A Correspondence between Nuprl, the Ramified Theory of Types and Pure Type Systems*

Fairouz Kamareddine†

University of Glasgow
Department of Computing Science
17 Lilybank Gardens
Glasgow G12 8QQ, Scotland
FAX: +44 141 330 4913
fairouz@dcs.gla.ac.uk
http://www.dcs.gla.ac.uk/research/beauty

Twan Laan‡

Mathematics and Computing Science
Eindhoven University of Technology
P.O.Box 513
5600 MB Eindhoven
The Netherlands
FAX: +31 40 2463992
laan@win.tue.nl

Abstract

In Russell's Ramified Theory of Types RTT, two hierarchical concepts dominate: *orders* and *types*. The use of orders has as a consequence that the logic part of RTT is predicative. The concept of order however, is almost dead since Ramsey eliminated it from RTT. This is why we find Church's simple theory of types (which uses the type concept without the order one) at the bottom of the Barendregt Cube rather than RTT. Despite the disappearence of orders which have a strong correlation with predicativity, predicative logic still plays an influential role in Computer Science. An important example is the proof checker Nuprl, which is based on Martin-Löf's Type Theory which uses type universes. Those type universes, and also degrees of expressions in AUTOMATH, are closely related to orders. In this paper, we show that orders have not disappeared from modern logic and computer science, rather, orders play a crucial role in understanding the hierarchy of modern systems. In order to achieve our goal, we concentrate on Nuprl.

The novelty of our paper lies in: 1) the revival of Russell's orders, 2) the placing of the historical system RTT underlying the famous Principia Mathematica in a context with a modern system of computer mathematics (Nuprl) and modern type theories (Martin-Löf's type theory and PTSs), and 3) the presentation of a complex type system (Nuprl) as a simple and compact PTS.

1 Introduction

The Ramified Theory of Types (RTT) was developed by Bertrand Russell [21, 25] in order to solve the paradoxes that resulted from Frege's "Grundgesetze der Arithmetik" [6]. It has a double hierarchy: one of types (which can be seen as an elementary version of Church's well-known Simple Theory of Types [2]) and one

*This work is partially supported by EPSRC grant GR/K 25014. The authors are grateful for some useful discussions with Roel Bloo, Herman Geuvers, Jan Zwanenburg and Joe Wells. We are also grateful for the useful remarks and suggestions of an anonymous referee.

†Kamareddine is grateful to Assaf Kfoury and Joe Wells for their hospitality at Boston University during the preparation of this article.

‡Laan is supported by the Co-operation Centre Tilburg and Eindhoven Universities. He is grateful to the Department of Computing Science, University of Glasgow, for their hospitality and financial support when preparing this article.

of orders, which can be compared with Kripke's Hierarchy of Truths, see [13, 11]. The hierarchy of orders is less known, as it became unpopular when Ramsey [19] and Hilbert and Ackermann [7] showed that one can avoid the paradoxes without this hierarchy. Furthermore, even though it became widely acknowledged that the paradoxes can be avoided without the use of orders, we believe that many logicians are (maybe unconsiously) influenced by the hierarchy of orders when constructing (non-paradoxical) theories. Moreover, orders can elegantly explain some useful hierarchies. As an example, when Kripke wanted to build a logical theory [13] which has its own truth predicate (something not straightforward according to Tarski's hierarchy of truths [24], in which the truth predicate is not definable), he used a hierarchy of languages which could elegantly be explained via the notion of orders as is shown in [11]. Similarly, when Martin-Löf's impredicative type theory was shown to suffer from the paradox, he moved to the predicative version in [15] and has since, built layers of universes that again could be elegantly explained by orders. Also, orders are closely related to the degree of expression notion of AUTOMATH [18].

Logic based on the double hierarchy of orders and types is usually called *predicative*. The difference between predicative and impredicative logic may seem small, nevertheless, this small difference can have some drastic consequences in fundamental mathematics. When constructing the real numbers out of the rationals (with Dedekind-cuts), the Theorem of the Lowest Upper Bound[1], is not provable in predicative logic (see [23]). The Theorem of the Lowest Upper Bound is, however, one of the most fundamental theorems in real analysis.

Many modern type systems are impredicative. For instance, the systems of the Barendregt cube [1] that have the rule "$(\square, *)$" are all impredicative. Hence, a proof checker like Coq [5], based on the Calculus of Constructions [4], is itself founded on impredicative logic.

Nevertheless, mathematics with predicative logic is possible, and from a constructive point of view it is even attractive. For instance, the proof checker Nuprl [3, 10] is based on predicative logic yet many mathematical theories can be developed using this proof checker (see [9]).

Nuprl's type theory is related to type theories proposed by Martin-Löf [16], used as a foundation for constructive mathematics. Nuprl's logic is related to its type theory via the well-known propositions-as-types embedding, also known as the Curry-Howard-de Bruijn isomorphism (see [8]). It is constructive on two points: it is based on intuitionistic logic (as is the Curry-Howard-de Bruijn isomorphism) and it is based on predicative logic.

In this paper, we will try to establish the relation between predicative logic as present in modern type theory (we concentrate on Nuprl because Martin-Löf's type theory is one of the richest and most expressive predicative type theories) and Russell's Ramified Type Theory RTT. This has many advantages, the most important of which is the formulation of the informal notion of universe hierarchy in these modern predicative logics using Russell's notion of order. There are however many important bonuses that result from our study:

1. We give the first presentation of the proof checker Nuprl as a PTS. In Section 2 we give a formal description of a part of the type system of Nuprl as a Pure Type System (PTS) [22]. The systems of the Barendregt cube are examples of PTSs. Nuprl in PTS style enables us to formalize the concept of order in Nuprl and to show its correctness. This order classifies types and terms of Nuprl into their relevant hierarchy.

[1] This Theorem states that any non-empty set of real numbers with an upper bound has a least upper bound.

2. We give a formal presentation of RTT. Such a formal presentation is not given in "Principia" [25]. In Section 3 we present a simplified formalization of RTT, which is based on a more extensive formalization given in [14].

3. We give the first account of embedding RTT in a relevant modern type theory. This is done in Section 4, where we present an embedding of RTT in Nuprl's type system.

4. Our study is the first to connect RTT to the modern way of writing type theory as a PTS. As we present Nuprl within the framework of PTSs in Section 2, and as we present an embedding of RTT in Nuprl's type system in Section 4, we also obtain a description of RTT in PTS-style.

5. Our study is the first to show that orders in the historical system RTT correspond to orders in a very powerful modern system Nuprl.

6. Finally, our paper places the historical system underlying Principia Mathematica in a context with a modern system of computer mathematics (Nuprl) and modern type theories (Martin-Löf's type theory and Pure Type Systems).

2 The Nuprl type system

2a A fragment of Nuprl in PTS-style

We give a description of a part of the type system on which Nuprl is based (see [9, 3]). We don't give a full presentation of all of Nuprl's type constructors, as we will only need parts of it. The description of the typing rules is given in a natural deduction style similar to that used in the Barendregt Cube [1], and Pure Type Systems [22].

Below we assume $\mathbb{V}$ to be a set of variables, $\mathbb{Z}$ to be the set of integers, and $\mathbb{S} = \{*_1, *_2, \ldots\}$ a set of sorts. The intuition behind the sort $*_a$ is that it represents the propositions (and, more general, the types) of order $\leq a$. $*_a$ corresponds to the Universe of Types $\mathbb{U}_a$ in [9, 16]. $\perp$ represents the undefined or a contradiction. Application and abstraction (λ and Π) are familiar from PTSs. The remaining notions represent cartesian products, pairing, and first and second projections.

Definition 2.1 (Terms) The set of terms $\mathbb{T}$ is defined by the following abstract syntax:

$$\mathbb{T} ::= \mathbb{S} \mid \mathbb{V} \mid \perp \mid \mathbb{Z} \mid \mathbb{T}\mathbb{T} \mid \lambda\mathbb{V}{:}\mathbb{T}.\mathbb{T} \mid \Pi\mathbb{V}{:}\mathbb{T}.\mathbb{T} \mid \mathbb{T} \times \mathbb{T} \mid \langle\mathbb{T},\mathbb{T}\rangle \mid \pi_1(\mathbb{T}) \mid \pi_2(\mathbb{T})$$

We let $\alpha, \beta, x, y, z, \ldots$ range over $\mathbb{V}$; $m, n, \ldots$ over $\mathbb{Z}$ and A, B, M, N, a, b over $\mathbb{T}$. When x does not occur free in B, we write $A \to B$ for $\Pi x{:}A.B$. Free and bound variables are defined as usual. $\mathrm{FV}(A)$ and $\mathrm{BV}(A)$ denote the set of free and bound variables of A. $A[x{:=}B]$ denotes the term in which all the free occurrences of x in A have been replaced by B. Syntactic equality of terms is taken modulo renaming of bound variables. This allows us to assume the following:

Convention 2.2 (Barendregt's Convention) Names of bound variables differ from the free ones in a term. Moreover, we use different bound names for different bound variables.

We take the axioms:
$(\to_\beta) : (\lambda x{:}T.A)B \to_\beta A[x{:=}B]$
$(\to_\sigma) : \pi_1(\langle A,B\rangle) \to_\sigma A$ and $\pi_2(\langle A,B\rangle) \to_\sigma B$.

We define the reduction relations $\to_\beta$ and $\to_\sigma$ generated by the above two axioms respectively (with the usual compatibility rules of course). $\twoheadrightarrow_\beta$ and $\twoheadrightarrow_\sigma$ are the reflexive transitive closures of $\to_\beta$ and $\to_\sigma$. We define moreover $\to_{\beta\sigma}$ and $\twoheadrightarrow_{\beta\sigma}$ in the obvious way and take $=_{\beta\sigma}$ to be the symmetric closure of $\twoheadrightarrow_{\beta\sigma}$. We define contexts and some related properties:

Definition 2.3 (Contexts) A *context* is a finite list $x_1{:}A_1, \ldots, x_n{:}A_n$ of declarations $x_i{:}A_i$. $\{x_1, \ldots, x_n\}$ is called the *domain* of the context. If Γ, Δ are contexts then we write $\Gamma \subseteq \Delta$ if all declarations in Γ are also in Δ. We let Γ, Δ range over contexts.

