TWAN LAAN*
ROB NEDERPELT

A Modern Elaboration of the Ramified Theory of Types

Abstract. The paper first formalizes the ramified type theory as (informally) described in the Principia Mathematica [32]. This formalization is close to the ideas of the Principia, but also meets contemporary requirements on formality and accuracy, and therefore is a new supply to the known literature on the Principia (like [25], [19], [6] and [7]).
As an alternative, notions from the ramified type theory are expressed in a lambda calculus style. This situates the type system of Russell and Whitehead in a modern setting. Both formalizations are inspired by current developments in research on type theory and typed lambda calculus; see [3].

Key words: ramified type theory, typed lambda calculus, substitution

1. Introduction

In 1879, Gottlob Frege published the Begriffschrift [12], presenting logic as a formal system. In 1902, Russell wrote a letter to Frege ([18], pp. 124–125), claiming that the system of the Begriffschrift is inconsistent. Though Russell didn't make his claim exact (see the answer of Frege to Russell's letter in [18], pp. 126–128) and the inconsistency cannot be derived from the Begriffschrift, it *is* possible to deduce the claim from the system that is presented in Frege's Grundgesetze der Arithmetik [13, 14]. Russell's claim has become known under the name "Russell Paradox".
Though this paradox was certainly not the first or only one in the history of logic and mathematics (see [5], Chapter 17), Russell's paradox is regarded as the most fundamental one (cf. [11], pp. 1–7): it is a paradox that arises at an elementary level of logic.
Whitehead and Russell remark[1] that *"An analysis of the paradoxes to be avoided shows that they all result from a certain kind of vicious circle. The vicious circles in question arise from supposing that a collection of objects may contain members which can only be defined by means of the collection as a whole."*. They ban this kind of collections by postulating the "vicious circle principle":

*Supported by the Co-operation Centre Tilburg and Eindhoven Universities.
[1] [32], Introduction, Chapter II, Section I, p. 37.

Presented by **Wolfgang Rautenberg**; *Received* October 2, 1995

VICIOUS CIRCLE PRINCIPLE 1.1. *Whatever involves* all *of a collection must not be one of the collection.*

Whitehead and Russell show that the Russell paradox, and some other paradoxes, can not be derived if one accepts the vicious circle principle[2].

The vicious circle principle is implemented in [27] (1908) by the use of types, in particular the so-called *ramified* types. This theory was elaborated in the famous Principia Mathematica [32] (1910-1912).

The basic aim of the Ramified Theory of Types is to exclude the logical paradoxes by eliminating all self-references. An extended philosophical motivation for this theory can be found in the Principia [32], pages 38–55. In the Principia, Russell and Whitehead founded mathematics on logic, as far as possible.

Though this build-up of mathematics and logic in the Principia is formal and accurate, the type system is not presented in a formal way.[3] There is neither a formal definition of "propositional function", nor a formal definition of "type" in the Principia. Moreover, Russell's notion of substitution is left very vague. Several formal type systems roughly describe the type theory underlying the Principia (see for instance [6] and [7]), but there is no description of the ramified type theory in the current literature that is faithful to the original ideas in the Principia *and* formal. In this paper we fill up this gap in the literature. As a result, we

- Are able to compare Russell's original type theory with more modern type systems (like Pure Type Systems, see [3], [29], or the type theory on which a proof checker like Nuprl is based, see [8]);

- See that Russell's original ideas on typing, despite the vague way in which they are presented in the Principia, can be described in a precise and formal way.

A formalization close to the Principia is given in the first part of the paper (Section 2).

First, so-called "pseudoformulas" are introduced as a notion anticipating the "formulas", i.e. the typable terms, of the Principia[4]. We introduce types

[2][32], Introduction, Chapter II, Section VIII.

[3]This was already observed by Gödel, who said that the precision of Frege was lost in the writings of Russell, and who, due to the informality of some basic notions of the Principia, had to give his paper [16] the title "Über formal unentscheidbare Sätze der Principia Mathematica und *verwandter* Systeme".

[4][32], Introduction, Chapter II, Section V, pp. 50–52, and Part I, Section B, $*12$, pp. 162–164.

and give a formal definition of the notion "pseudoformula f is of type t" (2.21), motivating our definition by referring to passages in the Principia.

The comparison of the system of the Principia with current type systems in λ-calculus is pursued by introducing (in Section 3) a new, λ-calculus-like, notation for pseudoformulas. This new notation gives us also a possibility to present a compact definition of the notion of substitution as it is used in the Principia.

Furthermore, the lambda notation makes it possible to derive properties of the Ramified Theory of Types in an easy way, using properties of modern type systems. This will be done in Section 4. Due to the new notation it is easy to see that we have proven variants of well-known theorems from type theory, like Strong Normalization, Free Variable Lemma, Thinning Lemma, Unicity of Types and Subterm Lemma.

An important goal of Section 4 is to show that substitutions are well-definable, terminating operations. The Principia does not pay any attention to this important and non-trivial problem[5].

In Section 5, we answer in full detail the question which pseudoformulas are formulas. We also make a comparison between our notion of formula, and the notion of formula as defined in the Principia, and conclude that these two notions of formula coincide.

The Ramified Theory of Types is a very restrictive system for the development of mathematics (an effort has been made by Weyl [30]). Whitehead and Russell removed this restriction by postulating the "Axiom of Reducibility" (see Section 6). But it is almost impossible to give a non-technical motivation for this axiom.

Ramsey ([25], 1925) and Hilbert and Ackermann ([19], 1928) propose a simplification of the ramified type theory (the *simple* type theory), in which mathematics is easier to develop, without restrictions. They motivate this simplification by making the distinction between logical and semantical paradoxes. It turns out that it is very easy to carry out this simplification in our formal system.

2. Formalization of the Ramified Theory of Types

In this section we present a formalization of the Ramified Theory of Types (RTT). First we describe the set of propositions and propositional func-

[5]In [28] (1919), p. 151, Russell admits: *"The legitimacy of substitutions* [...] *has to be ensured by means of a non-formal principle of inference. No such principle is enunciated in* Principia Mathematica [...] *But this would seem to be an omission".* However, the second edition of the Principia, which appeared in 1927 (so after the publication of [28]), still doesn't contain such an "inference principle".

tions which Whitehead and Russell use in the Principia, and which we call "pseudoformulas". We give a modernised, formal definition which corresponds to the description in the Principia.

Then we introduce (ramified) types and present a system in natural deduction style which assigns types to certain pseudoformulas.

2a Pseudoformulas

At the basis of the system of our formalization there is

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

Since functions are represented as relations in the Principia, we will not introduce a special set of function symbols.

We assume that $\{\mathtt{a}_1, \mathtt{a}_2, \ldots\} \subseteq \mathcal{A}$; $\{\mathtt{x}, \mathtt{x}_1, \mathtt{x}_2, \ldots, \mathtt{y}, \mathtt{y}_1, \ldots, \mathtt{z}, \mathtt{z}_1, \ldots\} \subseteq \mathcal{V}$; $\{\mathtt{R}, \mathtt{R}_1, \ldots, \mathtt{S}, \mathtt{S}_1, \ldots\} \subseteq \mathcal{R}$. We use the letters $x, y, z, x_1, \ldots$ as meta-variables over $\mathcal{V}$, and $R, R_1, \ldots$ as meta-variables over $\mathcal{R}$.
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

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

and: for each variable $x \in \mathcal{V}$ there is $y \in \mathcal{V}$ with $x < y$.
We also have the logical symbols $\forall$, $\vee$ and $\neg$ in our alphabet, and some non-logical symbols: parentheses and the comma.

DEFINITION 2.1. We define a collection $\mathcal{P}$ of *pseudoformulas*, and for each element f of $\mathcal{P}$ we simultaneously define the collection $\mathrm{FV}(f)$ 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 $R(i_1, \ldots, i_{\mathfrak{a}(R)}) \in \mathcal{P}$.
   $\mathrm{FV}(R(i_1, \ldots, i_{\mathfrak{a}(R)})) \stackrel{\text{def}}{=} \{i_1, \ldots, i_{\mathfrak{a}(R)}\} \cap \mathcal{V}$;
2. If $z \in \mathcal{V}$, $n \in \mathbb{N}$ and $k_1, \ldots, k_n \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$, then $z(k_1, \ldots, k_n) \in \mathcal{P}$.
   $\mathrm{FV}(z(k_1, \ldots, k_n)) \stackrel{\text{def}}{=} \{z, k_1, \ldots, k_n\} \cap \mathcal{V}$.
   If $n = 0$ then we write $z()$ in order to distinguish the pseudoformula $z()$ from the variable z[6];

[6]It is important to note that a variable is not a pseudoformula. See for instance [26], Chapter VIII: "The variable", p.94 of the 7th impression.

3. If $f, g \in \mathcal{P}$ then $f \vee g \in \mathcal{P}$ and $\neg f \in \mathcal{P}$. $\text{FV}(f \vee g) \overset{\text{def}}{=} \text{FV}(f) \cup \text{FV}(g)$; $\text{FV}(\neg f) \overset{\text{def}}{=} \text{FV}(f)$;
4. If $f \in \mathcal{P}$ and $x \in \text{FV}(f)$ then $\forall x[f] \in \mathcal{P}$ and $\text{FV}(\forall x[f]) \overset{\text{def}}{=} \text{FV}(f) \setminus \{x\}$;
5. All pseudoformulas can be constructed by using the construction-rules 1, 2, 3 and 4 above.

We use the letters f, g, h as meta-variables over $\mathcal{P}$.

REMARK 2.2. The intuition behind a pseudoformula is the (naive) idea of *propositional function* as presented in the Principia[7] together with the idea of *proposition.* A propositional function can be seen as an expression with one or more free variables. It will turn into a proposition as soon as we assign values to all the free variables occurring in it. In this light, a *proposition* can be seen as a degenerated propositional function (having 0 free variables).

It will be clear now what the intuition behind pseudoformulas of the form $R(i_1, \ldots, i_{\mathfrak{a}(R)})$, $f \vee g$, $\neg f$ and $\forall x[f]$ is.

The intuition behind pseudoformulas of the second kind is not so obvious. $z(k_1, \ldots, k_n)$ represents a propositional function of *higher order: z is a variable for a pseudoformula with n free variables; the argument list $k_1, \ldots, k_n$ indicates what should be substituted*[8] *for these free variables as soon as one assigns such a pseudoformula to z.*

EXAMPLE 2.3. Such higher-order pseudoformulas occur in ordinary mathematics, too. For instance:

- The pseudoformulas $\mathsf{z(x)}$ and $\mathsf{z(y)}$ in the definition of Leibniz-equality:

$$\forall \mathsf{z}[\mathsf{z(x)} \leftrightarrow \mathsf{z(y)}]$$

- The pseudoformulas $\mathsf{z(0)}$, $\mathsf{z(x)}$ and $\mathsf{z(y)}$ in the formulation of the principle of complete induction:

$$\begin{aligned} \mathsf{z(0)} &\rightarrow (\forall \mathsf{x} \forall \mathsf{y}[\mathsf{z(x)} \rightarrow (\mathsf{S(x,y)} \rightarrow \mathsf{z(y)})]) \\ &\rightarrow \forall \mathsf{x}[\mathsf{z(x)}] \end{aligned}$$

[7]See for instance: Introduction, Chapter II, Section II about the nature of propositional functions.

[8]In the Principia, it is not made clear *how* we should carry out such substitutions. We must depend on our intuition and on the way in which substitution is *used* in the Principia. Unfortunately, there are no really complex examples of the use of substitution in the Principia.

- $\mathtt{z}()$ in the formulation of the law of the excluded middle:

$$\forall \mathtt{z}[\mathtt{z}() \vee \neg \mathtt{z}()]$$

REMARK 2.4. We must remark that Definition 2.1 is not completely to the letter of the Principia. Russell and Whitehead introduce pseudoformulas in two stages. First, they introduce quantifier-free pseudoformulas, which they call *matrices*. Then they form propositional functions, defining that

- Any matrix is a propositional function;
- If f is a propositional function and $x \in \mathrm{FV}(f)$ then $\forall x[f]$ is a propositional function with free variables $\mathrm{FV}(f) \setminus \{x\}$

This definition is a little different from our Definition 2.1. Nevertheless we feel that Russell and Whitehead intended to give our definition 2.1, as in the Principia ([32], $*54$) they define the natural number 0 as the propositional function $\mathtt{0}(\mathtt{z}) \overset{\text{def}}{=} \forall \mathtt{x}[\neg \mathtt{z}(\mathtt{x})]$. And in defining the principle of induction on natural numbers, one needs to express the property "$\mathtt{0}$ has the property $\mathtt{y}$", or: $\mathtt{y}(\mathtt{0})$. But $\mathtt{y}(\mathtt{0})$ is not a propositional function according to Russell's definition, as $\mathtt{0}$ contains quantifiers.

Therefore we feel that, albeit not to the *letter*, Definition 2.1 is surely to the *spirit* of the Principia.

REMARK 2.5. Note that pseudoformulas do not yet obey to the Vicious Circle Principle 1.1! For example, $\neg \mathtt{z}(\mathtt{z})$ (the pseudoformula that is at the basis of the Russell paradox) is a pseudoformula. In this section we will assign types to some pseudoformulas, and it will be shown (Remark 4.11) that no type can be assigned to the pseudoformula $\neg \mathtt{z}(\mathtt{z})$.

