z \notin \mathsf{F}(\lambda yTS)$, which yields $(\lambda yRS)_{[T/x]} = \lambda zR_{[T/x]}S_{[z/y][T/x]} \neq \lambda yRS$. This proves $\Rightarrow$ on T^{n+1} and hence on T, by induction. $\square$

We can now verify that substitutions have the desired interpretation.

Proposition 4.2. *For any interpretation $[\![\cdot]\!]$, $x \in \mathsf{V}$ and $S, T \in \mathsf{T}$,*

$$\text{(Substitutivity)} \qquad\qquad [\![S_{[T/x]}]\!] = [\![S]\!]_{\langle [\![T]\!], x \rangle}.$$

Proof. Certainly $[\![x]\!]_{\langle [\![T]\!], x \rangle} = [\![T]\!] = [\![x_{[T/x]}]\!]$ and, for any $t \in \mathsf{T}^0 \setminus \{x\}$, $[\![t]\!]_{\langle [\![T]\!], x \rangle} = [\![t]\!] = [\![t_{[T/x]}]\!]$, proving the result for $S \in \mathsf{T}^0$.

Now assume the result for all $S \in \mathsf{T}^n$ and take any $R, S \in \mathsf{T}^n$. Note

$$[\![\rho R]\!]_{\langle [\![T]\!], x \rangle} = \prod [\![R]\!]_{\langle [\![T]\!], x \rangle} = \prod [\![R_{[T/x]}]\!] = [\![\rho R_{[T/x]}]\!] = [\![(\rho R)_{[T/x]}]\!].$$

Likewise, $[\![\beta RS]\!]_{\langle [\![T]\!], x \rangle} = [\![\beta R_{[T/x]} S_{[T/x]}]\!] = [\![(\beta RS)_{[T/x]}]\!]$. Next note

$$\begin{aligned}
[\![\lambda x RS]\!]_{\langle [\![T]\!], x \rangle} &= \{ \langle [\![S]\!]_{\langle [\![T]\!], x \rangle \langle r, x \rangle}, r \rangle \mid r \in [\![R]\!]_{\langle [\![T]\!], x \rangle} \} \\
&= \{ \langle [\![S]\!]_{\langle r, x \rangle}, r \rangle \mid r \in [\![R_{[T/x]}]\!] \} \\
&= [\![\lambda x R_{[T/x]} S]\!] \\
&= [\![(\lambda x RS)_{[T/x]}]\!].
\end{aligned}$$

Now take any $y \in \mathsf{V} \setminus \{x\}$. If $x \notin \mathsf{F}(S)$ then Proposition 3.1 yields $[\![S]\!]_{\langle [\![T]\!], x \rangle \langle r, y \rangle} = [\![S]\!]_{\langle r, y \rangle \langle [\![T]\!], x \rangle} = [\![S]\!]_{\langle r, y \rangle}$, and so again

$$\begin{aligned}
[\![\lambda y RS]\!]_{\langle [\![T]\!], x \rangle} &= \{ \langle [\![S]\!]_{\langle [\![T]\!], x \rangle \langle r, y \rangle}, r \rangle \mid r \in [\![R]\!]_{\langle [\![T]\!], x \rangle} \} \\
&= \{ \langle [\![S]\!]_{\langle r, y \rangle}, r \rangle \mid r \in [\![R_{[T/x]}]\!] \} \\
&= [\![\lambda y R_{[T/x]} S]\!] \\
&= [\![(\lambda y RS)_{[T/x]}]\!].
\end{aligned}$$

Otherwise $y \neq x \in \mathsf{F}(S)$. For any $z \notin \mathsf{F}(\lambda y TS)(\ni x)$ and r, then $[\![T]\!] = [\![T]\!]_{\langle r, z \rangle}$. Also, for any interpretation $(\!|\cdot|\!)$, $(\!|S_{[z/y]}|\!)_{\langle r, z \rangle} = (\!|S|\!)_{\langle r, y \rangle}$ — If $z = y$ then this is immediate, otherwise $y \neq z \notin \mathsf{F}(S)$ and hence

$$(\!|S_{[z/y]}|\!)_{\langle r, z \rangle} = (\!|S|\!)_{\langle r, z \rangle \langle (\!|z|\!)_{\langle r, z \rangle}, y \rangle} = (\!|S|\!)_{\langle r, z \rangle \langle r, y \rangle} = (\!|S|\!)_{\langle r, y \rangle \langle r, z \rangle} = (\!|S|\!)_{\langle r, y \rangle}.$$

In particular we can take $(\!|\cdot|\!) = [\![\cdot]\!]_{\langle [\![T]\!], x \rangle} = [\![\cdot]\!]_{\langle [\![T]\!]_{\langle r, z \rangle}, x \rangle}$. Noting $x \neq z$,

$$\begin{aligned}
[\![S]\!]_{\langle [\![T]\!], x \rangle \langle r, y \rangle} &= [\![S_{[z/y]}]\!]_{\langle [\![T]\!]_{\langle r, z \rangle}, x \rangle \langle r, z \rangle} = [\![S_{[z/y]}]\!]_{\langle r, z \rangle \langle [\![T]\!]_{\langle r, z \rangle}, x \rangle} \\
&= [\![S_{[z/y][T/x]}]\!]_{\langle r, z \rangle}.
\end{aligned}$$

From this it follows that

$$\begin{aligned}
[\![\lambda y RS]\!]_{\langle [\![T]\!], x \rangle} &= \{ \langle [\![S]\!]_{\langle [\![T]\!], x \rangle \langle r, y \rangle}, r \rangle \mid r \in [\![R]\!]_{\langle [\![T]\!], x \rangle} \} \\
&= \{ \langle [\![S_{[z/y][T/x]}]\!]_{\langle r, z \rangle}, r \rangle \mid r \in [\![R_{[T/x]}]\!] \} \\
&= [\![\lambda z R_{[T/x]} S_{[z/y][T/x]}]\!] = [\![(\lambda y RS)_{[T/x]}]\!].
\end{aligned}$$

This completes the induction. $\qquad\square$

Here is another observation on substitutions that is sometimes useful.

Proposition 4.3. *For all $x \in \mathsf{V}$ and $S, T \in \mathsf{T}$,*

$$x \in \mathsf{F}(S) \quad \Rightarrow \quad \mathsf{F}(S_{[T/x]}) = \mathsf{F}(\lambda x T S).$$

Proof. If $t \in \mathsf{T}^0$ then $x \in \mathsf{F}(t) \subseteq \{t\}$ means $t_{[T/x]} = x_{[T/x]} = T$ so $\mathsf{F}(t_{[T/x]}) = \mathsf{F}(T) = \mathsf{F}(\lambda x T t)$. This proves the result on T^0. Now assume the result holds on T^n and take $R, S \in \mathsf{T}^n$. If $x \in \mathsf{F}(\rho R) = \mathsf{F}(R)$ then

$$\mathsf{F}((\rho R)_{[T/x]}) = \mathsf{F}(\rho R_{[T/x]}) = \mathsf{F}(R_{[T/x]}) = \mathsf{F}(\lambda x T R) = \mathsf{F}(\lambda x T \rho R).$$

Likewise, if $x \in \mathsf{F}(\beta R S) = \mathsf{F}(R) \cup \mathsf{F}(S)$ then either $x \in \mathsf{F}(S)$ or $x \in \mathsf{F}(R)$ but in either case $\mathsf{F}((\beta R S)_{[T/x]}) = \mathsf{F}(\beta R_{[T/x]} S_{[T/x]})$ contains $\mathsf{F}(T)$ so

$$\mathsf{F}((\beta R S)_{[T/x]}) = (\mathsf{F}(R) \setminus \{x\}) \cup (\mathsf{F}(S) \setminus \{x\}) \cup \mathsf{F}(T) = \mathsf{F}(\lambda x T \beta R S).$$

Now if $x \in \mathsf{F}(\lambda y R S)$ then either $y = x \in \mathsf{F}(R)$ or $y \neq x \in \mathsf{F}(R) \cup \mathsf{F}(S)$. In the former case

$$\begin{aligned}
\mathsf{F}((\lambda y R S)_{[T/x]}) &= \mathsf{F}(\lambda y R_{[T/x]} S) \\
&= \mathsf{F}(T) \cup (\mathsf{F}(R) \setminus \{x\}) \cup (\mathsf{F}(S) \setminus \{y\}) \\
&= \mathsf{F}(T) \cup (\mathsf{F}(\lambda y R S) \setminus \{x\}),
\end{aligned}$$

as required. Similarly, in the latter case

$$\begin{aligned}
\mathsf{F}((\lambda y R S)_{[T/x]}) &= \mathsf{F}(\lambda z R_{[T/x]} S_{[z/y][T/x]}) \\
&= \mathsf{F}(T) \cup (\mathsf{F}(R) \setminus \{x\}) \cup (\mathsf{F}(S) \setminus \{x, y\}) \\
&= \mathsf{F}(T) \cup (\mathsf{F}(\lambda y R S) \setminus \{x\}).
\end{aligned}$$

This proves the result on T^{n+1} and hence on T, by induction. $\square$

5. Conversion

We can now show that changing bound variables does not affect the interpretation of the resulting term.

Proposition 5.1. *For any interpretation $[\![\cdot]\!]$, $x, y \in \mathsf{V}$ and $R, S \in \mathsf{T}$,*

$$y \notin \mathsf{F}(S) \quad \Rightarrow \quad [\![\lambda x R S]\!] = [\![\lambda y R S_{[y/x]}]\!].$$

Proof. First note (Substitutivity) yields

$$[\![S_{[y/x]}]\!]_{\langle r, y \rangle} = [\![S]\!]_{\langle r, y \rangle \langle [\![y]\!]_{\langle r, y \rangle}, x \rangle} = [\![S]\!]_{\langle r, y \rangle \langle r, x \rangle} = [\![S]\!]_{\langle r, x \rangle},$$

by Proposition 3.1, as $y \notin \mathsf{F}(S)$. Thus

$$\begin{aligned}
[\![\lambda x R S]\!] &= \{ \langle [\![S]\!]_{\langle r, x \rangle}, r \rangle \mid r \in [\![R]\!] \} \\
&= \{ \langle [\![S_{[y/x]}]\!]_{\langle r, y \rangle}, r \rangle \mid r \in [\![R]\!] \} \\
&= [\![\lambda y R S_{[y/x]}]\!]. \qquad \square
\end{aligned}$$

If we even have $y \notin \mathsf{V}(S)$ then we can reverse this process.

Proposition 5.2. *For any $x, y \in \mathsf{V}$ and $S \in \mathsf{T}$,*

$$y \notin \mathsf{V}(S) \quad \Rightarrow \quad S = S_{[y/x][x/y]}.$$

Proof. Note $x_{[y/x][x/y]} = y_{[x/y]} = x$. If $t \in \mathsf{T}^0 \setminus \{x\}$ and $y \notin \mathsf{V}(t)$ and hence $y \neq t \neq x$ then $t_{[y/x][x/y]} = t_{[x/y]} = t$, proving the result for T^0. Now assume the result for T^n and take $R, S \in \mathsf{T}$. Then $y \notin \mathsf{V}(\lambda x R S)$ implies $x \neq y \notin \mathsf{F}(S)$ and hence

$$(\lambda x R S)_{[y/x][x/y]} = (\lambda x R_{[y/x]} S)_{[x/y]} = \lambda x R_{[y/x][x/y]} S_{[x/y]} = \lambda x R S.$$

Also, for any $z \in \mathsf{V} \setminus \{x\}$, $y \notin \mathsf{V}(\lambda z R S)$ implies $y \neq z$ and hence

$$(\lambda z R S)_{[y/x][x/y]} = (\lambda z R_{[y/x]} S_{[y/x]})_{[x/y]} = \lambda z R_{[y/x][x/y]} S_{[y/x][x/y]} = \lambda x R S.$$

This completes the proof for λ-terms in T^{n+1}, while for ρ and β terms this is immediate. Now the result on T holds by induction. $\qquad \square$

This motivates the definition of $\equiv^\alpha \subseteq \mathsf{T} \times \mathsf{T}$ by

$$\equiv^\alpha = \{\langle \lambda x R S, \lambda y R S_{[y/x]} \rangle : x \notin \mathsf{B}(S) \text{ and } y \notin \mathsf{V}(S)\}.$$

We call this $\equiv^\alpha$ the *head α-conversion* relation.

Corollary 5.3. *The head α-conversion relation $\equiv^\alpha$ is symmetric.*

Proof. If $R, S \in \mathsf{T}$, $x \in \mathsf{V}$ and $y \in \mathsf{V} \setminus \mathsf{V}(S)$, Proposition 5.2 yields

$$\langle \lambda y R S_{[y/x]}, \lambda x R S \rangle = \langle \lambda y R S_{[y/x]}, \lambda x R S_{[y/x][x/y]} \rangle \in \equiv^\alpha.$$

Indeed $\mathsf{B}(S) = \mathsf{B}(S_{[y/x]})$, as $y \notin \mathsf{V}(S)$, so $y \notin \mathsf{V}(S) \supseteq \mathsf{B}(S) = \mathsf{B}(S_{[y/x]})$, $x \notin \mathsf{B}(S) = \mathsf{B}(S_{[y/x]})$ and $x \notin \mathsf{F}(S_{[y/x]})$, by Proposition 4.3. $\qquad \square$

Given $\rhd \subseteq \mathsf{T} \times \mathsf{T}$, define its *contextual closure* $\rhd^{\mathsf{c}} \subseteq \mathsf{T} \times \mathsf{T}$ by

$$R' \rhd^{\mathsf{c}} C' \quad \Leftrightarrow \quad \exists P, Q, R, C \ (R \sqsubseteq_Q^P R', C \sqsubseteq_Q^P C' \text{ and } R \rhd C).$$

Note P and Q above are not assumed to be terms, just strings (possibly even empty strings) which become terms when added to either side of R and C. Equivalently, $\rhd^{\mathsf{c}}$ is the smallest relation on T containing $\rhd$ such that, for all $x \in \mathsf{V}$ and $R, S, C \in \mathsf{T}$, $R \rhd^{\mathsf{c}} C$ always implies

$$\beta S R \rhd^{\mathsf{c}} \beta S C, \beta R S \rhd^{\mathsf{c}} \beta C S, \lambda x S R \rhd^{\mathsf{c}} \lambda x S C, \lambda x R S \rhd^{\mathsf{c}} \lambda x C S \text{ and } \rho R \rhd^{\mathsf{c}} \rho C.$$

We also define the reflexive closure of any $\rhd^{\mathsf{r}} \subseteq \mathsf{T} \times \mathsf{T}$ by

$$R \rhd^{\mathsf{r}} C \quad \Leftrightarrow \quad R \rhd C \text{ or } R = C.$$

Note that if $[\![R]\!] = [\![C]\!]$ whenever $R \lhd C$ then it also follows that $[\![R]\!] = [\![C]\!]$ whenever $R \lhd^{\mathsf{ctr}} C$. Defining *α-conversion* $\equiv_\alpha$ as the reflexive transitive contextual closure of head α-conversion

$$(\alpha\text{-Conversion}) \qquad\qquad\qquad \equiv_\alpha \ = \ \equiv^{\alpha\mathsf{ctr}},$$

the following is thus immediate from Proposition 5.1.

Corollary 5.4. *For any interpretation $\llbracket \cdot \rrbracket$ and $R, C \in \mathsf{T}$,*

$$R \equiv_\alpha C \qquad \Rightarrow \qquad \llbracket R \rrbracket = \llbracket C \rrbracket.$$

As $\equiv^\alpha$ is symmetric, so is $\equiv_\alpha$ and hence we could alternatively describe $\equiv_\alpha$ as the equivalence relation on T generated by $\equiv^{\alpha\mathsf{c}}$. Let us denote the corresponding equivalence class of any $S \in \mathsf{T}$ by

$$[S]_\alpha \;=\; \equiv_\alpha[S] \;=\; \{R \in \mathsf{T} \mid R \equiv_\alpha S\}.$$

Likewise, for any $\mathsf{T}' \subseteq \mathsf{T}$, let us define

$$[\mathsf{T}']_\alpha = \{[S]_\alpha \mid S \in \mathsf{T}'\}.$$

In particular, $[\mathsf{T}]_\alpha$ denotes the set of all equivalence classes of terms. These will be used below in §7 to define formal typing statements.

The symmetry of head α-conversion relies on the conditions we placed on the variables involved in the substitution. However, for more general α-conversion, these can be weakened as per Proposition 5.1.

Proposition 5.5. *For any $R, S \in \mathsf{T}$ and $x, y \in \mathsf{V}$,*

$$y \notin \mathsf{F}(S) \quad \Rightarrow \quad \lambda x R S \equiv_\alpha \lambda y R S_{[y/x]}.$$

Proof. See [HS08, Lemma A1.8]. $\qquad\qquad\qquad\qquad\qquad\qquad\square$

Later in §8, we will also need other basic syntactic properties of α-conversion, such as

$$\begin{aligned}
S \equiv_\alpha S' \text{ and } T \equiv_\alpha T' \quad &\Rightarrow \quad S_{[T/x]} \equiv_\alpha S'_{[T'/x]}, \\
x = w \text{ or } x \notin \mathsf{F}(R) \quad &\Rightarrow \quad R_{[S/w][T/x]} \equiv_\alpha R_{[S_{[T/x]}/w]} \quad \text{and} \\
x \neq w \notin \mathsf{F}(T) \quad &\Rightarrow \quad R_{[S/w][T/x]} \equiv_\alpha R_{[T/x][S_{[T/x]}/w]}
\end{aligned}$$

(again see [HS08, Appendix A], for example). Indeed, this is why we still need α-conversion even in such a bare bones type system.

6. Abbreviations

It is convenient to define various abbreviations for terms to make them easier to describe. For starters, let us adopt the more standard convention of replacing β's by parentheses, i.e. for any $t_0, \ldots, t_n \in \mathsf{T}$, let β^n denote β repeated n times and define

$$(t_0 t_1 \ldots t_{n-1} t_n) := \beta^n t_0 t_1 \ldots t_{n-1} t_n.$$

We even omit the parentheses if no ambiguity arises.