Definition 2.4 (Derivable statements) A statement $\Gamma \vdash A : B$ is *derivable* if it can be deduced by repeated application of the rules below:

$$\begin{array}{lcl}
\text{(Axioms)} & \begin{array}{lll} \vdash \bot : *_1 & \vdash *_n : *_{n+1} & (n \in \mathbb{N}) \\ \vdash \mathbb{Z} : *_1 & \vdash n : \mathbb{Z} & (n \in \mathbb{Z}) \end{array} & \\
\text{(Start)} & \dfrac{\Gamma \vdash A : *_n}{\Gamma, x{:}A \vdash x{:}A} & (x \text{ is } \Gamma\text{-fresh}) \\
\text{(Weak)} & \dfrac{\Gamma \vdash M : N \qquad \Gamma \vdash A : *_n}{\Gamma, x{:}A \vdash M : N} & (x \text{ is } \Gamma\text{-fresh}) \\
\text{(}\Pi\text{-form)} & \dfrac{\Gamma \vdash A : *_n \qquad \Gamma, x{:}A \vdash B : *_n}{\Gamma \vdash (\Pi x{:}A.B) : *_n} & \\
(\lambda) & \dfrac{\Gamma, x{:}A \vdash b : B \qquad \Gamma \vdash (\Pi x{:}A.B) : *_n}{\Gamma \vdash (\lambda x{:}A.b) : (\Pi x{:}A.B)} & \\
\text{(App)} & \dfrac{\Gamma \vdash M : (\Pi x{:}A.B) \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B[x{:=}N]} & \\
(\times\text{-form}) & \dfrac{\Gamma \vdash A : *_n \qquad \Gamma \vdash B : *_n}{\Gamma \vdash (A \times B) : *_n} & \\
\text{(Pairs)} & \dfrac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B \qquad \Gamma \vdash (A \times B) : *_m}{\Gamma \vdash \langle a, b \rangle : (A \times B)} & \\
\text{(Left)} & \dfrac{\Gamma \vdash M : (A \times B)}{\Gamma \vdash \pi_1(M) : A} & \\
\text{(Right)} & \dfrac{\Gamma \vdash M : (A \times B)}{\Gamma \vdash \pi_2(M) : B} & \\
\text{(Conv)} & \dfrac{\Gamma \vdash M : A \qquad \Gamma \vdash B : *_n \qquad A =_{\beta\sigma} B}{\Gamma \vdash M : B} & \\
(\subseteq) & \dfrac{\Gamma \vdash A : *_n}{\Gamma \vdash A : *_{n+1}} &
\end{array}$$

To those familiar with PTSs and/or Nuprl, the above rules are straightforward. Some remarks are due however:

1. The rule (Π-form) may look restrictive. This is not the case however due to the inclusion rule ($\subseteq$). Rather, it is fair to say that ($\subseteq$) simplifies the formulation without sacrifying expressivity.

2. A type universe $\mathbb{U}_n$ of Nuprl is closed under the construction of *dependent* cartesian products. We use non-dependent cartesian products ($\times$-form) . We refrain from introducing dependent cartesian products for two reasons: they are not needed for the purpose of the paper and they involve many complications that will obscure our main objectives.

3. The inclusion rule ($\subseteq$) is interesting on its own. We will see below that it leads to the loss of unicity of types. However, unicity of types is valued in many PTSs but not in Nuprl or Martin-Löf's type theory. We will in any case derive a version of unicity of types that is faithful to this idea of a term

having many types in Nuprl. That is, we will derive that if we collapse the orders, then a term will have only one type.

4. Nuprl itself is implicitly rather than explicitly typed. That is, Nuprl uses terms of the form $\lambda x.B$ rather than $\lambda x{:}A.B$. There is a huge literature in programming language theory and design which discusses the tradeoffs between both styles. Our reason for the explicitly typed style in Nuprl is due to the fact that PTSs deal with explicitly typed systems and it is not obvious how to extend them to the implicitly typed style.

Now we define some notions familiar from PTSs.

Definition 2.5

- Γ is called *legal* if there are A, B such that $\Gamma \vdash A : B$;
- A is called *legal* if there are Γ, B such that $\Gamma \vdash A : B$ or $\Gamma \vdash B : A$;
- A is called a Γ-*term* if there is B such that $\Gamma \vdash A : B$ or $\Gamma \vdash B : A$;
- A is called a Γ-*type* if there is n such that $\Gamma \vdash A : *_n$.

We now show some PTS properties of the Nuprl type system. Omitted proofs are as in [1].

Theorem 2.6 (Church Rosser Theorem for $\to_\beta$ and $\to_\sigma$)

1. If $A \twoheadrightarrow_\beta B_1$ and $A \twoheadrightarrow_\beta B_2$ then there is C such that $B_1 \twoheadrightarrow_\beta C$ and $B_2 \twoheadrightarrow_\beta C$.

2. If $A \twoheadrightarrow_\sigma B_1$ and $A \twoheadrightarrow_\sigma B_2$ then there is C such that $B_1 \twoheadrightarrow_\sigma C$ and $B_2 \twoheadrightarrow_\sigma C$.

PROOF: 2: any orthogonal term rewrite system (hence $(\mathbb{T}, \to_\sigma)$) is Church Rosser (see [12]). $\square$

Theorem 2.7 (Church Rosser Theorem for $\to_{\beta\sigma}$)

1. If $A \to_\beta B_1$ and $A \to_\sigma B_2$ then $\exists C$ such that $B_1 \twoheadrightarrow_\sigma C$, and either $B_2 \to_\beta C$ or $B_2 \equiv C$;

2. If $A \to_\beta B_1$ and $A \twoheadrightarrow_\sigma B_2$ then $\exists C$ such that $B_1 \twoheadrightarrow_\sigma C$, and either $B_2 \to_\beta C$ or $B_2 \equiv C$;

3. If $A \twoheadrightarrow_\beta B_1$ and $A \twoheadrightarrow_\sigma B_2$ then $\exists C$ such that $B_1 \twoheadrightarrow_\sigma C$ and $B_2 \twoheadrightarrow_\beta C$;

4. $\to_{\beta\sigma}$ has the Church Rosser property.

PROOF: 1: induction on the structure of A. 2: use 1. 3: use 2. 4: use 3 and Theorem 2.6. $\square$

Lemma 2.8 (Free Variable Lemma) *Assume $x_1{:}A_1, \ldots, x_n{:}A_n \vdash B : C$. Then*

- *The $x_1, \ldots, x_n$ are distinct;*
- $\mathrm{FV}(B) \cup \mathrm{FV}(C) \subseteq \{x_1, \ldots, x_n\}$;
- *For each i there is m such that $x_1{:}A_1, \ldots, x_{i-1}{:}A_{i-1} \vdash A_i : *_m$.* $\square$

Lemma 2.9 (Start Lemma) *Assume Γ is a legal context. Then $\Gamma \vdash \bot : *_1$, $\Gamma \vdash \mathbb{Z} : *_1$, $\Gamma \vdash n : \mathbb{Z}$ for any $n \in \mathbb{Z}$, and $\Gamma \vdash *_n{:}*_{n+1}$ for any $n \geq 1$. Moreover, $\Gamma \vdash x{:}C$ for all $x{:}C \in \Gamma$.* $\square$

Lemma 2.10 (Transitivity Lemma) *Let* Γ, Δ *be legal contexts such that* $\Gamma \vdash x : C$ *for all* $x{:}C \in \Delta$. *Then* $\Delta \vdash A : B \Rightarrow \Gamma \vdash A : B$.

Lemma 2.11 (Substitution Lemma)
If $\Gamma, x{:}A, \Delta \vdash B : C$ *and* $\Gamma \vdash D : A$ *then* $\Gamma, \Delta[x{:=}D] \vdash B[x{:=}D] : C[x{:=}D]$.

Lemma 2.12 (Thinning Lemma)
Let Γ, Δ *be legal contexts,* $\Gamma \subseteq \Delta$. $\Gamma \vdash A : B \Rightarrow \Delta \vdash A : B$.

Lemma 2.13 (Generation Lemma)

1. *If* $\Gamma \vdash *_n : C$ *then* $C =_{\beta\sigma} *_m$ *for a* $m > n$, *and if* $C \not\equiv *_m$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
2. *If* $\Gamma \vdash \bot : C$ *then* $C =_{\beta\sigma} *_m$ *for some* $m \geq 1$, *and if* $C \not\equiv *_m$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
3. *If* $\Gamma \vdash \mathbb{Z} : C$ *then* $C =_{\beta\sigma} *_m$ *for some* $m \geq 1$, *and if* $C \not\equiv *_m$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
4. *If* $\Gamma \vdash n : C$ *then* $C =_{\beta\sigma} \mathbb{Z}$, *and if* $C \not\equiv \mathbb{Z}$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
5. *If* $\Gamma \vdash x : C$ *then there is* B *such that* $x{:}B \in \Gamma$, *and either* $B =_{\beta\sigma} C$, *or there are* m, n *with* $m < n$ *and* $B =_{\beta\sigma} *_m$, $C =_{\beta\sigma} *_n$. *If* $C \not\equiv B$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
6. *If* $\Gamma \vdash (\Pi x{:}A.B) : C$ *then there is* m *such that* $\Gamma \vdash A : *_m$, $\Gamma, x{:}A \vdash B : *_m$ *and* $C =_{\beta\sigma} *_n$ *for a* $n \geq m$. *If* $C \not\equiv *_n$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
7. *If* $\Gamma \vdash (\lambda x{:}A.b) : C$ *then there are* m, B *such that* $\Gamma \vdash (\Pi x{:}A.B) : *_m$, $\Gamma, x{:}A \vdash b : B$ *and* $C =_{\beta\sigma} \Pi x{:}A.B$. *If* $C \not\equiv \Pi x{:}A.B$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
8. *If* $\Gamma \vdash AB : C$ *then there are* x, P, Q *such that* $\Gamma \vdash A : (\Pi x{:}P.Q)$, $\Gamma \vdash B : P$ *and either* $C =_{\beta\sigma} Q[x{:=}B]$, *or there are* m, n *with* $m < n$ *and* $Q[x{:=}B] =_{\beta\sigma} *_m$ *and* $C =_{\beta\sigma} *_n$. *If* $C \not\equiv Q[x{:=}B]$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
9. *If* $\Gamma \vdash (A \times B) : C$ *then there is* m *such that* $\Gamma \vdash A : *_m$, $\Gamma \vdash B : *_m$ *and* $C =_{\beta\sigma} *_n$ *for a* $n \geq m$. *If* $C \not\equiv *_n$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
10. *If* $\Gamma \vdash \langle a, b\rangle : C$ *then there are* m, A, B *such that* $\Gamma \vdash (A \times B) : *_m$, $\Gamma \vdash a : A$, $\Gamma \vdash b : B$ *and* $C =_{\beta\sigma} A \times B$. *If* $C \not\equiv A \times B$ *then* $\Gamma \vdash C : *_p$ *for some* $p \geq 1$.
11. *If* $\Gamma \vdash \pi_i(M) : C$ *then there are* A_1, A_2 *such that* $\Gamma \vdash M : (A_1 \times A_2)$ *and either* $C =_{\beta\sigma} A_i$ *or there are* m, n *with* $m < n$ *and* $A_i =_{\beta\sigma} *_m$ *and* $C =_{\beta\sigma} *_n$.

PROOF: Tedious but straightforward induction on the derivation $\Gamma \vdash M : C$. We only show two cases:

(Conversion:) $\Gamma \vdash M : C$ because $\Gamma \vdash C : *_p$, $\Gamma \vdash M : C'$ and $C =_{\beta\sigma} C'$. We treat only the case $M \equiv AB$, the others are similar or easier. With the induction hypothesis, determine x, P, Q such that $\Gamma \vdash A : (\Pi x{:}P.Q)$, $\Gamma \vdash B : P$. If $Q[x{:=}B] =_{\beta\sigma} C'$ then also $Q[x{:=}B] =_{\beta\sigma} C$; if $m < n$ such that $Q[x{:=}B] =_{\beta\sigma} *_m$ and $C' =_{\beta\sigma} *_n$ then also $C =_{\beta\sigma} *_n$.

