Remarks on Isomorphisms in Typed Lambda Calculi with Empty and Sum Types*†

Marcelo Fiore‡
Computer Laboratory
University of Cambridge

Roberto Di Cosmo
PPS - Université Paris 7
and
INRIA-Roquencourt

Vincent Balat
PPS - Université Paris 7

Abstract

Tarski asked whether the arithmetic identities taught in high school are complete for showing all arithmetic equations valid for the natural numbers. The answer to this question for the language of arithmetic expressions using a constant for the number one and the operations of product and exponentiation is affirmative, and the complete equational theory also characterises isomorphism in the typed lambda calculus, where the constant for one and the operations of product and exponentiation respectively correspond to the unit type and the product and arrow type constructors. This paper studies isomorphisms in typed lambda calculi with empty and sum types from this viewpoint. Our main contribution is to show that a family of so-called Wilkie-Gurevič identities, that plays a pivotal role in the study of Tarski's high school algebra problem, arises from type-theoretic isomorphisms. We thus close an open problem by establishing that the theory of type isomorphisms in the presence of product, arrow, and sum types (with or without the unit type) is not finitely axiomatisable. Further, we observe that for type theories with arrow, empty and sum types the correspondence between isomorphism and arithmetic equality generally breaks down, but that it still holds in some particular cases including that of type isomorphism with the empty type and equality with zero.

Introduction

We study isomorphisms in typed lambda calculi with empty and sum types from the viewpoint of programming-language and type theory, category theory, and mathematical logic.

Type isomorphism and programming languages. Two data types are isomorphic if it is possible to convert data between them without loss of information. The equivalence relation on types induced by the notion of isomorphism allows to abstract from inessential details in the representation of data in programming languages.

In Functional Programming, type isomorphisms provide a means to search functions by type (see [24, 25, 23, 26, 12, 11, 13]) and to match modules by specifications [1]. In proof assistants they are used to find proofs in libraries up to irrelevant syntactical details [10, 4].

*Preliminary version in *Proceedings of the 17th Symposium on Logic in Computer Science (LICS)*, pages 147–156. IEEE, Computer Society Press, 2002.

†Part of this work was done during a visit of Marcelo Fiore to PPS - Université Paris 7 in July 2001 supported by the CNRS.

‡Research supported by an EPSRC Advanced Research Fellowship.

A characterisation of type isomorphisms has been obtained for monomorphic type systems with various combinations of the unit, product, and arrow type constructors [29, 6, 5], as well as for ML-style [11] and second-order polymorphism [14], and for linear lambda calculus [30] and multiplicative linear logic [2].

Type isomorphism and category theory. Type isomorphism in foundational theories of functional programming languages, like typed lambda calculi, can be studied by their associated categorical models. From this perspective, our investigations fall in the context of Lawvere and Schanuel's *Objective Number Theory* [27, 28], which is the study of addition, multiplication, and exponentiation of objects in suitable categories (see also [21, § 2]). Indeed, we will be considering the equational theory of arithmetic expressions in the objective number theory of free bicartesian closed categories and relating it to that of the category of finite sets, where numerical equalities acquire a combinatorial meaning given by isomorphisms that provide so-called *combinatorial* or *bijective* proofs.

Type isomorphism and Tarski's high school algebra problem. There is a connection between the characterisation of type isomorphisms in typed lambda calculi and some results in mathematical logic related to Tarski's high school algebra problem [15]: for types built out of type constructors chosen amongst the unit, product, and arrow, two types are isomorphic if and only if their associated arithmetic expressions (obtained by interpreting the unit by the number one, product by multiplication, and arrow by exponentiation) are equal in the standard model of natural numbers. In these cases, type isomorphism (and numerical equality) is finitely axiomatisable and decidable; hence so is the equational theory of isomorphisms in cartesian closed categories. In the same vein, Soloviev [30], gave a complete axiomatisation of isomorphisms in symmetric monoidal closed categories, and Dosen and Petric [16] provided the arithmetic structure that exactly corresponds to these isomorphisms.

The question has been open as to whether such correspondence was limited to the case of the well-behaved unit, product, and arrow type constructors and, in particular, if it could be extended to more problematic types involving the empty type and the sum type constructor. (From a practical perspective, one is interested in knowing whether type isomorphisms in the presence of sums are finitely axiomatisable, a definitive advantage when implementing decision procedures in library search tools like those described in [24, 13].) To investigate these problems we devised a method for establishing normal forms in the typed lambda calculus with empty and sum types [17, 3] and used it to study type isomorphism in this setting; this was crucial for establishing (a) below.

Summary of results and organisation of the paper.

(a) We show that a family of so-called Wilkie-Gurevič identities arise from type-theoretic isomorphisms and subsequently establish that the equational theory of type isomorphisms in the presence of the product, arrow, and sum type constructors is not finitely axiomatisable.

(b) We observe that in the presence of arrow, empty and sum types, type isomorphism and arithmetic equality no longer coincide. However, the coincidence still holds for certain classes of arithmetic equations.

Section 1 recalls the basic definitions. Section 2 establishes the non finite axiomatisability and separation results (a) and (b). Section 3 concludes with remarks and directions for further work.

1 The type theories

1.1 Product and sum types

Syntax. We present the type theory of product and sum types, including the empty ones. The set of types contains two distinguished type constants $\mathtt{1}$ and $\mathtt{0}$, a countable set of base types, and is closed under the product and sum type constructors. That is, types are defined by the grammar

$$\tau \quad ::= \quad \theta \mid \mathtt{1} \mid \tau_1 \times \tau_2 \mid \mathtt{0} \mid \tau_1 + \tau_2 \tag{1}$$

where θ ranges over base types.

Raw terms are defined by the grammar

$$t \quad ::= \quad x \mid \langle\rangle \mid \langle t_1, t_2\rangle \mid \pi_1(t) \mid \pi_2(t) \mid \bot_\tau(t) \mid \iota_1^{\tau_1,\tau_2}(t) \mid \iota_2^{\tau_1,\tau_2}(t) \mid \delta(t, x_1 : \tau_1.\, t_1, x_2 : \tau_2.\, t_2) \tag{2}$$

where x ranges over (a countable set of) variables, $\iota_1^{\tau_1,\tau_2}$ and $\iota_2^{\tau_1,\tau_2}$ are the left and right injections into the sum type $\tau_1 + \tau_2$, and δ is the usual binary case analysis (discriminating on the first argument and branching according to the second or third).

We write $\Gamma \vdash t : \tau$ for the judgement "the term t has type τ in context Γ". As usual, typing contexts, are lists of distinct type variables together with type declarations. A term t is well typed in a context Γ if the judgement $\Gamma \vdash t : \tau$ is derivable from the standard rules given in (the top part of) Figure 1. The associated equational theory is given in Figure 2. (Note that the congruence rule

$$\frac{\Gamma \vdash t = t' : \mathtt{0}}{\Gamma \vdash \bot_\tau(t) = \bot_\tau(t') : \tau}$$

is admissible.)

Semantics. A *bicartesian category* is a category with finite products $(1, \times)$ and finite coproducts $(0, +)$. Bicartesian categories for which the canonical map

$$(A \times B) + (A \times C) \longrightarrow A \times (B + C)$$

is an isomorphism for all objects A, B, C are called *distributive categories* [9, 8]. We remark that in a distributive category the canonical map

$$0 \longrightarrow A \times 0$$

is an isomorphism for every object A (see, *e.g.*, [9]).

We let $\mathcal{D}[T]$ be the category with objects given by types over base types in T and morphisms $\tau_1 \longrightarrow \tau_2$ given by equivalence classes $[x : \tau_1 \vdash t : \tau_2]$ of well-typed terms under the equivalence identifying $(x : \tau_1 \vdash t : \tau_2)$ and $(x' : \tau_1 \vdash t' : \tau_2)$ iff the judgement $x : \tau_1 \vdash t = t'[^x/_{x'}] : \tau_2$ is derivable. Composition is by substitution

$$[x' : \tau_2 \vdash t' : \tau_3] \circ [x : \tau_1 \vdash t : \tau_2] \;=\; [x : \tau_1 \vdash t'[^t/_{x'}] : \tau_3] \tag{3}$$

$$\frac{}{\Gamma,\ x:\tau,\ \Gamma' \vdash x:\tau} \qquad \frac{}{\Gamma \vdash \langle\rangle : \mathtt{1}}$$

$$\frac{\Gamma \vdash t_i : \tau_i \quad (i=1,2)}{\Gamma \vdash \langle t_1, t_2\rangle : \tau_1 \times \tau_2} \qquad \frac{\Gamma \vdash t : \tau_1 \times \tau_2}{\Gamma \vdash \pi_i(t) : \tau_i}\ (i=1,2)$$

$$\frac{\Gamma \vdash t : \mathtt{0}}{\Gamma \vdash \bot_\tau(t) : \tau}$$

$$\frac{\Gamma \vdash t : \tau_i}{\Gamma \vdash \iota_i^{\tau_1,\tau_2}(t) : \tau_1 + \tau_2}\ (i=1,2) \qquad \frac{\Gamma \vdash t : \tau_1 + \tau_2 \qquad \Gamma,\ x_i : \tau_i \vdash t_i : \tau \quad (i=1,2)}{\Gamma \vdash \delta(t, x_1 : \tau_1.\ t_1, x_2 : \tau_2.\ t_2) : \tau}$$

$$\frac{\Gamma,\ x:\tau_1 \vdash t : \tau}{\Gamma \vdash \lambda x : \tau_1.\ t : \tau_1 \to \tau} \qquad \frac{\Gamma \vdash t : \tau_1 \to \tau \qquad \Gamma \vdash t_1 : \tau_1}{\Gamma \vdash t(t_1) : \tau}$$

Figure 1: Typing rules.

with identities given by

$$[x : \tau \vdash x : \tau] \quad . \tag{4}$$

The category $\mathcal{D}[T]$ is canonically a distributive category, with terminal object $\mathtt{1}$ and initial object $\mathtt{0}$, and with products and sums respectively given by the projections

$$[x : \tau_1 \times \tau_2 \vdash \pi_i(x) : \tau_i] : \tau_1 \times \tau_2 \longrightarrow \tau_i \quad (i=1,2)$$

and the injections

$$[x : \tau_i \vdash \iota_i^{\tau_1,\tau_2}(x) : \tau_1 + \tau_2] : \tau_i \longrightarrow \tau_1 + \tau_2 \quad (i=1,2)\ .$$

Further, $\mathcal{D}[T]$ is the free distributive category on the set of base types T in the sense that for every distributive category $\mathcal{S}$ with distinguished products and sums and interpretation $\mathcal{I} : T \longrightarrow \mathcal{S}$, there exists a unique functor $\mathcal{I}[\![\_]\!] : \mathcal{D}[T] \longrightarrow \mathcal{S}$ preserving the product and sum structures. Thus, the interpretation of the type theory in distributive categories is sound and complete.