EXAMPLE 2.6. Look at the pseudoformula $\mathtt{z}_1(\mathtt{x}_1, \mathtt{x}_2, \mathtt{S}(\mathtt{x}_3, \mathtt{x}_4))$, having free variable set $\{\mathtt{x}_1, \mathtt{x}_2, \mathtt{z}_1\}$ according to Definition 2.1. Note that $\mathtt{x}_3$ and $\mathtt{x}_4$ do not belong to this free variable set. We illustrate with an example that this is in line with the intuition on pseudoformulas and free variables of Remark 2.2. According to this intuition, substituting appropriate values for $\mathtt{z}_1$, $\mathtt{x}_1$ and $\mathtt{x}_2$ should result in a proposition.

Assume, we substitute the pseudoformula $\mathtt{z}_2(\mathtt{x}_5, \mathtt{x}_6)$ for $\mathtt{z}_1$,[9] simultaneously with the substitution of $\mathtt{a}_1$ for $\mathtt{x}_1$ and $\mathtt{a}_2$ for $\mathtt{x}_2$.

[9] $\mathtt{z}_2(\mathtt{x}_5, \mathtt{x}_6)$ is intuitively an "appropriate value" for $\mathtt{z}_1$: The argument list $\mathtt{x}_1, \mathtt{x}_2, \mathtt{S}(\mathtt{x}_3, \mathtt{x}_4)$ behind $\mathtt{z}_1$ consists of three objects, of which one is a pseudoformula, and $\mathtt{z}_2(\mathtt{x}_5, \mathtt{x}_6)$ has three variables, of which $\mathtt{z}_2$ is a variable for a pseudoformula. When we have introduced types, the notion of "appropriate" can be formally expressed with "being of the same type".

This gives $(\mathtt{z}_2(\mathtt{x}_5,\mathtt{x}_6))(\mathtt{a}_1,\mathtt{a}_2,\mathtt{S}(\mathtt{x}_3,\mathtt{x}_4))$, and according to the intuition given in the italicized sentence of Remark 2.2, the expressions $\mathtt{a}_1,\mathtt{a}_2,\mathtt{S}(\mathtt{x}_3,\mathtt{x}_4)$ should be substituted for the free variables of $\mathtt{z}_2(\mathtt{x}_5,\mathtt{x}_6)$. A reasonable assignment[10] is: $\mathtt{S}(\mathtt{x}_3,\mathtt{x}_4)$ to $\mathtt{z}_2$ ($\mathtt{z}_2$ being a variable for a pseudoformula), and (e.g.) $\mathtt{a}_1$ to $\mathtt{x}_5$ and $\mathtt{a}_2$ to $\mathtt{x}_6$.

This results in $(\mathtt{S}(\mathtt{x}_3,\mathtt{x}_4))(\mathtt{a}_1,\mathtt{a}_2)$ and, again using the intuition of Remark 2.2, the expressions $\mathtt{a}_1$ and $\mathtt{a}_2$ should be substituted for the free variables $\mathtt{x}_3$ and $\mathtt{x}_4$ of $\mathtt{S}(\mathtt{x}_3,\mathtt{x}_4)$. This leads to $\mathtt{S}(\mathtt{a}_1,\mathtt{a}_2)$ (or $\mathtt{S}(\mathtt{a}_2,\mathtt{a}_1)$; see footnote 10), and there is no new substitution left. Apparently, $\mathtt{S}(\mathtt{a}_1,\mathtt{a}_2)$ is the final result of the substitution.

$\mathtt{S}(\mathtt{a}_1,\mathtt{a}_2)$ is indeed a proposition. It has not been necessary to substitute values for $\mathtt{x}_3$ and $\mathtt{x}_4$ at the beginning: These two variables obtained their values "in the way of the substitution process". We come back to this example in Example 3.8.4, after we have formalized substitution.

Remark 2.7. In the previous example, we see that the variables $\mathtt{x}_3$ and $\mathtt{x}_4$ are neither free, nor bound by a quantifier.

We conclude our discussion of pseudoformulas with defining a number of related notions:

Definition 2.8. Let f and g be pseudoformulas. We say that f and g are *α-equal*, notation $f =_\alpha g$, if there is a bijection $\varphi : \mathcal{V} \to \mathcal{V}$ such that g can be obtained from f by replacing each variable that occurs in f by its φ-image.

Definition 2.9. Assume f is a pseudoformula, and $k \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$. We define, inductively, the notion *k is an argument of f*.

- If $f = R(i_1,\ldots,i_{\mathfrak{a}(R)})$ then $i_1,\ldots,i_{\mathfrak{a}(R)}$ are the arguments of f.
- If $f = z(k_1,\ldots,k_n)$ then $k_1,\ldots,k_n$ are the arguments of f.
- If $f = f_1 \vee f_2$ then the arguments of f are the arguments of f_1 together with the arguments of f_2;
  if $f = \neg f'$ then the arguments of f are the arguments of f';
- If $f = \forall x[f']$ then the arguments of f are the arguments of f'.

We write $\textsc{arg}(f)$ for the set of arguments of f.

Example 2.10. Let $f \stackrel{\text{def}}{=} \mathtt{z}(\mathtt{x}_1,\mathtt{a}_2,\mathtt{S}(\mathtt{x}_1,\mathtt{a}_3))$. $\mathtt{a}_2$, $\mathtt{S}(\mathtt{x}_1,\mathtt{a}_3)$ and the leftmost occurrence of $\mathtt{x}_1$ are the arguments of f, but neither $\mathtt{a}_3$ nor the rightmost occurrence of $\mathtt{x}_1$ is an argument of f.

[10] At this stage there is no rule that determines which expression should be substituted for which variable. Such a rule will be provided by the formal definition of substitution.

$\mathtt{a}_3$ and the rightmost occurrence of $\mathtt{x}_1$ are not arguments of f, but they are *recursive* arguments according to the following definition:

DEFINITION 2.11. Assume f is a pseudoformula. We define, inductively, the set of *recursive arguments* of f, $\mathrm{RA}(f)$:

- $\mathrm{RA}(R(i_1, \ldots, i_{\mathfrak{a}(R)})) \overset{\mathrm{def}}{=} \{i_1, \ldots, i_{\mathfrak{a}(R)}\}$;
- $\mathrm{RA}(z(k_1, \ldots, k_n)) \overset{\mathrm{def}}{=} \{k_1, \ldots, k_n\} \cup \bigcup_{k_i \in \mathcal{P}} \mathrm{RA}(k_i)$;
- $\mathrm{RA}(\neg f) \overset{\mathrm{def}}{=} \mathrm{RA}(f)$; $\mathrm{RA}(f_1 \vee f_2) \overset{\mathrm{def}}{=} \mathrm{RA}(f_1) \cup \mathrm{RA}(f_2)$;
- $\mathrm{RA}(\forall x[f]) \overset{\mathrm{def}}{=} \mathrm{RA}(f)$.

2b Ramified Types

Firstly, we introduce the notion of *simple type*:

DEFINITION 2.12. (Simple types)

1. $\mathsf{0}$ is a simple type;
2. If $t_1, \ldots, t_n$ are simple types, then also $(t_1, \ldots, t_n)$ is a simple type. $n = 0$ is allowed: then we obtain the simple type $()$.
3. All simple types can be constructed using the rules 1 and 2.

REMARK 2.13. (Simple) types are not explicitly introduced in the Principia. But there is a definition of "being of the same type"[11], on which definition 2.12 is based: Individuals are of the same type. This will be the type $\mathsf{0}$. And one can think of $(t_1, \ldots, t_n)$ as being the type of the propositional functions with n free variables, say $x_1, \ldots, x_n$, such that if we assign values u_1 of type t_1 to x_1, ..., u_n of type t_n to x_n, then we obtain a proposition. The type $()$ stands for the type of the propositions.

Note that Whitehead and Russell do not give an order in $(t_1, \ldots, t_n)$: They consider $(t_1, \ldots, t_n)$ as a *multiset* of types, being in some bijective relation with the free variables occurring in a propositional function of type $(t_1, \ldots, t_n)$. For technical reasons however, we want to keep a distinction between, e.g., the types (t_1, t_2) and (t_2, t_1) in this paper. See Remark 2.23.7, where we justify this different view.

EXAMPLE 2.14. Look again at the pseudoformula $\mathtt{z}_1(\mathtt{x}_1, \mathtt{x}_2, \mathtt{S}(\mathtt{x}_3, \mathtt{x}_4))$. The calculation in Example 2.6 shows, that the pseudoformula $\mathtt{S}(\mathtt{x}_3, \mathtt{x}_4)$ is of type

[11] [32], $*9{\cdot}131$, p. 133

$(0,0)$: When two individuals are substituted, the pseudoformula becomes a proposition.

Now we can also decide about the type of pseudoformulas that can be substituted for z_1: For z_1 we can only substitute pseudoformulas with three free variables. Two of these free variables must be of the same type as x_1 and x_2 (as we substituted individuals for them, we can take type 0); the third one must have the type of $S(x_3, x_4)$: $(0,0)$. That means that for z_1, pseudoformulas of type $(0,0,(0,0))$ can be substituted. Now look at the full pseudoformula $z_1(x_1, x_2, S(x_3, x_4))$. This pseudoformula has three free variables: x_1 and x_2, for which things of type 0 can be substituted, and z_1, for which things of type $(0,0,(0,0))$ can be substituted.

Hence, the pseudoformula $z_1(x_1, x_2, S(x_3, x_4))$ is of type $(0,0,(0,0,(0,0)))$.

The intuition presented in Remark 2.13 and Example 2.14 will be made formal in 2.21. Theorem 4.2 shows that this formalization follows the intuition.

Simple types can be divided into orders[12]. Doing this we obtain *ramified types*:

DEFINITION 2.15. (Ramified types)

1. 0^0 is a ramified type;
2. If $t_1^{a_1}, \ldots, t_n^{a_n}$ are ramified types, and $a \in \mathbb{N}$, $a > \max(a_1, \ldots, a_n)$, then $(t_1^{a_1}, \ldots, t_n^{a_n})^a$ is a ramified type (if $n = 0$ then take $a \geq 0$);
3. All ramified types can be constructed using the rules 1 and 2.

If t^a is a ramified type, then a is called the *order* of t^a.

EXAMPLE 2.16. 0^0, $(0^0)^1$, $((0^0)^1, (0^0)^4)^5$, and $(0^0, ()^2, (0^0, (0^0)^1)^2)^7$ are ramified types.
$(0^0, (0^0, (0^0)^2)^2)^7$ is not a ramified type.

In the rest of this paper we simply speak of *types* when we mean: *ramified* types, as long as no confusion arises.

Now we have introduced types, it is possible to indicate behind a quantifier $\forall x$ over *which type* is quantified. When quantification is over a type t^a, we write $\forall x{:}t^a$.

In the type $(0^0)^1$, all orders are "minimal", i.e., not higher than strictly necessary. This is, for instance, not the case in the type $(0^0)^2$. Types in which all orders are minimal are called *predicative* and play a special role in the Ramified Theory of Types. A formal definition:

[12]More about orders can be found in [32], *12, and Introduction, Chapter II, Section v, pp. 51-55

DEFINITION 2.17. (Predicative types)

1. 0^0 is a predicative type;
2. If $t_1^{a_1}, \ldots, t_n^{a_n}$ are predicative types, and $a = 1 + \max(a_1, \ldots, a_n)$ (take $a = 0$ if $n = 0$), then $(t_1^{a_1}, \ldots, t_n^{a_n})^a$ is a predicative type.

DEFINITION 2.18. (Basic formulas) If $R \in \mathcal{R}$ and $c_1, \ldots, c_{\mathfrak{a}(R)} \in \mathcal{A}$, then the pseudoformula $R(c_1, \ldots, c_{\mathfrak{a}(R)})$ is called a *basic formula.*

The basic formulas are formalizations of the "elementary judgements" in the Principia[13].

DEFINITION 2.19. (Contexts) Let $x_1, \ldots, x_n \in \mathcal{V}$ be distinct variables, and assume $t_1^{a_1}, \ldots, t_n^{a_n}$ are ramified types. Then $\{x_1{:}t_1^{a_1}, \ldots, x_n{:}t_n^{a_n}\}$ is a *context.* $\{x_1, \ldots, x_n\}$ is called the *domain* $\mathrm{dom}(\{x_1{:}t_1^{a_1}, \ldots, x_n{:}t_n^{a_n}\})$ of the context.

These contexts will take over implicit presuppositions, like the assumption that we want to substitute an individual for the variable $\mathtt{x}_1$ in Example 2.14. In the formal system that we are going to build, such information will be made explicit by the addition of contexts. Contexts, common in modern type systems, are not used in the Principia.We will use the letters Γ, Δ for meta-variables over contexts.

We restrict α-equality in two senses. Firstly, α-equal variables must have the same type with respect to a context Γ, secondly, the alphabetic order on variables must be conserved by the α-equality. This leads to the notion of α_Γ-equality.

DEFINITION 2.20. Let Γ be a context and f and g pseudoformulas. f and g are called α_Γ*-equal,* if there is a bijection $\varphi : \mathcal{V} \to \mathcal{V}$ such that g can be obtained from f by replacing each variable x that occurs in f by $\varphi(x)$, and for all x that occur in f it holds that $x : t^a \in \Gamma$ iff $\varphi(x) : t^a \in \Gamma$, and for all x, y that occur in f: $x < y$ iff $\varphi(x) < \varphi(y)$.