($\subseteq$): $\Gamma \vdash M : *_{k+1}$ because $\Gamma \vdash M : *_k$. Notice that, by the induction hypothesis, the cases $M \equiv n$ and $M \equiv \lambda x{:}A.b$ are impossible. We treat the case $M \equiv AB$; the other cases are similar or easier. By the induction hypothesis, there are x, P, Q such that $\Gamma \vdash A : (\Pi x{:}P.Q)$, $\Gamma \vdash B : P$. If $*_k =_{\beta\sigma} Q[x{:=}B]$ then take $m = k$ and $n = k + 1$; if there are $m' < n'$ such that $Q[x{:=}B] =_{\beta\sigma} *_{m'}$ and $*_k =_{\beta\sigma} *_{n'}$ then notice that $k = n'$ by the Church Russer Theorem, and take $m = m'$ and $n = k + 1$. □

Corollary 2.14 (Correctness of Types)
If $\Gamma \vdash A : B$ *then there is* $n \geq 1$ *such that* $\Gamma \vdash B : *_n$.

PROOF: Induction on $\Gamma \vdash A : B$ with the help of the Generation Lemma and the Substitution Lemma for the cases $A \equiv MN$, $A \equiv \pi_1(M)$ and $A \equiv \pi_2(M)$. $\square$

Theorem 2.15 (Subject Reduction)
If $\Gamma \vdash A : B$ *and* $A \to_{\beta\sigma} A'$ *then* $\Gamma \vdash A' : B$.

PROOF: As is usual in the literature, we use induction on $\Gamma \vdash A : B$ to prove simultaneously

- $\Gamma \vdash A : B, \Gamma \to_\beta \Gamma' \Rightarrow \Gamma' \vdash A : B$;
- $\Gamma \vdash A : B, A \to_\beta A' \Rightarrow \Gamma \vdash A' : B$. $\square$

Corollary 2.16 ($\twoheadrightarrow_{\beta\sigma}$ preserves Γ-terms)
If A *is a* Γ*-term and* $A \twoheadrightarrow_{\beta\sigma} A'$ *then* A' *is a* Γ*-term.*

PROOF: We only prove the case $A \to_{\beta\sigma} A'$. If $\Gamma \vdash A : B$ then by Subject Reduction, $\Gamma \vdash A' : B$ and A' is a Γ-term. If $\Gamma \vdash B : A$ then by correctness of types $\Gamma \vdash A : *_n$ for some n and we use Subject Reduction. $\square$

Due to ($\subseteq$), Unicity of Types doesn't hold for Nuprl. For example, $\perp : *_1$ and $\perp : *_2$. A weak version however, is possible. This version collapses the different levels of $*$'s into $*_1$:

Definition 2.17 For each term A we define a term $|A|$ as follows:

$$\begin{array}{ll}
|\,*_m\,| = *_1 & |\Pi x{:}A.B| = \Pi x{:}A.|B| \\
|x| = x & |A \times B| = |A| \times |B| \\
|\perp| = \perp & |\,\langle A, B\rangle\,| = \langle |A|, |B|\rangle \\
|\mathbb{Z}| = \mathbb{Z} & |\pi_1(M)| = \pi_1(|M|) \\
|MN| = |M||N| & |\pi_2(M)| = \pi_2(|M|) \\
|\lambda x{:}A.b| = \lambda x{:}A.|b| &
\end{array}$$

Theorem 2.18 (Weak Unicity of Types)
If $\Gamma \vdash A : B_1$ *and* $\Gamma \vdash A : B_2$ *then* $|B_1| =_{\beta\sigma} |B_2|$.

PROOF: Induction on the structure of A. We only treat $A \equiv (\lambda x{:}M.N)$. By Lemma 2.13, $\exists D_1$, D_2 with $B_j =_{\beta\sigma} \Pi x{:}M.D_j$, and $\Gamma, x{:}M \vdash N : D_j$. By the induction hypothesis, $|D_1| =_{\beta\sigma} |D_2|$. Hence, $|B_1| =_{\beta\sigma} |\Pi x{:}M.D_1| \equiv \Pi x{:}M.|D_1| =_{\beta\sigma} \Pi x{:}M.|D_2| \equiv |\Pi x{:}M.D_2| =_{\beta\sigma} |B_2|$. $\square$

2b Orders in Nuprl

Correctness of Types makes the following lemma and definition possible:

Lemma 2.19 *If* A *is a* Γ*-term then* $\exists$ *a* Γ*-term* B, $\exists n \geq 1$ *such that* $\Gamma \vdash A : B : *_n$.

PROOF: A is a Γ-term $\Rightarrow$ $\exists$ Γ-term B with $\Gamma \vdash A : B$ or $\Gamma \vdash B : A$. If $\Gamma \vdash A : B$, then by Correctness of Types $\exists n \geq 1$ where $\Gamma \vdash A : B : *_n$. If $\Gamma \vdash B : A$ then again by Correctness of Types $\exists n \geq 1$ where $\Gamma \vdash A : *_n$ and hence by Start and Thinning, $\Gamma \vdash A : *_n : *_{n+1}$. $\square$

Note that by Corollary 2.16, if A is a Γ-term then for any A' where $A \twoheadrightarrow_{\beta\sigma} A'$, A' is a Γ-term. There are also $A' =_{\beta\sigma} A$ where $A \not\twoheadrightarrow_{\beta\sigma} A'$ yet A' is a Γ-term. For example, take $A = (\lambda x : \Pi z : *_1. *_1 .xa)b$ and $A' = (\lambda y : *_1.by)a$. For this reason, we introduce the following definition:

Definition 2.20 (Γ-terms modulo A)
We define $[A]_\Gamma = \{A' | A' \text{ is } \Gamma\text{-term and } A =_{\beta\sigma} A'\}$.

Now, we define the order of a term:

Definition 2.21 (Order of a Term)
Assume A is a Γ-term. We define $\mathrm{ord}_\Gamma(A)$, the *order* of A in Γ, as the smallest natural number a (i.e. $a \geq 0$) for which there are $A' \in [A]_\Gamma$ and B such that $\Gamma \vdash A' : B : *_{a+1}$.

Let us explain the intuition behind this definition. The order of a term A must be the smallest natural number n such that the type of A is of type $*_{n+1}$. By ($\subseteq$), we get that for any $m > n$, the type of A is also of type $*_m$. This captures the notion of orders à la Russell. If A itself is a type and n is the order of of A, then not only the type of A is of type $*_{n+1}$, but also $A \twoheadrightarrow_{\beta\sigma} A'$ for some A' of type $*_n$ (see Lemma 2.29). Moreover, $*_n$ can be regarded as the type of types of order $\leq n$ (Corollary 2.30) and a term is always of a lower order than its type (Corollary 2.31). More importantly also, is the fact that a function can never take arguments of a higher order than itself (Lemma 2.33).

Of course, we want to make sure that any element $=_{\beta\sigma}$ to A has the same order as A. For this reason, we defined order as above by finding one A' in $[A]_\Gamma$ which gives us the minimal n in question. Even better, there is such an A' where $A \twoheadrightarrow_{\beta\sigma} A'$ rather than only $A =_{\beta\sigma} A'$. The following lemma shows this:

Lemma 2.22 *Let A be a Γ-term and $\mathrm{ord}_\Gamma(A) = a$. The following holds:*

1. *If $A' \in [A]_\Gamma$ then $\mathrm{ord}_\Gamma(A) = \mathrm{ord}_\Gamma(A')$.*
2. *There are A' and B such that $\Gamma \vdash A' : B : *_{a+1}$ and $A \twoheadrightarrow_{\beta\sigma} A'$.*

PROOF: 1: easy. 2: by definition of $\mathrm{ord}_\Gamma(A)$, $\exists A'' =_{\beta\sigma} A$ and B where $\Gamma \vdash A'' : B : *_{a+1}$. By Church Rosser, A, A'' have a common reduct, say A'. By Subject Reduction, $\Gamma \vdash A' : B : *_{a+1}$. □

Corollary 2.23
For a Γ-term A in $\beta\sigma$-normal form and $\mathrm{ord}_\Gamma(A) = a$, $\exists B$ where $\Gamma \vdash A : B : *_{a+1}$.

PROOF: Determine, with Lemma 2.22, A' and B such that $A \twoheadrightarrow_{\beta\sigma} A'$ and $\Gamma \vdash A' : B : *_{a+1}$. As A is in normal form, $A' \equiv A$. □

In what follows, we prove some elementary properties of $\mathrm{ord}_\Gamma(A)$. The first such property states that the order of a term does not change if the context is expanded:

Lemma 2.24 (Orders are invariant under context expansion)
If $\Gamma \vdash A : B$ and $\Gamma, x{:}C$ is legal, then $\mathrm{ord}_\Gamma(A) = \mathrm{ord}_{\Gamma,x:C}(A)$.

PROOF: Let $a = \mathrm{ord}_{\Gamma,x:C}(A)$. ($\geq$) By Thinning, $\Gamma \vdash A' : P \Rightarrow \Gamma, x{:}C \vdash A' : P$ for all $A' =_{\beta\sigma} A$ and P, so $\mathrm{ord}_\Gamma(A) \geq a$. ($\leq$) $\exists A' =_{\beta\sigma} A$ and P with $\Gamma, x{:}C \vdash A' : P : *_{a+1}$. By Lemma 2.22, assume $A \twoheadrightarrow_{\beta\sigma} A'$. By Lemma 2.11, $\Gamma \vdash A'[x{:=}C] : P[x{:=}C] : *_{a+1}$. As $\mathrm{FV}(A') \subseteq \mathrm{FV}(A) \subseteq \mathrm{dom}(\Gamma)$, $x \notin \mathrm{FV}(A')$. Hence $A' \equiv A'[x{:=}C]$, so $\Gamma \vdash A' : P[x{:=}C] : *_{a+1}$ and $\mathrm{ord}_\Gamma(A) \leq \mathrm{ord}_{\Gamma,x:C}(A)$. □

Corollary 2.25 *If A is a Γ-term and $\Delta \supseteq \Gamma$ is legal then $\mathrm{ord}_\Delta(A) = \mathrm{ord}_\Gamma(A)$.*

The order of a term does not increase under substitution:

Lemma 2.26 (Substitution does not lead to order increase)
If $\Gamma, x{:}A, \Delta \vdash B : C$ and $\Gamma \vdash D : A$ then $\mathrm{ord}_{\Gamma,x:A,\Delta}(B) \geq \mathrm{ord}_{\Gamma,\Delta[x:=D]}(B[x{:=}D])$.

PROOF: $\Gamma' = \Gamma, x{:}A, \Delta$; $\Gamma'' = \Gamma, \Delta[x{:=}D]$; $b = \mathrm{ord}_{\Gamma'}(B)$. $\exists P, B' =_{\beta\sigma} B$ s.t. $\Gamma' \vdash B' : P : *_{b+1}$. By Lemma 2.11 $\Gamma'' \vdash B'[x{:=}D] : P[x{:=}D] : *_{b+1}$. $B[x{:=}D] =_{\beta\sigma} B'[x{:=}D]$, so $b \geq \mathrm{ord}_{\Gamma''}(B[x{:=}D])$. □

Note here that $\mathrm{ord}_{\Gamma,x:A,\Delta}(B) = \mathrm{ord}_{\Gamma,\Delta[x:=D]}(B[x{:=}D])$ does not hold in general: take $\Gamma \equiv y{:}*_1$. Then $\Gamma, x{:}*_2 \vdash x{:}*_2$ and $\Gamma \vdash y{:}*_2$, and (by Lemma 2.32 below) $\mathrm{ord}_{\Gamma,x:*_2}(x) = 2$ and $\mathrm{ord}_\Gamma(x[x{:=}y]) = \mathrm{ord}_\Gamma(y) = 1$.