Note, in particular, that the interpretation of types induced by $\mathcal{I} : T \longrightarrow \mathcal{S}$ is given by

$$\begin{gathered}\mathcal{I}[\![\theta]\!] = \mathcal{I}(\theta)\ (\theta \in T)\ , \quad \mathcal{I}[\![\mathtt{1}]\!] = 1\ , \quad \mathcal{I}[\![\tau_1 \times \tau_2]\!] = \mathcal{I}[\![\tau_1]\!] \times \mathcal{I}[\![\tau_2]\!] \\ \mathcal{I}[\![\mathtt{0}]\!] = 0\ , \quad \mathcal{I}[\![\tau_1 + \tau_2]\!] = \mathcal{I}[\![\tau_1]\!] + \mathcal{I}[\![\tau_2]\!] \quad .\end{gathered} \tag{5}$$

1.2 Product, arrow, and sum types

Syntax. The typed lambda calculus with sums (see,*e.g.*, [20]) has types as in (1) extended as follows:

$$\tau \ ::= \ \cdots \mid \tau_1 \to \tau_2 \quad .$$

$$\frac{\Gamma \vdash t : \tau}{\Gamma \vdash t = t : \tau} \qquad \frac{\Gamma \vdash t = t' : \tau}{\Gamma \vdash t' = t : \tau} \qquad \frac{\Gamma \vdash t_1 = t_2 : \tau \qquad \Gamma \vdash t_2 = t_3 : \tau}{\Gamma \vdash t_1 = t_3 : \tau}$$

$$\frac{\Gamma \vdash t : \mathtt{1}}{\Gamma \vdash t = \langle\rangle : \mathtt{1}}$$

$$\frac{\Gamma \vdash t_i = t_i' : \tau_i \quad (i = 1, 2)}{\Gamma \vdash \langle t_1, t_2\rangle = \langle t_1', t_2'\rangle : \tau_1 \times \tau_2} \qquad \frac{\Gamma \vdash t = t' : \tau_1 \times \tau_2}{\Gamma \vdash \pi_i(t) = \pi_i(t') : \tau_i} \quad (i = 1, 2)$$

$$\frac{\Gamma \vdash t_1 : \tau_1 \qquad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash \pi_i(\langle t_1, t_2\rangle) = t_i : \tau_i} \quad (i = 1, 2) \qquad \frac{\Gamma \vdash t : \tau_1 \times \tau_2}{\Gamma \vdash t = \langle \pi_1(t), \pi_2(t)\rangle : \tau_1 \times \tau_2}$$

$$\frac{\Gamma \vdash t = t' : \tau_i}{\Gamma \vdash \iota_i^{\tau_1,\tau_2}(t) = \iota_i^{\tau_1,\tau_2}(t') : \tau_1 + \tau_2} \quad (i = 1, 2)$$

$$\frac{\Gamma \vdash t = t' : \tau_1 + \tau_2 \qquad \Gamma, x_i : \tau_i \vdash t_i = t_i' : \tau \quad (i = 1, 2)}{\Gamma \vdash \delta(t, x_1.\, t_1, x_2.\, t_2) = \delta(t', x_1.\, t_1', x_2.\, t_2') : \tau}$$

$$\frac{\Gamma \vdash t : \mathtt{0} \qquad \Gamma \vdash t' : \tau}{\Gamma \vdash \bot_\tau(t) = t' : \tau}$$

$$\frac{\Gamma \vdash t : \tau_j \qquad \Gamma, x_i : \tau_i \vdash t_i : \tau \quad (i = 1, 2)}{\Gamma \vdash \delta(\iota_j^{\tau_1,\tau_2}(t), x_1.\, t_1, x_2.\, t_2) = t_j[^t/_{x_j}] : \tau} \quad (j = 1, 2)$$

$$\frac{\Gamma \vdash t : \tau_1 + \tau_2 \qquad \Gamma, x : \tau_1 + \tau_2 \vdash t' : \tau}{\Gamma \vdash \delta(t, x_1.\, t'[\iota_1^{\tau_1,\tau_2}(x_1)/x], x_2.\, t'[\iota_2^{\tau_1,\tau_2}(x_2)/x]) = t'[^t/_x] : \tau}$$

Figure 2: Equational rules of the type theory for product and sum types

$$\frac{\Gamma \vdash t : \tau}{\Gamma \vdash t = t : \tau} \qquad \frac{\Gamma \vdash t = t' : \tau}{\Gamma \vdash t' = t : \tau} \qquad \frac{\Gamma \vdash t_1 = t_2 : \tau \qquad \Gamma \vdash t_2 = t_3 : \tau}{\Gamma \vdash t_1 = t_3 : \tau}$$

$$\frac{\Gamma \vdash t : \mathtt{1}}{\Gamma \vdash t = \langle\rangle : \mathtt{1}}$$

$$\frac{\Gamma \vdash t_1 : \tau_1 \qquad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash \pi_i\langle t_1, t_2\rangle = t_i : \tau_i}\ (i = 1, 2) \qquad \frac{\Gamma \vdash t : \tau_1 \times \tau_2}{\Gamma \vdash t = \langle \pi_1(t), \pi_2(t)\rangle : \tau_1 \times \tau_2}$$

$$\frac{\Gamma, x : \tau_1 \vdash t : \tau \qquad \Gamma \vdash t_1 : \tau_1}{\Gamma \vdash (\lambda x : \tau_1.t)(t_1) = t[{}^{t_1}/_{x}] : \tau} \qquad \frac{\Gamma \vdash t : \tau_1 \to \tau}{\Gamma \vdash t = \lambda x : \tau_1.t(x) : \tau_1 \to \tau}\ (x \notin \mathrm{FV}(t))$$

$$\frac{\Gamma \vdash t : \tau_1 \to \tau \qquad \Gamma \vdash t_1 = t_1' : \tau_1}{\Gamma \vdash t(t_1) = t(t_1') : \tau} \qquad \frac{\Gamma, x : \tau_1 \vdash t = t' : \tau}{\Gamma \vdash \lambda x : \tau_1.t = \lambda x : \tau_1.t' : \tau_1 \to \tau}$$

$$\frac{\Gamma \vdash t : \mathtt{0} \qquad \Gamma \vdash t' : \tau}{\Gamma \vdash \bot_\tau(t) = t' : \tau}$$

$$\frac{\Gamma \vdash t : \tau_j \qquad \Gamma, x_i : \tau_i \vdash t_i : \tau \quad (i = 1, 2)}{\Gamma \vdash \delta(\iota_j^{\tau_1,\tau_2}(t), x_1 : \tau_1.t_1, x_2 : \tau_2.t_2) = t_j[{}^{t}/_{x_j}] : \tau}\ (j = 1, 2)$$

$$\frac{\Gamma \vdash t : \tau_1 + \tau_2 \qquad \Gamma, x : \tau_1 + \tau_2 \vdash t' : \tau}{\Gamma \vdash \delta(t, x_1 : \tau_1.t'[{}^{\iota_1^{\tau_1,\tau_2}(x_1)}/_{x}], x_2 : \tau_2.t'[{}^{\iota_2^{\tau_1,\tau_2}(x_2)}/_{x}]) = t'[{}^{t}/_{x}] : \tau}$$

Figure 3: Equational rules of the typed lambda calculus with sum types.

The raw terms are as in (2) extended by abstractions and applications:

$$t \quad ::= \quad \cdots \mid \lambda x : \tau.\, t \mid t_1(t_2) \quad .$$

A term t is well typed in a context Γ if the judgement $\Gamma \vdash t : \tau$ is derivable from the rules given in Figure 1. The associated equational theory is given in Figure 3. (Note that the congruence rule

$$\frac{\Gamma, x : \tau \vdash t : \tau' \qquad \Gamma \vdash t_1 = t_2 : \tau}{\Gamma \vdash t[{}^{t_1}/_{x}] = t[{}^{t_2}/_{x}] : \tau'}$$

is admissible.)

Semantics. A *cartesian closed category* (CCC) is a category with finite products and exponentials ($\Rightarrow$). *Bicartesian closed categories* are referred to as BiCCCs; they are, of course,

distributive (see, *e.g.*, [20]).

We let $\mathcal{F}_{0,+}[T]$ be the category with objects given by types over base types in T and morphisms $\tau_1 \longrightarrow \tau_2$ given by equivalence classes $[x : \tau_1 \vdash t : \tau_2]$ of well-typed terms under the equivalence identifying $(x : \tau_1 \vdash t : \tau_2)$ and $(x' : \tau_1 \vdash t' : \tau_2)$ iff the judgement $x : \tau_1 \vdash t = t'[^x/_{x'}] : \tau_2$ is derivable. Composition and identities are as in (3) and (4). Further, we define $\mathcal{F}_+[T]$ analogously by omitting the empty type together with its associated constructor and equations throughout.