Next let us define $\pi := \rho\lambda$ so, for all $x \in \mathsf{V}$ and $R, S \in \mathsf{T}$,

$$\llbracket \pi x R S \rrbracket = \llbracket \rho\lambda x R S \rrbracket = \prod \llbracket \lambda x R S \rrbracket = \prod_{r \in \llbracket R \rrbracket} \llbracket \lambda x R S \rrbracket(r) = \prod_{r \in \llbracket R \rrbracket} \llbracket S \rrbracket_{\langle r, x \rangle}.$$

So $[\![\pi x RS]\!]$ is a set of functions on $[\![R]\!]$ and accordingly we will also sometimes use the more suggestive notation $(x : R) \to S$, i.e.

$$(x : R) \to S \;:=\; \pi x RS.$$

As usual, multiple arrows associate to the right by default, i.e.

$$(x : R) \to (y : S) \to T := (x : R) \to ((y : S) \to T) = \pi x R \pi y S T.$$

When x is not free in S, we can even omit it, i.e.

$$R \to S \;:=\; \pi x RS, \quad \text{where } x \notin \mathsf{F}(S).$$

As before, it does not really matter which $x \in \mathsf{V} \setminus \mathsf{F}(S)$ we choose, although by default we could again choose the smallest, for example. By Proposition 3.1, $R \to S$ is interpreted as the set of all functions $\phi : [\![R]\!] \to [\![S]\!]$, i.e. such that $\mathrm{dom}(\phi) = [\![R]\!]$ and $\mathrm{ran}(\phi) \subseteq [\![S]\!]$, as

$$[\![R \to S]\!] = [\![\pi x RS]\!] \;=\; \prod_{r \in [\![R]\!]} [\![S]\!]_{\langle r, x \rangle} = \prod_{r \in [\![R]\!]} [\![S]\!] \;=\; [\![S]\!]^{[\![R]\!]}.$$

The 'propositions as types' philosophy views any set as an intuitionistic/constructive 'proposition' with its elements being the 'proofs' of the proposition. So from this viewpoint, $R \to S$ is interpreted as a function that takes proofs of $[\![R]\!]$ to proofs of $[\![S]\!]$, which is the constructive view of logical implication, as per the Brouwer–Heyting–Kolmogorov interpretation of intuitionistic logic.

This viewpoint also dictates that the empty set has no proofs and thus represents the unique false proposition, while any non-empty set represents a true proposition. Accordingly, we can view any function as encoding a relation which holds precisely when the result of applying the function is non-empty. In other words, for any sets R and S, any function $F \in S^R$ yields a corresponding subset/unary relation on R defined by $\underline{F} = \{r \in R : F(r) \neq \emptyset\}$. For example, if $R = Q \times Q$, for some set Q, then $\underline{F}$ is a binary relation on Q such that, for all $p, q \in Q$,

$$p \; \underline{F} \; q \qquad \Leftrightarrow \qquad F(p, q) \neq \emptyset.$$

To encode relations as functions in this way, it is convenient to distinguish a constant $* \in \mathsf{C}$ to denote a default family of sets we wish to consider as propositions. We then define $\bot$ as an abbreviation for the term $\pi \mathsf{v} * \mathsf{v}$, which will be interpreted as $\emptyset$ as long as $\emptyset \in [\![*]\!]$, as then

$$(6.1) \qquad [\![\bot]\!] = [\![\pi \mathsf{v} * \mathsf{v}]\!] = \prod_{s \in [\![*]\!]} [\![\mathsf{v}]\!]_{\langle s, \mathsf{v} \rangle} = \emptyset \times \prod_{s \in [\![*]\!] \setminus \{\emptyset\}} [\![\mathsf{v}]\!]_{\langle s, \mathsf{v} \rangle} = \emptyset.$$

So in this case $\bot$ is interpreted as 'false'. We then define $\top := \bot \to \bot$, which is interpreted as a true proposition as long as $\emptyset, \{\emptyset\} \in [\![*]\!]$, as

$$[\![\top]\!] = [\![\bot \to \bot]\!] = [\![\bot]\!]^{[\![\bot]\!]} = \emptyset^\emptyset = \{\emptyset\}.$$

For any term S, it thus makes sense to define its *negation* $\neg S$ by

$$\neg S \ := \ S \to \bot.$$

As desired, this has the effect of switching true and false, as

(6.2) $\qquad [\![\neg S]\!] = [\![S \to \bot]\!] = [\![\bot]\!]^{[\![S]\!]} = \emptyset^{[\![S]\!]} = \begin{cases} \emptyset & \text{if } [\![S]\!] \neq \emptyset \\ \{\emptyset\} & \text{if } [\![S]\!] = \emptyset. \end{cases}$

As long as $\emptyset, \{\emptyset\} \in [\![*]\!]$, this shows that negation also has the effect turning an arbitrary set into one of our default propositions. Let us further define the *truncation* of any $S \in \mathsf{T}$ as its double negation $\neg\neg S$. From the 'propositions as types' perspective, truncation has the effect of identifying all proofs of the proposition in question. Indeed,

(6.3) $\qquad [\![\neg\neg S]\!] = \begin{cases} \emptyset & \text{if } [\![\neg S]\!] \neq \emptyset \\ \{\emptyset\} & \text{if } [\![\neg S]\!] = \emptyset \end{cases} = \begin{cases} \emptyset & \text{if } [\![S]\!] = \emptyset \\ \{\emptyset\} & \text{if } [\![S]\!] \neq \emptyset. \end{cases}$

We can also define other logical operations that are consistent with the 'propositions as types' viewpoint. For example, we define the 'and' operation $\wedge$ by

$$R \wedge S \ := \ \neg(R \to \neg S).$$

Then $R \wedge S$ is 'true' precisely when R and S are, as

$$[\![R \wedge S]\!] \neq \emptyset \quad \Leftrightarrow \quad [\![R \to \neg S]\!] = \emptyset \quad \Leftrightarrow \quad [\![R]\!] \neq \emptyset \neq [\![S]\!].$$

From the 'propositions as types' perspective, we also see that π acts like the universal quantifier. Indeed, for all $x \in \mathsf{V}$ and $R, S \in \mathsf{T}$, we see that $\pi x R S$ is 'true' precisely when S is, for all values of $[\![R]\!]$, as

$$[\![\pi x R S]\!] \neq \emptyset \quad \Leftrightarrow \quad \prod_{r \in [\![R]\!]} [\![S]\!]_{\langle r,x \rangle} \neq \emptyset \quad \Leftrightarrow \quad [\![S]\!]_{\langle r,x \rangle} \neq \emptyset, \text{ for all } r \in [\![R]\!].$$

The only issue is that $\pi x R S$ may not be a default proposition, even when S is, i.e. we can have $[\![\pi x R S]\!] \notin [\![*]\!] \ni [\![S]\!]_{\langle r,x \rangle}$, for all $r \in [\![R]\!]$. To deal with this, we can define $\forall$ as the truncation of π, i.e. $\forall := \neg\neg\pi$, so that $[\![\forall x R S]\!] \in [\![*]\!]$, as long as $\emptyset, \{\emptyset\} \in [\![*]\!]$, and still

$$[\![\forall x R S]\!] \neq \emptyset \quad \Leftrightarrow \quad [\![S]\!]_{\langle r,x \rangle} \neq \emptyset, \text{ for all } r \in [\![R]\!].$$

Remark 6.1. Many logical systems based on dependent types take $\forall$ to be exactly the same as π and introduce an inference rule to the effect that π-terms of propositions remain propositions, i.e. of type $*$. From a syntactic point of view, this 'impredicativity' of $*$ might seem convenient, but unfortunately it is difficult if not impossible to justify semantically in any satisfactory way. To show that such an impredicative $*$ is semantically sound in [MW03] and [Car19], for example, they first have to split the language into propositional and non-propositional

parts. This breaks the key selling point of 'propositions as types', namely that types and propositions can be treated in a uniform way. These papers also rely on more complicated interpretations that depend not just on an assignment of sets to the variables and constants, but also on a given set of assumptions Γ that can change the way terms are interpreted. We feel it is better to keep the language unified and the semantics simple and just use truncation to simulate impredicativity by defining $\forall$ as $\neg\neg\pi$.

Alternatively, logical operators (including $\forall$) can be defined as constant symbols satisfying appropriate specifications. This is more in the spirit of our second logical system and is discussed below in §13.

7. Consequences

Any interpretation $[\![\cdot]\!]$ yields a *typing relation* $\in_{[\![\cdot]\!]}$ on T defined by

$$\in_{[\![\cdot]\!]} = \{\langle S, P\rangle \mid S, P \in \mathsf{T} \text{ and } [\![S]\!] \in [\![P]\!]\}.$$

Alternatively, we can see $\in_{[\![\cdot]\!]}$ as a unary relation on (*typing*) *statements* S, which are formally ordered pairs of $\equiv_\alpha$-equivalence classes, i.e.

(Statements) $$\mathsf{S} = [\mathsf{T}]_\alpha \times [\mathsf{T}]_\alpha.$$

As usual, for any $S, P \in \mathsf{T}$, we denote the corresponding statement by

$$(S : P) = \langle [S]_\alpha, [P]_\alpha\rangle$$

or just $S : P$ when convenient, reading this as 'S is of type P'. Here S is the *subject* of the statement, while P is the *predicate* of the statement. We invoke Corollary 5.4 to view $\in_{[\![\cdot]\!]}$ as a unary relation on statements, which we again denote by $[\![\cdot]\!]$, i.e. for all $S, P \in \mathsf{T}$,

$$[\![S : P]\!] \qquad \Leftrightarrow \qquad S \in_{[\![\cdot]\!]} P \qquad \Leftrightarrow \qquad [\![S]\!] \in [\![P]\!].$$

In this case we say $(S : P)$ is *satisfied* by the given interpretation $[\![\cdot]\!]$. We further extend this to sets of statements $\Gamma \subseteq \mathsf{S}$ by defining

$$[\![\Gamma]\!] \quad \Leftrightarrow \quad [\![S]\!] \in [\![P]\!] \text{ whenever } (S : P) \in \Gamma.$$

When $[\![\Gamma]\!]$ holds, we say that the interpretation $[\![\cdot]\!]$ is a *model* for Γ.

Definition 7.1. The *consequence relation* $\vDash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$ is defined by

$$\Gamma \vDash (S : P) \quad \Leftrightarrow \quad [\![\Gamma]\!] \Rightarrow [\![S : P]\!], \text{ for every interpretation } [\![\cdot]\!].$$

So $\Gamma \vDash (S : P)$ means every model of Γ is a model of $(S : P)$. We could already view $\vDash$ as a kind of 'semantic foundation' of mathematics, in the sense that any mathematical problem can be reduced to proving a particular instance of $\vDash$. But of course this may not make the problem any easier, because as yet we have no obvious way of checking $\vDash$. What

we really want is some kind of syntactic equivalent of $\vDash$, or at least a good enough syntactic approximation of $\vDash$, obeying rules that can be checked mechanically by a computer, for example. To find such an approximation, the idea is to investigate properties of $\vDash$ that we can later take as inference rules defining a syntactic 'inference' relation $\vdash \subseteq \vDash$. This is the goal of the present section.

The first basic properties to look at are those common to all classical logical systems. First let us extend any $\Vdash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$ to a binary relation on $\mathcal{P}(\mathsf{S})$ so that, for all $\Gamma, \Delta \subseteq \mathsf{S}$ and $S, P \in \mathsf{T}$,

$$\Gamma \Vdash \Delta \qquad \Leftrightarrow \qquad \Gamma \Vdash (S : P), \text{ for all } (S : P) \in \Delta.$$

We call $\Vdash$ a *sequent* if this extension defines a preorder on $\mathcal{P}(\mathsf{S})$. More explicitly this means that, for all $\Gamma, \Delta \subseteq \mathsf{S}$ and $S, P \in \mathsf{T}$,

(Sequent) $\qquad (S : P) \in \Gamma \text{ or } \Gamma \Vdash \Delta \Vdash (S : P) \quad \Rightarrow \quad \Gamma \Vdash (S : P).$

Proposition 7.2. *The consequence relation $\vDash$ is a sequent.*

Proof. If $(S : P) \in \Gamma$ and $[\![\Gamma]\!]$ holds then, in particular, $[\![S : P]\!]$ holds, showing that $\Gamma \vDash (S : P)$. Likewise, if $\Gamma \Vdash \Delta \Vdash (S : P)$ and $[\![\Gamma]\!]$ holds then so does $[\![\Delta]\!]$ and hence $[\![S : P]\!]$, showing that $\Gamma \vDash (S : P)$. $\qquad\square$

Let us call a relation $\Vdash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$ *monotone* if $\supseteq \circ \Vdash \subseteq \Vdash$, i.e. if, for all $\Gamma, \Delta \subseteq \mathsf{S}$ and $S, P \in \mathsf{T}$,

(Monotone) $\qquad\qquad \Gamma \supseteq \Delta \Vdash (S : P) \quad \Rightarrow \quad \Gamma \Vdash (S : P).$

We call $\Vdash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$ *reflexive* if, for all $S, P \in \mathsf{T}$,

(Reflexive) $\qquad\qquad\qquad (S : P) \Vdash (S : P).$

Proposition 7.3. *Every sequent $\Vdash$ is monotone and reflexive.*

Proof. Note $(S : P) \in \{(S : P)\}$ and hence $(S : P) \Vdash (S : P)$, proving reflexivity. Also if $\Gamma \supseteq \Delta \Vdash (S : P)$ then $\Gamma \Vdash \Delta \Vdash (S : P)$ and hence $\Gamma \Vdash (S : P)$, proving monotonicity. $\qquad\square$

The following is now immediate from Propositions 7.2 and 7.3.

Corollary 7.4. *The consequence relation $\vDash$ is monotone and reflexive.*

Next we have the following property of $\vDash$ relating to ρ-terms and β-terms. For any $A, B \in \mathsf{S}$, let us abbreviate $\{A, B\}$ to A, B.

Proposition 7.5. *For any $F, P, R, S \in \mathsf{T}$ and $x \in \mathsf{V}$,*

$$(F : \pi x P R), (S : P) \vDash (FS : R_{[S/x]}).$$

Proof. Assume that $[\![\cdot]\!]$ is a model for $(F : \pi x PR)$ and $(S : P)$. This means $[\![F]\!] \in \prod_{p \in [\![P]\!]} [\![R]\!]_{\langle p, x \rangle}$ and $[\![S]\!] \in [\![P]\!]$ and hence

$$[\![FS]\!] = [\![F]\!]([\![S]\!]) \in [\![R]\!]_{\langle [\![S]\!], x \rangle} = [\![R_{[S/x]}]\!],$$

by (Substitutivity). This means $[\![FS : R_{[S/x]}]\!]$, as required. $\square$

We denote the free variables in any set of statements $\Gamma \subseteq \mathsf{S}$ by

$$\mathsf{F}(\Gamma) = \bigcup_{(S:P) \in \Gamma} \mathsf{F}(S) \cup \mathsf{F}(P)$$

We also write $\Gamma, (x : Q)$ as an abbreviation for $\Gamma \cup \{x : Q\}$.

Proposition 7.6. *For $\Gamma \subseteq \mathsf{S}$, $P, Q, S \in \mathsf{T}$ and $x \in \mathsf{V} \setminus (\mathsf{F}(\Gamma) \cup \mathsf{F}(Q))$,*

$$\Gamma, (x : Q) \vDash (S : P) \quad \Rightarrow \quad \Gamma \vDash (\lambda x Q S : \pi x Q P).$$

Proof. Assume $\Gamma, (x : Q) \vDash (S : P)$ and take a model $[\![\cdot]\!]$ for Γ. If $q \in [\![Q]\!] = [\![Q]\!]_{\langle q, x \rangle}$, as $x \notin \mathsf{F}(Q)$, then $[\![x : Q]\!]_{\langle q, x \rangle}$ and $[\![\Gamma]\!]_{\langle q, x \rangle}$, as $x \notin \mathsf{F}(\Gamma)$. Thus $[\![S : P]\!]_{\langle q, x \rangle}$, i.e. $[\![S]\!]_{\langle q, x \rangle} \in [\![P]\!]_{\langle q, x \rangle}$ and hence

$$[\![\lambda x Q S]\!] = \{\langle [\![S]\!]_{\langle q, x \rangle}, q \rangle : q \in [\![Q]\!]\} \in \prod_{q \in [\![Q]\!]} [\![P]\!]_{\langle q, x \rangle} = [\![\pi x Q P]\!].$$

This shows that $[\![\lambda x Q S : \pi x Q P]\!]$ and hence $\Gamma \vDash (\lambda x Q S : \pi x Q P)$. $\square$

8. INFERENCES

The classic system of predicate logic has just two inference rules, modus ponens and universal generalisation (as in [CK90, 1.3.8 and 1.3.9], for example). Viewing propositions as types, we see that these rules are analogous to the properties of $\vDash$ proved in Propositions 7.5 and 7.6. Indeed, when $x \notin \mathsf{F}(R)$, Proposition 7.5 becomes

$$(F : P \to R), (S : P) \vDash (FS : R)$$

which turns into modus ponens once the subjects are erased. Likewise, erasing the subjects in Proposition 7.6 and replacing π with $\forall$ yields a rule of predicate logic that follows from universal generalisation.

Accordingly, we take Propositions 7.5 and 7.6 as the sole inference rules defining our *inference relation* $\vdash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$.

Definition 8.1. Let $\vdash$ be the smallest sequent such that, for all $\Gamma \subseteq \mathsf{S}$, $F, P, Q, R, S \in \mathsf{T}$ and $x \in \mathsf{V} \setminus (\mathsf{F}(\Gamma) \cup \mathsf{F}(Q))$,

(App$'$) $(F : \pi x PR), (S : P) \vdash (FS : R_{[S/x]})$ and

(Ab) $\Gamma, (x : Q) \vdash (S : P) \quad \Rightarrow \quad \Gamma \vdash (\lambda x Q S : \pi x Q P).$

The following is now immediate from Propositions 7.2, 7.5 and 7.6.

Proposition 8.2. *The inference relation is sound, i.e.* $\vdash \ \subseteq \ \vDash$.

Many type systems in the literature restrict the left side of $\vdash$ to special finite sequences of statements known as 'legal contexts'. We make no such restriction here, although any particular instance of $\vdash$ is already determined by a finite subset of assumptions on the left, i.e.

$$\Gamma \vdash (S : P) \qquad \Rightarrow \qquad \exists \text{ finite } \Phi \subseteq \Gamma \ (\Phi \vdash (S : P)),$$

for any $\Gamma \subseteq \mathsf{S}$ and $S, P \in \mathsf{T}$. To prove this and various other properties of the inference relation, it will be convenient to stratify $\vdash$ based on the number of times the inference rules need to be applied.