2c Evaluating the order of a Nuprl term

In this subsection, we attempt to provide a procedure that evaluates the order of almost any Nuprl term. We use the word almost because we are able to say how the order of almost all complex terms (like $A \times B$) is evaluated in term of the orders of the components (A and B). The only case that fails is that of an application. We cannot evaluate the order of AB precisely in terms of the orders of A and B. Rather, in the case of an application AB, we can only establish that the order of AB is $\leq$ the order of A.

We begin by evaluating the order of the first and second projections:

Lemma 2.27 (Order of Projections)
For a Γ*-term* $\langle A, B\rangle$, $\mathrm{ord}_\Gamma(\pi_1(\langle A, B\rangle)) = \mathrm{ord}_\Gamma(A)$ *and* $\mathrm{ord}_\Gamma(\pi_2(\langle A, B\rangle)) = \mathrm{ord}_\Gamma(B)$.

PROOF: This is a direct corollary of Lemma 2.22. □

The orders of constants and sorts are easy to calculate:

Lemma 2.28 (Orders of constants and sorts) *Let* Γ *be a legal context. Then* $\mathrm{ord}_\Gamma(*_a) = a + 1$, $\mathrm{ord}_\Gamma(\bot) = 1$, $\mathrm{ord}_\Gamma(\mathbb{Z}) = 1$, *and* $\mathrm{ord}_\Gamma(n) = 0$.

PROOF:

- As $\Gamma \vdash *_a : *_{a+1} : *_{a+2}$, $\mathrm{ord}_\Gamma(*_a) \leq a + 1$. Now assume $\Gamma \vdash A' : P : *_b$ for an $A' =_{\beta\sigma} *_a$ (hence $A' \twoheadrightarrow_{\beta\sigma} *_a$). By repeated Subject Reduction, $\Gamma \vdash *_a : P : *_b$. By Generation, $P =_{\beta\sigma} *_c$ for a $c > a$ (hence $P \twoheadrightarrow_{\beta\sigma} *_c$). By repeated Subject Reduction, $\Gamma \vdash *_c : *_b$, so again by Generation, $\exists d > c$ where $*_b =_{\beta\sigma} *_d$. Hence $d = b$, so $a < c < b$, so $b \geq a + 2$, so $\mathrm{ord}_\Gamma(*_a) \geq a + 1$.
- Notice that by the Start Lemma, $\Gamma \vdash \bot : *_1 : *_2$ so $\mathrm{ord}_\Gamma(\bot) \leq 1$. Now assume $\Gamma \vdash A' : P : *_1$ for an $A' =_{\beta\sigma} \bot$. Notice that $\bot$ is in normal form, so $A' \twoheadrightarrow_{\beta\sigma} \bot$ and by repeated Subject Reduction, $\Gamma \vdash \bot : P : *_1$. By the Generation Lemma, $P =_{\beta\sigma} *_1$, and as $*_1$ is in normal form, $P \twoheadrightarrow_{\beta\sigma} *_1$. By repeated Subject Reduction, $\Gamma \vdash *_1 : *_1$, which contradicts the fact that $\mathrm{ord}_\Gamma(*_1) = 2$.
- The proof for $\mathbb{Z}$ is similar to that for $\bot$.
- By the Start Lemma, $\Gamma \vdash n : \mathbb{Z} : *_1$, so $\mathrm{ord}_\Gamma(n) \leq 0$. $\mathrm{ord}_\Gamma(n) < 0$ is not possible. □

The following lemma and its corollaries are not only needed for evaluating the order of the remaining items, but they are also informative about the order of a term. This lemma says that for any Γ-type B, there is always B' of type $*_{\mathrm{ord}_\Gamma(B)}$ such that $B \twoheadrightarrow_{\beta\sigma} B'$. It also confirms that $*_a$ can be seen as the type of types (propositions) of order $\leq a$ (Corollary 2.30) and that a term is always of a lower order than its type (Corollary 2.31).

Lemma 2.29 (A type B reduces to a type B' of type $*_{\mathrm{ord}(B)}$)
Let B *be a* Γ*-type and* $b = \mathrm{ord}_\Gamma(B)$. $\exists B'$ *such that* $\Gamma \vdash B' : *_b$ *and* $B \twoheadrightarrow_{\beta\sigma} B'$.

PROOF: Assume $\Gamma \vdash B : *_p$. By Lemma 2.22, $\exists B'$ and P such that $\Gamma \vdash B' : P : *_{b+1}$ and $B \twoheadrightarrow_{\beta\sigma} B'$. By Weak Unicity of Types 2.18, $|P| =_{\beta\sigma} | *_p |$, say: $P =_{\beta\sigma} *_q$. Hence $P \twoheadrightarrow_{\beta\sigma} *_q$.

- By repeated Subject Reduction, $\Gamma \vdash *_q : *_{b+1} : *_{b+2}$. By Lemma 2.28, $b+1 \geq q+1$, so $b \geq q$.
- By the Conversion Rule, $\Gamma \vdash B' : *_q : *_{q+1}$, so by definition of b, $q \geq b$.

We find: $q = b$, so $P =_{\beta\sigma} *_b$, so $\Gamma \vdash B' : *_b$. □

Corollary 2.30 ($*_a$ is the type of types of order $\leq a$)
If P is a Γ-type in $\beta\sigma$-normal form, then $\Gamma \vdash P : *_a \Leftrightarrow \mathrm{ord}_\Gamma(P) \leq a$.

PROOF: Let $p = \mathrm{ord}_\Gamma(P)$. "$\Rightarrow$" is by definition of $\mathrm{ord}_\Gamma(P)$; for "$\Leftarrow$", by Lemma 2.29, $\exists P'$ where $\Gamma \vdash P' : *_p$ and $P \twoheadrightarrow_{\beta\sigma} P'$. As P is in normal form, $P' \equiv P$, so $\Gamma \vdash P : *_p$. Since $p \leq a$, repeated use of $(\subseteq)$ derives $\Gamma \vdash P : *_a$. □

Corollary 2.31 (A term is of a lower order than its type)
If $\Gamma \vdash A : B$ then $\mathrm{ord}_\Gamma(A) < \mathrm{ord}_\Gamma(B)$.

PROOF: Let $a = \mathrm{ord}_\Gamma(A)$, $b = \mathrm{ord}_\Gamma(B)$. B is a type, so by Lemma 2.29, $\exists B'$ where $\Gamma \vdash B' : *_b$ and $B \twoheadrightarrow_{\beta\sigma} B'$. $\Gamma \vdash A : B$, so by conversion, $\Gamma \vdash A : B' : *_b$. By definition of a, $b \geq a+1$, so $b > a$. □

In the above corollary, $\mathrm{ord}_\Gamma(A) = \mathrm{ord}_\Gamma(B) - 1$ does not hold: take $\Gamma = \emptyset$, $A \equiv *_1$ and $B \equiv *_3$. This is as expected because, by the inclusion rule $(\subseteq)$, once A is of type $*_n$, it is of type $*_m$ for any $m \geq n$.

So far, we can calculate the order of projections (Lemma 2.27) and the order of sorts and constants (Lemma 2.28). Now, we present methods to calculate the order of almost all the other terms:

Lemma 2.32 *Let C be a Γ-term. The following holds:*

1. *If $C \equiv x$ where $x{:}A \in \Gamma$ then $\mathrm{ord}_\Gamma(x) = \mathrm{ord}_\Gamma(A) - 1$.*
2. *If $C \equiv \Pi x{:}A.B$ then $\mathrm{ord}_\Gamma(\Pi x{:}A.B) = \max(\mathrm{ord}_\Gamma(A), \mathrm{ord}_{\Gamma,x:A}(B))$.*
3. *If $C \equiv \lambda x{:}A.b$ then $\mathrm{ord}_\Gamma(\lambda x{:}A.b) = \max(\mathrm{ord}_\Gamma(A) - 1, \mathrm{ord}_{\Gamma,x:A}(b))$.*
4. *If $C \equiv A \times B$ or $C \equiv \langle A, B\rangle$ then $\mathrm{ord}_\Gamma(C) = \max(\mathrm{ord}_\Gamma(A), \mathrm{ord}_\Gamma(B))$.*

PROOF: 1: Let $m = \mathrm{ord}_\Gamma(x)$. ¿From Corollary 2.23, $\exists B$ with $\Gamma \vdash x : B : *_{m+1}$. As $m+1$ is minimal, $\mathrm{ord}_\Gamma(B) = m+1$. By the Generation Lemma, $A =_{\beta\sigma} B$. Hence, $\mathrm{ord}_\Gamma(A) = m+1$. Note that the case $A =_{\beta\sigma} *_n$, $P =_{\beta\sigma} *_p$ with $n < p$ does not hold as m is minimal.
2: Let $a = \mathrm{ord}_\Gamma(A)$, $b = \mathrm{ord}_{\Gamma,x:A}(B)$, and $p = \mathrm{ord}_\Gamma(\Pi x{:}A.B)$. By Lemma 2.29, as $\Pi x : A.B$ is a Γ-type, $\exists P$ with $\Gamma \vdash P : *_p$ and $\Pi x{:}A.B \twoheadrightarrow_{\beta\sigma} P$. P must be of the form $\Pi x{:}A_1.B_1$, where $A \twoheadrightarrow_{\beta\sigma} A_1$ and $B \twoheadrightarrow_{\beta\sigma} B_1$. By Lemmas 2.29 and 2.13, $\exists A_2$ and B_2 such that $\Gamma \vdash A_2 : *_a$, $\Gamma, x{:}A \vdash B_2 : *_b$, $A \twoheadrightarrow_{\beta\sigma} A_2$ and $B \twoheadrightarrow_{\beta\sigma} B_2$. By Church Rosser, A_1 and A_2 have a common reduct A_3; B_1 and B_2 have a common reduct B_3. By repeated Subject Reduction: $\Gamma \vdash A_3{:}*_a$; $\Gamma, x{:}A \vdash B_3{:}*_b$. As $A \twoheadrightarrow_{\beta\sigma} A_3$ and $B \twoheadrightarrow_{\beta\sigma} B_3$, Subject Reduction gives $\Gamma \vdash (\Pi x{:}A_3.B_3) : *_p$. Now, $p = \max(a, b)$ as follows:

- By Generation $\exists m \leq p$ with $\Gamma \vdash A_3 : *_m$ and $\Gamma, x{:}A_3 \vdash B_3 : *_m$. By Transitivity, $\Gamma, x{:}A \vdash B_3 : *_m$. Hence $a, b \leq m \leq p$.

- As $\Gamma \vdash A_3 : *_a$ and $\Gamma, x{:}A_3 \vdash B_3 : *_b$, so by repeated application of ($\subseteq$), $\Gamma \vdash A_3 : *_{\max(a,b)}$ and $\Gamma, x{:}A_3 \vdash B_3 : *_{\max(a,b)}$. By ($\Pi$-form), $\Gamma \vdash (\Pi x{:}A_3.B_3) : *_{\max(a,b)}$, and so $p \leq \max(a,b)$.