The category $\mathcal{F}_{0,+}[T]$ is canonically a BiCCC, with terminal object $\mathtt{1}$ and initial object $\mathtt{0}$, and with products, exponentials, and sums respectively given by the projections

$$[x : \tau_1 \times \tau_2 \vdash \pi_i(x) : \tau_i] : \tau_1 \times \tau_2 \longrightarrow \tau_i \quad (i = 1, 2) \ ,$$

the evaluation map

$$[x : (\tau_1 \to \tau_2) \times \tau_1 \vdash (\pi_1(x))(\pi_2(x)) : \tau_2] : (\tau_1 \Rightarrow \tau_2) \times \tau_1 \longrightarrow \tau_2 \ ,$$

and the injections

$$[x : \tau_i \vdash \iota_i^{\tau_1,\tau_2}(x) : \tau_1 + \tau_2] : \tau_i \longrightarrow \tau_1 + \tau_2 \quad (i = 1, 2) \ .$$

Further, $\mathcal{F}_{0,+}[T]$ is the free BiCCC on the set of base types T in the sense that for every BiCCC $\mathcal{S}$ with distinguished structure and interpretation $\mathcal{I} : T \longrightarrow \mathcal{S}$, there exists a unique functor $\mathcal{I}[\![\_]\!] : \mathcal{F}_{0,+}[T] \longrightarrow \mathcal{S}$ preserving the BiCC structure. Thus, the interpretation of the type theory in BiCCCs is sound and complete (see, *e.g.*, [20]). Of course, an analogous result holds for $\mathcal{F}_+[T]$ with respect to CC structure with binary sums.

Note that the interpretation of types induced by $\mathcal{I} : T \longrightarrow \mathcal{S}$ is as in (5) together with

$$\mathcal{I}[\![\tau_1 \to \tau_2]\!] \ = \ \mathcal{I}[\![\tau_1]\!] \Rightarrow \mathcal{I}[\![\tau_2]\!] \quad .$$

2 Tarski's high school algebra problem and type isomorphisms

Tarski's high school algebra problem. Tarski [15] asked if the equational theory $\mathcal{E}$ of the usual arithmetic identities

$$(\mathcal{E}_1) \quad 1 \cdot x = x \qquad (\mathcal{E}_2) \quad x \cdot y = y \cdot x \qquad (\mathcal{E}_3) \quad (x \cdot y) \cdot z = x \cdot (y \cdot z)$$

$$(\mathcal{E}_4) \quad x^1 = x \qquad\qquad (\mathcal{E}_5) \quad 1^x = 1$$
$$(\mathcal{E}_6) \quad x^{y \cdot z} = (x^y)^z \qquad\qquad (\mathcal{E}_7) \quad (x \cdot y)^z = x^z \cdot y^z$$

$$(\mathcal{E}_8) \quad x + y = y + x \qquad\qquad (\mathcal{E}_9) \quad (x + y) + z = x + (y + z)$$
$$(\mathcal{E}_{10}) \quad x \cdot (y + z) = x \cdot y + x \cdot z \qquad\qquad (\mathcal{E}_{11}) \quad x^{(y+z)} = x^y \cdot x^z$$

that are taught in high school are complete for the standard model of positive natural numbers. He conjectured that they were, but was not able to prove the result. Martin [22] showed that the equation $(x^y)^z = (x^z)^y$ is complete for the standard model $\langle N, \uparrow \rangle$ of positive natural numbers with exponentiation, and that the identities $(\mathcal{E}_2)$, $(\mathcal{E}_3)$, $(\mathcal{E}_6)$, and $(\mathcal{E}_7)$ are complete for the standard model $\langle N, \cdot, \uparrow \rangle$ of positive natural numbers with multiplication and exponentiation. Further, he exhibited the identity

$$(x^u + x^u)^v \cdot (y^v + y^v)^u = (x^v + x^v)^u \cdot (y^u + y^u)^v$$

that is not provable in the restriction of $\mathcal{E}$ to the language without the constant 1.

Wilkie [31] established Tarski's conjecture in the negative. Indeed, by a proof-theoretic analysis, he showed that the identity

$$(A^x + B^x)^y \cdot (C^y + D^y)^x = (A^y + B^y)^x \cdot (C^x + D^x)^y$$

where

$$A \;=\; 1 + x \;,\quad B \;=\; 1 + x + x^2 \;,\quad C \;=\; 1 + x^3 \;,\quad D \;=\; 1 + x^2 + x^4$$

is not provable in $\mathcal{E}$.

Gurevič later gave an argument by an ad hoc countermodel [18] and, more importantly, showed that there is no finite axiomatisation for the valid equations in the standard model $\langle N, 1, \cdot, \uparrow, + \rangle$ of positive natural numbers with one, multiplication, exponentiation, and addition [19]. He did this by producing an infinite family of equations such that for every sound finite set of axioms one of the equations can be shown not to follow. Gurevič's identities are the following

$$(A^x + {B_n}^x)^{2^x} \cdot \; ({C_n}^{2^x} + {D_n}^{2^x})^x \;\;=\;\; (A^{2^x} + {B_n}^{2^x})^x \cdot \; ({C_n}^x + {D_n}^x)^{2^x} \qquad (n \geq 3 \text{ odd}) \quad (6)$$

where

$$\begin{array}{lcl} A & = & 1 + x \;, \\ B_n & = & 1 + x + \cdots + x^{n-1} \;=\; \sum_{i=0}^{n-1} \; x^i \;, \\ C_n & = & 1 + x^n \;, \\ D_n & = & 1 + x^2 + \cdots + x^{2(n-1)} \;=\; \sum_{i=0}^{n-1} \; x^{2i} \;. \end{array}$$

Type isomorphisms. The equations in $\mathcal{E}$ together with the following ones

$$(\mathcal{D}_1) \quad x \cdot 0 = 0 \qquad\qquad (\mathcal{D}_2) \quad x + 0 = x$$

have a clear combinatorial interpretation which is made evident when interpreting them as isomorphisms in the category of finite sets and functions $\mathbf{F}$, or indeed in any BiCCC, under the obvious translation given below:

$$\overline{x} = x \text{ (x a variable)} \;, \qquad \overline{\mathtt{1}} = \mathtt{1} \;, \qquad \overline{e_1 \cdot e_2} = \overline{e_1} \times \overline{e_2} \;, \qquad \overline{e_2{}^{e_1}} = \overline{e_1} \to \overline{e_2} \;,$$
$$\overline{\mathtt{0}} = \mathtt{0} \;, \qquad \overline{e_1 + e_2} = \overline{e_1} + \overline{e_2} \;.$$

The isomorphisms realising the translations of the equations in $\mathcal{E}$ are well-known. For instance, those associated with the equation $(\mathcal{E}_{11})$ are given by

$$f : \tau_1 + \tau_2 \to \tau \vdash \langle \; \lambda x_1 : \tau_1 .\, f(\iota_1^{\tau_1,\tau_2}(x_1)) \;,\; \lambda x_2 : \tau_2 .\, f(\iota_2^{\tau_1,\tau_2}(x_2)) \; \rangle : (\tau_1 \to \tau) \times (\tau_2 \to \tau)$$

and

$$p : (\tau_1 \to \tau) \times (\tau_2 \to \tau) \vdash \lambda x : \tau_1 + \tau_2 .\, \delta(\; x \;,\; x_1 : \tau_1 .(\pi_1(p))(x_1) \;,\; x_2 : \tau_2 .(\pi_2(p))(x_2) \;) : \tau_1 + \tau_2 \to \tau \quad .$$

(Note that the equation

$$x \cdot 0^x = 0$$

corresponding to the type isomorphism

$$\theta \times (\theta \to 0) \cong 0$$

has no obvious combinatorial meaning; though it corresponds logically to the intuitionistic tautology $(p \wedge \neg p) \leftrightarrow \bot$.)

Thus, for arithmetic expressions e_1 and e_2 in the language with the constant 1, the operations $\cdot$, $\uparrow$, $+$, and unknowns in a set U, we have the chain of implications

$$\begin{aligned} \mathcal{E} \vdash e_1 = e_2 &\implies \overline{e_1} \cong \overline{e_2} \text{ in } \mathcal{F}_+[U] \\ &\implies \mathbf{F} \models \overline{e_1} = \overline{e_2} \\ &\implies N \models e_1 = e_2 \end{aligned} \tag{7}$$

where, for types τ_1 and τ_2 and a category $\mathcal{S}$, we write $\mathcal{S} \models \tau_1 = \tau_2$ whenever the identity $\tau_1 = \tau_2$ holds as an isomorphism in $\mathcal{S}$; that is, if for all interpretations $\mathcal{I}$ of base types in $\mathcal{S}$, it holds that $\mathcal{I}[\![\tau_1]\!] \cong \mathcal{I}[\![\tau_2]\!]$ in $\mathcal{S}$. (Note that the statements $\tau_1 \cong \tau_2$ in $\mathcal{F}_{0,+}[U]$ and $\mathcal{F}_{0,+}[U] \models \tau_1 = \tau_2$ are equivalent, and that they amount to type isomorphism in the equational theory of the typed lambda calculus with empty and sum types.) The last implication in (7) is easily established by observing that two finite sets are isomorphic iff they have the same cardinality, and that the type constructors on finite sets coincide with cardinal arithmetic.

The implications as in (7) for the cartesian closed case have been shown to be equivalences [29, 5]. The rest of the paper is devoted to study the extent to which these implications can be reversed in type theories with empty and sum types.

2.1 Product and sum types

We consider the case of distributive categories; the categorical counterpart of the type theory with unit and empty types, and product and sum type constructors.