Accordingly, let $\vdash_0$ be the smallest monotone reflexive relation, i.e.

$$\Gamma \vdash_0 (S : P) \qquad \Leftrightarrow \qquad (S : P) \in \Gamma.$$

Given $\vdash_n$, let $\vdash_{n+1}$ be the smallest monotone reflexive relation such that, for all $\Gamma \subseteq \mathsf{S}$, $F, P, Q, R, S \in \mathsf{T}$, $x \in \mathsf{V}$ and $y \in \mathsf{V} \setminus (\mathsf{F}(\Gamma) \cup \mathsf{F}(Q))$,

$(\mathrm{App}_n) \quad \Gamma \vdash_n (F : \pi x P R), (S : P) \quad \Rightarrow \quad \Gamma \vdash_{n+1} (FS : R_{[S/x]}) \quad$ and

$(\mathrm{Ab}_n) \qquad \Gamma, (y : Q) \vdash_n (S : P) \quad \Rightarrow \quad \Gamma \vdash_{n+1} (\lambda y Q S : \pi y Q P).$

Let us also define corresponding *typing functions* $\lceil \cdot \rceil_n^\Gamma \in \mathcal{P}(\mathsf{T})^\mathsf{T}$, for all $\Gamma \subseteq \mathcal{P}(\mathsf{S})$ and $n \in \omega$, as follows. First, for any $S \in \mathsf{T}$, let

$$\lceil S \rceil_0^\Gamma = \Gamma^{-1}([S]_\alpha) = \{P \mid (S : P) \in \Gamma\}.$$

Once $\lceil \cdot \rceil_n^\Gamma$ has been defined, we define $\lceil \cdot \rceil_{n+1}^\Gamma$ by

$$\lceil t \rceil_{n+1}^\Gamma = \lceil t \rceil_0^\Gamma, \text{ if } t \in \mathsf{T}^0 \cup \rho\mathsf{T},$$

$$\lceil FS \rceil_{n+1}^\Gamma = \lceil FS \rceil_0^\Gamma \cup \bigcup \{[R_{[S/x]}]_\alpha \mid \exists P \in \lceil S \rceil_n^\Gamma \ (\pi x P R \in \lceil F \rceil_n^\Gamma)\} \text{ and}$$

$$\lceil \lambda x Q S \rceil_{n+1}^\Gamma = \lceil \lambda x Q S \rceil_0^\Gamma \cup \bigcup \{[\pi x Q P]_\alpha \mid \exists \Delta \subseteq \Gamma \ (x \notin \mathsf{F}(\Delta) \cup \mathsf{F}(Q) \text{ and } P \in \lceil S \rceil_n^{\Delta,(x:Q)})\}.$$

First we observe $\lceil \cdot \rceil_n^\Gamma$ is determined by its restriction to finite Γ.

Proposition 8.3. *For any $n \in \omega$ and $\Gamma \subseteq \mathsf{S}$,*

$$\lceil S \rceil_n^\Gamma = \bigcup \{\lceil S \rceil_n^\Phi \mid \Phi \subseteq \Gamma \text{ is finite}\}.$$

Proof. First we claim that, for all $n \in \omega$, $\Gamma, \Delta \subseteq \mathsf{S}$ and $S \in \mathsf{T}$,

$$(8.1) \qquad\qquad \Delta \subseteq \Gamma \qquad \Rightarrow \qquad \lceil S \rceil_n^\Delta \subseteq \lceil S \rceil_n^\Gamma.$$

Indeed, if $\Delta \subseteq \Gamma$ then $\lceil S \rceil_0^\Delta = \Delta^{-1}([S]_\alpha) \subseteq \Gamma^{-1}([S]_\alpha) = \lceil S \rceil_0^\Gamma$. Also $\Delta' \subseteq \Delta$ then implies $\Delta' \subseteq \Gamma$ and hence $\lceil \lambda x Q S \rceil_n^\Delta \subseteq \lceil \lambda x Q S \rceil_n^\Gamma$, for all $\Delta' \subseteq \mathsf{S}$, $x \in \mathsf{V}$, $Q, S \in \mathsf{T}$ and $n \in \omega$. Assuming (8.1) holds for n, it also then follows that $P \in \lceil S \rceil_n^\Delta$ and $\pi x P R \in \lceil F \rceil_n^\Delta$ implies $P \in \lceil S \rceil_n^\Gamma$ and

$\pi x P R \in \lceil F \rceil_n^\Gamma$ and hence $\lceil F S \rceil_{n+1}^\Delta \subseteq \lceil F S \rceil_{n+1}^\Gamma$, for all $F, S \in \mathsf{T}$. This proves (8.1) for $n+1$ and hence all $n \in \omega$, by induction. In particular,

$$\bigcup \{ \lceil S \rceil_n^\Phi \mid \Phi \subseteq \Gamma \text{ is finite} \} \subseteq \lceil S \rceil_n^\Gamma.$$

For the reverse inclusion, we must show that

$$(8.2) \qquad P \in \lceil S \rceil_n^\Gamma \qquad \Rightarrow \qquad \exists \text{ finite } \Phi \subseteq \Gamma \ (P \in \lceil S \rceil_n^\Phi).$$

By definition, $P \in \lceil S \rceil_0^\Gamma$ implies $(S : P) \in \Gamma$ and $P \in \lceil S \rceil_0^{(S:P)}$, proving (8.2) for $n = 0$. Now assume (8.2) for n and take $P \in \lceil S \rceil_{n+1}^\Gamma$. If $(S : P) \in \Gamma$ then again $P \in \lceil S \rceil_0^{(S:P)} \subseteq \lceil S \rceil_{n+1}^{(S:P)}$. Otherwise S is a β-term or a λ-term. In the former case, we have $F, P', R, S' \in \mathsf{T}$ such that $S = F S'$, $P = R_{[S'/x]}$, $P' \in \lceil S' \rceil_n^\Gamma$ and $\pi x P' R \in \lceil F \rceil_n^\Gamma$. Then we have finite $\Sigma, \Delta \subseteq \Gamma$ with $P' \in \lceil S' \rceil_n^\Sigma$ and $\pi x P' R \in \lceil F \rceil_n^\Delta$. But then $\Phi = \Sigma \cup \Delta$ is also finite with $P' \in \lceil S' \rceil_n^\Phi$ and $\pi x P' R \in \lceil F \rceil_n^\Phi$, by (8.1), and hence $P \in \lceil S \rceil_{n+1}^\Phi$. The latter case is proved similarly, thus proving the result for $n+1$ and hence all $n \in \omega$, by induction. □

Next we observe that the typing functions do indeed characterise the corresponding inference relations.

Proposition 8.4. *For all $n \in \omega$, $\Gamma \subseteq \mathsf{S}$ and $S, P \in \mathsf{T}$,*

$$\Gamma \vdash_n (S : P) \qquad \Leftrightarrow \qquad P \in \lceil S \rceil_n^\Gamma.$$

Proof. The $n = 0$ case holds by definition. Now assume the result for n. For the $\Leftarrow$ part, first note that if $P \in \lceil S \rceil_0^\Gamma$ and hence $(S : P) \in \Gamma$ then $\Gamma \vdash_{n+1} (S : P)$, as $\vdash_{n+1}$ is monotone and reflexive. In particular, $P \in \lceil t \rceil_{n+1}^\Gamma$ implies $\Gamma \vdash_n (t : P)$ when $t \in \mathsf{T}^0 \cup \rho \mathsf{T}$. Next note that if $P \in \lceil S \rceil_n^\Gamma$ and $\pi x P R \in \lceil F \rceil_n^\Gamma$ then, by the inductive hypothesis, $\Gamma \vdash_n (F : \pi x P R), (S : P)$ and hence $\Gamma \vdash_{n+1} (F S : R_{[S/x]})$. Thus $P' \in \lceil F S \rceil_{n+1}^\Gamma$ implies $\Gamma \vdash_{n+1} (F S : P')$. Finally note that if we have $\Delta \subseteq \Gamma$, $x \in \mathsf{V} \setminus (\mathsf{F}(\Delta) \cup \mathsf{F}(Q))$ and $P \in \lceil S \rceil_n^{\Delta,(x:Q)}$ then, again by the inductive hypothesis, $\Delta, (x : Q) \vdash_n (S : P)$ so $\Delta \vdash_{n+1} (\lambda x Q S : \pi x Q P)$ and hence $\Gamma \vdash_{n+1} (\lambda x Q S : \pi x Q P)$, as $\vdash_{n+1}$ is monotone. This means $P' \in \lceil \lambda x Q S \rceil_{n+1}^\Gamma$ implies $\Gamma \vdash_{n+1} (\lambda x Q S : P')$. Thus the $n+1$ case also holds and the $\Leftarrow$ part for all $n \in \omega$ follows by induction.

Conversely, first note that a simple inductive argument shows that each $\lceil \cdot \rceil_n^\Gamma$ is α-invariant in the sense that $S \equiv_\alpha S'$ implies $\lceil S \rceil_n^\Gamma = \lceil S' \rceil_n^\Gamma$ and $P' \equiv_\alpha P \in \lceil S \rceil_n^\Gamma$ implies $P' \in \lceil S \rceil_n^\Gamma$. For each $n \in \omega$, we may thus temporarily define $\Gamma \vdash_n' (S : P)$ to mean $P \in \lceil S \rceil_n^\Gamma$. As $\lceil S \rceil_0^\Gamma \subseteq \lceil S \rceil_n^\Gamma$, we immediately see that $\vdash_n'$ is reflexive. Also $\vdash_n'$ is monotone, by (8.1). The definition of $\lceil \cdot \rceil_n^\Gamma$ also immediately shows that $\vdash_n'$ satisfies (App_n) and (Ab_n), showing that $\vdash_n \subseteq \vdash_n'$ and hence $\vdash_n = \vdash_n'$, as required. □

Now we observe $\vdash_n$ is invariant under substitution. Here we define

$$\Gamma_{[T/x]} = \{(S_{[T/x]} : P_{[T/x]}) \mid (S : P) \in \Gamma\}.$$

Proposition 8.5. *For any $n \in \omega$, $\Gamma \subseteq \mathsf{S}$, $P, S, T \in \mathsf{T}$ and $x \in \mathsf{V}$,*

$$\Gamma \vdash_n (S : P) \qquad \Rightarrow \qquad \Gamma_{[T/x]} \vdash_n (S_{[T/x]} : P_{[T/x]}).$$

Proof. First note $(S : P) \in \Gamma$ implies $(S_{[T/x]} : P_{[T/x]}) \in \Gamma_{[T/x]}$ or, equivalently, $\Gamma \vdash_0 (S : P)$ implies $\Gamma_{[T/x]} \vdash_0 (S_{[T/x]} : P_{[T/x]})$.

Now assume the result for n. If $\Gamma \vdash_n (S : P)$ and $\Gamma \vdash_n (F : \pi x P R)$ then this means $\Gamma_{[T/x]} \vdash_n (S_{[T/x]} : P_{[T/x]})$. Moreover, if $y = x$ or $x \notin \mathsf{F}(R)$ then $\Gamma_{[T/x]} \vdash_n (F_{[T/x]} : (\pi y P R)_{[T/x]}) = (F_{[T/x]} : \pi y P_{[T/x]} R)$ so

$$\Gamma_{[T/x]} \vdash_{n+1} (F_{[T/x]} S_{[T/x]} : R_{[S_{[T/x]}/y]}) = ((FS)_{[T/x]} : R_{[S/y][T/x]}).$$

On the other hand, if $y \neq x \in \mathsf{F}(R)$ then we have $z \in \mathsf{V} \setminus (\mathsf{V}(RT))$ with $\Gamma_{[T/x]} \vdash_n (F_{[T/x]} : (\pi y P R)_{[T/x]}) = (F_{[T/x]} : \pi z P_{[T/x]} R_{[z/y][T/x]})$ so

$$\Gamma_{[T/x]} \vdash_{n+1} (F_{[T/x]} S_{[T/x]} : R_{[z/y][T/x][S_{[T/x]}/z]}) = ((FS)_{[T/x]} : R_{[z/y][S/z][T/x]})$$
$$= ((FS)_{[T/x]} : R_{[S/y][T/x]}).$$

Thus $\Gamma \vdash_{n+1} (FS : P')$ implies $\Gamma_{[T/x]} \vdash_{n+1} ((FS)_{[T/x]} : P'_{[T/x]})$.

Now say Δ is finite, $y \in \mathsf{V} \setminus (\mathsf{F}(\Delta) \cup \mathsf{F}(Q))$ and $\Delta, (y : Q) \vdash_n (S : P)$. If $y \neq x$ then, for any $z \in \mathsf{V} \setminus (\mathsf{F}(\Delta) \cup \mathsf{F}(xPQST))$, it follows that $\Delta, (z : Q) \vdash_n (S_{[z/y]} : P_{[z/y]})$ so

$$\Delta_{[T/x]}, (z : Q_{[T/x]}) \vdash_n (S_{[z/y][T/x]} : P_{[z/y][T/x]})$$

and hence

$$\Delta_{[T/x]} \vdash_{n+1} (\lambda z Q_{[T/x]} S_{[z/y][T/x]} : \pi z Q_{[T/x]} P_{[z/y][T/x]})$$
$$= ((\lambda y Q S)_{[T/x]} : (\pi y Q P)_{[T/x]})$$

On the other hand, if $y = x$ then we immediately see that

$$\Delta \vdash_{n+1} (\lambda y Q S : \pi y Q P) = (\lambda y Q_{[T/x]} S : \pi y Q_{[T/x]} P)$$
$$= ((\lambda y Q S)_{[T/x]} : (\pi y Q P)_{[T/x]}).$$

Thus $\Gamma \vdash_{n+1} (\lambda y Q S : P')$ implies $\Gamma_{[T/x]} \vdash_{n+1} ((\lambda y Q S)_{[T/x]} : P'_{[T/x]})$. The general result now follows by induction. $\square$

Using this we can show that $\vdash_m \circ \vdash_n \subseteq \vdash_{m+n}$, for all $m, n \in \omega$.

Proposition 8.6. *For all $m, n \in \omega$, $\Gamma, \Delta \subseteq \mathsf{S}$ and $S, P \in \mathsf{T}$,*

$$\Gamma \vdash_m \Delta \vdash_n (S : P) \qquad \Rightarrow \qquad \Gamma \vdash_{m+n} (S : P).$$

Proof. The proof is by induction. First note that $\Gamma \vdash_m \Delta \vdash_0 (S : P)$ means $\Gamma \vdash_m \Delta \ni (S : P)$, which immediately yields $\Gamma \vdash_m (S : P)$. Now assume the result holds for m and n and say $\Gamma \vdash_m \Delta \vdash_{n+1} (S : P)$. If $(S : P) \in \Delta$ then $\Gamma \vdash_m (S : P)$, as we just noted, and hence $\Gamma \vdash_{m+n} (S : P)$, as $\vdash_m \subseteq \vdash_{m+n}$. Otherwise S must be a β-term or a λ-term. In the former case, Proposition 8.4 yields $F, T, Q \in \mathsf{T}$ such that $S \equiv_\alpha FT$, $P \equiv_\alpha Q_{[T/x]}$ and $\Delta \vdash_n (F : \pi xRQ), (T : R)$. From the inductive assumption, it then follows that $\Gamma \vdash_{m+n} (F : \pi xRQ), (T : R)$ and hence $\Gamma \vdash_{m+n+1} (FT : Q_{[T/x]}) = (S : P)$, by (App_{m+n}). On the other hand, in the latter case Propositions 8.3 and 8.4 yield finite $\Sigma \subseteq \Delta$, $Q, S', P' \in \mathsf{T}$ and $x \in \mathsf{V} \setminus (\mathsf{F}(\Sigma) \cup \mathsf{F}(Q))$ such that $S \equiv_\alpha \lambda xQS'$, $P \equiv_\alpha \pi xQP'$ and $\Sigma, (x : Q) \vdash_n (S' : P')$. As $\Gamma \vdash_m \Delta \supseteq \Sigma$, we also have finite $\Phi \subseteq \Gamma$ with $\Phi \vdash_m \Sigma$. Taking any $y \in \mathsf{V} \setminus (\mathsf{F}(\Phi) \cup \mathsf{F}(P'QS'))$, Proposition 8.5 then yields $\Sigma, (y : Q) \vdash_n (S'_{[y/x]} : P'_{[y/x]})$. As $\vdash_m$ is monotone and reflexive, $\Phi, (y : Q) \vdash_m \Sigma, (y : Q)$ and so the inductive assumption yields $\Phi, (x : Q) \vdash_{m+n} (S'_{[y/x]} : P'_{[y/x]})$. Then (Ab_{m+n}) yields $\Gamma \supseteq \Phi \vdash_{m+n+1} (\lambda yQS'_{[y/x]} : \pi yQP'_{[y/x]}) = (S : P)$. $\square$

Now we can show $\vdash$ is the smallest monotone reflexive relation such that, for all $\Gamma \subseteq \mathsf{S}$, $F, Q, R, S, T \in \mathsf{T}$ and $x, y \in \mathsf{V}$ with $y \notin \mathsf{F}(\Gamma) \cup \mathsf{F}(Q)$,

(App) $\Gamma \vdash (F : \pi xRS), (T : R)$ $\Rightarrow$ $\Gamma \vdash (FT : S_{[T/x]})$ and

(Ab) $\Gamma, (y : Q) \vdash (S : P)$ $\Rightarrow$ $\Gamma \vdash (\lambda yQS : \pi yQP)$.

Corollary 8.7. *We have* $\vdash\ =\ \vdash_\omega := \bigcup_{n \in \omega} \vdash_n$.

Proof. Certainly $\vdash_0 \subseteq \vdash$. Now assume that $\vdash_n \subseteq \vdash$. Therefore if $\Gamma \vdash_n (F : \pi xRS), (T : R)$ then

$$\Gamma \vdash (F : \pi xRS), (T : R) \vdash (FT : S_{[T/x]})$$

so $\Gamma \vdash (FT : S_{[T/x]})$. And if $y \notin \mathsf{F}(\Gamma) \cup \mathsf{F}(Q)$ and $\Gamma, (y : Q) \vdash_n (S : P)$ then $\Gamma, (y : Q) \vdash (S : P)$ so $\Gamma \vdash (\lambda yQS : \pi yQP)$. Proposition 7.3 then yields $\vdash_{n+1} \subseteq \vdash$ so $\vdash_\omega \subseteq \vdash$, by induction.

Conversely, $\vdash_\omega$ is transitive, by (8.6). Also $\vdash_\omega$ contains $\vdash_0$ and is thus a sequent. But also $(F : \pi xRS), (T : R) \vdash_1 (FT : S_{[T/x]})$ and, for any $y \notin \mathsf{F}(\Gamma) \cup \mathsf{F}(Q)$, we see that $\Gamma, (y : Q) \vdash_\omega (S : P)$ implies $\Gamma \vdash_\omega (\lambda yQS : \pi yQP)$. This shows that $\vdash \subseteq \vdash_\omega$ as well. $\square$

Similarly, let us define $\lceil \cdot \rceil^\Gamma \in \mathcal{P}(\mathsf{T})^\mathsf{T}$ so that, for all $\Gamma \subseteq \mathsf{S}$ and $S \in \mathsf{T}$,

$$\lceil S \rceil^\Gamma = \bigcup_{n \in \omega} \lceil S \rceil_n^\Gamma.$$

The defining properties of $\lceil \cdot \rceil_n^\Gamma$ then also apply to $\lceil \cdot \rceil^\Gamma$, i.e.

$$\lceil t \rceil^\Gamma = \lceil t \rceil_0^\Gamma, \text{ if } t \in \mathsf{T}^0 \cup \rho\mathsf{T},$$

$$\lceil FS \rceil^\Gamma = \lceil FS \rceil_0^\Gamma \cup \bigcup \{[R_{[S/x]}]_\alpha \mid \exists P \in \lceil S \rceil^\Gamma \ (\pi x P R \in \lceil F \rceil^\Gamma)\} \text{ and}$$

$$\lceil \lambda x Q S \rceil^\Gamma = \lceil \lambda x Q S \rceil_0^\Gamma \cup \bigcup \{[\pi x Q P]_\alpha \mid \exists \Delta \subseteq \Gamma \ (x \notin \mathsf{F}(\Delta) \cup \mathsf{F}(Q) \text{ and } P \in \lceil S \rceil^{\Delta,(x:Q)})\}.$$

It follows that if Γ is finite then, for all $S \in \mathsf{T}$, $\lceil S \rceil^\Gamma$ is also finite, modulo α-conversion, and hence computable. In other words, given S and Γ, we can mechanically compute all the predicates/types P such that $\Gamma \vdash (S : P)$. More succinctly, we say that $\vdash$ has 'type inference'.