3: Let $a = \mathrm{ord}_\Gamma(A)$, $m = \mathrm{ord}_\Gamma(\lambda x{:}A.b)$, $n = \mathrm{ord}_{\Gamma,x:A}(b)$. By Lemma 2.22, $\exists P, Q$ where $\Gamma \vdash P : Q : *_{m+1}$ and $\lambda x{:}A.b \twoheadrightarrow_{\beta\sigma} P$. Observe that $P \equiv \lambda x{:}A'.b'$ for some A', b' with $A \twoheadrightarrow_{\beta\sigma} A'$ and $b \twoheadrightarrow_{\beta\sigma} b'$. By the Generation Lemma, $\exists B$ such that $\Gamma, x{:}A' \vdash b' : B$ and $Q =_{\beta\sigma} \Pi x{:}A'.B$. Now $m+1 = \mathrm{ord}_\Gamma(Q) = \mathrm{ord}_\Gamma(\Pi x{:}A'.B) = \mathrm{ord}_\Gamma(\Pi x{:}A.B) = \max(a, \mathrm{ord}_{\Gamma,x:A}(B))$ by 2 above. Now $m = \max(a-1, n)$ because $m+1 = \max(a, n+1)$ as is seen by the two cases:

- $m+1 = a$. By the Transitivity Lemma, $\Gamma, x{:}A \vdash b' : B$. By Corollary 2.31: $\mathrm{ord}_{\Gamma,x:A}(b') = n < \mathrm{ord}_{\Gamma,x:A}(B)$, so $m+1 = \max(a, n+1)$.
- $m+1 = \mathrm{ord}_{\Gamma,x:A}(B) > a$. $\exists B', b''$ with $\Gamma, x{:}A' \vdash b'' : B' : *_{n+1}$ and $b' \twoheadrightarrow_{\beta\sigma} b''$. By Transitivity, $\Gamma, x{:}A \vdash b'' : B' : *_{n+1}$. With the Π and λ rule: $\Gamma \vdash (\lambda x{:}A.b'') : (\Pi x{:}A.B') : *_{\max(a,n+1)}$. Hence, $\max(a, n+1) \geq m+1$, and as $a < m+1$, $n+1 \geq m+1$ and $n \geq m$. As $\Gamma, x{:}A \vdash b' : B$, $n < \mathrm{ord}_{\Gamma,x:A}(B) = m+1$. Hence $n = m$ and $m+1 = \max(a, n+1)$.

4: Case $C \equiv A \times B$ is similar to 2. Case $C \equiv \langle A, B \rangle$ is similar to 3. $\square$

As MN may be a redex, its order is harder to determine. We can, however, prove the following:

Lemma 2.33 (The order of an application)
If $\Gamma \vdash M : \Pi x{:}P.Q$ *and* $\Gamma \vdash N : P$ *then* $\mathrm{ord}_\Gamma(N), \mathrm{ord}_\Gamma(MN) \leq \mathrm{ord}_\Gamma(M)$.

PROOF: Let $m = \mathrm{ord}_\Gamma(M)$. $\exists M', R$ such that $\Gamma \vdash M' : R : *_{m+1}$ and $M \twoheadrightarrow_{\beta\sigma} M'$. By Subject Reduction, $\Gamma \vdash M' : \Pi x{:}P.Q$, so by Weak Unicity of Types, $|R| =_{\beta\sigma} |\Pi x{:}P.Q| \equiv \Pi x{:}P.|Q|$. By Church Rosser $\exists R'$ such that $R \twoheadrightarrow_{\beta\sigma} R'$ and $\Pi x{:}P.|Q| \twoheadrightarrow_{\beta\sigma} |R'|$. Also, R' must be of the form $\Pi x{:}P'.Q'$, where $P \twoheadrightarrow_{\beta\sigma} P'$ and $|Q| \twoheadrightarrow_{\beta\sigma} |Q'|$. By Subject Reduction and Conversion, $\Gamma \vdash M' : (\Pi x{:}P'.Q') : *_{m+1}$. As m is minimal, $\mathrm{ord}_\Gamma(\Pi x{:}P'.Q') = m+1$. Now, $m = \mathrm{ord}_\Gamma(M) = \mathrm{ord}_\Gamma(\Pi x{:}P'.Q') - 1 = \max(\mathrm{ord}_\Gamma(P') - 1, \mathrm{ord}_{\Gamma,x:P'}(Q') - 1) \geq \mathrm{ord}_\Gamma(P') - 1 = \mathrm{ord}_\Gamma(P) - 1 \geq \mathrm{ord}_\Gamma(N)$. By conversion, $\Gamma \vdash N : P'$, so $\Gamma \vdash M'N : Q'[x{:=}N]$. As $MN =_{\beta\sigma} M'N$, we have $\mathrm{ord}_\Gamma(MN) = \mathrm{ord}_\Gamma(M'N) < \mathrm{ord}_\Gamma(Q'[x{:=}N]) \leq \mathrm{ord}_{\Gamma,x:P'}(Q') \leq \mathrm{ord}_\Gamma(\Pi x{:}P'.Q') = m+1$, so $\mathrm{ord}_\Gamma(MN) \leq m$. $\square$

This shows that a function can never take an argument of higher order, and that the order of a term can not increase when applying an argument to that term.

3 The Ramified Theory of Types RTT

In this section we give a short, formal description of Russell's Ramified Theory of Types (RTT). This formalisation is both faithful to Russell's original informal presentation and compatible with the present formulations of type theories. The basic aim of RTT is to exclude the logical paradoxes from logic by eliminating all self-references. An extended philosophical motivation for this theory can be found in [25], pages 38–55. We will not go into the full details of the formalisation of RTT (these details can be found in [14], the presentation by Russell himself in "Principia" is informal).

In Subsection 3a we introduce propositional functions. In Subsection 3b we assign types to some of these propositional functions. Paradoxical propositional functions are, of course, not typeable.

3a Propositional Functions

In this section we shall describe the set of propositions and propositional functions which Whitehead and Russell use in "Principia". We give a modernised, formal definition which corresponds to the description in "Principia". At the basis of the system of our formalization there is

- an infinite set $\mathcal{A}$ of *individual-symbols* and an infinite set $\mathcal{V}$ of *variables*;
- an infinite set $\mathcal{R}$ of *relation-symbols* together with an arity map $\mathfrak{a} : \mathcal{R} \to \mathbb{N}^+$.

0-ary relations are not explicitly used in "Principia" but could be added without problems. Since functions are relations in Principia, we will not introduce a special set of function symbols.

We assume that $\{\mathtt{a}_1, \mathtt{a}_2, \ldots\} \subseteq \mathcal{A}$; $\{\mathtt{x}, \mathtt{x}_1, \mathtt{x}_2, \ldots, \mathtt{y}, \mathtt{y}_1, \ldots, \mathtt{z}, \mathtt{z}_1, \ldots\} \subseteq \mathcal{V}$; and that $\{\mathtt{R}, \mathtt{R}_1, \ldots, \mathtt{S}, \mathtt{S}_1, \ldots\} \subseteq \mathcal{R}$. We will use the letters $x, y, z, x_1, \ldots$ as meta-variables over $\mathcal{V}$, and $R, R_1, \ldots$ as meta-variables over $\mathcal{R}$. Note that variables are written in `typewriter` style and that meta-variables are written in *italics*: $\mathtt{x}$ denotes one, *fixed* object in $\mathcal{V}$ whilst x denotes an *arbitrary* object of $\mathcal{V}$.

We assume that there is an *order* (e.g. alphabetical) on the collection $\mathcal{V}$, and write $x < y$ if the variable x is ordered before the variable y. In particular, we assume that

$$\mathtt{x} < \mathtt{x}_1 < \ldots < \mathtt{y} < \mathtt{y}_1 < \ldots < \mathtt{z} < \mathtt{z}_1 < \ldots$$

We also have the logical symbols $\wedge$, $\neg$ and $\forall$ in our alphabet, and the non-logical symbols: parentheses and the comma. Note that Russell used classical logic (intuitionistic logic wasn't widespread when "Principia" appeared) and hence he didn't need to make symbols like $\vee$, $\rightarrow$, $\exists$ primitive.

Definition 3.1 (Propositional functions)
We define a collection $\mathcal{F}$ of *propositional functions*, and for each element f of $\mathcal{F}$ we simultaneously define the collection $\textsc{fv}(f)$ of *free variables* of f:

1. If $R \in \mathcal{R}$ and $i_1, \ldots, i_{\mathfrak{a}(R)} \in \mathcal{A} \cup \mathcal{V}$ then $R(i_1, \ldots, i_{\mathfrak{a}(R)}) \in \mathcal{F}$.
   $\textsc{fv}(R(i_1, \ldots, i_{\mathfrak{a}(R)})) \stackrel{\text{def}}{=} \{i_1, \ldots, i_{\mathfrak{a}(R)}\} \cap \mathcal{V}$;
2. If $z \in \mathcal{V}$, $n \in \mathbb{N}$ and $k_1, \ldots, k_n \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{F}$, then $z(k_1, \ldots, k_n) \in \mathcal{F}$.
   $\textsc{fv}(z(k_1, \ldots, k_n)) \stackrel{\text{def}}{=} \{z, k_1, \ldots, k_n\} \cap \mathcal{V}$.
   If $n = 0$, we write $z()$ so as to distinguish the propositional function $z()$ from the variable z;[2]
3. If $f, g \in \mathcal{F}$ then $f \wedge g \in \mathcal{F}$ and $\neg f \in \mathcal{F}$. $\textsc{fv}(f \wedge g) \stackrel{\text{def}}{=} \textsc{fv}(f) \cup \textsc{fv}(g)$; $\textsc{fv}(\neg f) \stackrel{\text{def}}{=} \textsc{fv}(f)$;
4. If $f \in \mathcal{F}$ and $x \in \textsc{fv}(f)$ then $\forall x[f] \in \mathcal{F}$. $\textsc{fv}(\forall x[f]) = \textsc{fv}(f) \setminus \{x\}$.
5. All propositional functions can be constructed by using the rules 1, 2, 3 and 4 above.

We use the letters f, g, h as meta-variables over $\mathcal{F}$ and similar to Convention 2.2, we assume that bound variables differ from free ones and that different bound variables have different names.

A propositional function f is a proposition in which some parts (the free variables) have been left undetermined. It will turn into a proposition as soon as we

[2] A variable is not a propositional function. See [20], Chapter VIII: "The variable", p.94 of the 7th impression.

assign values to all its free variables. In this light, a *proposition* can be seen as a degenerated propositional function (with 0 free variables).

It will be clear now what the intuition behind propositional function of the form $R(i_1,\ldots,i_{\mathfrak{a}(R)})$, $f\wedge g$, $\neg f$ and $\forall x[f]$ is. The intuition behind propositional functions of the second kind is not so obvious. $z(k_1,\ldots,k_n)$ is a propositional function of *higher order: z is a variable for a propositional function with n free variables; the argument list $k_1,\ldots,k_n$ indicates what should be substituted*[3] *for these free variables as soon as one assigns such a propositional function to z.*

Notice that there are propositional functions of the form $z(k_1,\ldots,k_n)$ (where $z\in\mathcal{V}$) but that expressions of the form $f(k_1,\ldots,k_n)$, where $f\in\mathcal{F}$, are not propositional functions. Even substituting f for z in $z(k_1,\ldots,k_n)$ does not lead to $f(k_1,\ldots,k_n)$, as the notion of substitution in RTT is quite different from the usual notion of substitution in first order logic .