2c RTT

We will now define what we mean by $\Gamma \vdash f : t^a$, or, in words: f is of type t^a in the context Γ. In this definition we will try to follow the line of the Principia as much as possible. If $\Gamma = \emptyset$ then we will write $\vdash f : t^a$, as usual.

We explain some aspects of the following definition in Remarks 2.23–2.25.

[13][32], Introduction, Chapter II, Section III, pp. 43-45, and Introduction to the 2nd edition, Section I, p. xv.

DEFINITION 2.21. (Ramified Theory of Types: RTT) The *judgement* $\Gamma \vdash f : t^a$ is inductively defined as follows:

1. (start) For all individual symbols a:

$$\vdash a : 0^0$$

For all basic formulas f:

$$\vdash f : ()^0$$

2. (connectives) Assume $\Gamma \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$, $\Delta \vdash g : (u_1^{b_1}, \ldots, u_m^{b_m})^b$, and for all $x \in \mathrm{dom}(\Gamma)$ and $y \in \mathrm{dom}(\Delta)$, $x < y$. Then

$$\Gamma \cup \Delta \vdash f \vee g : (t_1^{a_1}, \ldots, t_n^{a_n}, u_1^{b_1}, \ldots, u_m^{b_m})^{\max(a,b)}$$

$$\Gamma \vdash \neg f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$$

3. (introduction of variables for arguments) If $\Gamma \vdash f : (t_1^{a_1}, \ldots, t_m^{a_m})^a$, $t_{m+1}^{a_{m+1}}$ is a *predicative* type, $g \in \mathcal{A} \cup \mathcal{P}$ is an argument of f, and $\Gamma \vdash g : t_{m+1}^{a_{m+1}}$, and $x < y$ for all $x \in \mathrm{dom}(\Gamma)$, then

$$\Gamma' \vdash h : (t_1^{a_1}, \ldots, t_{m+1}^{a_{m+1}})^{\max(a, a_{m+1}+1)}$$

Here, h is a pseudoformula obtained by replacing all arguments g' of f which are α_Γ-equal to g by y. Moreover, Γ' is the subset of the context $\Gamma \cup \{y : t_{m+1}^{a_{m+1}}\}$ such that $\mathrm{dom}(\Gamma')$ contains exactly all the variables that occur in h[14].

4. (introduction of variables for formulas) If $(t_1^{a_1}, \ldots, t_m^{a_m})^a$ is a *predicative* type, $\Gamma \vdash f : (t_1^{a_1}, \ldots, t_m^{a_m})^a$, $x < z$ for all $x \in \mathrm{dom}(\Gamma)$, and $y_1 < \ldots < y_n$ are the free variables of f, then

$$\Gamma' \vdash z(y_1, \ldots, y_n) : (t_1^{a_1}, \ldots, t_m^{a_m}, (t_1^{a_1}, \ldots, t_m^{a_m})^a)^{a+1}$$

where Γ' is the subset of $\Gamma \cup \{z{:}(t_1^{a_1}, \ldots, t_m^{a_m})^a\}$ such that $\mathrm{dom}(\Gamma') = \{y_1, \ldots, y_n, z\}$[14].

5. (weakening) If Γ, Δ are contexts, $\Gamma \subseteq \Delta$, and $\Gamma \vdash f : t^m$, then also $\Delta \vdash f : t^m$;

6. (substitution) If y is the ith free variable in f (according to the order on variables), and $\Gamma \cup \{y : t_i^{a_i}\} \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$, and $\Gamma \vdash k : t_i^{a_i}$ then

$$\Gamma' \vdash f[y{:=}k] : (t_1^{a_1}, \ldots, t_{i-1}^{a_{i-1}}, t_{i+1}^{a_{i+1}}, \ldots, t_n^{a_n})^b$$

[14]In Lemma 2.26 we prove that this context always exists.

Here, $b = 1 + \max(a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n)$, ($b = 0$ if $n = 1$) and once more, Γ' is the subset of $\Gamma \cup \{y : t_i^{a_i}\}$ such that $\mathrm{dom}(\Gamma')$ contains exactly all the variables that occur in $f[y{:=}k]$[14].
By $f[y{:=}k]$ we mean the pseudoformula where k is "substituted" for all free occurrences of y (we come back to substitution in Section 3.)

7. (permutation) If y is the ith free variable in f (according to the order on variables), and $\Gamma \cup \{y{:}t_i^{a_i}\} \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$, and $x < y'$ for all $x \in \mathrm{dom}(\Gamma)$, then
$$\Gamma' \cup \{y'{:}t_i^{a_i}\} \vdash f[y{:=}y'] : (t_1^{a_1}, \ldots, t_{i-1}^{a_{i-1}}, t_{i+1}^{a_{i+1}}, \ldots, t_n^{a_n}, t_i^{a_i})^a.$$
$\Gamma' = \Gamma$ if y does not occur in $f[y{:=}y']$; $\Gamma' = \Gamma \cup \{y{:}t_i^{a_i}\}$ if y occurs in $f[y{:=}y']$.
8. (quantification) If y is the ith free variable in f (according to the order on variables), and $\Gamma \cup \{y{:}t_i^{a_i}\} \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$, then
$$\Gamma \vdash \forall y{:}t_i^{a_i}[f] : (t_1^{a_1}, \ldots, t_{i-1}^{a_{i-1}}, t_{i+1}^{t_{i+1}}, \ldots, t_n^{a_n})^a$$

DEFINITION 2.22. A pseudoformula f is called a *formula*, if there is a context Γ and a ramified type t^a such that $\Gamma \vdash f : t^a$.

REMARK 2.23. We will motivate RTT (Definition 2.21) by referring to the Principia:

1. Individuals and elementary judgements (basic formulas) are, also in the Principia, the basic ingredients for creating formulas.[15]
2. We only introduce the logical connectives $\vee$ and $\neg$. This is in line with the Principia, where only negation and disjunction are taken as primitive ideas[16], and implication[17] and conjunction[18] are defined in terms of negation and disjunction.
We can see rule 2 "at work" in *12, p. 163 of the Principia[19]: *"We can build up a number of new formulas, such as* [...] $\varphi!x \vee \varphi!y$, $\varphi!x \vee \psi!x$, $\varphi!x \vee \psi!y$, [...] *and so on."*
The restriction about contexts that we make in rule 2 is not made in the Principia, and will be discussed in 2.24.

[15]As for individuals: see [32], *9, p. 132, where "Individual" is presented as a primitive idea. As for elementary judgements: See [32], Introduction, Chapter II, Section III, pp. 43-45.
[16]cf. [32], *1, pp. 93-94
[17]cf. [32], *1·01, p. 94
[18]cf. [32], *3·01, p. 107
[19]In the Principia, Russell and Whitehead write $\varphi!x$ instead of φx to indicate that φx is not only (what we would call) a pseudoformula, but even a formula.

3. Rule 3 is justified by $*9{\cdot}14$ and $*9{\cdot}15$ in the Principia. The restriction to variables of *predicative* type is in line with the Principia (cf. [32], Chapter II, Section V, p. 54).

4. Rule 4 is based on the Introduction, Chapter II, Section V, p. 51. There, formulas are constructed, and "*the first matrices that occur are those whose values are of the forms* $\varphi x, \psi(x,y), \chi(x,y,z,\ldots)$, i.e. *where the arguments, however many there may be, are all individuals.* [...] *Such functions we will call* 'first-order *functions.' We may now introduce a notation to express 'any first-order function.'* " This, then, results in second order functions, and this process is repeated to obtain functions of higher orders.
Rule 4 makes it possible to introduce variables of higher order. In fact, leaving out rule 4 would lead to first-order predicate logic, as without rule 4 it is impossible to introduce variables of types that differ from 0^0.
The use of predicative types only is inspired by the Principia, again.

5. The weakening rule cannot be found in the Principia, because no formal contexts are used there. It is implicitly present, however: the addition of an extra variable to the set of variables does not affect the well-typedness of formulas that were already constructed.

6. The rule of substitution is based on $*9{\cdot}14$ and $*9{\cdot}15$ of the Principia. We must notice that substitution is not defined in the Principia, so it is not completely clear yet what $f[y{:=}k]$ should mean. Clarification of the notion of substitution is given in Section 3.

7. In the system above, the (sequential) order of the t_is is related to the alphabetic order of the free variables of the pseudoformula f that has type $(t_1,\ldots,t_n)$ (see Theorem 4.2). We need this alphabetic order for a clear presentation of results like Theorem 4.2.
With rule 7 we want to express that the order of the t_is in $(t_1,\ldots,t_n)$ and the alphabetic order of the variables are not characteristics of the Principia, but are only introduced for the technical reasons explained in this remark. This is worked out in Corollary 4.3.

8. Notice that in the quantification rule, both f and $\forall x{:}t_i^{a_i}.f$ have order a. The intuition is that the order of a propositional function f equals one plus the maximum of the orders of *all* the variables (either free or bound by a quantifier) in f. See also Lemma 4.4 below.

REMARK 2.24. In rule 2 of RTT, we make the assumption that the variables of Γ must all come before the variables of Δ. The reason for this is that we want to prevent undesired results like

$$\mathtt{x}_1{:}0^0 \vdash \mathtt{R}_1(\mathtt{x}_1) \vee \mathtt{R}_2(\mathtt{x}_1) : (0^0, 0^0)^1$$

In fact, $\mathtt{R}_1(\mathtt{x}_1) \vee \mathtt{R}_2(\mathtt{x}_1)$ has *only one* free variable, so its type should be $(0^0)^1$ and not $(0^0, 0^0)^1$ (see Example 2.27, second part). For technical reasons (the order of the $t_i^{a_i}$s; see also Theorem 4.2) we strengthen the assumption such that for $x \in \mathrm{dom}(\Gamma)$ and $y \in \mathrm{dom}(\Delta)$, $x < y$ must hold.

As Russell and Whitehead do not have a formal notation for types, they do not forbid this kind of constructions in [32]. In 5.4 we show that our limitation to non-overlapping contexts as made in rule 2 is not a real limitation: all the desired judgements can still be derived for overlapping contexts.

REMARK 2.25. Contexts as used in RTT contain, in a sense, too much information: not only information on all free variables, but also information on non-free variables (Cf. rules 3, 6 and 7. The set of non-free variables contains more than only the variables that are bound by a quantifier; see Remark 2.7). We will discuss this in Remark 3.6, where we also propose a solution to this problem. For the moment, we only prove the following lemma.

LEMMA 2.26. *Let* Γ *be a context,* f *a pseudoformula, and assume* $\Gamma \vdash f : t^a$. *Then*

1. *All variables of* f *that are not bound by a quantifier are in* $\mathrm{dom}(\Gamma)$*;*
2. *If* Δ *is the (unique) subset of* Γ *such that* $\mathrm{dom}(\Delta)$ *contains exactly all the variables of* f *that are not bound by a quantifier, then* $\Delta \vdash f : t^a$.

PROOF. An easy induction on the definition of $\Gamma \vdash f : t^a$. ∎

Part 1 of this lemma shows that the contexts Γ' in 2.21.3, 2.21.4 and 2.21.6 really exist.

EXAMPLE 2.27. We will give some examples, in order to illustrate how our system works. We will use a notation of the form $\dfrac{X_1, \ldots, X_n}{Y} N$, indicating that from the judgements $X_1, \ldots, X_n$, we can infer the judgement Y by using the RTT-rule of Definition 2.21 with number N. The types in the examples below are all predicative. To avoid too much notation, we omit the orders.

- $$\vdash \mathtt{S}(\mathtt{a}_1, \mathtt{a}_2) : ()$$

- $$\frac{\vdash R_1(a_1) : () \qquad \vdash R_2(a_1) : ()}{\vdash R_1(a_1) \vee R_2(a_1) : ()}\,2$$

  but *not*:

  $$\frac{x_1 : 0 \vdash R_1(x_1) : (0) \qquad x_1 : 0 \vdash R_2(x_1) : (0)}{x_1 : 0 \vdash R_1(x_1) \vee R_2(x_1) : (0,0)}\,2$$

  To obtain $R_1(x_1) \vee R_2(x_1)$ we must make a different start:

  $$\frac{\dfrac{\vdash R_1(a_1) : () \qquad \vdash R_2(a_1) : ()}{\vdash R_1(a_1) \vee R_2(a_1) : ()}\,2 \qquad \vdash a_1 : 0}{x_1 : 0 \vdash R_1(x_1) \vee R_2(x_1) : (0)}\,3$$

- $$\frac{\begin{array}{rcl} x_1 : 0, x_2 : 0, z_1 : ((0),(0)) & \vdash & z_1(R(x_1), R(x_2)) : (((0),(0))) \\ x_1 : 0, x_2 : 0, z_1 : ((0),(0)) & \vdash & R(x_1) : (0) \end{array}}{z_1 : ((0),(0)), z_2 : (0) \vdash z_1(z_2, z_2) : (((0),(0)),(0))}\,3$$

- $$\frac{x_1 : 0, x_2 : 0 \vdash S(x_1, x_2) : (0,0)}{x_1 : 0, x_2 : 0, z : (0,0) \vdash z(x_1, x_2) : (0,0,(0,0))}\,4$$

- $$\frac{\dfrac{x_1 : 0 \vdash R_1(x_1) \vee R_2(x_1) : (0) \qquad \vdash a_1 : 0}{\vdash R_1(a_1) \vee R_2(a_1) : ()}\,6}{x_1 : 0 \vdash R_1(a_1) \vee R_2(a_1) : ()}\,5$$

- $$\frac{\begin{array}{rcl} x_1 : 0, x_2 : 0, x_3 : (0,0) & \vdash & R(x_1) \vee \neg x_3(x_1, x_2) : (0,0,(0,0)) \\ x_1 : 0, x_2 : 0 & \vdash & T(x_1, x_1, x_2) : (0,0) \end{array}}{x_1 : 0, x_2 : 0 \vdash R(x_1) \vee \neg T(x_1, x_1, x_2) : (0,0)}\,6$$

EXAMPLE 2.28. Rule 6 of RTT is problematic:

$$\frac{\begin{array}{rcl} x_1 : 0, x_2 : (0,(0)), x_3 : 0, x_4 : (0) & \vdash & x_2(x_1, R(x_1)) : (0,(0,(0))) \\ x_1 : 0, x_3 : 0, x_4 : (0) & \vdash & x_4(x_3) : (0,(0)) \end{array}}{x_1 : 0 \vdash R(x_1) : (0)}\,6$$

It is not very clear what happens here. However, this example turns out to be all right (see Example 3.8, part 5) after we have formalized substitution.