For $\mathcal{D}$ the equational theory consisting of the identities $(\mathcal{E}_1)$, $(\mathcal{E}_2)$, $(\mathcal{E}_3)$, $(\mathcal{E}_8)$, $(\mathcal{E}_9)$, $(\mathcal{E}_{10})$ and $(\mathcal{D}_1)$, $(\mathcal{D}_2)$, we have the following result.

Proposition 2.1 *For arithmetic expressions e_1 and e_2 in the language given by 1, 0, $\cdot$, and $+$ and with unknowns in a set U, the following statements are equivalent.*

1. $\mathcal{D} \vdash e_1 = e_2$.
2. $\overline{e_1} \cong \overline{e_2}$ *in* $\mathcal{D}[U]$.
3. $\mathbf{F} \models \overline{e_1} = \overline{e_2}$.
4. $N_0 \models e_1 = e_2$.

The chain of implications (1) $\Rightarrow$ (2) $\Rightarrow$ (3) $\Rightarrow$ (4) are straightforward, whilst the implication (4) $\Rightarrow$ (1) amounts to the completeness of $\mathcal{D}$ for the standard model of natural numbers. This result is folklore (see, for instance, [7]) and we include it below.

Lemma 2.2 *The equational theory $\mathcal{D}$ is complete for the standard model $\langle N_0, 1, 0, \cdot, + \rangle$ of natural numbers.*

PROOF: Any expression in the language of variables, 1, 0, $\cdot$, and $+$ can be rewritten, using the equations in $\mathcal{D}$, into a canonical polynomial form, which can in fact be made unique.

Now, $N_0 \models e_1 = e_2$ iff, for the corresponding canonical polynomial forms p_1 and p_2 of e_1 and e_2 respectively, $N_0 \models p_1 = p_2$, and this is equivalent to p_1 and p_2 being the same polynomial (*i.e.*, syntactically equal) as the polynomial $p_1 - p_2$ with coefficients in the field of rational numbers has an infinite number of zeroes and hence it is null.

Thus, the lemma follows: if $e_1 = e_2$ holds in the standard model then, using $\mathcal{D}$, the expression e_1 can be transformed into its canonical polynomial form which can in turn be transformed into the expression e_2. □

Notice that the argument in the above proof shows that $\mathcal{D}$ is decidable. As a further corollary, we have the following *multiplicative cancellation* property.

Corollary 2.3 *For every non-empty type τ (viz., $\tau \ncong \mathsf{0}$) in $\mathcal{D}[T]$, we have that*

$$\tau_1 \times \tau \cong \tau_2 \times \tau \text{ in } \mathcal{D}[T] \implies \tau_1 \cong \tau_2 \text{ in } \mathcal{D}[T]$$

for every pair of types τ_1 and τ_2 in $\mathcal{D}[T]$.

It is interesting to note that in the further presence of exponentials, the above multiplicative cancellation property does not always hold. Indeed, for all types τ in $\mathcal{F}_{0,+}[T]$, we have the isomorphism

$$(\tau \to \mathsf{0}) \times \tau \cong \mathsf{0} \times \tau \text{ in } \mathcal{F}_{0,+}[T] \ ,$$

from which the cancellation of τ does not generally yield an isomorphism (for instance, $\theta \to \mathsf{0} \ncong \mathsf{0}$ for all $\theta \in T$).

2.2 Product, arrow, and sum types

Aiming at understanding type isomorphism in the presence of product, arrow, and sum types, it is natural to ask whether Gurevič's equations are also type isomorphisms.

We first consider two ways of establishing the identities (6) for the natural numbers respectively due to Wilkie and Gurevič.

Wilkie's method. Use that $C_n = A \cdot E_n$ and $D_n = B_n \cdot E_n$ for $E_n = 1 - x + \cdots + x^{n-1} = \sum_{i=0}^{n-1} (-1)^i x^i$.

Gurevič's method. Multiply the left hand side of (6) by $({D_n}^x)^{2^x}$ and using the equation $A \cdot D_n = B_n \cdot C_n$ twice establish the right hand side of (6) multiplied by $({D_n}^{2^x})^x$; conclude (6) by multiplicative cancellation of $({D_n}^x)^{2^x} = ({D_n}^{2^x})^x$.

Since the above methods respectively use negative numbers (which do not have a type-theoretic counterpart) and multiplicative cancellation (which is not known to be type theoretically sound), we speculated that Gurevič's identities did not hold as isomorphisms. Hence, we set out to prove that no term between the types corresponding to Gurevič's equations is an isomorphism by a careful study and analysis of normal forms in the typed lambda calculus with empty and sum types [17, 3].

Generalised Wilkie-Gurevič identities. To simplify the study of the normal forms between the types induced by the expressions in (6), we introduced the following generalised Wilkie-Gurevič identities with no constants

$$(A^u + {B_n}^u)^v \cdot ({C_n}^v + {D_n}^v)^u \;=\; (A^v + {B_n}^v)^u \cdot ({C_n}^u + {D_n}^u)^v \qquad (n \geq 3 \text{ odd}) \tag{8}$$

where

$$\begin{array}{lcl} A & = & y + x \\ B_n & = & y^{n-1} + y^{n-2}x + \cdots + x^{n-1} \;=\; \sum_{i=0}^{n-1} y^{n-(i+1)}x^i \\ C_n & = & y^n + x^n \\ D_n & = & y^{2(n-1)} + y^{2(n-2)}x^2 + \cdots + x^{2(n-1)} \;=\; \sum_{i=0}^{n-1} y^{2(n-(i+1))}x^{2i} \end{array} \tag{9}$$

Notice that replacing y by 1, u by x, and v by 2^x in (8) one obtains the equations (6). Hence the non finite axiomatisability result does not depend on the presence of constants in the language.

Corollary 2.4 *The equational theory of $\langle N, \cdot, \uparrow, + \rangle$ is not finitely axiomatisable.*

The generalised Wilkie-Gurevič identities are isomorphisms of types. Analysing the normal forms between the types corresponding to the generalised Wilkie-Gurevič identities (8) for the case $n = 3$ we found an isomorphism. The lemma below gives a general construction, which provides a type theoretic method for establishing the identities and that further exhibits their combinatorial content. (Note that this construction is more general than what is needed to establish that the Generalised Wilkie-Gurevič family of identities are isomorphisms.)

Lemma 2.5 *For types $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, $\mathcal{U}$, $\mathcal{V}$, $\mathcal{X}$, $\mathcal{Y}$ and mutually inverse closed terms $\varphi : \mathcal{A} \times \mathcal{D} \to \mathcal{C} \times \mathcal{B}$, $\phi : \mathcal{C} \times \mathcal{B} \to \mathcal{A} \times \mathcal{D}$, and $\varepsilon : \mathcal{U} \times \mathcal{V} \to \mathcal{X} \times \mathcal{Y}$, $\epsilon : \mathcal{X} \times \mathcal{Y} \to \mathcal{U} \times \mathcal{V}$ in the typed lambda calculus with empty and sum types, the following terms (where we write τ^σ for $\sigma \to \tau$)*

$$t_{\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D},\mathcal{U},\mathcal{V},\mathcal{X},\mathcal{Y}}[\varphi, \phi, \varepsilon, \epsilon] : \; (\mathcal{A}^{\mathcal{U}} + \mathcal{B}^{\mathcal{U}})^{\mathcal{V}} \times (\mathcal{C}^{\mathcal{X}} + \mathcal{D}^{\mathcal{X}})^{\mathcal{Y}} \to (\mathcal{A}^{\mathcal{X}} + \mathcal{B}^{\mathcal{X}})^{\mathcal{Y}} \times (\mathcal{C}^{\mathcal{U}} + \mathcal{D}^{\mathcal{U}})^{\mathcal{V}}$$

and

$$t_{\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D},\mathcal{X},\mathcal{Y},\mathcal{U},\mathcal{V}}[\varphi, \phi, \epsilon, \varepsilon] : \; (\mathcal{A}^{\mathcal{X}} + \mathcal{B}^{\mathcal{X}})^{\mathcal{Y}} \times (\mathcal{C}^{\mathcal{U}} + \mathcal{D}^{\mathcal{U}})^{\mathcal{V}} \to (\mathcal{A}^{\mathcal{U}} + \mathcal{B}^{\mathcal{U}})^{\mathcal{V}} \times (\mathcal{C}^{\mathcal{X}} + \mathcal{D}^{\mathcal{X}})^{\mathcal{Y}}$$

given by

$$\begin{array}{l} t_{\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D},\mathcal{U},\mathcal{V},\mathcal{X},\mathcal{Y}}[\varphi, \phi, \varepsilon, \epsilon] \\ \quad \stackrel{\text{def}}{=} \lambda h : (\mathcal{A}^{\mathcal{U}} + \mathcal{B}^{\mathcal{U}})^{\mathcal{V}} \times (\mathcal{C}^{\mathcal{X}} + \mathcal{D}^{\mathcal{X}})^{\mathcal{Y}}. \langle\; p_{\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D},\mathcal{U},\mathcal{V},\mathcal{X},\mathcal{Y}}[\varphi, \phi, \epsilon, \pi_1 h, \pi_2 h] \;, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad p_{\mathcal{C},\mathcal{D},\mathcal{A},\mathcal{B},\mathcal{X},\mathcal{Y},\mathcal{U},\mathcal{V}}[\phi, \varphi, \varepsilon, \pi_2 h, \pi_1 h] \;\rangle \end{array}$$