Example 3.2 Here are some higher-order propositional functions (pfs) from mathematics:
1. The pfs $\mathtt{z(x)}$ and $\mathtt{z(y)}$ in the definition of Leibniz-equality: $\forall\mathtt{z}[\mathtt{z(x)}\leftrightarrow\mathtt{z(y)}]$.
2. The pfs $\mathtt{z(0)}$, $\mathtt{z(x)}$ and $\mathtt{z(y)}$ in the formulation of complete induction:
$$[\mathtt{z(0)}\to(\forall\mathtt{x}\forall\mathtt{y}[\mathtt{z(x)}\to(\mathtt{S(x,y)}\to\mathtt{z(y)})])]\to\forall\mathtt{x}[\mathtt{z(x)}].$$
3. The pf $\mathtt{z()}$ in the formulation of the law of the excluded middle: $\forall\mathtt{z}[\mathtt{z()}\vee\neg\mathtt{z()}]$.

3b Ramified Types

Not all propositional functions should be allowed in our language. For instance, the expression $\neg x(x)$ is a perfectly legal element of $\mathcal{F}$, nevertheless, it is the propositional function that makes it possible to derive the Russell Paradox. Therefore, types are introduced.

Definition 3.3 (Ramified Types)
The ramified types $\mathcal{T}$ are defined inductively as follows:

1. ι^0 is a ramified type ($\mathtt{0}$ is called the *order* of this type);
2. If $t_1,\ldots,t_n$ are ramified types of orders $a_1,\ldots,a_n$ respectively, and $a>\max(a_1,\ldots,a_n)$, then $(t_1,\ldots,t_n)^a$ is a ramified type of order a (if $n=0$ then take $a\geq 1$);
3. All ramified types can be constructed using the rules 1 and 2.

ι^0 is the type of individuals, and $(t_1,\ldots,t_n)^a$ is the type of the propositional functions with n free variables, say $x_1,\ldots,x_n$, such that if we assign values k_1 of type t_1 to x_1, …, k_n of type t_n to x_n, then we obtain a proposition. The type $()^a$ is the type of propositions of order a.

Russell strictly divides his propositional functions in orders. For instance, both $\forall\mathtt{p}[\mathtt{p()}\wedge\neg\mathtt{p()}]$ and $\mathtt{R(a)}$ are propositions, but of different level: The first presumes a full collection of propositions, hence it cannot belong to the same collection of propositions as the propositions $\mathtt{p}$ over which it quantifies (among which $\mathtt{R(a)}$). This led Russell to make $\forall\mathtt{p}[\mathtt{p()}\wedge\neg\mathtt{p()}]$ belong to a type of a *higher* order (level) than the order of $\mathtt{R(a)}$. This can already be seen in the definition of ramified types: $(t_1,\ldots,t_n)^a$ can only be a type if a is strictly greater than each of the orders of the t_is.

[3] In Principia, it is not clear *how* such substitutions are carried out. One must depend on intuition and on how substitution is *used* in the Principia. It is quite hard and elaborate to give a proper definition of substitution.

Definition 3.4 Let $x_1, \ldots, x_n$ be a list of distinct variables, and $t_1, \ldots, t_n$ be a list of ramified types. We call $x_1{:}t_1, \ldots, x_n{:}t_n$ a *context* and call $\{x_1, \ldots, x_n\}$ its *domain.*
We write $\Gamma \vdash f : t$ to express that $f \in \mathcal{F}$ has type t in context Γ, and extend the variable convention to contexts: If x is bound in f, then x does not occur in the domain of Γ.

We use Γ, Δ to range over contexts and $t_1, t_2, \ldots$ to range over types. To avoid confusion we sometimes write $\vdash_{\mathrm{N}}$ for derivability in the Nuprl type system, and $\vdash_{\mathrm{R}}$ for derivability in RTT.

We now present the typing rules for RTT. These rules are derived from and equivalent to the rules in [14], which are as close as possible to Russell's original ideas. We change our notation for propositional functions slightly: Instead of $\forall x[f]$ we write $\forall x{:}t[f]$, where t is some ramified type.

Definition 3.5 (Typing Rules for RTT)

- If $c \in \mathcal{A}$, then $\Gamma \vdash c : \iota^0$ for any context Γ;
- If $f \in \mathcal{F}$, and $x_1 < \ldots < x_n$ are the free variables of f, and $t_1, \ldots, t_n$ are types such that $x_i{:}t_i \in \Gamma$, then $\Gamma \vdash f : (t_1, \ldots, t_n)^a$ if and only if
  - If $f \equiv R(i_1, \ldots, i_{\mathfrak{a}(R)})$ then $t_i = \iota^0$ for all i, and $a = 1$;
  - If $f \equiv z(k_1, \ldots, k_m)$ then there are $u_1, \ldots, u_m$ such that $z{:}(u_1, \ldots, u_m)^{a-1} \in \Gamma$, and $\Gamma \vdash k_i{:}u_i$ for all $k_i \in \mathcal{A} \cup \mathcal{F}$, and $k_i{:}u_i \in \Gamma$ for all $k_i \in \mathcal{V}$;
  - If $f \equiv f_1 \wedge f_2$ then there are $u_1^{a_1}, u_2^{a_2}$ such that $\Gamma \vdash f_i : u_i^{a_i}$ and $a = \max(a_1, a_2)$;
    if $f \equiv \neg f'$ then $\Gamma \vdash f' : (t_1, \ldots, t_n)^a$.
  - If $f \equiv \forall x{:}t_0[f']$ then $\exists j$ where $\Gamma, x{:}t_0 \vdash f' : (t_1, \ldots, t_{j-1}, t_0, t_j, \ldots, t_n)^a$.

Example 3.6 $\neg\mathtt{x}(\mathtt{x})$ is not typeable in any context Γ. If $\Gamma \vdash \neg\mathtt{x}(\mathtt{x}) : t$ then t must be of the form $(u)^a$, with $\mathtt{x}{:}u \in \Gamma$, as $\neg\mathtt{x}(\mathtt{x})$ has one free variable. Hence $\Gamma \vdash \mathtt{x}(\mathtt{x}) : (u)^a$, and by Unicity of Types below, $u \equiv (u')^{a-1}$, with $\mathtt{x} : u' \in \Gamma$. As Γ is a context, $u \equiv u'$, hence $u \equiv (u)^{a-1}$. Absurd.

An important result (whose proof follows directly from the definition of $\Gamma \vdash f : t$) is the following:

Theorem 3.7 (Unicity of Types) *If* $\Gamma \vdash f : t$ *and* $\Gamma \vdash f : u$ *then* $t \equiv u$.

4 RTT in Nuprl

We present a straightforward embedding of RTT in the type theory of Nuprl written as a PTS (Section 2). The embedding will consist of two parts: First we give a representation of the ramified types in Nuprl (Subsection 4a), then we represent the typable propositional functions in Nuprl (Subsection 4b).

4a Ramified Types in Nuprl

The main clue to our embedding is the interpretation of $*_n$ as the sort containing all order-n-propositions. There is a small difference in that Nuprl considers any term of type $*_n$ to be of type $*_{n+1}$ as well. This means that any proposition of order n can be interpreted as a proposition of order $n+1$ as well. This inclusion is not a feature of RTT; yet it isn't a serious extension.

Another small point is that Russell doesn't specify his underlying set of "individuals" and that we want to use $\mathbb{Z}$ as translation of this underlying set. Therefore, we will assume that the set $\mathcal{A}$ of RTT-individuals is equal to the set $\mathbb{Z}$ of integers. Recall that, when $x \notin \mathrm{FV}(B)$, we write $\Pi x : A.B$ as $A \to B$.

Definition 4.1 Define a mapping $T : \mathcal{T} \to \mathbb{T}$ as follows:

$$T(\iota^0) \stackrel{\mathrm{def}}{=} \mathbb{Z} \text{ and } T((t_1^{a_1}, \ldots, t_n^{a_n})^a) \stackrel{\mathrm{def}}{=} T(t_1^{a_1}) \to \ldots T(t_n^{a_n}) \to *_a$$

Note that $T(()^a) = *_a$ and T does indeed interpret the type of order-a-propositions as $*_a$. Moreover, translations of ramified types are typable in Nuprl:

Lemma 4.2 *If t^a is a ramified type of order a then* $\vdash_{\mathrm{N}} T(t^a) : *_{a+1}$.

PROOF: Induction on the construction of ramified types. □

When we speak of a ramified type t^a of order a, we actually mean that the terms that are of type t^a have order a. $T(t^a)$ itself should, therefore, have order $a+1$ in Nuprl. Indeed, we can prove:

Lemma 4.3 *If Γ is a legal context then* $\mathrm{ord}_\Gamma(T(t^a)) = a + 1$.

PROOF: Induction on ramified types. $T(\iota^0) = \mathbb{Z}$ and $\mathrm{ord}_\Gamma(\mathbb{Z}) = 1$ by Lemma 2.28. Now assume $\mathrm{ord}_\Gamma(T(t_i^{a_i})) = a_i + 1$ for $i = 1, \ldots, n$. Notice that

$$\begin{array}{lcl}
\mathrm{ord}_\Gamma(T((t_1^{a_1}, \ldots, t_n^{a_n})^a)) & = & \mathrm{ord}_\Gamma(T(t_1^{a_1}) \to \ldots \to T(t_n^{a_n}) \to *_a) \\
& \stackrel{2.32}{=} & \max(\mathrm{ord}_\Gamma(T(t_1^{a_1})), \ldots, \mathrm{ord}_\Gamma(T(t_n^{a_n})), \mathrm{ord}_\Gamma(*_a)) \\
& \stackrel{2.28,\ \mathrm{IH}}{=} & \max(a_1 + 1, \ldots, a_n + 1, a + 1) \stackrel{a \geq a_i}{=} a + 1 \square
\end{array}$$

4b Propositional Functions of RTT in Nuprl

We extend the mapping T of Definition 4.1 so that a propositional function with free variables $x_1 < \ldots < x_n$ will be translated into a λ-term of the form $\lambda x_1{:}t_1 \cdots x_n{:}t_n.A$, where A itself is not of the form $\lambda x{:}t.A'$. For notational convenience, T is extended to $\mathcal{A}$ and $\mathcal{V}$ as well.

Definition 4.4 Let Γ be a RTT-context. We extend T to the sets $\mathcal{A}$, $\mathcal{V}$ and $\mathcal{F}$. If $i \in \mathcal{A} \cup \mathcal{V}$ then $T(i) \stackrel{\mathrm{def}}{=} i$. Now let $f \in \mathcal{F}$ and assume f has free variables $x_1 < \ldots < x_n$, such that $x_i{:}t_i \in \Gamma$.

- If $f = R(i_1, \ldots, i_{\mathfrak{a}(R)})$ then $T(f) \stackrel{\mathrm{def}}{=} \lambda x_1{:}T(t_1) \cdots x_n{:}T(t_n).Ri_1 \cdots i_{\mathfrak{a}(R)}$
- If $f = z(k_1, \ldots, k_m)$ then $T(f) \stackrel{\mathrm{def}}{=} \lambda x_1{:}T(t_1) \cdots x_n{:}T(t_n).zT(k_1) \cdots T(k_m)$;
- If $f = g_1 \wedge g_2$, and g_i has free variables $y_{i1} < \ldots < y_{im_i}$, then $T(g_i) \equiv \lambda y_{i1}{:}u_{i1} \cdots y_{im_i}{:}u_{im_i}.G_i$ for some term G_i.