If we also want unique typing (again up to α-conversion) then we can restrict to certain subsets of S on the left.

Definition 8.8. We call $\Gamma \subseteq \mathsf{S}$ a *context* if Γ^{-1} is a function with

$$\mathrm{dom}(\Gamma^{-1}) \subseteq [\mathsf{T}^0 \cup \rho\mathsf{T}]_\alpha.$$

Put another way, a context Γ is just a set of type declarations for variables, constants and ρ-terms, at most one declaration for each term, i.e. if $P, Q, S \in \mathsf{T}$ then $(S : P) \in \Gamma$ implies $S \in \mathsf{T}^0 \cup \rho\mathsf{T}$ and also $(S : P), (S : Q) \in \Gamma$ implies $P \equiv_\alpha Q$.

Proposition 8.9. *If Γ is a context and $S, P, Q \in \mathsf{T}$ then*

$$\Gamma \vdash (S : P), (S : Q) \qquad \Rightarrow \qquad P \equiv_\alpha Q.$$

Proof. As Γ is a context, we know $\lceil S \rceil_0^\Gamma$ is empty unless $S \in \mathsf{T}^0 \cup \rho\mathsf{T}$, in which case all terms in $\lceil S \rceil_0^\Gamma$ are α-convertible. Now assume that this is true for n. Then $\lceil S \rceil_n^\Gamma$ and $\lceil F \rceil_n^\Gamma$ can contain at most one term up to α-convertibility and so the same is true of $\lceil FS \rceil_{n+1}^\Gamma$. On the other hand, say we have finite $\Delta, \Delta' \subseteq \Gamma$, $P, P', Q, S \in \mathsf{T}$ and $x \in \mathsf{V}$ such that $x \notin \mathsf{F}(\Delta) \cup \mathsf{F}(\Delta') \cup \mathsf{F}(Q)$, $P \in \lceil S \rceil_n^{\Delta,(x:Q)}$ and $P' \in \lceil S \rceil_n^{\Delta',(x:Q)}$. Then $P, P' \in \lceil S \rceil_n^{\Delta \cup \Delta',(x:Q)}$ so $P \equiv_\alpha P'$ and hence $\pi x Q P \equiv_\alpha \pi x Q P'$, showing that $\lceil \lambda x Q S \rceil_{n+1}^\Gamma$ also contains at most one term up to α-convertibility. This completes the proof for $\lceil \cdot \rceil_{n+1}^\Gamma$ and hence for $\vdash$ by induction. $\square$

9. Axioms

This completes the set-up of our bare bones type system and the examination of its basic syntactic properties. However, to use it as a foundational system, we also need some axioms. These are statements that are satisfied by all the interpretations we wish to consider and which we should thus be free to add to Γ when trying to verify a particular instance $\Gamma \vdash (S : P)$ of the inference relation.

9.1. **Equality.** So far we have not specified how to handle equality, even though this is of vital importance for any foundational system. Coming from classical predicate logic, the temptation would be to add equality statements $(R \equiv S)$ as a primitive notion, in addition to the typing statements $(R : S)$ we already have. One would then add an inference rule for substitution, e.g. stating that $\Gamma, (R \equiv S) \vdash (T : U)$ always implies $\Gamma, (R \equiv S) \vdash (T' : U')$, where T' and U' are obtained by replacing one or more instances of R by S. Indeed, if we were aiming to make $\vdash$ not just sound but also complete w.r.t. $\vDash$ in the traditional sense of $\vDash \subseteq \vdash$ then we would be forced to do something like this.

The problem is that these substitutions can break type inference and make even type checking an undecidable problem in general. In other words, there could cease to be any mechanical way of verifying whether $\Gamma \vdash (R : S)$ holds as there will potentially be infinitely many ways of getting from Γ to $(R : S)$ via sequences of substitutions. Also, when looking at a typing statement $(S : P)$ from the 'propositions as types' perspective, it is really only substitution in the predicate P that we care about. Indeed, from this viewpoint S is just code for a proof of P – we do not care if the code is longer than necessary if that means it is easier to decode in a mechanical way. And with this in mind, we would thus be happy with a weaker form of completeness, namely

$$\text{(Predicate Completeness)} \quad \Gamma \vDash (S : P) \quad \Rightarrow \quad \exists R \, (\Gamma \vdash (R : P)).$$

The standard solution is to instead take equality as a proposition, one which will allow us to do substitution in the predicate of a typing statement, at the cost of changing the subject. First recall that we have a constant $* \in \mathsf{C}$ to denote our default propositions. Given a term R on which we want to define equality, we then take further constants $\mathsf{eq}^*_R, \mathsf{sub}^*_R \in \mathsf{C}$ and introduce the axioms $(\mathsf{eq}^*_R : R \to R \to *)$ and

$$(\mathsf{sub}^*_R : (t : R) \to (u : R) \to \mathsf{eq}^*_R tu \to (P : R \to *) \to Pt \to Pu).$$

So if Γ contains these axioms and

$$\Gamma \vdash (g : \mathsf{eq}^*_R TU), (P : R \to *), (h : PT)$$

then the (App) rule yields $\Gamma \vdash (\mathsf{sub}^*_R TUgh : PU)$, showing that the T in PT can indeed be replaced by a U, at the cost of replacing the subject h with the longer term $\mathsf{sub}^*_R TUgh$.

In particular, P above could be of the form $\lambda x RS$. Given that we will then have $[\![(\lambda x RS)U]\!] = [\![S_{[U/x]}]\!]$, we would naturally like to replace the predicate $(\lambda x RS)U$ by $S_{[U/x]}$ as well. Again this can be achieved by

adding a propositional form of β-reduction as another axiom. Specifically, if $x \notin \mathsf{F}(R)$, $\mathsf{beta}_R^S \in \mathsf{C}$, $(\mathsf{beta}_R^S : (x : R) \to (\lambda x R S)x \to S) \in \Gamma$ and $\Gamma \vdash (h : (\lambda x R S)U), (U : R)$ then $\Gamma \vdash (\mathsf{beta}_R^S U h : S_{[U/x]})$.

9.2. Polymorphism.

One problem with the above approach is that we have to distinguish new constants and their axioms for each new term R that we wish to work with. If R happens to have free variables then another problem is that adding the axioms to Γ will then prevent us from applying the (Ab) rule with those variables. To some extent this is mitigated by the monotonicity of $\vdash$, which allows us to add the relevant axioms to Γ at the very end of a deduction. However, this is only true when $\Gamma \vdash (S : P)$ can already be proved without including those axioms in Γ.

To avoid these problems, it is natural to make the axioms 'polymorphic'. Specifically, we first distinguish a constant $\square$ which is meant to denote some universe of sets on which we would like to define various notions. The corresponding polymorphic notion of equality would then be defined by replacing the term R with a variable r and adding $(r : \square) \to$ to the beginning of the relevant predicates, e.g.

$$(\mathsf{eq}^* : (r : \square) \to r \to r \to *).$$

Likewise, to make β-reduction polymorphic in R, consider the axiom

$$(\mathsf{beta}^S : (r : \square) \to (x : r) \to (\lambda x r S)x \to S).$$

Note, however, that we can not make the β-reduction axiom polymorphic in S. One might naïvely consider the axiom

$$(9.1) \qquad (\mathsf{beta} : (s : \square) \to (r : \square) \to (x : r) \to (\lambda x r s)x \to s),$$

but when $x \in \mathsf{F}(S)$ (which is the situation we really care about), the substitution that would result from the (App) rule would end up changing the bound variable x. In other words, given Γ containing (9.1) and satisfying $\Gamma \vdash (S : \square)$, the (App) rule would result in $\Gamma \vdash (\mathsf{beta}S : (r : \square) \to (x : r) \to (\lambda y r S)x \to S)$, for some $y \neq x$.

This might be considered as one argument in favour of taking β-reduction as an inference rule rather than an axiom (as is usually done in pure type systems, for example) even though it could break the decidability of type checking, as mentioned above, or at least make it more difficult to prove. A possibly more compelling argument would be that, in the presence of a β-reduction rule like

$$(9.2) \qquad (S : Q), (T : \lambda x Q P S) \vdash (T : P_{[S/x]}),$$

we can modify the (App) rule to avoid substitution, making it

$$(F : \pi x Q R), (G : \rho F), (S : Q) \vdash (GS : FS).$$

Indeed, taking F to be $\lambda x Q P$ above and then applying (9.2) would then yield the original (App) rule.

We would naturally like polymorphic notions to apply to sets of functions as well, i.e. to terms of the form ρF, for some $F \in \mathsf{T}$. In an individual instance we could achieve this by simply adding $(\rho F : \square)$ as an axiom, so that our new polymorphic notions can indeed by applied on ρF. But again it would be nice if we did not have to add a new axiom like this for every new term F that we wish to work with. What we would really like is to be able to type ρ as $(R \to \square) \to \square$, for all terms R. But of course this does not make sense because syntactically ρ is not even a properly formed term on its own.

9.3. **Logic.** Before moving on, it is perhaps also worth mentioning the axioms that we do *not* need, namely most of the logical tautologies that are usually taken as axioms in (0th order) sentential logic or (1st order) predicate logic. For example, for any $R, S \in \mathsf{T}$, the tautology $R \to (R \to S) \to S$ is already witnessed by a term T, i.e. we can construct $T \in \mathsf{T}$ such that $\emptyset \vdash (T : R \to (R \to S) \to S)$. Indeed, taking any distinct $x, y \in \mathsf{V} \setminus \mathsf{F}(RS)$, we see that

$$(x : R), (y : R \to S) \vdash (yx : S), \text{ by (App)},$$
$$\text{so} \quad (x : R) \vdash (\lambda y (R \to S)(yx) : (R \to S) \to S), \text{ by (Ab)},$$
$$\text{and hence} \quad \emptyset \vdash (\lambda x R \lambda y (R \to S)(yx) : R \to (R \to S) \to S),$$

again by (Ab). In particular, taking $S = \bot$ yields $T \in \mathsf{T}$ with

$$\emptyset \vdash (T : R \to \neg\neg R).$$

However, in general we do not have any witness to the converse, i.e. we may not have any $S \in \mathsf{T}$ with $\emptyset \vdash (S : \neg\neg R \to R)$. In fact, if we are looking at the corresponding polymorphic term

$$(9.3) \qquad\qquad\qquad (x : \square) \to \neg\neg x \to x$$

then we see that any $\phi \in [\![(x : \square) \to \neg\neg x \to x]\!]$ is a function such that $\phi(\emptyset) = \emptyset$ while $\phi(r)(\emptyset) \in r$, for all $r \in [\![\square]\!] \setminus \{\emptyset\}$. Put another way, for each non-empty $r \in [\![\square]\!]$, the function ϕ is choosing an element of r. So if we are considering $\square$ as denoting a universe of sets then (9.3) is naturally viewed as the axiom of choice, while if we are instead taking $\square$ to denote a default set of propositions then (9.3) is a form of the law of the excluded middle. This could be viewed as a type theoretic manifestation of Diaconescu's theorem.

9.4. **Improvements.** Motivated by the remarks in §9.2, it is natural to build a new type system with inbuilt polymorphism as follows. From the outset, we distinguish a constant $\square$ to denote a universe of sets we wish to apply polymorphic definitions to. We then distinguish another constant p to act as a product operator on the terms of $\square$, replacing all formal product terms in the inference rules with terms involving p instead. To ensure that polymorphism also applies to product terms, we can type p as $(r : \square) \to (r \to \square) \to \square$. The only issue is that the $\to$ symbols here should also be abbreviations for product terms, but unfolding them using the same product symbol p would only be valid if $\square : \square$, which is not satisfied by any interpretation (because sets are $\in$-well-founded, by the axiom of foundation in ZFC). Thus we are naturally led to distinguishing another constant $\square'$ to denote another universe with $\square : \square'$ as well as another product operator constant $\mathsf{p}' : (r : \square') \to (r \to \square') \to \square'$ which we can use to unfold the above arrows. But to unfold these new arrows in turn requires another constant $\square''$ for an even higher universe, etc.. In the end we are led to distinguishing a countably infinite collection of constants to denote universes and products between them. It is also natural then to take the lowest universe as a default set of propositions.

Our new system will also have (sub-)reduction statements and rules allowing us to replace the corresponding terms in typing statements. We will usually take all β-sub-reduction statements as axioms and accordingly our (App) rule will be replaced with a substitution-free version, as outlined above. The computation rules usually associated to inductive definitions can also be handled by adding them as (sub-)reduction statements. Of course, the more of these statements we add, the more likely type-checking will become undecidable or difficult to prove. But given the prevalence of reduction/computation rules resulting from 'definitional equality' in the type systems underlying modern proof assistants, it seems that this is a price many are willing to pay.

Part 2. **A System with Polymorphic Product Operators**

The preliminaries of our next system will be much like the first. Thus we will be more terse and just review what is needed, pointing out the crucial differences – see Part 1 for more details.

10. THE LANGUAGE

The language of our second system is just like the first minus the ρ-terms. Specifically, the alphabet has five symbols denoted by

$$\mathsf{A} = \{\mathsf{v}, \mathsf{c}, {}', \beta, \lambda\}.$$

The strings we are interested in are defined by

(Constants) $\mathsf{C} ::= \mathsf{c} \mid \mathsf{C}'$

(Variables) $\mathsf{V} ::= \mathsf{v} \mid \mathsf{V}'$

(Terms) $\mathsf{T} ::= \mathsf{C} \mid \mathsf{V} \mid \beta\mathsf{TT} \mid \lambda\mathsf{VTT}$

So $\mathsf{C} = \{\mathsf{c}, \mathsf{c}', \mathsf{c}'', \dots\}$, $\mathsf{V} = \{\mathsf{v}, \mathsf{v}', \mathsf{v}'', \dots\}$ and T is minimal with

$$\mathsf{C} \cup \mathsf{V} \cup \beta\mathsf{TT} \cup \lambda\mathsf{VTT} \subseteq \mathsf{T}.$$

Let $\mathsf{T}^0 = \mathsf{C} \cup \mathsf{V}$ and $\mathsf{T}^{n+1} = \mathsf{T}^0 \cup \beta\mathsf{T}^n\mathsf{T}^n \cup \lambda\mathsf{VT}^n\mathsf{T}^n$ so $\mathsf{T} = \bigcup_{n\in\omega} \mathsf{T}^n$.

As outlined in §9.4, from the outset we distinguish a non-repeating sequence of constants $(\mathsf{u}_n)_{n\in\omega} \subseteq \mathsf{C}$ that we call *sorts* to denote universes of domains of polymorphic functions. We denote these sorts by

(Sorts) $\mathsf{U} = \{\mathsf{u}_n \mid n \in \omega\} \subseteq \mathsf{C}.$

For each $m, n \in \omega$, we further distinguish a unique constant $\mathsf{p}_m^n \in \mathsf{C}$ that is meant to denote the polymorphic product operator between the corresponding universes. We denote these operators by

(Operators) $\mathsf{P} = \{\mathsf{p}_m^n \mid m, n \in \omega\} \subseteq \mathsf{C}.$

Remark 10.1. While the original systems of Barendregt's λ-cube (see [HS08, §13.E]) distinguished only two sorts $*$ and $\square$, modern pure type systems often distinguish a sequence of them as done here, such as Coquand and Paulin's Calculus of Inductive Constructions (see [CP90]) and Luo's Extended Calculus of Constructions (see [Luo94]). We are just taking the next natural step and further distinguishing constants to use as polymorphic product operators on the sorts as well.

Sequences of β's at the start of a term will often be omitted as usual, i.e. for all $R_0, \dots, R_n \in \mathsf{T}$, we define the abbreviation

$$R_0 \dots R_n := \beta^n R_0 \dots R_n.$$

Define $\mathsf{F} \in \mathcal{P}(\mathsf{V})^{\mathsf{T}}$ so that, for all $a \in \mathsf{C}$, $x \in \mathsf{V}$ and $R, S \in \mathsf{T}$,

$$\mathsf{F}(a) = \emptyset.$$
$$\mathsf{F}(x) = \{x\}.$$
$$\mathsf{F}(RS) = \mathsf{F}(R) \cup \mathsf{F}(S).$$
$$\mathsf{F}(\lambda x RS) = \mathsf{F}(R) \cup (\mathsf{F}(S) \setminus \{x\}).$$

The variables in $\mathsf{F}(S)$ are said to be *free*. Abbreviations for product terms are then defined as follows

$$\pi_m^n x RS := \mathsf{p}_m^n R \lambda x RS.$$
$$(x : R) \to_m^n S := \pi_m^n x RS.$$
$$R \to_m^n S := \pi_m^n x RS, \text{ where } x \notin \mathsf{F}(S).$$

We will even drop the indices when they can be inferred from context, e.g. writing $\mathsf{p}RG$, $\pi x RS$ and $R \to S$ for $\mathsf{p}_m^n RG$, $\pi_m^n x RS$ and $R \to_m^n S$.