where

$$
\begin{aligned}
&p_{\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D},\mathcal{U},\mathcal{V},\mathcal{X},\mathcal{Y}}[\varphi,\phi,\gamma,f,g] \\
&\quad \stackrel{\text{def}}{=} \lambda y:\mathcal{Y}.\,\delta(\ g(y)\ , \\
&\qquad\qquad g_1:\mathcal{C}^{\mathcal{X}}.\,\iota_1^{\mathcal{A}^{\mathcal{X}},\mathcal{B}^{\mathcal{X}}}(\ \lambda x:\mathcal{X}.\,\delta(\ f(\pi_2(\gamma\langle x,y\rangle))\ , \\
&\qquad\qquad\qquad\qquad f_1:\mathcal{A}^{\mathcal{U}}.\,f_1(\pi_1(\gamma\langle x,y\rangle))\ , \\
&\qquad\qquad\qquad\qquad f_2:\mathcal{B}^{\mathcal{U}}.\,\pi_1(\phi\langle g_1(x),f_2(\pi_1(\gamma\langle x,y\rangle))\rangle)\ \ )\ \ )\ , \\
&\qquad\qquad g_2:\mathcal{D}^{\mathcal{X}}.\,\iota_2^{\mathcal{A}^{\mathcal{X}},\mathcal{B}^{\mathcal{X}}}(\ \lambda x:\mathcal{X}.\,\delta(\ f(\pi_2(\gamma\langle x,y\rangle))\ , \\
&\qquad\qquad\qquad\qquad f_1:\mathcal{A}^{\mathcal{U}}.\,\pi_2(\varphi\langle f_1(\pi_1(\gamma\langle x,y\rangle)),g_2(x)\rangle)\ , \\
&\qquad\qquad\qquad\qquad f_2:\mathcal{B}^{\mathcal{U}}.\,f_2(\pi_1(\gamma\langle x,y\rangle))\ \ )\ \ )\ \ )
\end{aligned}
$$

are mutually inverse.

Proof: See Appendix A. □

For the expressions (9) the identity $A \cdot D_n = C_n \cdot B_n$ can be proved from the standard axioms. Thus, there are mutually inverse closed terms $\varphi_n : \mathcal{A} \times \mathcal{D}_n \to \mathcal{C}_n \times \mathcal{B}_n$ and $\phi_n : \mathcal{C}_n \times \mathcal{B}_n \to \mathcal{A} \times \mathcal{D}_n$, where we write $\mathcal{A}$ for $\overline{A}$, $\mathcal{B}_n$ for $\overline{B_n}$, $\mathcal{C}_n$ for $\overline{C_n}$, and $\mathcal{D}_n$ for $\overline{D_n}$. It follows that,

$$t_{\mathcal{A},\mathcal{B}_n,\mathcal{C}_n,\mathcal{D}_n,\mathcal{U},\mathcal{V},\mathcal{V},\mathcal{U}}[\varphi_n,\phi_n,\lambda x:\mathcal{U}\times\mathcal{V}.\,\langle\pi_2 x,\pi_1 x\rangle,\lambda x:\mathcal{V}\times\mathcal{U}.\,\langle\pi_2 x,\pi_1 x\rangle]$$

is an isomorphism with inverse

$$t_{\mathcal{A},\mathcal{B}_n,\mathcal{C}_n,\mathcal{D}_n,\mathcal{V},\mathcal{U},\mathcal{U},\mathcal{V}}[\varphi_n,\phi_n,\lambda x:\mathcal{V}\times\mathcal{U}.\,\langle\pi_2 x,\pi_1 x\rangle,\lambda x:\mathcal{U}\times\mathcal{V}.\,\langle\pi_2 x,\pi_1 x\rangle] \quad .$$

Hence we have the following result.

Corollary 2.6 *The equational theory of type isomorphism in cartesian closed categories with binary coproducts is not finitely axiomatisable.*

2.3 Empty and sum types

In the presence of arrow and *both* empty and sum types we observe that not all equations that hold as isos in the category of finite sets hold in the type theory. Indeed, writing $\neg\tau$ for $\tau \to \mathtt{0}$, we have that

$$\mathbf{F} \models \neg\neg\theta \to \theta = \theta + \neg\theta$$

but as the formula $(\neg\neg p \to p) \to (\neg p \vee p)$ is a classical tautology which is not intuitionistically valid there is no term of type $(\neg\neg\theta \to \theta) \to (\neg\theta + \theta)$. Another such equation, derived from the above by taking the base type to be negated and using that $\mathtt{1} \cong \neg\neg\neg\theta \to \neg\theta$, is

$$\mathbf{F} \models \mathtt{1} = \neg\theta + \neg\neg\theta \quad . \tag{10}$$

Thus, in general, $\mathbf{F} \models \tau_1 = \tau_2$ does not imply $\tau_1 \cong \tau_2$ in $\mathcal{F}_{0,+}[T]$. However, we have the following result.

Proposition 2.7 *For every pair of types τ_1 and τ_2 in the typed lambda calculus with empty and sum types over a set of base types T, the following statements are equivalent.*

1. $\neg\tau_1 \cong \neg\tau_2$ *in* $\mathcal{F}_{0,+}[T]$.

2. *The formula* $\neg\overline{\tau_1} \leftrightarrow \neg\overline{\tau_2}$ *is an intuitionistic tautology, where*

$$\overline{\theta} = \theta \ (\theta \in T) \ , \qquad \overline{1} = \top \ , \qquad \overline{\tau_1 \times \tau_2} = \overline{\tau_1} \wedge \overline{\tau_2} \ , \qquad \overline{\tau_1 \to \tau_2} = \overline{\tau_1} \to \overline{\tau_2} \ ,$$
$$\overline{0} = \bot \ , \qquad \overline{\tau_1 + \tau_2} = \overline{\tau_1} \vee \overline{\tau_2} \ .$$

3. *For every interpretation* $\mathcal{I}$ *of base types in the category of finite sets* $\mathbf{F}$, $\mathcal{I}[\![\tau_1]\!] = 0$ *iff* $\mathcal{I}[\![\tau_2]\!] = 0$.

4. $\mathbf{F} \models \neg\tau_1 = \neg\tau_2$.

5. $N_0 \models 0^{e_1} = 0^{e_2}$, *where* $\overline{e_1} = \tau_1$ *and* $\overline{e_2} = \tau_2$.

PROOF: The equivalences (1)⇔(2) and (3)⇔(4)⇔(5), and the implication (1) ⇒ (4) are straightforward.

To establish the implication (3) ⇒ (2) we will use the (Gödel-Gentzen) negative translation $\tau^\star$ of types τ given by

$$\theta^\star = \neg\neg\theta \ (\theta \in T) \ , \quad \mathtt{1}^\star = \mathtt{1} \ , \quad (\tau_1 \times \tau_2)^\star = {\tau_1}^\star \times {\tau_2}^\star \ , \quad (\tau_1 \to \tau_2)^\star = {\tau_1}^\star \to {\tau_2}^\star \ ,$$
$$\mathtt{0}^\star = \mathtt{0} \ , \quad (\tau_1 + \tau_2)^\star = \neg(\neg{\tau_1}^\star \times \neg{\tau_2}^\star)$$

and the following facts:

(i) For every type τ and every interpretation $\mathcal{I}$ of base types in $\mathbf{F}$, the following hold

$$\mathcal{I}[\![\tau^\star]\!] \cong 0 \ \text{ or } \ \mathcal{I}[\![\tau^\star]\!] \cong 1$$

and

$$\begin{aligned} \mathcal{I}[\![\tau^\star]\!] \cong 1 &\iff \overline{\tau^\star} \text{ is a classical tautology} \\ &\iff \overline{\tau^\star} \text{ is an intuitionistic tautology} \quad . \end{aligned}$$

(ii) For every type τ, the formula $\overline{\tau^\star} \leftrightarrow \neg\neg\overline{\tau}$ is an intuitionistic tautology.

Indeed, assuming (3) it follows using (i) that $(\overline{\tau_1^\star} \to \overline{\tau_2^\star}) = \overline{(\tau_1 \to \tau_2)^\star}$ is an intuitionistic tautology. Thus, we have from (ii) that $\neg\neg\overline{\tau_1} \to \neg\neg\overline{\tau_2}$ is an intuitionistic tautology, from which we conclude that so is $\neg\overline{\tau_2} \to \neg\overline{\tau_1}$. Analogously, one sees that $\neg\overline{\tau_1} \to \neg\overline{\tau_2}$ is also an intuitionistic tautology, and we are done. □

It follows that the problem of whether two *negated* types are isomorphic in the theory of BiCCCs is decidable.

Corollary 2.8 *For all types* τ *in* $\mathcal{F}_{0,+}[T]$ *and arithmetic expressions* e *with* $\overline{e} = \tau$,

$$\tau \cong \mathtt{0} \text{ in } \mathcal{F}_{0,+}[T] \iff \mathbf{F} \models \tau = \mathtt{0} \iff N_0 \models e = 0 \quad .$$

3 Concluding remarks

The results of this paper are the first significant advance in the study of type isomorphisms in the presence of empty and sum types. Many questions still remain open, as for instance whether there are arithmetic equations in the language of $1, \cdot, \uparrow, 0$ or of $1, \cdot, \uparrow, +$ that do not correspond to type isomorphisms. We conjecture that Gurevič's result [19] for establishing the non-finite axiomatisability of the equational theory of the model of positive natural numbers $\langle N, 1, \cdot, \uparrow, + \rangle$ can be generalised to the case of the model of natural numbers $\langle N_0, 1, 0, \cdot, \uparrow, + \rangle$, and hence, by the results of this paper, that the equational theory of type isomorphism in bicartesian closed categories is not finitely axiomatisable. Decidability questions of the equational theory of type isomorphisms in the extensions of the typed lambda calculus with empty and/or sum types should be addressed. Finally, the observations in Subsection 2.3 suggest that the appropriate framework for characterising the type isomorphisms that hold in the category of finite sets for types with arrow, empty and sum constructors may be calculi for classical or intermediate logics.