3. RTT in λ-notation

The notation for pseudoformulas that we introduced in Definition 2.1 is based on the notations used in the Principia. In this section we present a new notation for pseudoformulas, using λ-calculus. This has several advantages:

- It makes comparisons of RTT with modern type systems easier;

- It opens the possibility to prove properties of RTT with techniques from (typed) lambda calculi;
- It will make the notion of free variable of a pseudoformula more clear (see also Example 2.6);
- The notion of substitution is quite complicated in RTT, but has been left unexplained in the Principia. It is possible to give such a definition with the pseudoformulas of Definition 2.1 (see for instance [23]), but β-reduction in λ-calculus makes it much easier to explain this notion.

We firstly introduce the λ-notation for pseudoformulas:

DEFINITION 3.1. Let Λ be the set of untyped λ-terms over a set $V \supseteq \mathcal{V}$ of variables and a set $C \supseteq \{\neg, \vee\} \cup \{\forall_{t^a} | t^a \text{ is a ramified type}\} \cup \mathcal{A} \cup \mathcal{R}$ of constants. We define a map $\widetilde{\ } : \mathcal{P} \to \Lambda$ inductively:

- If $f = R(i_1, \ldots, i_{\mathfrak{a}(R)})$ then $\widetilde{f} \stackrel{\text{def}}{=} Ri_1 \cdots i_{\mathfrak{a}(R)}$.
- If $f = z(k_1, \ldots, k_m)$ then define $\ell_i = k_i$ for $k_i \in \mathcal{A} \cup \mathcal{V}$, and $\ell_i = \lambda x_1 \cdots x_n.\widetilde{k_i}$ for $k_i \in \mathcal{P}$ with free variables $x_1 < \ldots < x_n$. Then $\widetilde{f} \stackrel{\text{def}}{=} z\ell_1 \ldots \ell_m$.
- If $f = f_1 \vee f_2$ then $\widetilde{f} \stackrel{\text{def}}{=} \vee\widetilde{f_1}\widetilde{f_2}$. If $f = \neg f'$ then $\widetilde{f} \stackrel{\text{def}}{=} \neg\widetilde{f'}$.
- If $f = \forall x{:}t^a.f'$ then $\widetilde{f} \stackrel{\text{def}}{=} \forall_{t^a}\lambda x.\widetilde{f'}$.

LEMMA 3.2. *The map* $\widetilde{\ }$ *is injective.*

PROOF. Easy. ∎

To make notation more convenient in the definition of ℓ_i in Definition 3.1 above, we also define a map $\overline{\ } : \mathcal{A} \cup \mathcal{V} \cup \mathcal{P} \to \Lambda$:

DEFINITION 3.3.

- $\overline{a} \stackrel{\text{def}}{=} a$ for $a \in \mathcal{A}$;
- $\overline{x} \stackrel{\text{def}}{=} x$ for $x \in \mathcal{V}$;
- Now assume $f \in \mathcal{P}$ has free variables $x_1 < \ldots < x_m$. Then $\overline{f} \stackrel{\text{def}}{=} \lambda x_1 \cdots x_m.\widetilde{f}$.

We use the notations and conventions on λ-terms presented in [2] throughout the rest of this paper.

EXAMPLE 3.4.

f	$\widetilde{f}$
$\mathtt{S(x_1,x_3)}$	$\mathtt{Sx_1x_3}$
$\mathtt{z_1(S(x_1,x_2),x_3,a_2)}$	$\mathtt{z_1(\lambda x_1x_2.Sx_1x_2)x_3a_2}$
$\mathtt{z_1(x_3,a_2,S(x_1,x_2)) \vee z_2(x_3)}$	$\mathtt{\vee(z_1x_3a_2(\lambda x_1x_2.Sx_1x_2))(z_2x_3)}$
$\mathtt{z_1(x_3,a_2,z_2(S(x_1,x_1)))}$	$\mathtt{z_1x_3a_2(\lambda z_2.z_2(\lambda x_1.Sx_1x_1))}$
$\mathtt{\forall x_1{:}0^0.S(x_1,x_3)}$	$\mathtt{\forall_{0^0}\lambda x_1.Sx_1x_3}$

The notions of free variable in new and old notation coincide:

LEMMA 3.5. *For all* $f \in \mathcal{P}$, $\mathrm{FV}(f) = \mathrm{FV}(\widetilde{f})$ *and* $\mathrm{FV}(\overline{f}) = \emptyset$.

PROOF. By induction on the construction of f. ■

REMARK 3.6. We see that the new notation is more clear, as from the λ-terms it can immediately be seen which variable is free, and which variable is not. (cf. the discussion in Example 2.6, and Remark 2.7). The new notation would also make it possible to give contexts a more elegant role (cf. 2.25), as one could decide to let them contain type information of the free variables only. When we take a preview at Theorem 4.1, we see that this would, for instance, simplify the description of the context Γ' in rule 2.21.6 considerably. We come back to the definition of $\widetilde{f}$ after the definition of substitution 3.7; see Remark 3.9.

With the λ-notation we can rely on the notions of β-reduction and β-normal form to give the following definition of substitution:

DEFINITION 3.7. (Substitution) Let $f \in \mathcal{P}$, assume $x_1, \ldots, x_n$ are distinct variables, and $g_1, \ldots, g_n \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$. Assume the λ-term

$$(\lambda x_1 \cdots x_n.\widetilde{f})\overline{g_1} \cdots \overline{g_n}$$

has a β-normal form H. Assume $h \in \mathcal{P}$ such that $\widetilde{h} \equiv H$. (This h is unique due to Lemma 3.2.)
Then $f[x_1, \ldots, x_n{:=}g_1, \ldots, g_n] \stackrel{\mathrm{def}}{=} h$.

In [23], the original definition based on Russell's notations is given, and it is proved to be equivalent with Definition 3.7 above.

Notice that $f[x_1, \ldots, x_n{:=}g_1, \ldots, g_n]$ should be seen as a *simultaneous* (non-curried) substitution of $g_1, \ldots, g_n$ for $x_1, \ldots, x_n$. As the $\overline{g_i}$s are either closed λ-terms, or individuals, or variables, it is no problem to define this simultaneous substitution via a list of (curried) applications that results in a list of *consecutive* substitutions.

EXAMPLE 3.8.

1. $\mathtt{S}(\mathtt{x}_1,\mathtt{x}_3)[\mathtt{x}_1{:=}\mathtt{a}_1] = \mathtt{S}(\mathtt{a}_1,\mathtt{x}_3)$, as $(\lambda\mathtt{x}_1.\mathtt{S}\mathtt{x}_1\mathtt{x}_3)\mathtt{a}_1 \to_\beta \mathtt{S}\mathtt{a}_1\mathtt{x}_3$;
2. $\mathtt{S}(\mathtt{x}_1,\mathtt{x}_3)[\mathtt{x}_2{:=}\mathtt{a}_2] = \mathtt{S}(\mathtt{x}_1,\mathtt{x}_3)$, as $(\lambda\mathtt{x}_2.\mathtt{S}\mathtt{x}_1\mathtt{x}_3)\mathtt{a}_2 \to_\beta \mathtt{S}\mathtt{x}_1\mathtt{x}_3$;
3. $\mathtt{z}(\mathtt{S}(\mathtt{x}_1,\mathtt{x}_2),\mathtt{x}_3,\mathtt{a}_2)[\mathtt{x}_1{:=}\mathtt{a}_1] = \mathtt{z}(\mathtt{S}(\mathtt{x}_1,\mathtt{x}_2),\mathtt{x}_3,\mathtt{a}_2)$ as $(\lambda\mathtt{x}_1.\mathtt{z}(\lambda\mathtt{x}_1\mathtt{x}_2.\mathtt{S}\mathtt{x}_1\mathtt{x}_2)\mathtt{x}_3\mathtt{a}_2)\mathtt{a}_1 \to_\beta \mathtt{z}(\lambda\mathtt{x}_1\mathtt{x}_2.\mathtt{S}\mathtt{x}_1\mathtt{x}_2)\mathtt{x}_3\mathtt{a}_2$ (in normal form). This illustrates that the λ-notation is more precise and convenient with respect to free variables. In $\mathtt{z}(\mathtt{S}(\mathtt{x}_1,\mathtt{x}_2),\mathtt{x}_3,\mathtt{a}_2)$, it is not immediately clear whether $\mathtt{x}_1$ is a free variable or not and one might tend to write $\mathtt{z}(\mathtt{S}(\mathtt{x}_1,\mathtt{x}_2),\mathtt{x}_3,\mathtt{a}_2)[\mathtt{x}_1{:=}\mathtt{a}_1] = \mathtt{z}(\mathtt{S}(\mathtt{a}_1,\mathtt{x}_2),\mathtt{x}_3,\mathtt{a}_2)$. The λ-notation is more explicit in showing that $\mathtt{x}_1 \notin \mathrm{FV}(\mathtt{z}(\mathtt{S}(\mathtt{x}_1,\mathtt{x}_2),\mathtt{x}_3,\mathtt{a}_2))$.
4. See Example 2.6.
   $\mathtt{z}_1(\mathtt{x}_1,\mathtt{x}_2,\mathtt{S}(\mathtt{x}_3,\mathtt{x}_4))[\mathtt{z}_1,\mathtt{x}_1,\mathtt{x}_2{:=}\mathtt{z}_2(\mathtt{x}_5,\mathtt{x}_6),\mathtt{a}_1,\mathtt{a}_2)] = \mathtt{S}(\mathtt{a}_1,\mathtt{a}_2)$, as
   $(\lambda\mathtt{z}_1\mathtt{x}_1\mathtt{x}_2.\mathtt{z}_1\mathtt{x}_1\mathtt{x}_2(\lambda\mathtt{x}_3\mathtt{x}_4.\mathtt{S}\mathtt{x}_3\mathtt{x}_4))(\lambda\mathtt{x}_5\mathtt{x}_6\mathtt{z}_2.\mathtt{z}_2\mathtt{x}_5\mathtt{x}_6)\mathtt{a}_1\mathtt{a}_2 \twoheadrightarrow_\beta$
   $(\lambda\mathtt{x}_5\mathtt{x}_6\mathtt{z}_2.\mathtt{z}_2\mathtt{x}_5\mathtt{x}_6)\mathtt{a}_1\mathtt{a}_2(\lambda\mathtt{x}_3\mathtt{x}_4.\mathtt{S}\mathtt{x}_3\mathtt{x}_4) \twoheadrightarrow_\beta$
   $(\lambda\mathtt{x}_3\mathtt{x}_4.\mathtt{S}\mathtt{x}_3\mathtt{x}_4)\mathtt{a}_1\mathtt{a}_2 \twoheadrightarrow_\beta \mathtt{S}\mathtt{a}_1\mathtt{a}_2$.
5. See Example 2.28.
   $\mathtt{x}_2(\mathtt{x}_1,\mathtt{R}(\mathtt{x}_1))[\mathtt{x}_2{:=}\mathtt{x}_4(\mathtt{x}_3)] = \mathtt{R}(\mathtt{x}_1)$ as
   $(\lambda\mathtt{x}_2.\mathtt{x}_2\mathtt{x}_1(\lambda\mathtt{x}_1.\mathtt{R}\mathtt{x}_1))(\lambda\mathtt{x}_3\mathtt{x}_4.\mathtt{x}_4\mathtt{x}_3) \twoheadrightarrow_\beta$
   $(\lambda\mathtt{x}_3\mathtt{x}_4.\mathtt{x}_4\mathtt{x}_3)\mathtt{x}_1(\lambda\mathtt{x}_1.\mathtt{R}\mathtt{x}_1) \twoheadrightarrow_\beta$
   $(\lambda\mathtt{x}_1.\mathtt{R}\mathtt{x}_1)\mathtt{x}_1 \twoheadrightarrow_\beta \mathtt{R}\mathtt{x}_1$.