  Let $T(f) \stackrel{\mathrm{def}}{=} \lambda x_1{:}T(t_1) \cdots x_n{:}T(t_n).G_1 \times G_2$.
- If $f = \neg g$, then $T(g) \equiv \lambda x_1{:}T(t_1) \cdots x_n{:}T(t_n).G$ for some term G.
  Let $T(f) \stackrel{\mathrm{def}}{=} \lambda x_1{:}T(t_1) \cdots x_n{:}T(t_n).G \to \perp$.
- If $f = \forall x{:}t.g$ then

  $$T(g) \equiv \lambda x_1{:}T(t_1) \cdots x_i{:}T(t_i).x{:}T(t).x_{i+1}{:}T(t_{i+1}) \cdots x_n{:}T(t_n).G$$

  for some term G. Let $T(f) \stackrel{\mathrm{def}}{=} \lambda x_1{:}T(t_1) \cdots x_n{:}T(t_n).\Pi x{:}T(t).G$.

The extension of T as defined above also depends on the context Γ. Normally it will be clear which context Γ is meant. If confusion arises, we write T_Γ to indicate the context in question.

It is important to notice that, for propositions f, $T(f)$ is exactly the interpretation of f provided by the Curry-Howard-de Bruijn isomorphism.

Finally, we define a special Nuprl-context Γ_0 which contains information on the relation and individual symbols of RTT by: $\Gamma_0 \stackrel{\text{def}}{=} \{R : \overbrace{\mathbb{Z} \to \ldots \to \mathbb{Z}}^{\mathfrak{a}(R) \text{ times } \mathbb{Z}} \to *_1 \mid R \in \mathcal{R}\}$.

We assume $\mathcal{R}$ to be finite for the moment, so that Γ_0 is finite as well, and therefore is a Nuprl-context. Γ_0 is legal, as we have $\vdash_{\mathrm{N}} \mathbb{Z} \to \ldots \to \mathbb{Z} \to *_1 : *_2$.

The following theorem states that the embedding T respects the type structure of RTT. This means that we can see Nuprl as an extension of the Ramified Theory of Types.

Theorem 4.5 (Nuprl extends RTT) *If $\Gamma \vdash_{\mathrm{R}} f : t$ then $\Gamma_0 \vdash_{\mathrm{N}} T(f) : T(t)$.*

PROOF: Induction on the definition of $\Gamma \vdash_{\mathrm{R}} f : t$. If $\Gamma \vdash c : \iota^0$ because $c \in \mathbb{Z}$ then $c{:}\mathbb{Z} \in \Gamma_0$, so $\Gamma_0 \vdash c : \mathbb{Z}$. Now assume $f \in \mathcal{F}$, f has free variables $x_1 < \ldots < x_n$, and $t_1, \ldots, t_n$ where $x_i{:}t_i \in \Gamma$ for $i = 1, \ldots, n$, and $\Gamma \vdash_{\mathrm{R}} f : (t_1, \ldots, t_n)^a$. By Lemma 4.2, $\vdash_{\mathrm{N}} T(t_i) : *_{a_i}$ for some a_i. Hence, by the Start and Weakening rules, we add $x_i{:}T(t_i)$ one by one to the context Γ_0, obtaining a legal context $\Gamma_1 = \Gamma_0, x_1{:}T(t_1), \ldots, x_n{:}T(t_n)$. We only treat the case $f = \forall x{:}t_0[g]$:

If $f = \forall x{:}t_0[g]$ then $\exists j$ such that $\Gamma, x{:}t_0 \vdash_{\mathrm{R}} g : (t_1, \ldots, t_{j-1}, t_0, t_j, \ldots, t_n)^a$. By the induction hypothesis, $\Gamma_0 \vdash T(g) : T(t_1) \to \cdots \to T(t_{j-1}) \to T(t_0) \to T(t_j) \to \cdots \to T(t_n) \to *_a$. By the Generation Lemma,
$\Gamma_0, x_1{:}T(t_1), \ldots, x_{j-1}{:}T(t_{j-1}), x{:}T(t_0), x_j{:}T(t_j), \ldots, x_n{:}T(t_n) \vdash_{\mathrm{N}} G : *_a$ where $g \equiv \lambda x_1 \cdots x_{j-1} x x_j \cdots x_n.G$. As the types of the variables in the context are independent from each other, we also have $\Gamma_1, x{:}T(t_0) \vdash_{\mathrm{N}} G : *_a$. As the order of type t_0 is smaller than a, we have $\Gamma_1 \vdash_{\mathrm{N}} T(t_0) : *_a$ (Lemma 4.2), so by (Π-form): $\Gamma_1 \vdash_{\mathrm{N}} \Pi x{:}T(t_0).G : *_a$. By λ-abstracting over all the variables in FV(f) we obtain $\Gamma_0 \vdash_{\mathrm{N}} T(f) : T(t)$. $\Box$

It would be nice if we could also prove a kind of opposite of Theorem 4.5. However, the statement "*If $\Gamma_0 \vdash_{\mathrm{N}} T(f) : T(t)$ then there is a context Γ such that $\Gamma \vdash_{\mathrm{R}} f : t$*" is not true. We can derive $\Gamma_0 \vdash_{\mathrm{N}} T(\forall x{:}\iota^0[R(x)]) : *_n$ for any $n \geq 1$. Nevertheless, we have $\Gamma \vdash_{\mathrm{R}} \forall x{:}\iota^0[R(x)] : ()^1$ for all RTT-contexts Γ, so by Unicity of Types 3.7 it is impossible that $\Gamma \vdash_{\mathrm{R}} \forall x{:}\iota^0[R(x)] : ()^n$ for any $n > 1$. It is clear that this difference between RTT and Nuprl is caused by the type inclusion rule $\subseteq$, which is only present in Nuprl, and not in RTT. We do have a partial result, however:

Lemma 4.6 *If $\Gamma \vdash_{\mathrm{R}} f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$ and $x_1 < \ldots < x_n$ are the free variables of f, then $\mathrm{ord}_{\Gamma_0}(T(\forall x_1{:}t_1^{a_1}. \cdots \forall x_n{:}t_n^{a_n}[f])) = a$.*

PROOF: Induction on the definition of $\Gamma \vdash_{\mathrm{R}} f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$. Note that $x_i{:}t_i^{a_i} \in \Gamma$ for all i, and $\Gamma \vdash_{\mathrm{R}} \forall x_1{:}t_1^{a_1} \cdots \forall x_n{:}t_n^{a_n}[f] : ()^a$. Let $\Gamma_i \equiv \Gamma_0, x_1{:}T(t_1^{a_1}), \ldots, x_i{:}T(t_i^{a_i})$. We only treat the case $f \equiv z(k_1, \ldots, k_m)$; the other cases are similar. $z \in$ FV(f), say: $z \equiv x_p$. As $x_p{:}t_p^{a_p} \in \Gamma$, $a_p = a - 1$. Hence $\mathrm{ord}_{\Gamma_n}(z) = \mathrm{ord}_{\Gamma_n}(T(t_p^{a_p})) - 1 = a - 1$. By 2.33, $\mathrm{ord}_{\Gamma_n}(zT(k_1) \cdots T(k_m)) \leq a - 1$. Hence

$$\begin{array}{lr}
\mathrm{ord}_{\Gamma_0}(T(\forall x_1{:}t_1^{a_1} \cdots \forall x_n{:}t_n^{a_n}[f])) & = \\
\mathrm{ord}_{\Gamma_0}(\Pi x_1{:}T(t_1^{a_1}). \cdots \Pi x_n{:}T(t_n^{a_n}).zT(k_1) \cdots T(k_m)) & = \\
\max(\mathrm{ord}_{\Gamma_n}(zT(k_1) \cdots T(k_m)), \max_{i \leq n}(\mathrm{ord}_{\Gamma_i}(T(t_i^{a_i}))) & = \\
\max(\mathrm{ord}_{\Gamma_n}(zT(k_1) \cdots T(k_m)), \max_{i \leq n}(a_i + 1)) = a_p + 1 = a & \Box
\end{array}$$

Corollary 4.7 *If $\Gamma \vdash_{\mathrm{R}} f : ()^a$ then $\mathrm{ord}_{\Gamma_0}(T(f)) = a$.*

5 Conclusions

In this paper we focus on Nuprl and describe a fragment of it as a Pure Type System λN. A type universe $\mathbb{U}_n$ $(n \geq 1)$ of Nuprl contains certain basis types, and is closed under the construction of dependent product types and cartesian products. Moreover, $\mathbb{U}_n$ is an element of $\mathbb{U}_{n+1}$, and all types in $\mathbb{U}_n$ also belong to $\mathbb{U}_{n+1}$. We represent the type universe $\mathbb{U}_n$ by the PTS sort $*_n$. Closure under the construction of dependent products is given by rule $(*_n, *_n)$, and the fact that $\mathbb{U}_n$ is element of $\mathbb{U}_{n+1}$ is represented by the PTS axiom $*_n : *_{n+1}$. We extend this PTS as follows:

- For cartesian products, we introduce the rule $\dfrac{\Gamma \vdash A_1 : *_n \qquad \Gamma \vdash A_2 : *_n}{\Gamma \vdash A_1 \times A_2 : *_n}$
  Canonical inhabitants of $A_1 \times A_2$ are terms of the form $\langle a_1, a_2 \rangle$, where $a_i : A_i$.
- We also introduce the projection functions π_i: $\dfrac{\Gamma \vdash a : A_1 \times A_2}{\Gamma \vdash \pi_i(a) : A_i}$
  together with a reduction relation generated by the axiom $\pi_i(\langle a_1, a_2 \rangle) \to_\sigma a_i$.
- As $\mathbb{U}_n \subseteq \mathbb{U}_{n+1}$, we introduce an inclusion rule $(\subseteq)$: $\dfrac{\Gamma \vdash A : *_n}{\Gamma \vdash A : *_{n+1}}$

A type universe $\mathbb{U}_n$ in Nuprl is closed under the construction of *dependent* cartesian products, but as we do not need dependent cartesian products in the paper, we don't introduce them.

The system λN thus obtained has many properties of usual PTSs, like Church-Rosser (for $\to_{\beta\sigma}$), Subject Reduction and Correctness of Types. With rule $(\subseteq)$, we lose Unicity of Types, but we can prove a weakened version of it.

Let Γ be a context for λN. Due to correctness of types, for each Γ-type A there is $n \geq 1$ such that $\Gamma \vdash A : *_n$. (compare this to Nuprl: each type in Nuprl belongs to some type universe $\mathbb{U}_n$). We call the smallest n for which $\Gamma \vdash A : *_n$ the *order* of A (in Γ), notation $\mathrm{ord}_\Gamma(A)$. We generalize this definition to arbitrary Γ-terms A: $\mathrm{ord}_\Gamma(A)$ is the minimal n for which there is B such that $\Gamma \vdash A : B : *_n$. We prove some elementary properties of $\mathrm{ord}_\Gamma(A)$:

- $\mathrm{ord}_\Gamma(A) = \mathrm{ord}_\Delta(A)$ if Δ is legal and $\Delta \supseteq \Gamma$;
- $\mathrm{ord}_\Gamma(*_n) = n + 1$;
- If $\Gamma \vdash A : B$ then $\mathrm{ord}_\Gamma(A) < \mathrm{ord}_\Gamma(B)$;
- If $x{:}A \in \Gamma$ then $\mathrm{ord}_\Gamma(x) = \mathrm{ord}_\Gamma(A) - 1$;
- $\mathrm{ord}_\Gamma(\Pi x{:}A.B) = \max(\mathrm{ord}_\Gamma(A), \mathrm{ord}_{\Gamma,x:A}(B))$;
- $\mathrm{ord}_\Gamma(\lambda x{:}A.b) = \max(\mathrm{ord}_\Gamma(A) - 1, \mathrm{ord}_{\Gamma,x:A}(b))$;
- $\mathrm{ord}_\Gamma(\langle A_1, A_2 \rangle) = \mathrm{ord}_\Gamma(A_1 \times A_2) = \max(\mathrm{ord}_\Gamma(A_1), \mathrm{ord}_\Gamma(A_2))$.