Acknowledgements. We are grateful to Claude Kirchner and Sergei Soloviev for interesting discussions on the subject, and to Alex Simpson for pointing out (10).

References

[1] M.-V. Aponte and R. Di Cosmo. Type isomorphisms for module signatures. In *Programming Languages Implementation and Logic Programming (PLILP)*, volume 1140 of *Lecture Notes in Computer Science*, pages 334–346. Springer-Verlag, 1996.

[2] V. Balat and R. Di Cosmo. A linear logical view of linear type isomorphisms. In *Computer Science Logic*, volume 1683 of *Lecture Notes in Computer Science*, pages 250–265. Springer-Verlag, 1999.

[3] V. Balat, R. Di Cosmo, and M. Fiore. Extensional normalisation and type-directed partial evaluation for typed lambda calculus with sums. In *31st Ann. ACM Symp. on Principles of Programming Languages (POPL)*, pages 64–76. ACM, 2004.

[4] G. Barthes and O. Pons. Type isomorphisms and proof reuse in dependent type theory. In F. Honsell and M. Miculan, editors, *FOSSACS*, number 2030 in LNCS, pages 57–71. Springer-Verlag, 2001.

[5] K. Bruce, R. Di Cosmo, and G. Longo. Provable isomorphisms of types. *Mathematical Structures in Computer Science*, 2(2):231–247, 1992.

[6] K. Bruce and G. Longo. Provable isomorphisms and domain equations in models of typed languages. *ACM Symposium on Theory of Computing (STOC 85)*, 1985.

[7] S. Burris and S. Lee. Tarski's high school identities. *American Mathematical Monthly*, 100(3):231–236, 1993.

[8] A. Carboni, S. Lack, and R. F. C. Walters. Introduction to extensive and distributive categories. *Journal of Pure and Applied Algebra*, 84:145–158, 1993.

[9] J. R. B. Cockett. Introduction to distributive categories. *Mathematical Structures in Computer Science*, 3:277–307, 1993.

[10] D. Delahaye, R. Di Cosmo, and B. Werner. Recherche dans une bibliothèque de preuves Coq en utilisant le type et modulo isomorphismes. In *PRC/GDR de programmation, Pôle Preuves et Spécifications Algébriques*, 1997.

[11] R. Di Cosmo. Type isomorphisms in a type assignment framework. In *19th Ann. ACM Symp. on Principles of Programming Languages (POPL)*, pages 200–210. ACM, 1992.

[12] R. Di Cosmo. Deciding type isomorphisms in a type assignment framework. *Journal of Functional Programming*, 3(3):485–525, 1993. (Special Issue on ML).

[13] R. Di Cosmo. *Isomorphisms of types: from λ-calculus to information retrieval and language design*. Birkhauser, 1995.

[14] R. Di Cosmo. Second order isomorphic types. A proof theoretic study on second order λ-calculus with surjective pairing and terminal object. *Information and Computation*, pages 176–201, 1995.

[15] J. Doner and A. Tarski. An extended arithmetic of ordinal numbers. *Fundamenta Mathematica*, 65:95–127, 1969.

[16] K. Dosen and Z. Petric. Isomorphic objects in symmetric monoidal closed categories. *Mathematical Structures in Computer Science*, 7(6):639–662, 1997.

[17] M. Fiore, R. Di Cosmo, and V. Balat. Extensional normalisation for typed lambda calculus with sums via Grothendieck logical relations. Manuscript, 2002.

[18] R. Gurevič. Equational theory of positive numbers with exponentiation. *Proceedings of the American Mathematical Society*, 94(1):135–141, 1985.

[19] R. Gurevič. Equational theory of positive numbers with exponentiation is not finitely axiomatizable. *Annals of Pure and Applied Logic*, 49:1–30, 1990.

[20] J. Lambek and P. Scott. *Introduction to higher order categorical logic*, volume 7 of *Cambridge studies in advanced mathematics*. Cambridge University Press, 1986.

[21] F. W. Lawvere. Left and right adjoint operations on spaces and data types. *Theoretical Computer Science*, 316:105–111, 2004.

[22] C. F. Martin. Axiomatic bases for equational theories of natural numbers. *Notices of the Am. Math. Soc.*, 19(7):778, 1972.

[23] M. Rittri. Retrieving library identifiers by equational matching of types. In *10th Int. Conf. on Automated Deduction*, volume 449 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990.

[24] M. Rittri. *Searching program libraries by type and proving compiler correctness by bisimulation*. PhD thesis, University of Göteborg, Göteborg, Sweden, 1990.

[25] M. Rittri. Using types as search keys in function libraries. *Journal of Functional Programming*, 1(1):71–89, 1991.

[26] C. Runciman and I. Toyn. Retrieving re-usable software components by polymorphic type. *Journal of Functional Programming*, 1(2):191–211, 1991.

[27] S. H. Schanuel. Objective number theory and the retract chain condition. *Journal of Pure and Applied Algebra*, 154:295–298, 2000.

[28] S. H. Schanuel. Transcendence in objective number theory. *Rendiconti del Circolo Matematico di Palermo*, Serie II, Suppl. 64:43–48, 2000.

[29] S. V. Soloviev. The category of finite sets and cartesian closed categories. *Journal of Soviet Mathematics*, 22(3):1387–1400, 1983.

[30] S. V. Soloviev. A complete axiom system for isomorphism of types in closed categories. In A. Voronkov, editor, *Logic Programming and Automated Reasoning, 4^{th} International Conference*, volume 698 of *Lecture Notes in Artificial Intelligence*, pages 360–371. Springer-Verlag, 1993.

[31] A. J. Wilkie. On exponentiation — A solution to Tarski's high school algebra problem. Math. Inst. Oxford University (preprint), 1981.

A Proof of Lemma 2.5

For the following definitions (where, to improve readability, all type information has been omitted)

$$\begin{aligned}
t[\varphi,\phi,\varepsilon,\epsilon] &\stackrel{\text{def}}{=} \lambda h.\,\langle\ p[\varphi,\phi,\epsilon,\pi_1 h,\pi_2 h]\ ,\ p[\phi,\varphi,\varepsilon,\pi_2 h,\pi_1 h]\ \rangle \\
p[\varphi,\phi,\gamma,f,g] &\stackrel{\text{def}}{=} \lambda y.\,\delta(\ g(y)\ , \\
&\qquad\qquad g_1.\ \iota_1(\lambda x.\,F[\phi,\gamma,f,g_1,x,y])\ , \\
&\qquad\qquad g_2.\ \iota_2(\lambda x.\,G[\varphi,\gamma,f,g_2,x,y]\ ) \\
F[\phi,\gamma,f,g,x,y] &\stackrel{\text{def}}{=} \delta(\ f(\pi_2(\gamma\langle x,y\rangle))\ , \\
&\qquad\quad f_1.\ f_1(\pi_1(\gamma\langle x,y\rangle))\ , \\
&\qquad\quad f_2.\ \pi_1(\ \phi\langle\ g(x)\ ,\ f_2(\pi_1(\gamma\langle x,y\rangle))\ \rangle\ )\ ) \\
G[\varphi,\gamma,f,g,x,y] &\stackrel{\text{def}}{=} \delta(\ f(\pi_2(\gamma\langle x,y\rangle))\ , \\
&\qquad\quad f_1.\ \pi_2(\ \varphi\langle\ f_1(\pi_1(\gamma\langle x,y\rangle))\ ,\ g(x)\ \rangle\ )\ , \\
&\qquad\quad f_2.\ f_2(\pi_1(\gamma\langle x,y\rangle))\ )\ )
\end{aligned}$$

we will establish the identity

$$\lambda h.\,t[\varphi,\phi,\epsilon,\varepsilon](t[\varphi,\phi,\varepsilon,\epsilon](h)) \quad = \quad \lambda h.\,h$$

in the equational theory of the typed lambda calculus with empty and sum types (see Figure 3).

The identities (A)–(G) below, all of which are valid in the equational theory, will be used throughout the proof.

$$\begin{aligned}
\text{(A)}\quad & \delta(\ \delta(t_0, x_1.\ t_1, x_2.t_2)\ ,\ y_1.\ u_1\ ,\ y_2.\ u_2\ ) \\
&\quad = \ \delta(\ t_0\ ,\ x_1.\ \delta(t_1, y_1.\ u_1, y_2.\ u_2)\ ,\ x_2.\ \delta(t_2, y_1.\ u_1, y_2.\ u_2)\ )
\end{aligned}$$

(B) $\delta(t_0, x_1.\ \delta(t_0, y_1.\ t_1, y_2.\ t_2), x_2.\ t) \ = \ \delta(t_0, x_1.\ t_1[^{x_1}/_{y_1}], x_2.\ t)$

$\delta(t_0, x_1.\ t, x_2.\ \delta(t_0, y_1.\ t_1, y_2.\ t_2)) \ = \ \delta(t_0, x_1.\ t_1, x_2.\ t_2[^{x_2}/_{y_2}])$

(C) $\delta(\ t_0\ ,\ x_1.\ \delta(u_0, y_1.\ u_1, y_2.\ u_2)\ ,\ x_2.\ t_2\ )$
$\quad = \ \delta(\ u_0\ ,\ y_1.\ \delta(t_0, x_1.\ u_1, x_2.\ t_2)\ ,\ y_2.\ \delta(t_0, x_1.\ u_2, x_2.\ t_2)\ )$

$\delta(\ t_0\ ,\ x_1.\ t_1\ ,\ x_2.\ \delta(u_0, y_1.\ u_1, y_2.\ u_2)\ )$
$\quad = \ \delta(\ u_0\ ,\ y_1.\ \delta(t_0, x_1.\ t_1, x_2.\ u_1)\ ,\ y_2.\ \delta(t_0, x_1.\ t_1, x_2.\ u_2)\ )$