For all $t \in \mathsf{T}^0$, $R, S, T \in \mathsf{T}$ and $x, y \in \mathsf{V}$, define

$$t_{[T/x]} := \begin{cases} T & \text{if } t = x \\ t & \text{if } t \in \mathsf{T}^0 \setminus \{x\}. \end{cases}$$

$$(RS)_{[T/x]} := R_{[T/x]} S_{[T/x]}.$$

$$(\lambda y RS)_{[T/x]} := \begin{cases} \lambda y R_{[T/x]} S & \text{if } y = x \text{ or } x \notin \mathsf{F}(S) \\ \lambda z R_{[T/x]} S_{[z/y][T/x]} & \text{otherwise, where } z \notin \mathsf{F}(\lambda y TS) \end{cases}$$

As in Proposition 4.1, as long as we choose z appropriately,

$$S_{[T/x]} \neq S \qquad \Leftrightarrow \qquad T \neq x \in \mathsf{F}(S).$$

11. INTERPRETATIONS

Denote the class of all sets by Set. We define *interpretations* to be functions $[\![\cdot]\!]_\psi \in \mathsf{Set}^{\mathsf{T}}$, for $\psi \subseteq \mathsf{Set} \times (\mathsf{T}^0 \setminus \mathsf{P})$ (where $[\![\cdot]\!] = [\![\cdot]\!]_\emptyset$) such that, for all $t \in \mathsf{T}^0$, $R, S \in \mathsf{T}$, $x \in \mathsf{V}$ and $m, n \in \omega$,

$$[\![t]\!]_\psi = \begin{cases} \psi(t) & \text{if } t \in \mathrm{dom}(f) \\ [\![t]\!] & \text{otherwise.} \end{cases}$$

$$[\![RS]\!] = [\![R]\!]([\![S]\!]).$$
$$[\![\lambda x RS]\!] = \{\langle [\![S]\!]_{\langle r, x \rangle}, r \rangle \mid r \in [\![R]\!]\}.$$

$$[\![\mathsf{p}_m^n]\!] = \{\langle \{\langle \prod \phi, \phi \rangle \mid \phi \in [\![\mathsf{u}_n]\!]^D\}, D \rangle \mid D \in [\![\mathsf{u}_m]\!]\}.$$

So interpretations can take arbitrary values on $\mathsf{T}^0 \setminus \mathsf{P}$ as before, but this time their values on the operators P are determined by their values on the sorts U. Specifically, for all $m, n \in \omega$, $[\![\mathsf{p}_m^n]\!]$ is the function on

$\llbracket u_m \rrbracket$ such that $\llbracket p_m^n \rrbracket(D)$ is itself a function, for each $D \in \llbracket u_m \rrbracket$, defined on each function $\phi \in \llbracket u_n \rrbracket^D$ by

$$\llbracket p_m^n \rrbracket(D)(\phi) = \prod \phi.$$

Unlike before, here we will be concerned not just with the actual interpretation of terms but whether the terms are *well-formed* with respect to an interpretation, meaning that only functions are applied to other sets and only to those lying in their domain. More precisely, let Fun denote the class of functions and, given any interpretation $\llbracket \cdot \rrbracket$, define $\mathsf{wf}_{\llbracket \cdot \rrbracket} \subseteq \mathsf{T}$ as the smallest set of terms such that $\mathsf{T}^0 \subseteq \mathsf{wf}_{\llbracket \cdot \rrbracket}$,

$$\{FS \mid F, S \in \mathsf{wf}_{\llbracket \cdot \rrbracket}, \ \llbracket F \rrbracket \in \mathsf{Fun} \text{ and } \llbracket S \rrbracket \in \mathrm{dom}\llbracket F \rrbracket\} \subseteq \mathsf{wf}_{\llbracket \cdot \rrbracket} \quad \text{and}$$

$$\{\lambda x R S \mid R \in \mathsf{wf}_{\llbracket \cdot \rrbracket} \text{ and } \forall r \in \llbracket R \rrbracket \ (S \in \mathsf{wf}_{\llbracket \cdot \rrbracket_{\langle r, x \rangle}})\} \subseteq \mathsf{wf}_{\llbracket \cdot \rrbracket}.$$

Equivalently, define a unary relation $\llbracket \cdot \rrbracket^{\mathsf{wf}}$ on terms T such that

$$\llbracket FS \rrbracket^{\mathsf{wf}} \quad \Leftrightarrow \quad \llbracket F \rrbracket^{\mathsf{wf}}, \llbracket S \rrbracket^{\mathsf{wf}}, \ \llbracket F \rrbracket \in \mathsf{Fun} \text{ and } \llbracket S \rrbracket \in \mathrm{dom}\llbracket F \rrbracket,$$

$$\llbracket \lambda x R S \rrbracket^{\mathsf{wf}} \quad \Leftrightarrow \quad \llbracket R \rrbracket^{\mathsf{wf}} \text{ and } \forall r \in \llbracket R \rrbracket \ (\llbracket S \rrbracket^{\mathsf{wf}}_{\langle r, x \rangle})$$

and $\llbracket t \rrbracket^{\mathsf{wf}}$, for all $t \in \mathsf{T}^0$. Then $\mathsf{wf}_{\llbracket \cdot \rrbracket} = \{R \mid \llbracket R \rrbracket^{\mathsf{wf}}\}$.

Proposition 11.1. *For all $S, T \in \mathsf{T}$ and $x \in \mathsf{V}$,*

$$\llbracket T \rrbracket^{\mathsf{wf}} \text{ and } \llbracket S \rrbracket^{\mathsf{wf}}_{\langle \llbracket T \rrbracket, x \rangle} \quad \Rightarrow \quad \llbracket S_{[T/x]} \rrbracket^{\mathsf{wf}}.$$

The converse also holds when $x \in \mathsf{F}(S)$.

Proof. This is certainly true for $S \in \mathsf{T}^0$ and can then be extended to all T^n by induction like in the proof of Proposition 4.2. $\square$

Given an interpretation $\llbracket \cdot \rrbracket$, we define a *typing relation* $\in_{\llbracket \cdot \rrbracket}$ on T as before except that we also require the terms to be well-formed, i.e.

$$S \in_{\llbracket \cdot \rrbracket} P \quad \Leftrightarrow \quad \llbracket S \rrbracket^{\mathsf{wf}}, \llbracket P \rrbracket^{\mathsf{wf}} \text{ and } \llbracket S \rrbracket \in \llbracket P \rrbracket.$$

Equivalently, $\in_{\llbracket \cdot \rrbracket}$ is a unary relation on *typing statements* given by

(Typing Statements) $\mathsf{S}_: = \mathsf{T} \times \{:\} \times \mathsf{T}.$

So this time typing statements are pairs of actual terms, not α-conversion classes as before, together with a formal : symbol added in the middle to distinguish them from (sub-)reduction statements below. Again a particular typing statement $\langle S, :, P \rangle$ will usually be abbreviated to $S : P$ and read as 'S is of type P', referring to S and P as the *subject* and *predicate* of the typing statement. Viewing $\in_{\llbracket \cdot \rrbracket}$ as a unary relation on typing statements, let us again denote it by $\llbracket \cdot \rrbracket$, i.e. for all $S, P \in \mathsf{T}$,

$$\llbracket S : P \rrbracket \quad \Leftrightarrow \quad S \in_{\llbracket \cdot \rrbracket} P \quad \Leftrightarrow \quad \llbracket S \rrbracket^{\mathsf{wf}}, \llbracket P \rrbracket^{\mathsf{wf}} \text{ and } \llbracket S \rrbracket \in \llbracket P \rrbracket.$$

Again if $[\![S : P]\!]$ holds, we say that the typing statement $(S : P)$ is *satisfied* by the interpretation $[\![\cdot]\!]$.

Proposition 11.2. *For any interpretation $[\![\cdot]\!]$, $x \in \mathsf{V}$ and $R, S \in \mathsf{T}$,*

$$[\![(x : R) \rightarrow_m^n S]\!]^{\mathsf{wf}} \quad \Leftrightarrow \quad [\![R : \mathsf{u}_m]\!] \text{ and } [\![S : \mathsf{u}_n]\!]_{\langle r,x \rangle}, \text{ for all } r \in [\![R]\!].$$

Proof. First note that $(x : R) \rightarrow_m^n S = \pi_m^n x R S = \mathsf{p}_m^n R(\lambda x R S)$ and

$$[\![\mathsf{p}_m^n R(\lambda x R S)]\!]^{\mathsf{wf}} \Leftrightarrow [\![\mathsf{p}_m^n R]\!]^{\mathsf{wf}}, \ [\![\lambda x R S]\!]^{\mathsf{wf}} \text{ and } [\![\lambda x R S]\!] \in \mathrm{dom}[\![\mathsf{p}_m^n R]\!].$$

But $[\![\mathsf{p}_m^n R]\!]^{\mathsf{wf}}$ means $[\![R]\!]^{\mathsf{wf}}$ and $[\![R]\!] \in \mathrm{dom}[\![\mathsf{p}_m^n]\!] = [\![\mathsf{u}_m]\!]$, i.e. $[\![R : \mathsf{u}_m]\!]$. In this case, $[\![\lambda x R S]\!]^{\mathsf{wf}}$ is equivalent to $[\![S]\!]^{\mathsf{wf}}_{\langle r,x \rangle}$, for all $r \in [\![R]\!]$. Also then $\mathrm{dom}[\![\mathsf{p}_m^n R]\!] = [\![\mathsf{u}_n]\!]^{[\![R]\!]}$ and so $[\![\lambda x R S]\!] \in \mathrm{dom}[\![\mathsf{p}_m^n R]\!]$ precisely when $[\![S]\!]_{\langle r,x \rangle} \in [\![\mathsf{u}_n]\!]$ and hence $[\![S : \mathsf{u}_n]\!]_{\langle r,x \rangle}$, for all $r \in [\![R]\!]$. $\qquad \square$

As mentioned in §9.4, we are primarily interested in interpretations satisfying extra conditions corresponding to the 'axioms' and 'rules' of pure type systems (see [Bar92, §5]), conditions like $[\![\mathsf{u}_n]\!] \in [\![\mathsf{u}_{n+1}]\!]$ and $[\![\mathsf{p}_m^n]\!](D)(\phi) \in [\![\mathsf{u}_{m \smile n}]\!]$, for all $D \in [\![\mathsf{u}_m]\!]$ and $\phi \in [\![\mathsf{u}_n]\!]^D$, where $\smile$ is some binary operation on ω. In other words, we are interested in interpretations $[\![\cdot]\!]$ satisfying typing statements like

$$(\mathsf{u}_n : \mathsf{u}_{n+1}) \qquad \text{and} \qquad (\mathsf{p}_m^n : (x : \mathsf{u}_m) \rightarrow (x \rightarrow \mathsf{u}_n) \rightarrow \mathsf{u}_{m \smile n}).$$

In keeping with idea that the sorts should denote 'universes' in which we can specify various mathematical structures (see §13 below), we will also generally want each $[\![\mathsf{u}_n]\!]$ to satisfy as much of ZFC as possible. If we are willing to accept the existence of infinitely many inaccessible cardinals then we can take each $[\![\mathsf{u}_n]\!]$ to satisfy all of ZFC, like in [Car19], for example. However, we can avoid any large cardinal assumptions and still take each $[\![\mathsf{u}_n]\!]$ to satisfy ZFC$-$P (i.e. ZFC without the power set axiom), which suffices for most specifications. This is done by taking universes to be sets of the form $\mathsf{H}(\kappa)$, the family of all sets of hereditary cardinality less than κ (see [Kun11]).

The one exception here is the bottom sort u_0 which we are taking to denote a default set of propositions, as suggested in §9.4. Accordingly, it is not so important for $[\![\mathsf{u}_0]\!]$ to satisfy any axioms of ZFC, rather we just want $[\![\mathsf{u}_0]\!]$ to include the default truth values $\emptyset$ (false) and $\{\emptyset\}$ (true), as in §6. If we want all 'proofs' of a particular proposition to be identical, then we should also require each member of $[\![\mathsf{u}_0]\!]$ to contain at most one element. On the other hand, if we want propositions to be identified with their truth values then we should make sure there is at most one non-empty member of $[\![\mathsf{u}_0]\!]$. This all motivates Definition 11.3.

First let us call a sequence (κ_n) of regular cardinals a *regular sequence* if, for all $n \in \omega$ and cardinals μ and ν,

$$\mu < \kappa_n \quad \text{and} \quad \nu < \kappa_{n+1} \qquad \Rightarrow \qquad |\nu^\mu| < \kappa_{n+1}.$$

For example, we could take $\kappa_0 = \aleph_0$ and just define $\kappa_{n+1} = |\kappa_n^{\kappa_n}|^+$, for all $n \in \omega$ (here $|S|$ denotes the cardinality of any set S and μ^+ denotes the successor of any cardinal μ). If the generalised continuum hypothesis holds, then we could also just take $\kappa_n = \aleph_n$, for all $n \in \omega$. Or if there exists infinitely many inaccessible cardinals then we could simply let (κ_n) be any strictly increasing sequence of them.

Definition 11.3. We call an interpretation $[\![\cdot]\!]$ *canonical* if we have a regular sequence (κ_n) such that, for all $n \in \omega$,

$$\{\emptyset, \{\emptyset\}\} \subseteq [\![u_0]\!] \subseteq \mathsf{H}(\kappa_0) \qquad \text{and} \qquad [\![u_n]\!] = \mathsf{H}(\kappa_n), \text{ for all } n \geq 1.$$

We also call an interpretation $[\![\cdot]\!]$
  (1) *proof-irrelevant* if $|s| \leq 1$, for all $s \in [\![u_0]\!]$.
  (2) *propositionally extensional* if $|[\![u_0]\!] \setminus \{\emptyset\}| \leq 1$.

In particular, a canonical interpretation $[\![\cdot]\!]$ is both proof-irrelevant and propositionally extensional precisely when $[\![u_0]\!] = \{\emptyset, \{\emptyset\}\}$.

The first basic properties of canonical interpretations are as follows.

Proposition 11.4. *If $[\![\cdot]\!]$ is a canonical interpretation and $m, n \in \omega$,*

$$m < n \qquad \Leftrightarrow \qquad [\![u_m : u_n]\!].$$

Moreover, $[\![u_n]\!]$ is a model of $\mathrm{ZFC} - \mathrm{P}$, *for all $n \geq 1$.*

Proof. First note $\mathsf{H}(\kappa)$ is a model of $\mathrm{ZFC}-\mathrm{P}$ for all uncountable regular κ, by [Kun11, II.2.1], which immediately yields the last statement.

If $m < n$ then certainly $[\![u_m]\!] \subseteq \mathsf{H}(\kappa_m) \subseteq \mathsf{H}(\kappa_n)$. By [Kun11, I.13.28], $|\mathsf{H}(\kappa_m)| \leq \sup_{\mu,\nu<\kappa_m} |\nu^\mu| < \kappa_{m+1} \leq \kappa_n$ and hence $[\![u_m]\!] \in \mathsf{H}(\kappa_n) = [\![u_n]\!]$, by [Kun11, I.13.32], i.e. $[\![u_m : u_n]\!]$. Conversely, if $[\![u_m]\!] \in [\![u_n]\!]$ then $m \neq n$, as sets are $\in$-well-founded, but also we can not have $m > n$ because that would imply $[\![u_m]\!] = \mathsf{H}(\kappa_m) \supseteq \mathsf{H}(\kappa_n) \supseteq [\![u_n]\!] \ni [\![u_m]\!]$, again contradicting the fact sets are $\in$-well-founded. $\qquad\square$

Canonicity also yields the following version of Proposition 11.2. Here $m \vee n$ denotes the maximum of m and n.

Proposition 11.5. *If $[\![\cdot]\!]$ is canonical, $m, n \in \omega$, $x \in \mathsf{V}$ and $R, S \in \mathsf{T}$,*

$$[\![(x : R) \to_m^n S : u_{(m+1)\vee n}]\!] \;\Leftrightarrow\; [\![R : u_m]\!] \text{ and } [\![S : u_n]\!]_{\langle r,x\rangle}, \text{ for } r \in [\![R]\!].$$

Proof. By Proposition 11.2, it suffices to show that the right side above implies $[\![(x : R) \to_m^n S]\!] \in [\![u_{(m+1)\vee n}]\!]$. But the right side implies $[\![R]\!] \in [\![u_m]\!] \subseteq \mathsf{H}(\kappa_{(m+1)\vee n})$ and $[\![S]\!]_{\langle r,x\rangle} \in [\![u_n]\!] \subseteq \mathsf{H}(\kappa_{(m+1)\vee n})$ and

hence $[\![S]\!]_{\langle r,x \rangle} \subseteq \mathsf{H}(\kappa_{(m+1)\vee n})$, for all $r \in [\![R]\!]$. Then by [Kun11, Lemma I.13.32], $\prod_{r \in [\![R]\!]} [\![S]\!]_{\langle r,x \rangle} \subseteq \mathsf{H}(\kappa_{(m+1)\vee n})$ and it suffices to show $|\prod_{r \in [\![R]\!]} [\![S]\!]_{\langle r,x \rangle}| < \kappa_{(m+1)\vee n}$. This follows because, for any function ϕ with $|D| < \kappa_m$ and $|\phi(d)| < \kappa_n$, for all $d \in D := \mathrm{dom}(\phi)$,

$$(11.1) \qquad \left| \prod \phi \right| \leq |(\sup_{d \in D} |\phi(d)|)^{|D|}| < \kappa_{(m+1)\vee n}.$$

Indeed, if $m > n$ then $\sup_{d \in D} |\phi(d)| \leq \kappa_n < \kappa_{m \vee n}$, while if $m \leq n$ then $\sup_{d \in D} |\phi(d)| < \kappa_n \leq \kappa_{m \vee n}$, by the regularity of κ_n. So in either case $\sup_{d \in D} |\phi(d)| < \kappa_{m \vee n} \leq \kappa_{(m+1)\vee n}$ and $|D| < \kappa_m \leq \kappa_{((m+1)\vee n)-1}$ and hence $|(\sup_{d \in D} |\phi(d)|)^{|D|}| < \kappa_{(m+1)\vee n}$, as (κ_j) is a regular sequence. $\square$

In particular, if $[\![\cdot]\!]$ is canonical then, for all $m, n \in \omega$ and $R, S \in \mathsf{T}$,

$$[\![R : \mathsf{u}_m]\!] \quad \text{and} \quad [\![S : \mathsf{u}_n]\!] \qquad \Leftrightarrow \qquad [\![R \to_m^n S : \mathsf{u}_{(m+1)\vee n}]\!].$$

We can now type the product operators (p_m^n) as follows.

Proposition 11.6. *If $[\![\cdot]\!]$ is a canonical interpretation and $m, n \in \omega$,*

$$[\![\mathsf{p}_m^n : (x : \mathsf{u}_m) \to (x \to \mathsf{u}_n) \to \mathsf{u}_{(m+1)\vee n}]\!],$$

specifically $[\![\mathsf{p}_m^n : (x : \mathsf{u}_m) \to_{m+1}^{(m\vee n)+2} (x \to_m^{n+1} \mathsf{u}_n) \to_{(m\vee n)+1}^{((m+1)\vee n)+1} \mathsf{u}_{(m+1)\vee n}]\!]$.