We show that the orders in λN (and thus the type universes in Nuprl) are closely related to orders in RTT by looking at translations of RTT propositions to λN types via a propositions-as-types embedding T: We prove that if f is an order-n proposition in RTT, then $\mathrm{ord}_{\Gamma_0}(T(f)) = n$. Here, Γ_0 is some basic context that contains only some type information of the relation symbols that are used in RTT. We conclude that our formulation of Nuprl as a PTS is faithful to the idea behind universes in Martin-Löf's type theory and our definition of order on Nuprl terms captures the hierarchy of universes in Nuprl and provides an elegant comparison between Nuprl and RTT. As a bonus, we get a description of RTT in a propositions-as-types style in which the notion of order is maintained.

There are more similarities between RTT and Nuprl. Both Nuprl and RTT have a kind of higher order substitution (see Chapter 5 of [10] and Section 3 of [14]). We are currently investigating the similarities between both notions of substitution.

Now we stop to explain the philosophy of our approach and the novelty of what we have provided. We also discuss future research that might be sparkled by our paper.

At the beginning of this century, the paradoxes led to many new formulations of logical systems and an amazing variety of ideas and approaches. Later on, some of these ideas where abandoned when they shouldn't have. Even more, some of the ideas proposed were found later to contribute nothing to the solution of the paradoxes. For example, even though ZF set theory uses the foundation axiom, it is quite clear now that it is the separation rather than the foundation axiom which was responsible for the avoidance of the paradoxes.

Our standpoint in this paper is not to defend one line against another. Rather, we aim to clarify the different notions and philosophies assumed in the foundation of logic. In this paper, our chosen notion is that of Russell's orders as found in the famous Ramified Theory of Types RTT. Russell, whose contribution to modern logic is historical, avoided the paradox (that he himself discovered) by adopting two layers: types and orders. Later it was found that orders contributed nothing to the avoidance of the paradox and Ramsey's work led to the abandonment of Russell's orders. It is not clear to us whether Russell did actually know that orders do not contribute to the avoidance of the paradox. We believe however that his intuition of using orders (as well as types) is a solid one and we have seen this intuition being repeated in many predicative styles logics. In [11], we show that Russell's orders come back in Kripke's account of levels of truths. In this paper, we show that Russell's orders are present in Martin-Löf's type theory and the proof checker Nuprl. Of course the word "orders" is not used by Kripke, Martin-Löf and Constable. Our study however shows that formally representing (with orders) the informal hierarchies of these systems is informative about these hierarchies, about the systems themselves and about the philosophies behind them.

Not only does our paper revive the "order" concept, and show its usefulness for explaining basic hierarchies and philosophies in modern systems, but also, our paper places the historical system underlying Principia Mathematica in a context with a modern system of computer mathematics (Nuprl) and modern type theories (Martin-Löf's type theory and PTSs). Our main results concerning the relationship between these various systems can be summarised as follows (we take $\vdash_{\mathrm{R}}$ (resp. $\vdash_{\mathrm{N}}$) to stand for type derivation in RTT (resp. in Nuprl), and assume a translation T from types and functions in RTT into Nuprl; also Γ_0 is a basic Nuprl-context which contains information on the relation and individual symbols of RTT):

1. The system (underlying) Nuprl can be seen as a simple extension of a PTS.
2. RTT can be embedded in Nuprl.
3. Hence RTT can be regarded as a PTS.
4. Nuprl extends RTT in the sense that if $\Gamma \vdash_{\mathrm{R}} f : t$ then $\Gamma_0 \vdash_{\mathrm{N}} T(f) : T(t)$.

A number of questions on extending these results remain open. These questions are as follows:

1. Since Martin-Löf's type theory, Nuprl and RTT have as aim to be a foundation of mathematics, one should have an interpretation of the most basic systems of logic: predicate logic (Pred) in RTT. This would be nice and the advantages of relating RTT, PTSs and Nuprl would carry over to Pred as well. Moreover, one would get the following picture: Pred $<$ RTT $<$ Nuprl $<$ PTSs.
2. We have shown that Nuprl extends RTT (see 4 above). It would be nice to answer whether Nuprl is a conservative extension of RTT.

Questions 1 and 2 are very interesting and must be the subject of future research. We have thought about them and up to this stage, no clear answer has been found. Question 1 causes difficulties precisely because Russell's notion of substitution is different from substitution as is used in modern logic and type theory. We have come a long way at formalising in modern style Russell's ideas and theory. There is still work to be done in this field and we believe that this work might prove very useful to modern computer science. It may be the case for example that parallel computation may well benefit from Russell's substitution. These are issues we are investigating at the moment.

Question 2 has been partially attempted in the paper. We have said that the converse of Theorem 4.5 does not hold. We have given as a reason for this the inclusion rule ($\subseteq$) which is only present in Nuprl and not in RTT. As shown in the paper, RTT enjoys the unicity of types property whereas Nuprl does not. Here we explain intuitively this problem caused by the difference between Nuprl and RTT and give our opinion of how future directions in establishing a form of conservativity must be followed.

We know from the fact that Nuprl extends RTT that $\Gamma \vdash_{\mathrm{R}} f : t$ then $\Gamma_0 \vdash_{\mathrm{N}} T(f) : T(t)$. Now, let us take this example:

$\Gamma \vdash_{\mathrm{R}} \forall x{:}\iota^0[R(x)] : ()^1 \Rightarrow$

$\Gamma_0 \vdash_{\mathrm{N}} T(\forall x{:}\iota^0[R(x)]) : *_1 \Rightarrow^{(\subseteq)}$

$\Gamma_0 \vdash_{\mathrm{N}} T(\forall x{:}\iota^0[R(x)]) : *_n \equiv T(()^n)$ for any $n \geq 1 \not\Rightarrow^{\text{unicity of types in RTT}}$

$\Gamma \vdash_{\mathrm{R}} \forall x{:}\iota^0[R(x)] : ()^n$ for any $n > 1$.

This means that we cannot go back from Nuprl to RTT.

We can however do something about that. The idea is to establish the order of the Nuprl term A and to only go in the opposite direction of Theorem 4.5 when the type of A is $*_a$ and a is the order of A. Hence in our example above, as 1 is the order of $T(\forall x{:}\iota^0[R(x)])$, we can only go back with $\Gamma_0 \vdash_{\mathrm{N}} T(\forall x{:}\iota^0[R(x)]) : *_1$ obtaining the valid typing $\Gamma \vdash_{\mathrm{R}} \forall x{:}\iota^0[R(x)] : ()^1$.

We have provided a partial result related to this question (given by Lemma 4.6 and Corollary 4.7) which says that for any Russell typable propositional function f of order a, we can establish that its Nuprl order is also a and hence when we try and mimick the Nuprl typing in RTT, we should only restrict ourselves to doing this when the Nuprl type is $*_a$ and a is the order of the Nuprl term avoiding the inclusion rule as much as possible. This is already a powerful result. Of course, it remains that we fully work out a translation from Nuprl to RTT and show in what way it can be said that RTT extends Nuprl. This will involve a huge technicality concerning RTT's substitution and free variables. It is left as a subject for future research.

References

[1] H.P. Barendregt. Lambda calculi with types. In S. Abramsky, Dov Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science 2: Background: Computational Structures*, 117–309. OUP, 1992.

[2] A. Church. A formulation of the simple theory of types. *The Journal of Symbolic Logic*, 5:56–68, 1940.

[3] R.L. Constable et al. *Implementing Maths with the Nuprl Proof Development System.* Prentice-Hall, 1986.

[4] T. Coquand and G. Huet. The calculus of constructions. *Information and Computation*, 76:95–120, 1988.

[5] G. Dowek et al. The Coq proof assistant version 5.6, Users guide. Rapport de recherche 134, INRIA, 1991.

[6] G. Frege. *Grundgesetze der Arithmetik, begriffsschriftlich abgeleitet*, I + II. Pohle, Jena, 1892 and 1903.

[7] D. Hilbert and W. Ackermann. *Grundzüge der Theoretischen Logik.* Die Grundlehren der Mathematischen Wissenschaften in Einzeldarstellungen, Band XXVII. Springer Verlag, Berlin, first edition, 1928.

[8] W.A. Howard. The formulas-as-types notion of construction. In J.P. Seldin and J.R. Hindley, editors, *To H.B. Curry: Essays on Combinatory Logic, λ-Calculus and Formalism*, 479–490, 1980. Academic Press.

[9] P.B. Jackson. *Enhancing the Nuprl Proff Development System and Applying it to Computational Abstract Algebra.* PhD thesis, Cornell University, Ithaca, New York, 1995.

[10] P.B. Jackson. The Nuprl proof development system, Version 4.1 reference manual and user's guide. Cornell University, Department of Computing Science, Ithaca, New York., 1995.

[11] F. Kamareddine and T. Laan. A reflection on Russell's ramified types and Kripke's hierarchy of truths. *Journal of the Interest Group in Pure and Applied Logic 4(2)*, 1996.

[12] J.W. Klop. Term rewriting systems. In S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science 2: Background: Computational Structures*, pages 1–116. OUP, 1992.

[13] S. Kripke. Outline of a theory of truth. *Journal of Philosophy*, 72:690–716, 1975.

[14] T.D.L. Laan. A modern elaboration of the Ramified Theory of Types. *Studia Logica*, 57:243–278, 1996.

[15] P. Martin-Löf. An intuitionistic theory of types: predicative part. In H.E. Rose and J. Shepherdson, editors, *logic Colloquium '73.* North Holland, 1975.

[16] P. Martin-Löf. Constructive mathematics and computer programming. In *Sixth International Congress for Logic, Methodology and Philosophy of Science*, 153–175, Amsterdam, 1982. North-Holland.

[17] P. Martin-Löf. *Intuitionistic Type Theory.* Bibliopolis, 1984.

[18] R.P. Nederpelt, J.H. Geuvers, and R.C. de Vrijer, editors. *Selected Papers on Automath.* Studies in Logic and the Foundations of Mathematics **133**. North-Holland, Amsterdam, 1994.

[19] F.P. Ramsey. The foundations of mathematics. *Proc. of the London Mathematical Society*, 338–384, 1925.

[20] B. Russell. *The Principles of Mathematics.* Allen & Unwin, London, 1903.

[21] B. Russell. Mathematical logic as based on the theory of types. *American Journal of Mathematics*, 30, 1908.

[22] J. Terlouw. Een nadere bewijstheoretische analyse van GSTT's. Technical report, Department of Computer Science, University of Nijmegen, 1989.

[23] H. Weyl. *Das Kontinuum.* Veit, Leipzig, 1918. German; also in: Das Kontinuum und andere Monographien, Chelsea Pub.Comp., New York, 1960.

[24] A. Tarski. Der Wahrheitsbegriff in den formalisierten Sprachen. *Studia Philosophica*, 1:261–405, 1936. German translation by L. Blauwstein from the Polish original (1933) with a postscript added.

[25] A.N. Whitehead and B. Russell. *Principia Mathematica.* Cambridge University Press, 1910^1, 1927^2.