(D) $\lambda x.\, \delta(t_0, x_1.\ t_1, x_2.\ t_2) \ = \ \delta(t_0, x_1.\ \lambda x.\, t_1, x_2.\ \lambda x.\, t_2) \qquad (x \notin \mathrm{FV}(t_0) \cup \{\ x_1, x_2\ \})$

(E) $t(\delta(t_0, x_1.\ t_1, x_2.\ t_2)) \ = \ \delta(t_0, x_1.\ t(t_1), x_2.\ t(t_2)) \qquad (x_1, x_2 \notin \mathrm{FV}(t))$

(F) $\langle\ \delta(t_0, x_1.\ t_1, x_2.\ t_2)\ ,\ t\ \rangle \ = \ \delta(\ t_0\ ,\ x_1.\ \langle t_1, t\rangle\ ,\ x_2.\ \langle t_2, t\rangle\ )$
$\langle\ t\ ,\ \delta(t_0, x_1.\ t_1, x_2.\ t_2)\ \rangle \ = \ \delta(\ t_0\ ,\ x_1.\ \langle t, t_1\rangle\ ,\ x_2.\ \langle t, t_2\rangle\ ) \qquad (x_1, x_2 \notin \mathrm{FV}(t))$

(G) $\delta(t_0, x_1.\ t, x_2.\ t) \ = \ t \qquad (x_1, x_2 \notin \mathrm{FV}(t))$

We start the proof by observing that

$$t[\varphi, \phi, \epsilon, \varepsilon](t[\varphi, \phi, \varepsilon, \epsilon](h)) \ = \ \langle\ p[\varphi, \phi, \varepsilon, p[\varphi, \phi, \epsilon, \pi_1 h, \pi_2 h], p[\phi, \varphi, \varepsilon, \pi_2 h, \pi_1 h]]\ ,\ p[\phi, \varphi, \epsilon, p[\phi, \varphi, \varepsilon, \pi_2 h, \pi_1 h], p[\varphi, \phi, \epsilon, \pi_1 h, \pi_2 h]]\ \rangle$$

and show that the first and second components of the above pair are the respective projections of h; *i.e.*, that

$$p[\varphi, \phi, \varepsilon, p[\varphi, \phi, \epsilon, \pi_1 h, \pi_2 h], p[\phi, \varphi, \varepsilon, \pi_2 h, \pi_1 h]] \ = \ \pi_1 h \quad ,$$

$$p[\phi, \varphi, \epsilon, p[\phi, \varphi, \varepsilon, \pi_2 h, \pi_1 h], p[\varphi, \phi, \epsilon, \pi_1 h, \pi_2 h]] \ = \ \pi_2 h \quad .$$

To this end it will be enough to establish

(1) $p[\varphi, \phi, \varepsilon, p[\varphi, \phi, \epsilon, f, g], p[\phi, \varphi, \varepsilon, g, f]] \ = \ f \quad .$

The left hand side of (1) equals

(2) $\lambda y.\, \delta(\ p[\phi, \varphi, \varepsilon, g, f](y)\ ,$
$\qquad g_1.\ \iota_1(\ \lambda x.\, F[\phi, \varepsilon, p[\varphi, \phi, \epsilon, f, g], g_1, x, y]\ )\ ,$
$\qquad g_2.\ \iota_2(\ \lambda x.\, G[\varphi, \varepsilon, p[\varphi, \phi, \epsilon, f, g], g_2, x, y]\ )\ )$

and, as the discriminator of (2) equals

(3) $\delta(\ f(y)\ ,$
$\quad g_1'.\ \iota_1(\ \lambda x'.\, F[\varphi, \varepsilon, g, g_1', x', y]\ )\ ,$
$\quad g_2'.\ \iota_2(\ \lambda x'.\, G[\phi, \varepsilon, g, g_2', x', y]\ )\ ) \quad ,$

we have, from (2) and (3) using (A), that the left hand side of (1) equals

(4) $\lambda y.\, \delta(\ f(y)\ ,$
$\qquad g_1'.\ \iota_1(\ \lambda x.\, F[\phi, \varepsilon, p[\varphi, \phi, \epsilon, f, g], \lambda x'.\, F[\varphi, \varepsilon, g, g_1', x', y], x, y]\ )\ ,$
$\qquad g_2'.\ \iota_2(\ \lambda x.\, G[\varphi, \varepsilon, p[\varphi, \phi, \epsilon, f, g], \lambda x'.\, G[\phi, \varepsilon, g, g_2', x', y], x, y]\ )\ ) \quad .$

We now calculate, in turn, the following terms appearing in the conditional branches of (4):

(5) $F[\phi, \varepsilon, p[\varphi, \phi, \epsilon, f, g], \lambda x'.\, F[\varphi, \varepsilon, g, g'_1, x', y], x, y]$,

(6) $G[\varphi, \varepsilon, p[\varphi, \phi, \epsilon, f, g], \lambda x'.\, G[\phi, \varepsilon, g, g'_2, x', y], x, y]$.

The term (5) equals

(7) $\delta(\ p[\varphi, \phi, \epsilon, f, g](\pi_2(\varepsilon\langle x, y\rangle))\ ,$
$\quad f_1.\ f_1(\pi_1(\varepsilon\langle x, y\rangle))\ ,$
$\quad f_2.\ \pi_1(\ \phi\langle\ F[\varphi, \varepsilon, g, g'_1, x, y]\ ,\ f_2(\pi_1(\varepsilon\langle x, y\rangle))\ \rangle\ )\ )$

and, as the discriminator of (7) equals

(8) $\delta(\ g(\pi_2(\varepsilon\langle x, y\rangle))\ ,$
$\quad g''_1.\ \iota_1(\ \lambda x''.\, F[\phi, \epsilon, f, g''_1, x'', \pi_2(\varepsilon\langle x, y\rangle)]\ )\ ,$
$\quad g''_2.\ \iota_2(\ \lambda x''.\, G[\varphi, \epsilon, f, g''_2, x'', \pi_2(\varepsilon\langle x, y\rangle)]\ )\ )$

we have, from (7) and (8) using (A), that (5) equals

(9) $\delta(\ g(\pi_2(\varepsilon\langle x, y\rangle))\ ,$
$\quad g''_1.\ F[\phi, \epsilon, f, g''_1, \pi_1(\varepsilon\langle x, y\rangle), \pi_2(\varepsilon\langle x, y\rangle)]\ ,$
$\quad g''_2.\ \pi_1(\ \phi\langle\ F[\varphi, \varepsilon, g, g'_1, x, y]\ ,\ G[\varphi, \epsilon, f, g''_2, \pi_1(\varepsilon\langle x, y\rangle), \pi_2(\varepsilon\langle x, y\rangle)]\ \rangle\ )\ )$.

We calculate, in turn, the following terms appearing in the conditional branches of (9):

(10) $F[\phi, \epsilon, f, g''_1, \pi_1(\varepsilon\langle x, y\rangle), \pi_2(\varepsilon\langle x, y\rangle)]$,

(11) $G[\varphi, \epsilon, f, g''_2, \pi_1(\varepsilon\langle x, y\rangle), \pi_2(\varepsilon\langle x, y\rangle)]$.

The term (10) equals

$\delta(\ f(\pi_2(\ \epsilon\langle\ \pi_1(\varepsilon\langle x, y\rangle)\ ,\ \pi_2(\varepsilon\langle x, y\rangle)\ \rangle\ ))\ ,$
$\quad f_1.\ f_1(\ \pi_1(\ \epsilon\langle\ \pi_1(\varepsilon\langle x, y\rangle)\ ,\ \pi_2(\varepsilon\langle x, y\rangle)\ \rangle\ )\ )\ ,$
$\quad f_2.\ \pi_1(\ \phi\langle\ g''_1(\pi_1(\varepsilon\langle x, y\rangle))\ ,\ f_2(\ \pi_1(\ \epsilon\langle\ \pi_1(\varepsilon\langle x, y\rangle)\ ,\ \pi_2(\varepsilon\langle x, y\rangle)\ \rangle\ )\ )\ \rangle\ )\ )$

from which, applying the surjective-pairing law $\langle \pi_1(z), \pi_2(z)\rangle = z$ three times, we obtain

$\delta(\ f(\pi_2(\ \epsilon\ \varepsilon\langle x, y\rangle))\ ,$
$\quad f_1.\ f_1(\ \pi_1(\ \epsilon\ \varepsilon\langle x, y\rangle\ )\ )\ ,$
$\quad f_2.\ \pi_1(\ \phi\langle\ g''_1(\pi_1(\varepsilon\langle x, y\rangle))\ ,\ f_2(\ \pi_1(\ \epsilon\ \varepsilon\langle x, y\rangle)\ )\ )\ \rangle\ )\ )$.

Further, since ε and ϵ are mutual inverses, after an application of the first-projection law $\pi_1(\langle z_1, z_2\rangle) = z_1$, we get that this term equals the following one

(12) $\delta(\ f(y)\ ,$
$\quad f_1.\ f_1(x)\ ,$
$\quad f_2.\ \pi_1(\ \phi\langle\ g''_1(\pi_1(\varepsilon\langle x, y\rangle))\ ,\ f_2(x)\ \rangle\ )\ )$.