REMARK 3.9. One might wonder why we use the translation $\widetilde{f}$ instead of $\overline{f}$ for $f \in \mathcal{P}$. At first sight, the translation $\overline{f}$ looks more natural: For e.g. $f = \mathtt{S}(\mathtt{x}_1,\mathtt{x}_2)$, $\widetilde{f} \equiv \mathtt{S}\mathtt{x}_1\mathtt{x}_2$ but $\overline{f} \equiv \lambda\mathtt{x}_1\mathtt{x}_2.\mathtt{S}\mathtt{x}_1\mathtt{x}_2$, and the latter formula seems a logical choice as f is a propositional function.
However, using $\overline{f}$ as translation enforces us to make a choice for the order in which the variables $\mathtt{x}_1$, $\mathtt{x}_2$ are "curried". We would have got problems with the definition of $f[\mathtt{x}_2{:=}\mathtt{a}_2]$: As the variable $\mathtt{x}_1$ is curried "before" $\mathtt{x}_2$ in $\overline{f}$, we would have to remove $\mathtt{x}_1$ first, then carrying out the substitution, and finally adding it again. This is all possible (for instance, we could work with the term $\lambda\mathtt{x}_1.(\lambda\mathtt{x}_1\mathtt{x}_2.\mathtt{S}\mathtt{x}_1\mathtt{x}_2)\mathtt{x}_1\mathtt{a}_2)$, but this method seems less natural than the method using $\widetilde{f}$ proposed above.

REMARK 3.10. $f[x_1,\ldots,x_n{:=}g_1,\ldots,g_n]$ is not always defined. For its existence we need:

- The existence of the normal form H in Definition 3.7. For instance, this normal form does not exist if we choose $f = \mathtt{x}_1(\mathtt{x}_1)$ and $g_1 = \mathtt{x}_1(\mathtt{x}_1)$: then we obtain for the calculation of $f[\mathtt{x}_1{:=}g_1]$ the famous λ-term $(\lambda\mathtt{x}_1.\mathtt{x}_1\mathtt{x}_1)(\lambda\mathtt{x}_1.\mathtt{x}_1\mathtt{x}_1)$;

- The existence of a (unique) h such that $\widetilde{h} \equiv H$.

So for the moment, the Substitution Rule 2.21.6 has to carry as a premise (notation of 2.21.6): "if $f[y{:=}k]$ exists". In the next section we will prove that, as long as we are within the system RTT, both H and h always exist uniquely (Theorem 4.17).

We end with an observation that will help us to simplify proofs at some points:

LEMMA 3.11. *If $f \in \mathcal{P}$ then $\widetilde{f}$ is a λI-term, i.e. for any subterm $\lambda x.M$ of $\widetilde{f}$ we have $x \in \mathrm{FV}(M)$.*

PROOF. An easy induction on the definition of $\widetilde{f}$, using Lemma 3.5. ∎

4. Metaproperties of RTT

In this section we treat some meta-properties of RTT. Since we use the λ-notation of Section 3, we can often refer to known results in typed λ-calculus[20].

THEOREM 4.1. (First Free Variable Theorem)
$\mathrm{FV}(f[x_1,\ldots,x_n{:=}g_1,\ldots,g_n]) = (\mathrm{FV}(f)\setminus\{x_1,\ldots,x_n\})\cup\{g_i \in \mathcal{V}|x_i \in \mathrm{FV}(f)\}$.

PROOF. Write $h = f[x_1,\ldots,x_n{:=}g_1,\ldots,g_n]$. We know that $\widetilde{h}$ is the β-normal form of the λ-term $(\lambda x_1\cdots x_n.\widetilde{f})\overline{g_1}\cdots\overline{g_n}$.
By Lemma 3.11 we know that $\widetilde{f},\overline{g_1},\ldots,\overline{g_n}$ are all λI-terms. Therefore $\widetilde{f}[x_1{:=}\overline{g_1}]\cdots[x_n{:=}\overline{g_n}]$ is also a λI-term. As we have

$$(\lambda x_1\cdots x_n.\widetilde{f})\overline{g_1}\cdots\overline{g_n} \twoheadrightarrow_\beta \widetilde{f}[x_1{:=}\overline{g_1}]\cdots[x_n{:=}\overline{g_n}]$$

we have by the Church-Rosser Theorem that $\widetilde{f}[x_1{:=}\overline{g_1}]\cdots[x_n{:=}\overline{g_n}] \twoheadrightarrow_\beta \widetilde{h}$, and therefore:

$$\begin{aligned}
\mathrm{FV}(h) &\overset{3.5}{=} \mathrm{FV}(\widetilde{h})\\
&\overset{(1)}{=} \mathrm{FV}(\widetilde{f}[x_1{:=}\overline{g_1}]\cdots[x_n{:=}\overline{g_n}])\\
&= (\mathrm{FV}(\widetilde{f})\setminus\{x_1,\ldots,x_n\})\cup\bigcup\{\mathrm{FV}(\overline{g_i})|x_i \in \mathrm{FV}(\widetilde{f})\}\\
&\overset{(2)}{=} (\mathrm{FV}(\widetilde{f})\setminus\{x_1,\ldots,x_n\})\cup\{\overline{g_i} \in V|x_i \in \mathrm{FV}(\widetilde{f})\}\\
&\overset{3.5}{=} (\mathrm{FV}(f)\setminus\{x_1,\ldots,x_n\})\cup\{g_i \in \mathcal{V}|x_i \in \mathrm{FV}(f)\}
\end{aligned}$$

At (1) we use that $\widetilde{f}[x_1{:=}\overline{g_1}]\cdots[x_n{:=}\overline{g_n}]$ is a λI-term that β-reduces to $\widetilde{h}$; at (2) we use the fact that $\mathrm{FV}(\overline{g_i}) = \emptyset$ whenever $g_i \in \mathcal{A}\cup\mathcal{P}$ (by definition of $\overline{g_i}$). ∎

[20]The meta-properties can also be proved directly, without λ-calculus: see [23].

THEOREM 4.2. (Second Free Variable Theorem) *Assume Γ is a context, f is a pseudoformula such that $\Gamma \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$, and $x_1 < \ldots < x_m$ are the free variables of f.*
Then $m = n$ and $x_i : t_i^{a_i} \in \Gamma$ for all $i \leq n$.

PROOF. An easy induction on $\Gamma \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$. For rules 6 and 7, use Theorem 4.1. ■

We can now prove a corollary that we promised in Remark 2.23.7:

COROLLARY 4.3. *If $\Gamma \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$ and φ is a bijection $\{1, \ldots, n\} \to \{1, \ldots, n\}$ then there is a context Γ' and a pseudoformula f' which is α-equal to f such that*

$$\Gamma' \vdash f' : (t_{\varphi(1)}^{a_{\varphi(1)}}, \ldots, t_{\varphi(n)}^{a_{\varphi(n)}})^a.$$

PROOF. By the second Free Variable Theorem, we can assume that f has n free variables $x_1 < \ldots < x_n$, and that $x_i{:}t_i^{a_i} \in \Gamma$ for all $i \in \{1, \ldots, n\}$.
Take n new free variables $z_1 < \ldots < z_n$ such that $z_i > y$ for all $y \in \mathrm{dom}(\Gamma)$.
Now apply rule 7 of RTT n times. ■

We can also prove unicity of types and unicity of orders. Orders are unique in the following sense:

LEMMA 4.4. *Assume $\Gamma \vdash f : t^a$. If $x : u^b \in \Gamma$, then u^b is predicative. Moreover, if also $\Gamma \vdash f : t'^{a'}$, then $a = a'$.*

PROOF. By induction on the derivation of $\Gamma \vdash f : t^a$ one shows that a context can only contain predicative types, and that both a and a' equal one plus the maximum of the orders of all the (free and non-free) variables that occur in f. ■

COROLLARY 4.5. (Unicity of types for pseudoformulas) *Assume, Γ is a context, f is a pseudoformula, $\Gamma \vdash f : t^a$ and $\Gamma \vdash f : u^b$.*
Then $t^a = u^b$.

PROOF. $t = u$ follows from Theorem 4.2; $a = b$ from Lemma 4.4. ■

REMARK 4.6. We can not omit the context Γ in Corollary 4.5. For example, the pseudoformula $\mathtt{z(x)}$ can have different types in different contexts, as is illustrated by the following derivations (we have omitted the orders as they can be calculated via Lemma 4.4):

$$\dfrac{\dfrac{\vdash \mathtt{R(a_1)} : () \qquad \vdash \mathtt{a_1} : 0}{\mathtt{x} : 0 \vdash \mathtt{R(x)} : (0)}\,3}{\mathtt{x} : 0, \mathtt{z} : (0) \vdash \mathtt{z(x)} : (0, (0))}\,4$$

versus

$$\dfrac{\dfrac{\vdash \mathtt{R}(\mathtt{a}_1) : ()}{\mathtt{x} : () \vdash \mathtt{x}() : (())}\,4}{\mathtt{x} : (), \mathtt{z} : (()) \vdash \mathtt{z}(\mathtt{x}) : ((), (()))}\,4$$

Theorem 4.2 and Corollary 4.5 show that our system RTT makes sense, in a certain way: The type of a pseudoformula only depends on the context and does not depend on the way in which we derived the type of that pseudoformula.

Finally, we investigate the problem whether there exists (in the situation of Definition 2.21.6) a pseudoformula h such that $h = f[y{:=}k]$. We show that this is the case, in Theorem 4.17.

The nonexistence of $f[x_i{:=}g_i]$ can have two reasons:

- The λ-term $(\lambda x_1 \cdots x_n.\widetilde{f})\overline{g_1} \cdots \overline{g_n}$ has no β-normal form;
- The λ-term $(\lambda x_1 \cdots x_n.\widetilde{f})\overline{g_1} \cdots \overline{g_n}$ has β-normal form H, but there is no $h \in \mathcal{P}$ such that $\widetilde{h} \equiv H$.

However, these two things do not occur if we use substitution under the restrictions of Definition 2.21.6. For a proof of this we use the simply typed λ-calculus of Church [6] as presented in [3], Section 3.2. This system is called λ-Church, or $\lambda{\rightarrow}$. Especially, $\mathbb{V}$ denotes the set of type variables, and $\mathbb{T}$ denotes the set of simple types in $\lambda{\rightarrow}$. We assume that ι and o are distinct elements of $\mathbb{V}$, which should be interpreted as in [6]: ι is the type of the individuals and o is the type of the propositions.

We write $\vdash_{\lambda\rightarrow}$ to denote derivability in λ-Church.

DEFINITION 4.7. Define for each ramified type t^a a type $T(t^a) \in \mathbb{T}$ as follows:

- $T(0^0) \stackrel{\text{def}}{=} \iota$;
- $T((t_1^{a_1}, \ldots, t_n^{a_n})^a) \stackrel{\text{def}}{=} T(t_1^{a_1}) \rightarrow \ldots \rightarrow T(t_n^{a_n}) \rightarrow o$.

The mapping T is injective when restricted to predicative types:

LEMMA 4.8. *If t^a and u^b are predicative types, and $T(t^a) = T(u^b)$ then $t^a = u^b$.*

PROOF. Induction on the definition of predicative type. ■

DEFINITION 4.9. We define a standard context Γ_0 in $\lambda\!\to$, in which type information on $\vee$, $\neg$ and elements of $\mathcal{A}$ and $\mathcal{R}$ is stored:

$$\begin{aligned}\Gamma_0 \overset{\text{def}}{=} & \{\neg : o \to o, \vee : o \to o \to o\} \cup \\ & \{a : \iota | a \in \mathcal{A}\} \cup \\ & \{\forall_{t^a} : T(t^a) \to o | t^a \text{ is a ramified type}\} \cup \\ & \{R : \underbrace{\iota \to \ldots \to \iota}_{m \text{ times } \iota} \to o | R \in \mathcal{R}, \mathfrak{a}(R) = m\}\end{aligned}$$

If Γ is a context in RTT then $T(\Gamma) \overset{\text{def}}{=} \Gamma_0 \cup \{x : T(t^a) | x{:}t^a \in \Gamma\}$.

As we use explicit typing, the mappings $\tilde{}$ and $\bar{}$ have to be adapted a bit, and become dependant from the context Γ.
In Definition 3.1, $\ell_i = \lambda x_1 \cdots x_n.\widetilde{k_i}$ now changes to

$$\ell_i = \lambda x_1{:}T(t_1^{a_1}) \cdots x_n{:}T(t_n^{a_n}).\widetilde{k_i},$$

where $t_j^{a_j}$ is such that $x_j{:}t_j^{a_j} \in \Gamma$. In the same definition, $\forall_{t^a}\lambda x.\widetilde{f'}$ changes to $\forall_{t^a}\lambda x{:}T(t^a).\widetilde{f'}$. Similar changes have to be made in the definition of $\bar{}$. These changes do not affect the results of Section 3.

THEOREM 4.10. *If* $\Gamma \vdash f : t^a$ *then*

1. $T(\Gamma) \vdash_{\lambda\to} \widetilde{f} : o$;
2. $\emptyset \vdash_{\lambda\to} \overline{f} : T(t^a)$.

PROOF. A straightforward induction on $\Gamma \vdash f : t^a$ with the use of Theorem 4.2 and the Subject Reduction property for λ-Church. ∎

REMARK 4.11. Observe that the above theorem immediately excludes the pseudoformula that leads to the Russell Paradox from the well-typed formulas: If $\neg\mathbf{z}(\mathbf{z})$ were a formula then the λ-term $\neg(\mathbf{zz})$ would be typable in λ-Church.