Proof. For starters, we should show that

$$[\![(x : \mathsf{u}_m) \to_{m+1}^{(m\vee n)+2} (x \to_m^{n+1} \mathsf{u}_n) \to_{(m\vee n)+1}^{((m+1)\vee n)+1} \mathsf{u}_{(m+1)\vee n}]\!]^{\mathsf{wf}}.$$

By Proposition 11.4, $[\![\mathsf{u}_m : \mathsf{u}_{m+1}]\!]$ so, by Proposition 11.2, we just have to show that, for all $r \in [\![\mathsf{u}_m]\!]$,

$$[\![(x \to_m^{n+1} \mathsf{u}_n) \to_{(m\vee n)+1}^{((m+1)\vee n)+1} \mathsf{u}_{(m+1)\vee n} : \mathsf{u}_{(m\vee n)+2}]\!]_{\langle r,x \rangle}.$$

Noting that $((m \vee n) + 1 + 1) \vee (((m+1) \vee n) + 1) = (m \vee n) + 2$ and $[\![\mathsf{u}_{(m+1)\vee n} : \mathsf{u}_{((m+1)\vee n)+1}]\!]$, again by Proposition 11.4, we must just show

$$[\![(x \to_m^{n+1} \mathsf{u}_n) : \mathsf{u}_{(m\vee n)+1}]\!]_{\langle r,x \rangle},$$

again by Proposition 11.5. Noting that $(m+1) \vee (n+1) = (m \vee n) + 1$ and $[\![x]\!]_{\langle r,x \rangle} = r \in [\![\mathsf{u}_m]\!]$ so $[\![x : \mathsf{u}_m]\!]_{\langle r,x \rangle}$, this amounts to showing that $[\![\mathsf{u}_n : \mathsf{u}_{n+1}]\!]$, which again follows from Proposition 11.4.

Now it only remains to show that

$$[\![\mathsf{p}_m^n]\!] \in [\![(x : \mathsf{u}_m) \to (x \to \mathsf{u}_n) \to \mathsf{u}_{(m+1)\vee n}]\!]$$

$$= \prod_{r \in [\![\mathsf{u}_m]\!]} [\![(x \to \mathsf{u}_n) \to \mathsf{u}_{(m+1)\vee n}]\!]_{\langle r,x \rangle}.$$

As $[\![\cdot]\!]$ is an interpretation, $\mathrm{dom}[\![\mathsf{p}_m^n]\!] = [\![\mathsf{u}_m]\!]$ so we just have to show

$$[\![\mathsf{p}_m^n]\!](r) \in [\![(x \to \mathsf{u}_n) \to \mathsf{u}_{(m+1)\vee n}]\!]_{\langle r,x \rangle} = [\![\mathsf{u}_{(m+1)\vee n}]\!]^{[\![\mathsf{u}_n]\!]^r},$$

for all $r \in [\![\mathsf{u}_m]\!]$. But again we already know $\operatorname{dom}([\![\mathsf{p}_m^n]\!](r)) = [\![\mathsf{u}_n]\!]^r$ because $[\![\cdot]\!]$ is an interpretation, so we just need to show that

$$\prod \phi = [\![\mathsf{p}_m^n]\!](r)(\phi) \in [\![\mathsf{u}_{(m+1)\vee n}]\!],$$

for all $\phi \in [\![\mathsf{u}_n]\!]^r$. As $r \in [\![\mathsf{u}_m]\!] = \mathsf{H}(\kappa_m)$ and $[\![\mathsf{u}_n]\!] = \mathsf{H}(\kappa_n)$, we know that $\phi \subseteq \mathsf{H}(\kappa_{m\vee n}) \subseteq \mathsf{H}(\kappa_{(m+1)\vee n})$ and hence $\prod \phi \subseteq \mathsf{H}(\kappa_{(m+1)\vee n})$. Also $|\prod \phi| < \kappa_{(m+1)\vee n}$, by (11.1), so $\prod \phi \in \mathsf{H}(\kappa_{(m+1)\vee n}) = [\![\mathsf{u}_{(m+1)\vee n}]\!]$, thus completing the proof. $\qquad\square$

In particular, if $[\![\cdot]\!]$ is canonical and $m \in \omega$ then

$$[\![\mathsf{p}_m^0 : (x : \mathsf{u}_m) \to (x \to \mathsf{u}_0) \to \mathsf{u}_{m+1}]\!].$$

So our canonical interpretations are never 'impredicative', i.e. we can not replace $m + 1$ here with 0, despite how tempting this might be. Indeed, if $[\![\cdot]\!]$ is proof-irrelevant then $|\prod \phi| \leq 1$ when $\operatorname{ran}(\phi) \subseteq [\![\mathsf{u}_0]\!]$ and so one might try and identify all singleton sets to enforce impredicativity. However, this would necessarily mean forgetting the domain of each such ϕ, which in turn would mean we could no longer tell which terms are well-formed with respect to $[\![\cdot]\!]$ (see [MW03] for further discussion on this point). In any case, as mentioned in Remark 6.1 and the comment after, we can easily simulate impredicativity in other ways, e.g. by specifying universal quantification for $[\![\mathsf{u}_0]\!]$ separately from π, as discussed below in §13.

12. REDUCTION

Every interpretation $[\![\cdot]\!]$ yields binary *reduction* and *sub-reduction* relations $\blacktriangleright_{[\![\cdot]\!]}$ and $\triangleright_{[\![\cdot]\!]}$ on T defined by

$$R \blacktriangleright_{[\![\cdot]\!]} C \quad\Leftrightarrow\quad \forall \psi \in \mathsf{Set}^\mathsf{V}\, ([\![R]\!]_\psi^\mathsf{wf} \Rightarrow ([\![C]\!]_\psi^\mathsf{wf} \text{ and } [\![R]\!]_\psi = [\![C]\!]_\psi)).$$

$$R \triangleright_{[\![\cdot]\!]} C \quad\Leftrightarrow\quad \forall \psi \in \mathsf{Set}^\mathsf{V}\, ([\![R]\!]_\psi^\mathsf{wf} \Rightarrow ([\![C]\!]_\psi^\mathsf{wf} \text{ and } [\![R]\!]_\psi \subseteq [\![C]\!]_\psi)).$$

Equivalently, we have unary relations on (*sub-*)*reduction statements*

(Reduction Statements) $\mathsf{S}_\blacktriangleright = \mathsf{T} \times \{\blacktriangleright\} \times \mathsf{T}.$

(Sub-Reduction Statements) $\mathsf{S}_\triangleright = \mathsf{T} \times \{\triangleright\} \times \mathsf{T}.$

Particular (sub-)reduction statements $\langle R, \blacktriangleright, C \rangle$ and $\langle R, \triangleright, C \rangle$ will usually be abbreviated to $R \blacktriangleright C$ and $R \triangleright C$ and read as 'R (sub-)reduces to C', referring to R and C as the *redex* and *contractum* of the (sub-)reduction statement. Viewing $\blacktriangleright_{[\![\cdot]\!]}$ and $\triangleright_{[\![\cdot]\!]}$ as a unary relations on (sub-)reduction statements, we again denote them by $[\![\cdot]\!]$, i.e.

$$[\![R \blacktriangleright C]\!] \quad\Leftrightarrow\quad \forall \psi \in \mathsf{Set}^\mathsf{V}\, ([\![R]\!]_\psi^\mathsf{wf} \Rightarrow ([\![C]\!]_\psi^\mathsf{wf} \text{ and } [\![R]\!]_\psi = [\![C]\!]_\psi)).$$

$$[\![R \triangleright C]\!] \quad\Leftrightarrow\quad \forall \psi \in \mathsf{Set}^\mathsf{V}\, ([\![R]\!]_\psi^\mathsf{wf} \Rightarrow ([\![C]\!]_\psi^\mathsf{wf} \text{ and } [\![R]\!]_\psi \subseteq [\![C]\!]_\psi)).$$

So $[\![R \blacktriangleright C]\!]$ and $[\![R \triangleright C]\!]$ mean that, no matter how we change the interpretation on the variables, if R is well-formed with respect to the interpretation then so is C and, in the case of $\blacktriangleright$, their interpretations then agree or, in the case of $\triangleright$, the interpretation of R is then contained in that of C. In particular, $[\![R \blacktriangleright C]\!]$ and $[\![R \triangleright C]\!]$ depend only on the values of $[\![\cdot]\!]$ on the constants C.

For some trivial examples note, for any canonical interpretation $[\![\cdot]\!]$,

$$m = n \qquad \Leftrightarrow \qquad [\![\mathsf{u}_m \blacktriangleright \mathsf{u}_n]\!] \quad \text{and}$$

$$m \leq n \qquad \Leftrightarrow \qquad [\![\mathsf{u}_m \triangleright \mathsf{u}_n]\!].$$

Another immediate example of reduction comes from α-conversion.

Proposition 12.1. *For any interpretation* $[\![\cdot]\!]$, $R, S \in \mathsf{T}$ *and* $x, y \in \mathsf{V}$,

$$y \notin \mathsf{F}(S) \qquad \Rightarrow \qquad [\![\lambda x R S \blacktriangleright \lambda y R S_{[y/x]}]\!].$$

Proof. If $y \notin \mathsf{F}(S)$ then Proposition 11.1 yields

$$[\![S]\!]^{\mathsf{wf}}_{\langle r, x \rangle} \quad \Rightarrow \quad [\![S]\!]^{\mathsf{wf}}_{\langle r, y \rangle \langle [\![y]\!]_{\langle r, y \rangle}, x \rangle} \quad \Rightarrow \quad [\![S_{[y/x]}]\!]^{\mathsf{wf}}_{\langle r, y \rangle}.$$

So if $[\![\lambda x R S]\!]^{\mathsf{wf}}$ then $[\![S]\!]^{\mathsf{wf}}_{\langle r, x \rangle}$ and hence $[\![S_{[y/x]}]\!]^{\mathsf{wf}}_{\langle r, y \rangle}$, for all $r \in [\![R]\!]$, i.e. $[\![\lambda y R S_{[y/x]}]\!]^{\mathsf{wf}}$. Also $[\![\lambda x R S]\!] = [\![\lambda y R S_{[y/x]}]\!]$, as in Proposition 5.1. As $[\![\cdot]\!]$ was arbitrary, the same applies to $[\![\cdot]\!]_\phi$, for any $f \in \mathsf{Set}^\mathsf{V}$. $\qquad\square$

Yet another example comes from β-reduction.

Proposition 12.2. *For any interpretation* $[\![\cdot]\!]$, $R, S \in \mathsf{T}$ *and* $x \in \mathsf{V}$,

$$[\![\beta \lambda x R S T \blacktriangleright S_{[T/x]}]\!].$$

Proof. We need to show $[\![\beta \lambda x R S T]\!]^{\mathsf{wf}}$ always implies $[\![S_{[T/x]}]\!]^{\mathsf{wf}}$ and $[\![\beta \lambda x R S T]\!] = [\![S_{[T/x]}]\!]$. Now $[\![\beta \lambda x R S T]\!]^{\mathsf{wf}}$ implies $[\![T]\!]^{\mathsf{wf}}$, $[\![\lambda x R S]\!]^{\mathsf{wf}}$ and $[\![T]\!] \in \mathrm{dom}[\![\lambda x R S]\!] = [\![R]\!]$. Thus $[\![S]\!]^{\mathsf{wf}}_{\langle r, x \rangle}$, for all $r \in [\![R]\!]$, so $[\![S]\!]^{\mathsf{wf}}_{\langle [\![T]\!], x \rangle}$ and hence $[\![S_{[T/x]}]\!]^{\mathsf{wf}}$, by Proposition 11.1. Then (Substitutivity) yields

$$[\![\beta \lambda x R S T]\!] = [\![\lambda x R S]\!]([\![T]\!]) = [\![S]\!]_{\langle [\![T]\!], x \rangle} = [\![S_{[T/x]}]\!]. \qquad \square$$

The other standard reduction notion is η-reduction. However, this is really an example of sub-reduction.

Proposition 12.3. *For any interpretation* $[\![\cdot]\!]$, $R, F \in \mathsf{T}$ *and* $x \in \mathsf{V}$,

$$x \notin \mathsf{F}(F) \qquad \Rightarrow \qquad [\![\lambda x R (F x) \triangleright F]\!].$$

Proof. Assume $[\![\lambda x R (F x)]\!]^{\mathsf{wf}}$ so $[\![F x]\!]^{\mathsf{wf}}_{\langle s, x \rangle}$, for all $s \in [\![R]\!]$. As $x \notin \mathsf{F}(F)$, it follows that $[\![F]\!] = [\![F]\!]_{\langle s, x \rangle} \in \mathsf{Fun}$ and $s \in \mathrm{dom}[\![F]\!]$, for all $s \in [\![R]\!]$,

i.e. $[\![R]\!] \subseteq \mathrm{dom}[\![F]\!]$. Thus

$$
\begin{aligned}
[\![\lambda x R(Fx)]\!] &= \{\langle [\![Fx]\!]_{\langle s,x\rangle}, s\rangle : s \in [\![R]\!]\} \\
&\subseteq \{\langle [\![F]\!](s), s\rangle : s \in \mathrm{dom}[\![F]\!]\} \\
&= [\![F]\!].
\end{aligned}
$$

As the interpretation was arbitrary, the same argument is valid for $[\![\cdot]\!]_\psi$, for any $\psi \in \mathsf{Set}^\mathsf{V}$, and so this shows that $[\![\lambda x R(Fx) \triangleright F]\!]$. $\square$

Denote the set of reduction, sub-reduction and typing statements by

(Statements) $\mathsf{S} = \mathsf{S}_{:} \cup \mathsf{S}_{\triangleright} \cup \mathsf{S}_{\blacktriangleright}.$

Extending the terminology above, if $X \in \mathsf{S}$ and $[\![X]\!]$ holds then we say the statement X is *satisfied* by the interpretation $[\![\cdot]\!]$. Likewise, if $[\![\cdot]\!]$ satisfies every statement in some given set $\Gamma \subseteq \mathsf{S}$ then we say $[\![\cdot]\!]$ is a *model* for Γ or again that $[\![\cdot]\!]$ *satisfies* Γ. Thus the results above are saying that every interpretation satisfies the *head α/β-reduction* and *head η-sub-reduction* statements defined by

$\blacktriangleright^\alpha = \{(\lambda x RS \blacktriangleright \lambda y RS_{[y/x]}) \mid R, S \in \mathsf{T}, x \in \mathsf{V} \setminus \mathsf{B}(S) \text{ and } y \in \mathsf{V} \setminus \mathsf{V}(S)\}.$

$\blacktriangleright^\beta = \{(\lambda x RS)T \blacktriangleright S_{[T/x]}) \mid R, S, T \in \mathsf{T} \text{ and } x \in \mathsf{V}\}.$

$\triangleright^\eta = \{(\lambda x R(Fx) \triangleright F) \mid R, F \in \mathsf{T} \text{ and } x \in \mathsf{V}\}.$

Like before, define the *contextual closure* of any $\Gamma \subseteq \mathsf{T} \times \mathsf{Set} \times \mathsf{T}$ by

$$\Gamma^\mathsf{c} = \{\langle PRQ, \bullet, PCQ\rangle : PRQ, PCQ \in \mathsf{T}, \langle R, \bullet, C\rangle \in \Gamma \text{ and } Q(0) \neq {}'\}.$$

Also define the transitive and reflexive closure of Γ as before just leaving the extra symbol in the middle fixed. We immediately see that any model for some $\Gamma \subseteq \mathsf{S}_\blacktriangleright$ is also a model for its reflexive transitive contextual closure Γ^ctr. In particular, any interpretation satisfies the general α/β-*reduction* statements defined by

$$\blacktriangleright_\alpha = \blacktriangleright^{\alpha\mathsf{ctr}} \qquad \text{and} \qquad \blacktriangleright_\beta = \blacktriangleright^{\beta\mathsf{ctr}}.$$

However, a model for some $\Gamma \subseteq \mathsf{S}_\triangleright$ is also only a model for its reflexive transitive closure Γ^rt, not its contextual closure Γ^c (as inclusions of sets need not be preserved by function application, for example). In particular, interpretations may not satisfy $\triangleright^{\eta\mathsf{crt}}$, which is how one would usually define general η-sub-reduction statements, rather they only satisfy $\triangleright^{\eta\mathsf{rt}}$.

13. SPECIFICATIONS

Here we look at how statements can be used to specify certain mathematical structures. So at the very least, any specification $\Gamma \subseteq \mathsf{S}$ should be *satisfiable*, meaning it should have at least one model, preferably a canonical one. But any canonical model for Γ should also be isomorphic in an appropriate sense to the structure we want to specify.

The simplest example to start with is the specification for 'false', i.e. the empty set $\emptyset$. First take $\bot \in \underline{\mathsf{C}} := \mathsf{C} \setminus (\mathsf{U} \cup \mathsf{P})$ and a canonical interpretation $[\![\cdot]\!]$ with $[\![\bot]\!] = \emptyset$. Certainly $[\![\bot : \mathsf{u}_0]\!]$, by the canonicity of $[\![\cdot]\!]$, but $[\![\mathsf{u}_0]\!]$ also contains non-empty sets so this is definitely not a complete specification. What distinguishes $\emptyset$ is that we have a map from $\emptyset$ to $\prod_{r \in [\![u_0]\!]} r = \emptyset$, namely the empty map $\emptyset$ again. Changing the interpretation at another constant $\natural \in \underline{\mathsf{C}}$ so that $[\![\natural]\!] = \emptyset$ too, this means $[\![\natural : \bot \to_0^2 (x : \mathsf{u}_0) \to_1^0 x]\!]$ or, leaving the indices implicit,

$$[\![\natural : \bot \to (x : \mathsf{u}_0) \to x]\!].$$

The completeness of this specification can now be stated as follows.

Proposition 13.1. *For any canonical* $[\![\cdot]\!]$, $\bot, \natural \in \underline{\mathsf{C}}$ *and* $x \in \mathsf{V}$,

$$[\![\bot : \mathsf{u}_0]\!] \quad and \quad [\![\natural : \bot \to (x : \mathsf{u}_0) \to x]\!] \quad \Rightarrow \quad [\![\bot]\!] = \emptyset.$$

Proof. Just note that $[\![(x : \mathsf{u}_0) \to x]\!] = \prod_{r \in [\![u_0]\!]} r = \emptyset$ because $\emptyset \in [\![\mathsf{u}_0]\!]$, as $[\![\cdot]\!]$ is canonical. Thus $[\![\bot \to (x : \mathsf{u}_0) \to x]\!] = \emptyset^{[\![\bot]\!]} = \emptyset$ unless $[\![\bot]\!] = \emptyset$. If $[\![\natural]\!] \in \emptyset^{[\![\bot]\!]}$ then this is the only possibility. $\square$

Note we do not even really need the assumption $[\![\bot : \mathsf{u}_0]\!]$ above, we are just adding it for clarity and so the implicit indices on the arrows in the statement $(\natural : \bot \to (x : \mathsf{u}_0) \to x)$ can be inferred.