Using the strong normalization of λ-Church, it is easy to solve the first problem:

THEOREM 4.12. *Assume* $\Gamma \cup \{y{:}t_i^{a_i}\} \vdash f : (t_1^{a_1}, \ldots, t_n^{a_n})^a$ *and* $\Gamma \vdash k : t_i^{a_i}$ *(so: the preconditions of rule 6 of* RTT *are fulfilled). Then* $(\lambda y{:}T(t_i^{a_i}).\widetilde{f})\overline{k}$ *is strongly normalizing.*

PROOF. The theorem is easy for $k \in \mathcal{A}$, so assume $k \in \mathcal{P}$. By Theorem 4.10.2 we know that $\emptyset \vdash \overline{k} : T(t_i^{a_i})$, hence $T(\Gamma) \vdash \overline{k} : T(t_i^{a_i})$ by weakening. As, by Theorem 4.10.1, $T(\Gamma) \cup \{y{:}T(t_i^{a_i})\} \vdash \widetilde{f} : o$ and therefore $T(\Gamma) \vdash \lambda y{:}T(t_i^{a_i}).\widetilde{f} : T(t_i^{a_i}) \to o$, we have $T(\Gamma) \vdash (\lambda y{:}T(t_i^{a_i}).\widetilde{f})\overline{k} : o$, so the term $(\lambda y{:}T(t_i^{a_i}).\widetilde{f})\overline{k}$, being a typable term in $\lambda{\to}$, is strongly normalizing. ■

The second problem is somewhat harder to tackle: substitution (Definition 3.7) is defined in terms of λ-calculus, and not every λ-term H has an equivalent h in $\mathcal{P}$ with $\widetilde{h} \equiv H$. This makes it hard to see what happens, especially in case of a substitution $z(h_1, \ldots, h_m)[x_1, \ldots, z, \ldots, x_n{:=}g_1, \ldots, g, \ldots, g_n]$. For this substitution we must calculate the β-normal form of

$$(\lambda x_1{:}\tau_1 \cdots z{:}\tau \cdots x_n{:}\tau_n.z\overline{h_1} \cdots \overline{h_m})\overline{g_1} \cdots \overline{g} \cdots \overline{g_n}.$$

This term reduces to $\overline{g}H_1 \cdots H_m$ for some H_js. If $g \in \mathcal{P}$ then this new term may not be in β-normal form, and it is not clear what will be the final result (Cf. Examples 3.8.4 and 5.).

The lemma below shows that $\overline{g}H_1 \cdots H_m$ represents, again, a substitution $g[y_1, \ldots, y_m{:=}h_1', \ldots, h_m']$, which is a crucial step in solving the second problem.

LEMMA 4.13. *Let* $f = z(h_1, \ldots, h_m)$, *assume* $\Gamma \vdash f : t^a$, $x_j{:}t_j^{a_j} \in \Gamma$ $(j = 1, \ldots, n)$, $z = x_p$, $g_1, \ldots, g_n \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$; $g = g_p \notin \mathcal{V}$. *Assume also that* $\Gamma \vdash g_j{:}t_j^{a_j}$ *for* $g_j \in \mathcal{A} \cup \mathcal{P}$ *and* $g_j{:}t_j^{a_j} \in \Gamma$ *for* $g_j \in \mathcal{V}$. *Then:*

1. $g \in \mathcal{P}$ *and* g *has exactly* m *free variables (say* $y_1 < \ldots < y_m$*);*
2. $g[y_j{:=}h_j']$ *exists (here* $h_j' = g_\ell$ *if* $h_j = x_\ell$; $h_j' = h_j$ *if* $h_j \notin \{x_1, \ldots, x_n\}$*), if and only if* $f[x_i{:=}g_i]$ *exists, and if they exist, they are equal.*

PROOF.

1. As $\Gamma \vdash f : t^a$, $T(\Gamma) \vdash_{\lambda\to} z\overline{h_1} \cdots \overline{h_m} : o$ by Theorem 4.10. Therefore $z : \tau_1 \to \cdots \to \tau_m \to o \in T(\Gamma)$ (for certain τ_i), so there are $u_1^{b_1}, \ldots, u_m^{b_m}, b$ such that $z : (u_1^{b_1}, \ldots, u_m^{b_m})^b \in \Gamma$. Therefore: $t_p^{a_p} = (u_1^{b_1}, \ldots, u_m^{b_m})^b$, and $\Gamma \vdash g : (u_1^{b_1}, \ldots, u_m^{b_m})^b$. Therefore, $g \in \mathcal{P}$, and by Theorem 4.2, g has m free variables (say: $y_1 < \ldots < y_m$).
2. Due to the definition of the h_j's we have:

$$\begin{aligned}&(\lambda x_1{:}T(t_1^{a_1}) \cdots x_n{:}T(t_n^{a_n}).\widetilde{f})\overline{g_1} \cdots \overline{g_n} \\ &\qquad \equiv \quad (\lambda x_1{:}T(t_1^{a_1}) \cdots x_n{:}T(t_n^{a_n}).z\overline{h_1} \cdots \overline{h_m})\overline{g_1} \cdots \overline{g_n} \\ &\qquad \twoheadrightarrow_\beta \quad \overline{g}\overline{h_1'} \cdots \overline{h_m'}\end{aligned}$$

as all the $\overline{g_i}$s are either closed λ-terms or variables different from the x_is. By (1), g has free variables $y_1 < \ldots < y_m$, so $\overline{g} \equiv \lambda y_1{:}\tau_1 \cdots y_m{:}\tau_m.\widetilde{g}$, where $\tau_j = T(u_j^{b_j})$ and $y_j{:}u_j^{b_j} \in \Gamma$. This means

$$(\lambda x_1{:}T(t_1^{a_1}) \cdots x_n{:}T(t_n^{a_n}).\widetilde{f})\overline{g_1} \cdots \overline{g_n} \\ \twoheadrightarrow_\beta \quad (\lambda y_1{:}\tau_1 \cdots y_m{:}\tau_m.\widetilde{g})\overline{h'_1} \cdots \overline{h'_m}$$

and therefore $f[x_i{:=}g_i] = g[y_j{:=}h'_j]$ whenever $f[x_i{:=}g]$ or $g_p[y_j{:=}h'_j]$ exist.

■

Another property that we need is the so-called subterm property, stating that all pseudoformulas which are recursive arguments of a typable term, are typable (for RA, see Definition 2.11):

DEFINITION 4.14. Assume $\Gamma \vdash f : t^a$. f has the *argument property* ($\mathrm{SP}(f)$) if for all $h \in \mathrm{RA}(f) \cap \mathcal{P}$ there is $\Delta \supseteq \Gamma$ and a predicative type u^b such that $\Delta \vdash h : u^b$.

Lemma 4.13 helps us in proving that the argument property is maintained under substitution:

LEMMA 4.15. *Assume* $\Gamma \vdash f : t^a$, $x_j{:}t_j^{a_j} \in \Gamma$ $(j = 1,\ldots,n)$, $g_1,\ldots,g_n \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$. *Assume also that* $\Gamma \vdash g_j{:}t_j^{a_j}$ *for* $g_j \in \mathcal{A} \cup \mathcal{P}$ *and* $g_j{:}t_j^{a_j} \in \Gamma$ *for* $g_j \in \mathcal{V}$.
Assume $\mathrm{SP}(f)$, *and* $\mathrm{SP}(g_i)$ *for all* $g_i \in \mathcal{P}$. *If* $f[x_i{:=}g_i]$ *exists then:*

1. $\mathrm{RA}(f[x_i{:=}g_i]) \subseteq \mathrm{RA}(f) \cup \{g_i | 1 \leq i \leq n\} \cup \bigcup_{g_i \in \mathcal{P}} \mathrm{RA}(g_i)$;
2. $\mathrm{SP}(f[x_i{:=}g_i])$.

PROOF. Clearly, (2) follows from (1). We prove (1) by induction on a, the order of f. If $a = 0$ then f has no free variables by Theorem 4.2, therefore $f[x_i{:=}g_i] = f$ and there is nothing to prove.

Now assume (1) holds for all orders $a' < a$. Use induction on the structure of f. We treat the only interesting case: $f = z(h_1,\ldots,h_m)$ with $z = x_p$ and $g = g_p \in \mathcal{P}$, and we use notations as in Lemma 4.13. By this lemma, g has exactly m free variables, say $y_1 < \ldots < y_m$, and $f[x_i{:=}g_i] = g[y_j{:=}h'_j]$. Notice that $z \in \mathrm{FV}(f)$ and therefore $a_p < a$ (a_p is the order of $x_p = z$ and g). We want to use the induction hypothesis, so we check that the y_js and the h'_js have the same type.

g is typable in Γ, so there are $u_j^{b_j}$ such that $y_j{:}u_j^{b_j} \in \Gamma$. Focus on the h'_j.

- If $h'_j \in \mathcal{A}$ then $\Gamma \vdash h'_j{:}0^0$;
- If $h'_j \in \mathcal{V}$ then $h'_j \in \mathrm{dom}(\Gamma)$: if $h'_j = g_\ell$ then this is by assumption; if $h'_j = h_j$ then this is because $h_j \in \mathrm{FV}(f)$.
- If $h'_j \in \mathcal{P}$ then $\Gamma \vdash h'_j{:}u'^{b'_j}_j$ for some predicative type $u'^{b'_j}_j$: if $h'_j = g_\ell$ then this is by assumption; if $h'_j = h_j$ then we use the fact $\mathrm{SP}(f)$.

In any case, h'_j is typable, let $u'^{b'_j}_j$ be its (predicative) type in Γ. Now look at the λ-terms $(\lambda x_1 \cdots x_n.\widetilde{f})\overline{g_1} \cdots \overline{g_n}$ and $\overline{g}\overline{h'_1} \cdots \overline{h'_m}$ in the proof of Lemma 4.13.2. By Theorem 4.10, $T(\Gamma) \vdash \widetilde{f} : o$. As the x_is and the g_is have the same type, we have:

$$T(\Gamma) \vdash_{\lambda\to} (\lambda x_1{:}T(t_1^{a_1}) \cdots x_n{:}T(t_n^{a_n}).\widetilde{f})\overline{g_1} \cdots \overline{g_n} : o$$

and by Subject Reduction for $\lambda\!\to$:

$$T(\Gamma) \vdash_{\lambda\to} \overline{g}\overline{h'_1} \cdots \overline{h'_m} : o,$$

therefore $T(\Gamma) \vdash \overline{h'_j} : T(u_j^{b_j})$. As h'_j has type $u'^{b'_j}_j$, we also have $T(\Gamma) \vdash \overline{h'_j} : T(u'^{b'_j}_j)$, so by unicity of types for $\lambda\!\to$: $T(u_j^{b_j}) = T(u'^{b'_j}_j)$, and as both types are predicative, $u_j^{b_j} = u'^{b'_j}_j$ by Lemma 4.8. So: The h'_js and the y_js have the same type.
$\mathrm{SP}(g)$ holds by assumption and as the h'_js are either arguments of f or equal to one of the g_ℓs, $\mathrm{SP}(h'_j)$ holds for all $h'_j \in \mathcal{P}$. This allows us to use the induction hypothesis:

$$\mathrm{RA}(g[y_j{:=}h'_j]) \subseteq \mathrm{RA}(g) \cup \{h'_j | 1 \le j \le m\} \cup \bigcup_{h'_j \in \mathcal{P}} \mathrm{RA}(h'_j)$$

and as $f[x_i{:=}g_i] = g[y_j{:=}h'_j]$ we have[21]

$$\begin{aligned} \mathrm{RA}(f[x_i{:=}g_i]) &\subseteq \mathrm{RA}(g) \cup \{h'_j | 1 \le j \le m\} \cup \bigcup_{h'_j \in \mathcal{P}} \mathrm{RA}(h'_j) \\ &\subseteq \mathrm{RA}(f) \cup \{g_i | 1 \le i \le n\} \cup \bigcup_{g_i \in \mathcal{P}} \mathrm{RA}(g_i) \end{aligned}$$

■

Now the technical work has been done:

[21]Note that $\mathrm{RA}(g) \subseteq \bigcup_{g_i \in \mathcal{P}} \mathrm{RA}(g_i)$, $\{h'_j | 1 \le j \le m\} \subseteq \mathrm{RA}(f) \cup \{g_i | 1 \le i \le n\}$ and $\bigcup_{h'_j \in \mathcal{P}} \mathrm{RA}(h'_j) \subseteq \mathrm{RA}(f) \cup \bigcup_{g_i \in \mathcal{P}} \mathrm{RA}(g_i)$.

COROLLARY 4.16. (Subterm Lemma) *If* $\Gamma \vdash f : t^a$ *then* SP(f).

PROOF. Induction on $\Gamma \vdash f : t^a$. All cases are easily checked except for the substitution rule, which is proved with Lemma 4.15. ■

COROLLARY 4.17. (Strong Normalisation) *Assume* $\Gamma \vdash f : t^a$, $x_i{:}t_i^{a_i} \in \Gamma$ $(i = 1,\dots,n)$, $g_1,\dots,g_n \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$. *Assume also that* $\Gamma \vdash g_i{:}t_i^{a_i}$ *for* $g_i \in \mathcal{A} \cup \mathcal{P}$ *and* $g_i{:}t_i^{a_i} \in \Gamma$ *for* $g_i \in \mathcal{V}$.
Then $f[x_i{:=}g_i]$ *exists.*

PROOF. We use induction on a. If $a = 0$ then f contains no free variables, and $f[x_i{:=}g_i] = f$. Now assume the corollary is proved for all $a' < a$. Use induction on the structure of f. The only interesting case is $f = z(h_1,\dots,h_m)$ with $z = x_p$ and $g = g_p \in \mathcal{P}$.