Another good example is equality. This time take $a, \mathsf{eq}_a \in \underline{\mathsf{C}}$ and a canonical interpretation $[\![\cdot]\!]$ such that, for all $r, s \in [\![a]\!]$,

$$[\![\mathsf{eq}_a]\!](r)(s) \neq \emptyset \quad \Leftrightarrow \quad r = s.$$

Thus we can change $[\![\cdot]\!]$ at another constant $\mathsf{rfl}_a \in \underline{\mathsf{C}}$ so $[\![\mathsf{rfl}_a]\!]$ takes each $r \in [\![a]\!]$ to an element of $[\![\mathsf{eq}_a]\!](r)(r)$, i.e. $[\![\mathsf{rfl}_a]\!] \in \prod_{r \in [\![a]\!]} [\![\mathsf{eq}_a]\!](r)(r)$.

So then $[\![\cdot]\!]$ is a canonical model for the typing statements $(a : \mathsf{u}_n)$, $(\mathsf{eq}_a : a \to a \to \mathsf{u}_0)$ and $(\mathsf{rfl}_a : (x : a) \to \mathsf{eq}_a xx)$. However, there are other models $(\!|\cdot|\!)$ of these statements where $(\!|\mathsf{eq}_a|\!)(r)(s)$ is non-empty not just when $r = s$ but for some distinct $r, s \in [\![a]\!]$ as well. To complete the specification, we can use the substitution property of equality, specifically the fact that, for any $\phi : [\![a]\!] \to [\![\mathsf{u}_0]\!]$, if $r = s$ and $\phi(r) \neq \emptyset$ then $\phi(s) \neq \emptyset$ too. So we can change $[\![\cdot]\!]$ at another constant

$\mathsf{sub}_a \in \underline{\mathsf{C}}$ so that $[\![\mathsf{sub}_a]\!] \in \prod_{r \in [\![a]\!]} \prod_{s \in [\![a]\!]} (\prod_{\phi \in [\![\mathsf{u}_0]\!]^{[\![a]\!]}} \phi(s)^{\phi(r)})^{[\![\mathsf{eq}_a]\!](r)(s)}$ and thus $[\![\cdot]\!]$ is a canonical model for the typing statement

$$(\mathsf{sub}_a : (x : a) \to (y : a) \to \mathsf{eq}_a xy \to (p : a \to \mathsf{u}_0) \to px \to py).$$

This completes the specification, as shown by the following.

Proposition 13.2. *Given $n \in \omega$, $x, y, p \in \mathsf{V}$ and $a, \mathsf{eq}_a, \mathsf{rfl}_a \in \underline{\mathsf{C}}$, if*

$$(a : \mathsf{u}_n), (\mathsf{eq}_a : a \to a \to \mathsf{u}_0), (\mathsf{rfl}_a : (x : a) \to \mathsf{eq}_a xx) \quad and$$
$$(\mathsf{sub}_a : (x : a) \to (y : a) \to \mathsf{eq}_a xy \to (p : a \to \mathsf{u}_0) \to px \to py)$$

have a canonical model $[\![\cdot]\!]$ then, for all $r, s \in [\![a]\!]$,

$$[\![\mathsf{eq}_a]\!](r)(s) \neq \emptyset \qquad \Leftrightarrow \qquad r = s.$$

Proof. Note $[\![\mathsf{rfl}_a : (x : a) \to \mathsf{eq}_a xx]\!]$ means $[\![\mathsf{rfl}_a]\!] \in \prod_{r \in [\![a]\!]} [\![\mathsf{eq}_a]\!](r)(r)$ and hence $[\![\mathsf{eq}_a]\!](r)(r) \neq \emptyset$, for all $r \in [\![a]\!]$.

Conversely, say we had distinct $r, s \in [\![a]\!]$ and $t \in [\![\mathsf{eq}_a]\!](r)(s)$. Take $\phi \in [\![\mathsf{u}_0]\!]^{[\![a]\!]}$ with $\phi(r) \neq \emptyset = \phi(s)$, so we have some $u \in \phi(r)$. Then $[\![\mathsf{sub}_a : (x : a) \to (y : a) \to \mathsf{eq}_a xy \to (p : a \to \mathsf{u}_0) \to px \to py]\!]$ implies

$$[\![\mathsf{sub}_a]\!](r)(s)(t)(\phi)(u) \in \phi(s) = \emptyset,$$

a contradiction, completing the proof. $\qquad\qquad\qquad\qquad\qquad\square$

As noted in the previous section, $[\![R \blacktriangleright S]\!]$ always implies $[\![(R \blacktriangleright S)^{\mathsf{c}}]\!]$ and hence $[\![(R \triangleright S)^{\mathsf{c}}]\!]$. When $[\![\cdot]\!]$ satisfies the above specification, we can also reverse this and derive equality from the contextual closure of a sub-reduction statement.

Proposition 13.3. *If $[\![\cdot]\!]$ satisfies the typing statements in Proposition 13.2 then, for all $R, S \in \mathsf{T}$,*

$$[\![R : a]\!] \quad and \quad [\![(R \triangleright S)^{\mathsf{c}}]\!] \qquad \Rightarrow \qquad [\![R]\!] = [\![S]\!].$$

Proof. If $[\![R : a]\!]$ then $[\![\mathsf{eq}_a RR]\!]^{\mathsf{wf}}$ so $[\![(R \triangleright S)^{\mathsf{c}}]\!]$ implies

$$\emptyset \neq [\![\mathsf{eq}_a RR]\!] \subseteq [\![\mathsf{eq}_a RS]\!]$$

and hence $[\![R]\!] = [\![S]\!]$. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$

The above specifications were for elements of/functions to our default set of propositions $[\![\mathsf{u}_0]\!]$ where we only care about their truth values, i.e. whether the sets in $[\![\mathsf{u}_0]\!]$ are empty or not. For specifications in higher universes, we care more about the precise sets and functions we are specifying. This is where we can use (sub-)reduction statements.

To take a basic example, let us consider the usual binary Cartesian product. First take $a, b \in \underline{\mathsf{C}}$ and $n \geq 1$ and let $[\![\cdot]\!]$ be a canonical model

of $(a : \mathsf{u}_n)$ and $(b : \mathsf{u}_n)$. We can then change the interpretation at another constant $\mathsf{pr}_{a,b} \in \underline{\mathsf{C}}$ to ensure that

$$\llbracket \mathsf{pr}_{a,b} \rrbracket = \llbracket a \rrbracket \times \llbracket b \rrbracket$$

and hence $\llbracket \mathsf{pr}_{a,b} \rrbracket \in \llbracket \mathsf{u}_n \rrbracket$, by the canonicity of $\llbracket \cdot \rrbracket$. To reflect the fact elements of $\llbracket a \rrbracket \times \llbracket b \rrbracket$ are made from those of $\llbracket a \rrbracket$ and $\llbracket b \rrbracket$, we can change the interpretation at another constant $\mathsf{mk}_{a,b} \in \underline{\mathsf{C}}$ to ensure that $\llbracket \mathsf{mk}_{a,b} \rrbracket \in (\llbracket \mathsf{pr}_{a,b} \rrbracket^{\llbracket b \rrbracket})^{\llbracket a \rrbracket}$ and, for all $r \in \llbracket a \rrbracket$ and $s \in \llbracket b \rrbracket$,

$$\llbracket \mathsf{mk}_{a,b} \rrbracket(r)(s) = \langle r, s \rangle.$$

This means $\llbracket \mathsf{mk}_{a,b} : a \to b \to \mathsf{pr}_{a,b} \rrbracket$, but of course we could change the interpretation at $\mathsf{mk}_{a,b}$ in many other ways and still satisfy this typing statement, as there are many other functions in $(\llbracket \mathsf{pr}_{a,b} \rrbracket^{\llbracket b \rrbracket})^{\llbracket a \rrbracket}$.

As a first step to completing the specification, we can use the key property of $\llbracket a \rrbracket \times \llbracket b \rrbracket$ that, for any function $\theta \in (\mathsf{Set}^{\llbracket b \rrbracket})^{\llbracket a \rrbracket}$, we have a (unique) function $\theta^\times \in \mathsf{Set}^{\llbracket a \rrbracket \times \llbracket b \rrbracket}$ such that $\theta^\times(r, s) = \theta(r)(s)$, for all $r \in \llbracket a \rrbracket$ and $s \in \llbracket b \rrbracket$. We can thus change the interpretation at another constant $\mathsf{rec}_{a,b} \in \underline{\mathsf{C}}$ to ensure that $\llbracket \mathsf{rec}_{a,b} \rrbracket$ is a function such that, for all $\phi \in \llbracket \mathsf{u}_n \rrbracket^{\llbracket \mathsf{pr}_{a,b} \rrbracket}$ and $\theta \in \prod_{r \in \llbracket a \rrbracket} \prod_{s \in \llbracket b \rrbracket} \phi(r, s)$,

$$\llbracket \mathsf{rec}_{a,b} \rrbracket(\phi)(\theta) = \theta^\times.$$

As long as we also make $\llbracket \mathsf{rec}_{a,b} \rrbracket$ undefined elsewhere, this means

$$\llbracket \mathsf{rec}_{a,b} \rrbracket \in \prod_{\phi \in \llbracket \mathsf{u}_n \rrbracket^{\llbracket \mathsf{pr}_{a,b} \rrbracket}} \left(\prod_{t \in \llbracket \mathsf{pr}_{a,b} \rrbracket} \phi(t) \right)^{\prod_{r \in \llbracket a \rrbracket} \prod_{s \in \llbracket b \rrbracket} \phi(r,s)}.$$

In particular, $\llbracket \cdot \rrbracket$ also satisfies the typing statement

$$\mathsf{rec}_{a,b} : (f : \mathsf{pr}_{a,b} \to \mathsf{u}_n) \to ((x : a) \to (y : b) \to f(\mathsf{mk}_{a,b}xy)) \to (z : \mathsf{pr}_{a,b}) \to fz.$$

But again there are plenty of other functions we could assign to $\mathsf{rec}_{a,b}$ that still satisfy this statement so our specification is still incomplete.

To really complete it note that, for any $\psi \in \mathsf{Set}^\mathsf{V}$ and $f, g, x, y \in \mathsf{V}$,

$$\llbracket \mathsf{rec}_{a,b}fg(\mathsf{mk}_{a,b}xy) \rrbracket^{\mathsf{wf}}_\psi \quad \Rightarrow \quad \llbracket \mathsf{rec}_{a,b}fg(\mathsf{mk}_{a,b}xy) \rrbracket_\psi = \llbracket g \rrbracket^\times_\psi(\llbracket x \rrbracket_\psi, \llbracket y \rrbracket_\psi)$$
$$= \llbracket g \rrbracket_\psi(\llbracket x \rrbracket_\psi)(\llbracket y \rrbracket_\psi)$$
$$= \llbracket gxy \rrbracket_\psi.$$

In other words, $\llbracket \cdot \rrbracket$ satisfies the reduction statement

$$\mathsf{rec}_{a,b}fg(\mathsf{mk}_{a,b}xy) \blacktriangleright gxy.$$

Now we can show that these statements completely specify the product in the sense that any canonical interpretation $\llbracket \cdot \rrbracket$ satisfying them gives us a natural bijection from $\llbracket \mathsf{pr}_{a,b} \rrbracket$ onto $\llbracket a \rrbracket \times \llbracket b \rrbracket$.

Proposition 13.4. *Given $n \geq 1$, $f, g, x, y, z \in \mathsf{V}$ as well as constants $a, b, \mathsf{pr}_{a,b}, \mathsf{mk}_{a,b}, \mathsf{rec}_{a,b} \in \underline{\mathsf{C}}$, if $[\![\cdot]\!]$ is a canonical model of*

$$(a, b, \mathsf{pr}_{a,b} : \mathsf{u}_n), (\mathsf{mk}_{a,b} : a \to b \to \mathsf{pr}_{a,b}), (\mathsf{rec}_{a,b} f g (\mathsf{mk}_{a,b} xy) \blacktriangleright gxy) \quad and$$

$$(\mathsf{rec}_{a,b} : (f : \mathsf{pr}_{a,b} \to \mathsf{u}_n) \to ((x : a) \to (y : b) \to f(\mathsf{mk}_{a,b} xy)) \to (z : \mathsf{pr}_{a,b}) \to fz)$$

then $\langle r, s \rangle \mapsto [\![\mathsf{mk}_{a,b}]\!](r)(s)$ is a bijection from $[\![a]\!] \times [\![b]\!]$ onto $[\![\mathsf{pr}_{a,b}]\!]$.

Proof. It suffices to show the map $\langle r, s \rangle \mapsto \mathsf{mk}_{a,b}(r)(s)$ is invertible. Specifically, we claim the inverse map is given by $p \times q$ where

$$p = [\![\mathsf{rec}_{a,b}]\!]([\![\lambda w \mathsf{pr}_{a,b} a]\!])([\![\lambda xa\lambda ybx]\!]) \quad \text{and}$$

$$q = [\![\mathsf{rec}_{a,b}]\!]([\![\lambda w \mathsf{pr}_{a,b} b]\!])([\![\lambda xa\lambda yby]\!]).$$

To see this, take any $r \in [\![a]\!]$ and $s \in [\![b]\!]$. We can then change the interpretation $[\![\cdot]\!]$ so that $[\![x]\!] = r$ and $[\![y]\!] = s$. Further changing it so that $[\![f]\!] = [\![\lambda w \mathsf{pr}_{a,b} a]\!]$ and $[\![g]\!] = [\![\lambda ya\lambda zby]\!]$ (so $[\![f]\!](t) = [\![a]\!]$, for all $t \in [\![\mathsf{pr}_{a,b}]\!]$, and $[\![g]\!](i)(j) = i$, for all $i \in [\![a]\!]$ and $j \in [\![b]\!]$), we see that

$$
\begin{aligned}
p(\mathsf{mk}_{a,b}(r)(s)) &= [\![\mathsf{rec}_{a,b}]\!]([\![\lambda w \mathsf{pr}_{a,b} a]\!])([\![\lambda ya\lambda zby]\!])(\mathsf{mk}_{a,b}(r)(s)) \\
&= [\![\mathsf{rec}_{a,b} f g (\mathsf{mk}_{a,b} xy)]\!] \\
&= [\![gxy]\!] = [\![x]\!] = r.
\end{aligned}
$$

Likewise, instead changing $[\![\cdot]\!]$ so $[\![f]\!] = [\![\lambda w \mathsf{pr}_{a,b} b]\!]$ and $[\![g]\!] = [\![\lambda ya\lambda zbz]\!]$, the same argument yields $q(\mathsf{mk}_{a,b}(r)(s)) = s$. This shows that $p \times q$ is a left inverse to the map $\langle r, s \rangle \mapsto \mathsf{mk}_{a,b}(r)(s)$.

To show that $p \times q$ is also a right inverse, it suffices to show that $[\![\mathsf{mk}_{a,b}]\!]$ is surjective. If not, then we could take $t \in [\![\mathsf{pr}_{a,b}]\!] \setminus \mathrm{ran}([\![\mathsf{mk}_{a,b}]\!])$. Then we could further define $\phi \in [\![\mathsf{u}_n]\!]^{[\![\mathsf{pr}_{a,b}]\!]}$ so

$$
\phi(r) = \begin{cases} \{\emptyset\} & \text{if } r \in \mathrm{ran}([\![\mathsf{mk}_{a,b}]\!]) \\ \emptyset & \text{otherwise.} \end{cases}
$$

Taking $\theta \in \prod_{r \in [\![a]\!]} \prod_{s \in [\![b]\!]} (\phi([\![\mathsf{mk}_{a,b}]\!](r)(s))) = \prod_{r \in [\![a]\!]} \prod_{s \in [\![b]\!]} \{\emptyset\}$, necessarily with $\theta(r)(s) = \emptyset$, for all $r \in [\![a]\!]$ and $s \in [\![b]\!]$, we see that

$$[\![\mathsf{rec}_{a,b}]\!](\phi)(\theta)(t) \in \phi(t) = \emptyset,$$

a contradiction. Thus $[\![\mathsf{mk}_{a,b}]\!]$ must indeed be surjective. $\qquad \square$

Note the proof of the last part above would have worked just as well if we had replaced u_n in the typing statement for $\mathsf{rec}_{a,b}$ with u_0 or any other u_l. Indeed, when it comes to adding specifications like this as axioms for the syntactic system (see Definition 14.6 below), it makes sense to add corresponding statements $(\mathsf{rec}_{a,b}^l : (f : \mathsf{pr}_{a,b} \to \mathsf{u}_l) \to \ldots)$, for all $l \in \omega$. In fact, even if we just wanted to extend the result above

to models of $(a : \mathsf{u}_m)$ and $(b : \mathsf{u}_n)$ for distinct $m, n \geq 1$, then we would need both $\mathsf{rec}^m_{a,b}$ and $\mathsf{rec}^n_{a,b}$ in the first part of the proof.

Again for the syntactic system, it is often better to add sub-reduction statements as axioms rather than reduction statements. However, as long as the interpretation also satisfies the specification for equality on the appropriate set, the contextual closure of the corresponding sub-reduction statement still suffices to complete the specification, as indicated by Proposition 13.3. We can also replace the individual variables in these sub-reduction statements with arbitrary terms.

We can also make everything polymorphic by replacing a and b with v and w while adding $(v : \mathsf{u}_m) \to (w : \mathsf{u}_n) \to$ at the start of all the typing statements. So the full polymorphic specification for binary Cartesian products would then consist of the following statements, for all $l, m, n \in \omega$ with $m, n \geq 1$ and $F, G, V, W, X, Y \in \mathsf{T}$ –

$$(\mathsf{pr}_{m,n} : (v : \mathsf{u}_m) \to (w : \mathsf{u}_n) \to \mathsf{u}_{m \vee n}),$$
$$(\mathsf{mk}_{m,n} : (v : \mathsf{u}_m) \to (w : \mathsf{u}_n) \to v \to w \to \mathsf{pr}_{m,n} vw),$$
$$(\mathsf{rec}^l_{m,n} : (v : \mathsf{u}_m) \to (w : \mathsf{u}_n) \to (f : \mathsf{pr}_{m,n} vw \to \mathsf{u}_l)$$
$$\to ((x : v) \to (y : w) \to f(\mathsf{mk}_{m,n} vwxy))$$
$$\to (z : \mathsf{pr}_{m,n} vw) \to fz) \quad \text{and}$$
$$(\mathsf{rec}^l_{m,n} VWFG(\mathsf{mk}_{m,n} VWXY) \triangleright GXY)^{\mathsf{c}}.$$

The logical 'and' operation on propositions can be specified in essentially the same way, just without the need for any sub-reduction statements. Indeed, the proof of the following is the same as the last part of the proof of Proposition 13.4 above.