Reasoning as in the proof of Lemma 4.15, we find that $\Gamma \vdash g{:}t_p^{a_p}$, $y_j{:}u_j^{b_j} \in \Gamma$ $(j = 1,\dots,m)$, $h'_1,\dots,h'_m \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$. Using Corollary 4.16, we know that the h'_js are typable and by similar reasoning as in the proof of 4.15, we find that $\Gamma \vdash h'_j{:}u_j^{b_j}$ for $h'_j \in \mathcal{A} \cup \mathcal{P}$ and $h'_j{:}u_j^{b_j} \in \Gamma$ for $h'_j \in \mathcal{V}$.

As the order of g is $< a$, we can use the induction hypothesis and conclude: $g[y_j{:=}h'_j]$ exists. Now we can use Lemma 4.13.2 to conclude: $f[x_i{:=}g_i]$ exists. ■

It is important to notice that Corollary 4.17 can be applied in the case of the substitution rule of RTT, so that the existence of $f[y{:=}k]$ is always guaranteed.

5. Formulas in the Principia Mathematica

We recall Definition 2.22: a pseudoformula f is called a *formula* if $\Gamma \vdash f : t^a$ for some Γ and t^a. We will check whether this definition of formula coincides with the definition of formula that was given in the Principia. For this purpose we prove a number of lemmas concerning the relation between formulas and predicative types.

We do not distinguish between pseudoformulas that are α-equal, nor between types $(t_1,\dots,t_n)$ and $(t_{\varphi(1)},\dots,t_{\varphi(n)})$ for a bijection φ. This is justified by Corollary 4.3 and by the fact, that formulas that are α-equal are supposed to be the same in the Principia, too.

We define the notion "up to α-equality" formally:

DEFINITION 5.1. Let $f \in \mathcal{P}$, Γ a context, t^a a type. f is of type t^a in the context Γ *up to α-equality*, notation $\Gamma \vdash f : t^a(\text{mod }\alpha)$, if there is $f' \in \mathcal{P}$, a context Γ' and a bijection $\varphi : \mathcal{V} \to \mathcal{V}$ such that

- $\Gamma' \vdash f' : t^a$;
- f' and f are α-equal via the bijection φ;
- $x < y \Rightarrow \varphi(x) < \varphi(y)$ for all $x, y \in \text{dom}(\Gamma)$;
- $\Gamma' = \{\varphi(x){:}u^b | x{:}u^b \in \Gamma\}$.

We say that f is a formula in the context Γ *up to α-equality* if there is a type u^b such that $\Gamma \vdash f : u^b(\text{mod }\alpha)$.
We say that f is a formula *up to α-equality* if there is a context Γ such that f is a formula in Γ up to α-equality.

The following lemma states that all predicative types are "inhabited":

LEMMA 5.2. *If t^a is predicative then there are f and Γ such that $\Gamma \vdash f : t^a$.*

PROOF. We use induction on predicative types. For the sake of clarity, we leave out the orders in this proof. This is justified by Lemma 4.4.

The case $t = 0$ is trivial.

Now assume $t = (t_1, \dots, t_m)$. By induction we have for all $i \leq m$: $\Gamma_i \vdash f_i : t_i$. Take variables $z_1 < \dots < z_m$ such that $x < z_i$ for all $x \in \text{dom}(\Gamma_i)$.

Take a fixed i. We shall find a context Δ_i and a formula g_i such that $\Delta_i \vdash g_i{:}(t_i)$. Let $x_1 < \dots < x_n$ be the free variables of f_i. Distinguish 2 cases:

- $t_i = 0$. Then make the following derivation:

$$\frac{\Gamma_i \vdash \mathtt{R}(f_i) : () \qquad \Gamma_i \vdash f_i : 0}{\Gamma_i, z_i{:}0 \vdash \mathtt{R}(z_i) : (0)}3$$

  Write $\Delta_i = \Gamma_i \cup \{z_i{:}0\}$, and $g_i = \mathtt{R}(z_i)$, then $\Delta_i \vdash g_i{:}(t_i)$.

- $t_i \neq 0$. Because of Theorem 4.2, there are types $u_1, \dots, u_n$ such that $x_j : u_j \in \Gamma_i$ and $t_i = (u_1, \dots, u_n)$.
  Now use rule 4 of RTT:

$$\frac{\Gamma_i \vdash f_i : t_i}{\Gamma_i, z_i{:}t_i \vdash z_i(x_1, \dots, x_n) : (u_1, \dots, u_n, t_i)}4 \qquad (*)$$

and rule 8 n times:

$$\Gamma_i \setminus \{x_j{:}u_j | 1 \leq j \leq n\} \vdash \forall x_1{:}u_1.\cdots\forall x_n{:}u_n.z_i(x_1,\ldots,x_n) : (t_i)$$

Write $\Delta_i = \Gamma_i \setminus \{x_j{:}u_j | 1 \leq j \leq n\}$ and

$$g_i = \forall x_1{:}u_1.\cdots\forall x_n{:}u_n.z_i(x_1,\ldots,x_n).$$

For arbitrary $i = 1,\ldots,m$ we now have: $\Delta_i \vdash g_i : (t_i)$.
We can assume that $x < y$ for $x \in \mathrm{dom}(\Delta_i)$, $y \in \mathrm{dom}(\Delta_j)$ with $i < j$.
Write $\Delta = \Delta_1 \cup \ldots \cup \Delta_m$.
Now apply rule 2 consecutively $m-1$ times, to obtain:

$$\Delta \vdash g_1 \vee (g_2 \vee \ldots \vee (g_{m-1} \vee g_m)\ldots) : (t_1,\ldots,t_m)$$

■

We can use the same techniques as in the preceding proof to show that $z(k_1,\ldots,k_m)$ is a formula if $k_1,\ldots,k_m$ are either formulas or variables, and z is "fresh".

LEMMA 5.3. *If $k_1,\ldots,k_n \in \mathcal{A} \cup \mathcal{V} \cup \mathcal{P}$, $t^a = (t_1^{a_1},\ldots,t_n^{a_n})^a$ is a predicative type, $\Gamma \vdash k_i : t_i^{a_i}$ for all $k_i \in \mathcal{A} \cup \mathcal{P}$ and $k_i : t_i^{a_i} \in \Gamma$ for all $k_i \in \mathcal{V}$, and $z \in \mathcal{V} \setminus \mathrm{dom}(\Gamma)$, then $z(k_1,\ldots,k_n)$ is a formula in the context $\Gamma \cup \{z : t^a\}$ (up to α-equality).*

PROOF. First, we make a derivation of

$$\{z{:}t^a\} \cup \{x_i{:}t_i^{a_i} | 1{\leq}i{\leq}n\} \vdash z(x_1,\ldots,x_n) : (t_1^{a_1},\ldots,t_n^{a_n},t^a)^{a+1} (\mathrm{mod}\ \alpha),$$

similarly to the derivation of $(*)$ in the proof of Lemma 5.2. Next, find (with Lemma 5.2) $k'_1,\ldots,k'_m$ such that

- $k'_i = k_i$ if $k_i \in \mathcal{A} \cup \mathcal{P}$;
- $k'_i \in \mathcal{A} \cup \mathcal{P}$ has type $t_i^{a_i}$ in a context Δ_i if $k_i \in \mathcal{V}$;
- Δ, the union of the contexts Γ and the Δ_is, is a context.
- For $k_i \in \mathcal{V}$: k'_i and k'_j are α_Γ-equal if and only if $k_i = k_j$.[22]

[22]One might wonder whether there are enough pseudoformulas of one type that are not α_Γ-equal. Lemma 5.2 provides only one pseudoformula for each type. But if we have that pseudoformula, say k, then we use rule 2 of RTT to create $\neg k$, $\neg\neg k$, $\neg\neg\neg k$, etc. k, $\neg k$, $\neg\neg k$,... are all α_Γ-different and of the same type as k.

Apply rule 6 m times (as in the proof of Lemma 5.2), and where necessary the weakening rule, to obtain:

$$\Delta \vdash z(k'_1, \ldots, k'_m) : (t^a)^{a+1}(\text{mod } \alpha).$$

Now introduce, with rule 3, new variables for the k'_i which are not equal to k_i, to obtain a formula that is α-equal to $z(k_1, \ldots, k_m)$. ■

It is also not hard to show that $f \vee g$ is a formula if f and g are so (see also Remark 2.24):

LEMMA 5.4. *If f and g are formulas in contexts Γ_1 and Γ_2, respectively, and $\Gamma_1 \cup \Gamma_2$ is a context, then $f \vee g$ is a formula in the context $\Gamma_1 \cup \Gamma_2$ (up to α-equality).*

PROOF. For reasons of clarity, we again leave out the orders of the ramified types. We can not simply apply rule 2 of RTT, as the contexts Γ_1 and Γ_2 may not obey to the condition on them in rule 2. Assume, $\Gamma_1 \vdash f : (t_1, \ldots, t_m)$, $\Gamma_2 \vdash g : (u_1, \ldots, u_n)$, and $x_1 < \ldots < x_m$ and $y_1 < \ldots y_n$ are the free variables of f and g, respectively. Write $t = (t_1, \ldots, t_m)$ and $u = (u_1, \ldots, u_n)$.
Take variables $x'_1 < \ldots x'_m < z_1 < y'_1 < \ldots < y'_n < z_2$ not occurring in the domain of $\Gamma_1 \cup \Gamma_2$; let

$$\Delta_1 = \{z_1{:}t, x'_1{:}t_1, \ldots x'_m{:}t_m\};$$

$$\Delta_2 = \{z_2{:}u, y'_1{:}u_1, \ldots y'_n{:}u_n\}.$$

Similar to the derivation $(*)$ in the proof of Lemma 5.2, we can derive

$$\Delta_1 \vdash z_1(x'_1, \ldots, x'_m) : (t_1, \ldots, t_m, t);$$

$$\Delta_2 \vdash z_2(y'_1, \ldots, y'_n) : (u_1, \ldots, u_n, u).$$

As Δ_1 and Δ_2 obey to the conditions of rule 2 of RTT, we can derive

$$\Delta_1 \cup \Delta_2 \vdash z_1(x'_1, \ldots, x'_m) \vee z_2(y'_1, \ldots, y'_n) : (t_1, \ldots, t_m, t, u_1, \ldots, u_n, u)$$

With similar techniques as in the proof of Lemma 5.3 we can now derive

$$\Gamma_1 \cup \Gamma_2 \cup \{z_1{:}t, z_2{:}u\} \vdash z_1(x_1, \ldots, x_m) \vee z_2(y_1, \ldots, y_n) : v(\text{mod } \alpha)$$

for a certain type v.
Use rule 6 twice: Substitute f for z_1 and substitute g for z_2. This gives a derivation of $f \vee g$ in the context $\Gamma_1 \cup \Gamma_2$. ■

The following lemma is easy to prove and will be used in the proof of the main result of this section.

LEMMA 5.5. *If $R(i_1,\ldots,i_{\mathfrak{a}(R)})$ is a pseudoformula with free variables $x_1 < \ldots < x_m$, then it is a formula in the context $\{x_j{:}0 \mid 1 \le j \le m\}$.*

PROOF. Write $f = R(i_1,\ldots,i_{\mathfrak{a}(R)})$. Let $a_1,\ldots,a_m \in \mathcal{A}$ be m different individuals that do not occur in f, and replace each variable x_j in f by a_j, calling the result f'.
By the first rule of RTT, f' is a formula in the empty context.
Re-introducing the variables $x_1,\ldots,x_m$ (by applying rule 3 of RTT m times) for the individuals $a_1,\ldots,a_m$, respectively, we obtain that f is a formula in the context $\{x_j{:}0 \mid 1 \le j \le m\}$. ∎

Finally, we can give a characterization of the pseudoformulas which are formulas:

THEOREM 5.6. *Let $f \in \mathcal{P}$. f is a formula (mod α) if and only if:*

- *$f = R(i_1,\ldots,i_{\mathfrak{a}(R)})$, or*
- *$f = z(k_1,\ldots,k_n)$, $z \neq k_j$ for all $k_j \in \mathcal{V}$ and z does not occur in any $k_j \in \mathcal{P}$, and there is Γ with $\mathrm{FV}(f) \supseteq \Gamma$ and for all $k_j \in \mathcal{P}$, $\Gamma \vdash k_j{:}t_j^{a_j}$ for some predicative type $t_j^{a_j}$, or*
- *$f = \neg f'$ and f' is a formula (mod α) or*
  $f = f_1 \vee f_2$, there are Γ_i and $t_i^{a_i}$ such that $\Gamma_i \vdash f_i{:}t_i^{a_i}$ (mod α) and $\Gamma_1 \cup \Gamma_2$ is a context, or
- *$f = \forall x{:}t^a.f'$ and f' is a formula.*

PROOF. Use induction on the structure of f:

- $f = R(i_1,\ldots,i_{\mathfrak{a}(R)})$. This is Lemma 5.5
- $f = z(k_1,\ldots,k_n)$. "$\Leftarrow$" is Lemma 5.3. "$\Rightarrow$": As $T(\Gamma) \vdash z\overline{k_1}\cdots\overline{k_n} : o$ (Theorem 4.10), $z : (u_1^{b_1},\ldots,u_n^{b_n})^b$ for a predicative $(u_1^{b_1},\ldots,u_n^{b_n})^b$, and $T(\Gamma) \vdash \overline{k_j} : T(u_j^{b_j})$. If $z = k_j$ then $z = \overline{k_j}$ and therefore $z{:}u_j^{b_j} \in \Gamma$, which is impossible. By Corollary 4.16, each $k_j \in \mathcal{P}$ is typable in Γ, and as $T(\Gamma) \vdash \overline{k_j} : T(u_j^{b_j})$ and the type of k_j is predicative, $\Gamma \vdash k_j{:}u_j^{b_j}$. Notice that $b_j < b$, so it is impossible that z occurs in a $k_j \in \mathcal{P}$.
- "$\Leftarrow$" is Rule 2 of RTT (for $\neg$) and Lemma 5.4 (for $\vee$). "$\Rightarrow$" (for $\vee$; the proof for $\neg$ is similar): Let Δ be the context containing all the variables of f (also those that are bound by a quantifier; we can assume that different quantifiers bind different variables) and their types. f

is built from several pseudoformulas of the form $R(i_1, \dots, i_{\mathfrak{a}(R)})$ and $z(k_1, \dots, k_m)$ (we will call these pseudoformulas the *constituents* of f), and the logical connectives $\neg$, $\vee$ and $\forall$. Reasoning as in the "$\Rightarrow$" part of the first two cases of the proof of this lemma, we can show the preconditions for Lemma 5.5 (for constituents of the form $R(i_1, \dots, i_{\mathfrak{a}(R)})$) and Lemma 5.3 (for constituents of the form $z(k_1, \dots, k_m)$). Applying these Lemmas, we find that any constituent h of f is typable in Δ. Using Rule 2 of RTT (for $\neg$), Rule 8 of RTT (for $\forall$) and Lemma 5.4 (for $\vee$), we find that f itself is typable.

- "$\Leftarrow$" is Rule 8 of RTT. "$\Rightarrow$" is similar to "$\Rightarrow$" in the previous case.

■

We can now answer the question whether our formulas (as defined in 2.22) are the same as the formulas of the Principia.

First of all, we must notice that all the formulas from Definition 2.22 are also formulas of the Principia: This was motivated directly after we defined system RTT.

Moreover, we proved (in 5.6) that if f is a pseudoformula, then the only reasons why f cannot be a formula (according to Definition 2.22) are:

- There is a constituent $z(k_1, \dots, k_m)$ of f in which z occurs in one of the k_i's;
- There is a constituent $z(k_1, \dots, k_m)$ of f and a $j \in \{1, \dots, m\}$ such that k_j is a pseudoformula, but not a formula.

Pseudoformulas of the first type cannot be formulas in the Principia, because of the vicious circle principle. The same holds for pseudoformulas of the second type, because also in the Principia, arguments cannot be untyped.

We conclude that we have described the formulas of the Principia Mathematica with the formal system RTT.

6. Conclusions

The main part of the Principia is devoted to the development of logic and mathematics using the formulas of the ramified type theory. It appears that the system isn't easy to use. The main reason for this is the use of the ramification. We illustrate this with an example.

EXAMPLE 6.1. (Equality) One tends to define the notion of *equality* in the style of Leibniz [15]:

$$\mathsf{x} = \mathsf{y} \overset{\text{def}}{\leftrightarrow} \forall \mathsf{z}.\mathsf{z}(\mathsf{x}) \leftrightarrow \mathsf{z}(\mathsf{y}),$$

or in words: Two individuals are equal if and only if they have exactly the same properties.

Unfortunately, in order to express this general notion in our formal system, we have to incorporate *all* formulas $\forall \mathsf{z} : (0^0)^n.\mathsf{z}(\mathsf{x}) \leftrightarrow \mathsf{z}(\mathsf{y})$ for $n > 1$.

Russell and Whitehead tried to solve this problem with the so-called *axiom of reducibility*.

AXIOM 6.2. (Axiom of Reducibility) *For each formula f, there is a formula g with a* predicative *type such that f and g are (logically) equivalent.*

Accepting this axiom, one may define equality on formulas of order 1 only:

$$\mathsf{x} = \mathsf{y} \overset{\text{def}}{\leftrightarrow} \forall \mathsf{z} : (0^0)^1.\mathsf{z}(\mathsf{x}) \leftrightarrow \mathsf{z}(\mathsf{y}).$$

If f is a function of type $(0^0)^n$ for some $n > 1$, and a and b are individuals for which $\mathsf{a} = \mathsf{b}$ holds then also $f(\mathsf{a}) \leftrightarrow f(\mathsf{b})$ holds: With the Axiom of Reducibility we can determine a predicative function g (so: of type $(0^0)^1$), equivalent to f. As g has order 1, $g(\mathsf{a}) \leftrightarrow g(\mathsf{b})$ holds. And because f and g are equivalent, also $f(\mathsf{a}) \leftrightarrow f(\mathsf{b})$ holds.

In the introduction to the 2nd edition, Whitehead and Russell admit: "*This axiom has a purely pragmatic justification: it leads to the desired results, and to no others. But clearly it is not the sort of axiom with which we can rest content.*" Moreover, in [31], Weyl states that "*if the properties are constructed there is no room for an axiom here; it is a question which ought to be decided on ground of the construction*", and that "*with his axiom of reducibility Russell therefore abandoned the road of logical analysis and turned from the constructive to the existential-axiomatic standpoint*".

The reason for the introduction of ramified type theory was to keep away the paradoxes that appeared in Frege's Begriffschrift. Ramsey [25] and Hilbert and Ackermann [19] divide these paradoxes in two parts:

- The contradictions that are removed by pointing out that (because of the Vicious Circle Principle 1.1) a propositional function cannot significantly take itself as argument, such as the antinomies of Russell, Burali-Forti and Cantor.

- The contradictions resulting from ambiguities of language (Richard, Epimenides). These contradictions are based on propositions that are partly expressed in the formal language, and partly in the meta-language over this formal language.

Now Ramsey and Hilbert and Ackermann showed that the first kind of paradoxes can be prevented *without* the use of orders in type theory. Moreover, they made the second class of paradoxes non-existing by separating the notion of language and metalanguage. Hence, they could do without the orders, thus obtaining the so-called *simple* type theory. Simple type theory has become the standard since the late twenties. See for example Church's formulation of the simple theory of types [6].

However, using the simple type theory, the Vicious Circle Principle is violated in another sense (for instance, first order and second order propositions are not distinguished any more)[23]. Some philosophical justification for this is given by Gödel (see [17]) and Quine ([24], sections 34 and 35).

The solution suggested by Ramsey and Hilbert and Ackermann and the development and the succes of the Simple Theory of Types insinuates that it is the *orders* (the ramification) that makes Russell's system unsuitable for practical use. In [21], however, it is shown that it is the *combination* of (simple) types and orders that causes the troubles. There are usefull type-free systems (like the system of [22]) in which a structure with close relation to the order-structure of the Ramified Theory of Types is present.

The ramified type theory not only prevents the paradoxes of Frege's Begriffschrift, but also guarantees the well-definedness of substitution, as we showed in Section 4.

There is a narrow relation between this substitution and β-reduction in λ-calculus (Section 3), and RTT has characteristics that are also the basic properties of modern type systems for λ-calculus (Thinning Lemma 2.26.2, Free Variable Theorem 4.2, Unicity of Types 4.5, Subterm Lemma 4.16 and Strong Normalisation 4.17). As there is no real reduction in RTT, we don't have an equivalent of the Subject Reduction theorem. However, the fact that the Free Variable property 4.2 is maintained under substitution can be seen as a (very weak) form of Subject Reduction.

The Ramified Theory of Types is a restrictive system for the development of mathematics, as only predicative formulas are allowed, whereas also impredicative formulas are used throughout mathematics. The first solution to this problem as given in [32] by postulating the Axiom of Reducibility is not acceptable. The second solution is the simple theory of types. By omitting the orders, RTT can be appended to become a formal system for this simple type theory.

With the following remark, we express our personal admiration for the au-

[23]See the motivation by Whitehead and Russell for the introduction of orders in the Principia [32], Introduction, Chapter II, Section V, p. 48–50

thors of the Principia: Although the description of the Ramified Theory of Types in the Principia is very informal, it is remarkable that an accurate formalization of this system can be made (see Theorem 5.6 and the discussion that follows it). The formalization shows that Russell and Whitehead's ideas on the notion of types, though very informal to modern standards, must have been very thorough and to the point.

We discussed the type theory of the Principia Mathematica in this paper, and didn't look more than superficially at other, more modern type systems. We are currently investigating an embedding of RTT in the type system that is at the basis of the proof checker Nuprl (see [20], [8]). This type system is predicative, and the embedding of RTT in Nuprl should maintain predicativity. Logic in Nuprl is based on propositions-as-types, so the embedding also results in a propositions-as-types description of RTT.

Acknowledgements

We would like to thank the following people for their help and their suggestions: Stuart Allen, Roel Bloo, Bob Constable, Herman Geuvers, Kees Hemerik (who gave the idea to make the above formalization), Fairouz Kamareddine, Eric Poll (for suggesting the lambda notation of Section 3) and the referees.

References

[1] S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, 1992, *Handbook of Logic in Computer Science, Volume 2: Background: Computational Structures.* Oxford Science Publications.

[2] H.P. Barendregt, 1984, *The Lambda Calculus: its Syntax and Semantics,* North-Holland, Amsterdam, revised edition.

[3] H.P. Barendregt, 1992, Lambda calculi with types, In [1], Oxford University Press, 117–309.

[4] P. Benacerraf and H. Putnam, editors, 1983, *Philosophy of Mathematics,* Cambridge University Press, second edition.

[5] E.W. Beth, 1959, *The Foundations of Mathematics,* Studies in Logic and the Foundations of Mathematics, North-Holland, Amsterdam.

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

[7] A. Church, 1976, Comparison of Russell's resolution of the semantic antinomies with that of Tarski, *The Journal of Symbolic Logic* **41**, 747–760.

[8] R.L. Constable et al, 1986, *Implementing Mathematics with the Nuprl Proof Development System*, Prentice-Hall, New Jersey.

[9] H.B. Curry and R. Feys, 1958, *Combinatory Logic*, volume I of *Studies in Logic and the Foundations of Mathematics*, North-Holland, Amsterdam.

[10] H.B. Curry, J.R. Hindley, and J.P. Seldin, 1972, *Combinatory Logic*, volume II of *Studies in Logic and the Foundations of Mathematics*, North-Holland, Amsterdam.

[11] A.A. Fraenkel, Y. Bar-Hillel, and A. Levy, second edition, 1973, *Foundations of Set Theory*, Studies in Logic and the Foundations of Mathematics 67, North Nolland, Amsterdam.

[12] G. Frege, 1879, *Begriffschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*, Nebert, Halle. Also in [18].

[13] G. Frege, 1892, *Grundgesetze der Arithmetik, begriffsschriftlich abgeleitet*, volume I, Jena. Reprinted 1962 (Olms, Hildesheim).

[14] G. Frege, 1903, *Grundgesetze der Arithmetik, begriffsschriftlich abgeleitet*, volume II, Pohle, Jena. Reprinted 1962 (Olms, Hildesheim).

[15] C.I. Gerhardt, editor, 1890, Die philosophischen Schriften von Gottfried Wilhelm Leibniz, Berlin.

[16] K. Gödel, 1931, Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I, *Monatshefte für Mathematik und Physik* **38**, 173–198, German; English translation in [18].

[17] K. Gödel, 1944, Russell's mathematical logic, in P.A. Schlipp, editor, *The Philosophy of Bertrand Russell*, Evanston & Chicago, Northwestern University. Also in [4].

[18] J. van Heijenoort, editor, 1967, *From Frege to Gödel: A Source Book in Mathematical Logic*, Harvard University Press, Cambridge, Massachusetts, 1879–1931.

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

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

[21] F. Kamareddine and T. Laan, 1996, A reflection on Russell's ramified types and Kripke's hierarchy of truths, *Journal of the Interest Group in Pure and Applied Logic* **4**, 195–213.

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

[23] T.D.L. Laan, 1994, A formalization of the Ramified Type Theory, Technical Report 33, TUE Computing Science Reports, Eindhoven University of Technology.

[24] W. Van Orman Quine, 1963, *Set Theory and its Logic*, Harvard University Press, Cambridge, Massachusetts.

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

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

[27] B. Russell, 1908, Mathematical logic as based on the theory of types, *American Journal of Mathematics* **30**. Also in [18].

[28] B. Russell, 1919, *Introduction to Mathematical Philosophy*, Allen & Unwin, London.

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

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

[31] H. Weyl, 1946, Mathematics and logic: A brief survey serving as preface to a review of "The Philosophy of Bertrand Russell", *American Mathematical Monthly.*

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

DEPARTMENT OF MATHEMATICS AND COMPUTING SCIENCE
EINDHOVEN UNIVERSITY OF TECHNOLOGY
P.O.BOX 513, 5600 MB EINDHOVEN
THE NETHERLANDS
{laan,wsinrpn}@win.tue.nl