Proposition 13.5. *Given* $f, g, v, w, x, y, z \in \mathsf{V}$ *as well as constants* $\mathsf{and}, \mathsf{mk}, \mathsf{rec} \in \mathsf{C}$, *if* $[\![\cdot]\!]$ *is a canonical model of*

$$(\mathsf{and} : (v : \mathsf{u}_0) \to (w : \mathsf{u}_0) \to \mathsf{u}_0),$$
$$(\mathsf{mk} : (v : \mathsf{u}_0) \to (w : \mathsf{u}_0) \to v \to w \to \mathsf{and} vw) \quad and$$
$$(\mathsf{rec} : (v : \mathsf{u}_0) \to (w : \mathsf{u}_0) \to (f : \mathsf{and} vw \to \mathsf{u}_0)$$
$$\to ((x : v) \to (y : w) \to f(\mathsf{mk} vwxy))$$
$$\to (z : \mathsf{and} vw) \to fz)$$

then, for all $p, q \in [\![\mathsf{u}_0]\!]$, $\langle r, s \rangle \mapsto \mathsf{mk}(r)(s)$ *maps* $p \times q$ *onto* $[\![\mathsf{and}]\!](p)(q)$. *In particular,*

$$[\![\mathsf{and}]\!](p)(q) \neq \emptyset \qquad \Leftrightarrow \qquad p \neq \emptyset \neq q.$$

Likewise, the complete specification of the universal quantifier for propositions over some set $[\![a]\!]$ can be seen from the following.

Proposition 13.6. *Given $n \in \omega$, $f, g, q, z \in \mathsf{V}$ and $a, \forall_a, \mathsf{mk}_a, \mathsf{rec}_a \in \underline{\mathsf{C}}$, if $[\![\cdot]\!]$ is a canonical model for $(a : \mathsf{u}_n)$, $(\forall_a : (q : a \to \mathsf{u}_0) \to \mathsf{u}_0)$,*

$$(\mathsf{mk}_a : (q : a \to \mathsf{u}_0) \to \mathsf{p}aq \to \forall_a q) \quad and$$

$$(\mathsf{rec}_a : (q : a \to \mathsf{u}_0) \to (f : \forall_a q \to \mathsf{u}_0)$$

$$\to ((g : \mathsf{p}aq) \to f(\mathsf{mk}_a qg)) \to (z : \forall_a q) \to fz) \quad then,$$

for all $\theta \in [\![\mathsf{u}_0]\!]^{[\![a]\!]}$, $\phi \mapsto \mathsf{mk}_a(\theta)(\phi)$ maps $\prod \theta$ onto $[\![\forall_a]\!](\theta)$ and hence

$$[\![\forall_a]\!](\theta) \neq \emptyset \qquad \Leftrightarrow \qquad \prod \theta \neq \emptyset.$$

Again the proof is just like the last part of the proof of Proposition 13.4 and the specification can be made polymorphic as above.

14. Consequences

We now define the semantic consequence relation $\vDash$ and examine its basic properties just like in §7.

Definition 14.1. Define the *consequence relation* $\vDash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$ by

$$\Gamma \vDash X \qquad \Leftrightarrow \qquad [\![\Gamma]\!] \Rightarrow [\![X]\!], \text{ for every interpretation } [\![\cdot]\!].$$

So $\Gamma \vDash X$ means every model of Γ is a model of X.

As before, we extend any $\Vdash \subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$ to a binary relation on $\mathcal{P}(\mathsf{S})$ so that, for all $\Gamma, \Delta \subseteq \mathsf{S}$ and $S, P \in \mathsf{T}$,

$$\Gamma \Vdash \Delta \quad \Leftrightarrow \quad \Gamma \Vdash (S : P), \text{ for all } (S : P) \in \Delta.$$

We call $\Vdash$ a *sequent* if this extension defines a preorder on $\mathcal{P}(\mathsf{S})$, which is again equivalent to saying that, for all $\Gamma, \Delta \subseteq \mathsf{S}$ and $X \in \mathsf{S}$,

(Sequent) $\qquad\qquad X \in \Gamma \text{ or } \Gamma \Vdash \Delta \Vdash X \qquad \Rightarrow \qquad \Gamma \Vdash X.$

We immediately see that $\vDash$ is a sequent and hence also monotone and reflexive, exactly as in Proposition 7.3 and Corollary 7.4. We also have the following property of $\vDash$ analogous to Proposition 7.5.

Proposition 14.2. *For any $m, n \in \omega$ and $F, G, R, S \in \mathsf{T}$,*

$$(S : R), (F : \mathsf{p}_m^n RG) \vDash (FS : GS).$$

Proof. Assume $[\![\cdot]\!]$ is a model for $(S : R)$ and $(F : \mathsf{p}_m^n RG)$ so $[\![S]\!] \in [\![R]\!]$ and $[\![F]\!] \in \prod[\![G]\!]$. But also $[\![\mathsf{p}_m^n RG]\!]^{\mathsf{wf}}$ so $[\![R]\!] \in [\![\mathsf{u}_m]\!]$ and $[\![G]\!] \in [\![\mathsf{u}_n]\!]^{[\![R]\!]}$. In particular, $\mathrm{dom}[\![F]\!] = \mathrm{dom}[\![G]\!] = [\![R]\!] \ni [\![S]\!]$ so $[\![FS]\!]^{\mathsf{wf}}$, $[\![GS]\!]^{\mathsf{wf}}$ and

$$[\![FS]\!] = [\![F]\!]([\![S]\!]) \in [\![G]\!]([\![S]\!]) = [\![GS]\!].$$

This shows that $[\![\cdot]\!]$ is a model of $(FS : GS)$, as required. $\qquad\qquad \square$

The free variables in any set of statements $\Gamma \subseteq \mathsf{S}$ are given by

$$\mathsf{F}(\Gamma) = \bigcup_{(R:S)\in\Gamma} \mathsf{F}(RS).$$

So $\mathsf{F}(\Gamma)$ consists of the free variables of the subjects and predicates of typing statements in Γ. Reduction and sub-reduction statements in Γ have no bearing on $\mathsf{F}(\Gamma)$, owing to the fact models of (sub-)reduction statements are invariant under any change to the values of variables.

We now have the following analog of Proposition 7.6.

Proposition 14.3. *If $x \notin \mathsf{F}(\Gamma) \cup \mathsf{F}(Q)$ and $\Gamma \vDash (Q : \mathsf{u}_m)$ then*

$$\Gamma, (x : Q) \vDash (S : P), (P : \mathsf{u}_n) \qquad \Rightarrow \qquad \Gamma \vDash (\lambda xQS : \pi_m^n xQP).$$

Proof. Assume $\Gamma, (x : Q) \vDash (S : P), (S : \mathsf{u}_n)$ and take a model $[\![\cdot]\!]$ for Γ. If $q \in [\![Q]\!] = [\![Q]\!]_{\langle q,x\rangle}$, as $x \notin \mathsf{F}(Q)$, then $[\![x : Q]\!]_{\langle q,x\rangle}$ and also $[\![\Gamma]\!]_{\langle q,x\rangle}$, as $x \notin \mathsf{F}(\Gamma)$ (and the fact that models of (sub-)reduction statements are invariant under changing the values of variables). Thus $[\![\cdot]\!]_{\langle q,x\rangle}$ is a model for Γ and $(x : Q)$ and hence for $(S : P)$ and $(P : \mathsf{u}_n)$ as well. In particular, $[\![S]\!]^{\mathsf{wf}}_{\langle q,x\rangle}$, for all $q \in [\![Q]\!]$, and hence $[\![\lambda xQS]\!]^{\mathsf{wf}}$. Likewise $[\![\lambda xQP]\!]^{\mathsf{wf}}$ and $[\![\lambda xQP]\!] \in [\![\mathsf{u}_n]\!]^{[\![Q]\!]}$. As $[\![\Gamma]\!]$ and hence $[\![Q]\!] \in [\![\mathsf{u}_m]\!]$ too, it follows that $[\![\mathsf{p}_m^n Q\lambda xQP]\!]^{\mathsf{wf}}$. By definition, $\pi_m^n xQP = \mathsf{p}_m^n Q\lambda xQP$ and, as in the proof of Proposition 7.6,

$$[\![\lambda xQS]\!] = \{\langle [\![S]\!]_{\langle q,x\rangle}, q\rangle : q \in [\![Q]\!]\} \in \prod_{q\in[\![Q]\!]} [\![P]\!]_{\langle q,x\rangle} = \prod [\![\lambda xQP]\!] = [\![\pi_m^n xQP]\!].$$

This shows $[\![\lambda xQS : \pi_m^n xQP]\!]$ and hence $\Gamma \vDash (\lambda xQS : \pi_m^n xQP)$. $\qquad\square$

We also observe that sub-reduction statements allow us to replace the predicates of typing statements as follows.

Proposition 14.4. *For all $P, R, S \in \mathsf{T}$,*

$$(S : R), (R \triangleright P) \vDash (S : P).$$

Proof. Just note that $[\![S : R]\!]$ implies $[\![S]\!] \in [\![R]\!]$ and $[\![R]\!]^{\mathsf{wf}}$ so $[\![R \triangleright P]\!]$ implies $[\![P]\!]^{\mathsf{wf}}$ and $[\![R]\!] \subseteq [\![P]\!]$ and hence $[\![S]\!] \in [\![P]\!]$, i.e. $[\![S : P]\!]$. $\qquad\square$

The same applies subjects under reduction.

Proposition 14.5. *For all $P, R, S \in \mathsf{T}$,*

$$(R : P), (R \blacktriangleright S) \vDash (S : P).$$

Proof. As above, $[\![R : P]\!]$ implies $[\![R]\!] \in [\![P]\!]$ and $[\![R]\!]^{\mathsf{wf}}$ so $[\![R \blacktriangleright S]\!]$ implies $[\![S]\!]^{\mathsf{wf}}$ and $[\![R]\!] = [\![S]\!]$ and hence $[\![S]\!] \in [\![P]\!]$, i.e. $[\![S : P]\!]$. $\qquad\square$

We now define our syntactic *inference relation* $\vdash$ as the smallest sequent satisfying the rules arising from Propositions 14.2 to 14.5.

Definition 14.6. Let $\vdash \,\subseteq \mathcal{P}(\mathsf{S}) \times \mathsf{S}$ be the smallest sequent such that, for all $F, G, P, Q, R, S \in \mathsf{T}$ and $m, n \in \omega$,

$$(\mathrm{Red}_{\blacktriangleright}) \qquad\qquad\qquad (R : P), (R \blacktriangleright S) \vdash (S : P),$$

$$(\mathrm{Red}_{\rhd}) \qquad\qquad\qquad (S : R), (R \rhd P) \vdash (S : P),$$

$$(\mathrm{App}) \qquad\qquad (S : R), (F : \mathsf{p}_m^n RG) \vdash (FS : GS) \quad \text{and}$$

$$(\mathrm{Ab}) \quad \Gamma, (x : Q) \vdash (S : P), (P : \mathsf{u}_n) \quad \Rightarrow \quad \Gamma \vdash (\lambda x Q S : \pi_m^n x Q P),$$

whenever $\Gamma \vdash (Q : \mathsf{u}_m)$ and $x \in \mathsf{V} \setminus (\mathsf{F}(\Gamma) \cup \mathsf{F}(Q))$.

As the rules defining $\vdash$ we were already derived from the properties of $\vDash$ proved previously, the following result is immediate.

Proposition 14.7. *The inference relation is sound, i.e. $\vdash \,\subseteq\, \vDash$.*

15. Future Work

So far we have set out the basic syntax and semantics of our simplified type system with polymorphic product operators. But of course this is just the beginning and there would be much further work to do if this system were to be used as a foundation for mathematics.

Firstly, we should investigate basic properties of the inference relation $\vdash$ as in §8. The extra $(\mathrm{Red}_{\blacktriangleright})$ and $(\mathrm{Red}_{\rhd})$ rules here complicate matters and accordingly we may want to restrict the reduction and sub-reduction statements that can be added as axioms. In other words, results concerning instances $\Gamma \vdash (S : P)$ of the inference relation may only apply when $\Gamma_{\blacktriangleright} := \Gamma \cap \mathsf{S}_{\blacktriangleright}$ and $\Gamma_{\rhd} := \Gamma \cap \mathsf{S}_{\rhd}$ are of a specified form.

The most restrictive approach would be to only allow α-conversion for the reduction statements while allowing both α-conversion and β-reduction as sub-reduction statements but nothing else, much like in pure type systems. Reduction statements in specifications like those of Proposition 13.4 would then have to be replaced by typing statements involving propositional equality. A less restrictive approach would be to instead replace the reduction statements in such specifications by sub-reduction statements, as suggested in the comments after Proposition 13.4. This is essentially the approach taken in the dependent type theory underlying proof assistants like Lean – see [Car19, Car24]. It is only when we are not at all concerned with type-checking being decidable that we would opt for an 'extensional' system allowing arbitrary reduction statements, not just those for α-conversion.

If we want unique typing as in Proposition 8.9 then we should also restrict the typing statements on the left side of $\vdash$ to *contexts*, i.e. to $\Gamma \subseteq \mathsf{S}$ such that $\Gamma_{:}^{-1}$ is a function with $\mathrm{dom}(\Gamma_{:}^{-1}) \subseteq \mathsf{T}_0$ (where $\Gamma_{:} = \mathsf{S}_{:} \cap \Gamma$). If we also want to have any hope of proving some normalisation

result (that the terms involved can only be reduced finitely many times by the $(\mathrm{Red}_{\blacktriangleright})$ and $(\mathrm{Red}_{\triangleright})$ rules, at least modulo α-conversion) then we would have to further restrict to appropriate 'legal' contexts. Usually these are taken to be certain finite sequences of statements, but we could also define them for (even infinite) sets along the following lines.

Definition 15.1. A context $\Gamma \subseteq \mathsf{S}$ is *legal* if we have an enumeration $(x_n : P_n)_{n \in \omega}$ of Γ, such that, for all $j \in \omega$,

$$x_j \notin \bigcup_{i \geq j} \mathsf{F}(P_i) \quad \text{and} \quad (\Gamma \setminus \bigcup_{i \leq j}(x_i : P_i)) \vdash (P_j : \mathsf{u}_k), \text{ for some } k \in \omega.$$

One more item on the to do list would be to verify the semantic completeness of various other specifications for important mathematical structures like in Section 13. Even better would be to do this in a more systematic way. Indeed, there are already systematic ways of defining inductive specifications from a given list of constructors – see [Car19, §2.6]. These suffice to specify many mathematical structures and so a general result on their semantic completeness would cover most specifications of interest.

Finally, it would be nice to know if our inference relation satisfies any kind of completeness result to complement soundness, even for the bare bones type system in Part 1. In other words, are there any suitable conditions under which $\Gamma \vDash (S : P)$ implies $\Gamma \vdash (S : P)$ or even $\Gamma \vdash (T : P)$ for some other term T? If not then what are the 'missing' inference rules that would be required for this to hold?

References

[Acz99] Peter Aczel. On relating type theories and set theories. In *Types for proofs and programs. International workshop, TYPES '98. Kloster Irsee, Germany, March 27–31, 1999. Selected papers*, pages 1–18. Berlin: Springer, 1999.

[Bar92] H. P. Barendregt. Lambda calculi with types. In *Handbook of logic in computer science, Vol. 2*, volume 2 of *Handb. Log. Comput. Sci.*, pages 117–309. Oxford Univ. Press, New York, 1992.

[Car19] Mario Carneiro. The type theory of Lean. Master's thesis, Carnegie Mellon University, 2019. URL: `https://github.com/digama0/lean-type-theory/releases/tag/v1.0`.

[Car24] Mario Carneiro. Lean4lean: Towards a verified typechecker for Lean, in Lean, 2024. `arXiv:2403.14064`.

[Chu32] A. Church. A set of postulates for the foundation of logic. I, II. *Ann. Math. (2)*, 33:346–366, 1932. `doi:10.2307/1968337`.

[Chu36] A. Church. An unsolvable problem of elementary number theory. *Am. J. Math.*, 58:345–363, 1936. `doi:10.2307/2371045`.

[CK90] C. C. Chang and H. J. Keisler. *Model theory.*, volume 73 of *Stud. Logic Found. Math.* Amsterdam etc.: North-Holland, 3rd rev. ed. edition, 1990.

[Coq94] Thierry Coquand. A new paradox in type theory. In *Logic, methodology and philosophy of science IX. Proceedings of the ninth international congress of logic, methodology and philosophy of science, Uppsala, Sweden, August 7–14, 1991*, pages 555–570. Amsterdam: North-Holland, 1994.

[CP90]   Thierry Coquand and Christine Paulin. Inductively defined types. Colog-88, Proc. Int. Conf., Tallinn/USSR 1988, Lect. Not. Comput. Sci. 417, 50-66 (1990)., 1990. URL: `https://david.darais.com/assets/Inductively_Defined_Types--Coquand+Paulin.pdf`.

[HS08]   J. Roger Hindley and Jonathan P. Seldin. *Lambda-calculus and combinators, an introduction*. Cambridge: Cambridge University Press, 2nd revised and updated ed. edition, 2008.

[Hur95] Antonius J. C. Hurkens. A simplification of Girard's paradox. In *Typed lambda calculi and applications. 2nd international conference, TLCA '95, Edinburgh, GB, April 10–12, 1995. Proceedings*, pages 266–278. Berlin: Springer-Verlag, 1995.

[KR35]   S. C. Kleene and J. B. Rosser. The inconsistency of certain formal logics. *Ann. Math. (2)*, 36:630–636, 1935. `doi:10.2307/1968646`.

[Kri65] S. A. Kripke. Semantical analysis of intuitionistic logic. I. Formal Syst. and Recurs. Funct., Proc. 8th Logic Colloquium, Oxford 1963, 92-130 (1965)., 1965.

[Kun11] Kenneth Kunen. *Set theory.*, volume 34 of *Stud. Log. (Lond.)*. London: College Publications, 2011.

[Lin02] Bernard Linsky. The substitutional paradox in Russell's 1907 letter to Hawtrey. *Russell: The Journal of Bertrand Russell Studies*, 22(1):47–55, 2002.

[Luo94] Zhaohui Luo. *Computation and reasoning. A type theory for computer science*, volume 11 of *Int. Ser. Monogr. Comput. Sci.* Oxford: Clarendon Press, 1994.

[Mis98] Michael W. Mislove. Topology, domain theory and theoretical computer science. *Topology Appl.*, 89(1-2):3–59, 1998. `doi:10.1016/S0166-8641(97)00222-8`.

[MW03]   Alexandre Miquel and Benjamin Werner. The not so simple proof-irrelevant model of CC. In *Types for proofs and programs. International workshop, TYPES 2002, Berg en Dal, The Netherlands, April 24–28, 2002. Selected papers*, pages 240–258. Berlin: Springer, 2003. URL: `link.springer.de/link/service/series/0558/bibs/2646/26460240.htm`.

[RW10]   B. Russell and A. N. Whitehead. Principia Mathematica. Vol. I. Cambridge: University Press. xv, 666 S. (1910)., 1910. URL: `www.hti.umich.edu/cgi/t/text/text-idx?c=umhistmath;idno=AAT3201`.

Institute of Mathematics of the Czech Academy of Sciences, Žitná 25, Prague

Email address: `bice@math.cas.cz`