On the other hand, the term (11) equals

$$\begin{array}{l}\delta(\ f(\ \pi_2(\ \epsilon\langle\ \pi_1(\varepsilon\langle x,y\rangle)\ ,\ \pi_2(\varepsilon\langle x,y\rangle)\ \rangle\ )\ )\ ,\\ \quad f_1.\ \pi_2(\ \varphi\langle\ f_1(\ \pi_1(\ \epsilon\langle\ \pi_1(\varepsilon\langle x,y\rangle)\ ,\ \pi_2(\varepsilon\langle x,y\rangle)\ \rangle\ )\ )\ ,\ g_2''(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ ,\\ \quad f_2.\ f_2(\ \pi_1(\ \epsilon\langle\ \pi_1(\varepsilon\langle x,y\rangle)\ ,\ \pi_2(\varepsilon\langle x,y\rangle)\ \rangle\ )\ )\ )\end{array}$$

which equals

$$\begin{array}{ll}(13) & \delta(\ f(y)\ ,\\ & \quad f_1.\ \pi_2(\ \varphi\langle\ f_1(x)\ ,\ g_2''(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ ,\\ & \quad f_2.\ f_2(x)\ )\quad .\end{array}$$

Thus, from (4), (9), (12), and (13), the left hand side of (1) equals

$$\begin{array}{l}\lambda y.\,\delta(\ f(y)\ ,\\ \quad g_1'.\ \iota_1(\ \lambda x.\,\delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ \qquad\qquad g_1''.\ \delta(\ f(y)\ ,\\ \qquad\qquad\quad f_1.\ f_1(x)\ ,\\ \qquad\qquad\quad f_2.\ \pi_1(\ \phi\langle\ g_1''(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ f_2(x)\ \rangle\ )\ )\ ,\\ \qquad\qquad g_2''.\ \pi_1(\ \phi\langle\ \delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ \qquad\qquad\qquad\quad f_1.\ f_1(\pi_1(\varepsilon\langle x,y\rangle))\ ,\\ \qquad\qquad\qquad\quad f_2.\ \pi_1(\ \varphi\langle\ g_1'(x)\ ,\ f_2(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ )\ ,\\ \qquad\qquad\qquad\ \delta(\ f(y)\ ,\\ \qquad\qquad\qquad\quad f_1.\ \pi_2(\ \varphi\langle\ f_1(x)\ ,\ g_2''(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ ,\\ \qquad\qquad\qquad\quad f_2.\ f_2(x)\ )\ \rangle\ )\ )\ )\ ,\\ \quad g_2'.\ \iota_2(\ \lambda x.\,G[\varphi,\varepsilon,p[\varphi,\phi,\epsilon,f,g],\lambda x'.\,G[\phi,\varepsilon,g,g_2',x',y],x,y]\ )\ )\end{array}$$

which, using (B)–(F), equals

$$\begin{array}{l}\lambda y.\,\delta(\ f(y)\ ,\\ \quad g_1'.\ \iota_1(\ \lambda x.\,\delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ \qquad\qquad g_1''.\ g_1'(x)\ ,\\ \qquad\qquad g_2''.\ \pi_1(\ \phi\langle\ \pi_1(\ \varphi\langle\ g_1'(x)\ ,\ g_2''(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ ,\\ \qquad\qquad\qquad\quad \pi_2(\ \varphi\langle\ g_1'(x)\ ,\ g_2''(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ \rangle\ )\ )\ )\ ,\\ \quad g_2'.\ \iota_2(\ \lambda x.\,G[\varphi,\varepsilon,p[\varphi,\phi,\epsilon,f,g],\lambda x'.\,G[\phi,\varepsilon,g,g_2',x',y],x,y]\ )\ )\end{array}$$

and that, using surjective pairing, the fact that φ and ϕ are mutual inverses, the first-projection law, the equation (G), and the extensionality law $\lambda x.\,t(x) = t$ $(x \notin \mathrm{FV}(t))$, further equals

$$\begin{array}{ll}(14) & \lambda y.\,\delta(\ f(y)\ ,\\ & \quad g_1'.\ \iota_1(g_1')\ ,\\ & \quad g_2'.\ \iota_2(\ \lambda x.\,G[\varphi,\varepsilon,p[\varphi,\phi,\epsilon,f,g],\lambda x'.\,G[\phi,\varepsilon,g,g_2',x',y],x,y]\ )\ )\quad .\end{array}$$

We now calculate (6), which appears in the second branch above. This term equals

$$\begin{array}{l}\delta(\ p[\varphi,\phi,\epsilon,f,g](\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ \quad f_1'.\ \pi_2(\ \varphi\langle\ f_1'(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ G[\phi,\varepsilon,g,g_2',x,y]\ \rangle\ )\ ,\\ \quad f_2'.\ f_2'(\pi_1(\varepsilon\langle x,y\rangle))\ )\end{array}$$

and from this and (8), using (A), it further equals

$$\begin{array}{l}\delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ \quad g_1''.\ \pi_2(\ \varphi\langle\ F[\phi,\epsilon,f,g_1'',\pi_1(\varepsilon\langle x,y\rangle),\pi_2(\varepsilon\langle x,y\rangle)]\ ,\\ \qquad\qquad G[\phi,\varepsilon,g,g_2',x,y]\ \rangle\ )\ ,\\ \quad g_2''.\ G[\varphi,\epsilon,f,g_2'',\pi_1(\varepsilon\langle x,y\rangle),\pi_2(\varepsilon\langle x,y\rangle)]\ )\end{array}$$

which, from (12) and (13), equals

$$\begin{array}{ll}(15) & \delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ & \quad g_1''.\ \pi_2(\ \varphi\langle\ \delta(\ f(y)\ ,\\ & \qquad\qquad f_1.\ f_1(x)\ ,\\ & \qquad\qquad f_2.\ \pi_1(\ \phi\langle\ g_1''(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ f_2(x)\ \rangle\ )\ )\ ,\\ & \qquad\quad \delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ & \qquad\qquad f_1.\ \pi_2(\ \phi\langle\ f_1(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ g_2'(x)\ \rangle\ )\ ,\\ & \qquad\qquad f_2.\ f_2(\pi_1(\varepsilon\langle x,y\rangle))\ )\ \rangle\ )\ ,\\ & \quad g_2''.\ \delta(\ f(y)\ ,\\ & \qquad f_1.\ \pi_2(\ \varphi\langle\ f_1(x)\ ,\ g_2''(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ ,\\ & \qquad f_2.\ f_2(x)\ )\ )\quad .\end{array}$$

Thus, from (14) and (15), the left hand side of (1) equals

$$\begin{array}{l}\lambda y.\,\delta(\ f(y)\ ,\\ \quad g_1'.\ \iota_1(g_1')\ ,\\ \quad g_2'.\ \iota_2(\ \lambda x.\,\delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ \qquad\qquad g_1''.\ \pi_2(\ \varphi\langle\ \delta(\ f(y)\ ,\\ \qquad\qquad\qquad\qquad f_1.\ f_1(x)\ ,\\ \qquad\qquad\qquad\qquad f_2.\ \pi_1(\ \phi\langle\ g_1''(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ f_2(x)\ \rangle\ )\ )\ ,\\ \qquad\qquad\qquad\quad \delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\ \qquad\qquad\qquad\qquad f_1.\ \pi_2(\ \phi\langle\ f_1(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ g_2'(x)\ \rangle\ )\ ,\\ \qquad\qquad\qquad\qquad f_2.\ f_2(\pi_1(\varepsilon\langle x,y\rangle))\ )\ \rangle\ )\ ,\\ \qquad\qquad g_2''.\ \delta(\ f(y)\ ,\\ \qquad\qquad\qquad f_1.\ \pi_2(\ \varphi\langle\ f_1(x)\ ,\ g_2''(\pi_1(\varepsilon\langle x,y\rangle))\ \rangle\ )\ ,\\ \qquad\qquad\qquad f_2.\ f_2(x)\ )\ )\ )\ )\end{array}$$

which, using (B)–(F), equals

$$\begin{aligned}
&\lambda y.\,\delta(\ f(y)\ ,\\
&\qquad g_1'.\ \iota_1(g_1')\ ,\\
&\qquad g_2'.\ \iota_2(\ \lambda x.\,\delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\
&\qquad\qquad\qquad g_1''.\ \pi_2(\ \varphi\langle\ \pi_1(\ \phi\langle\ g_1''(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ g_2'(x)\ \rangle\ )\ ,\\
&\qquad\qquad\qquad\qquad\qquad \pi_2(\ \phi\langle\ g_1''(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ g_2'(x)\ \rangle\ )\ \rangle\ )\ ,\\
&\qquad\qquad\qquad g_2''.\ g_2'(x)\ )\ )\ )\quad .
\end{aligned}$$

By surjective pairing, this is equal to

$$\begin{aligned}
&\lambda y.\,\delta(\ f(y)\ ,\\
&\qquad g_1'.\ \iota_1(g_1')\ ,\\
&\qquad g_2'.\ \iota_2(\ \lambda x.\,\delta(\ g(\pi_2(\varepsilon\langle x,y\rangle))\ ,\\
&\qquad\qquad\qquad g_1''.\ \pi_2(\ \varphi(\ \phi\langle\ g_1''(\pi_1(\varepsilon\langle x,y\rangle))\ ,\ g_2'(x)\ \rangle\ )\ )\\
&\qquad\qquad\qquad g_2''.\ g_2'(x)\ )\ )\ )\quad .
\end{aligned}$$

Moreover, as ϕ and φ are mutual inverses, an application of the second-projection law on the second branch yields

$$\begin{aligned}
&\lambda y.\,\delta(\ f(y)\ ,\\
&\qquad g_1'.\ \iota_1(g_1')\ ,\\
&\qquad g_2'.\ \iota_2(\ \lambda x.\,\delta(g(\pi_2(\varepsilon\langle x,y\rangle)),\ g_1''.\ g_2'(x),\ g_2''.\ g_2'(x)\ )\ )\ )
\end{aligned}$$

which, by (G) and the extensionality law, equals

$$\lambda y.\,\delta(\ f(y)\ ,\ g_1'.\ \iota_1(g_1')\ ,\ g_2'.\ \iota_2(g_2')\ )\quad .$$

Finally, by the weak sum-extensionality law $\delta(t, z_1.\ \iota_1(z_1), z_2.\ \iota_2(z_2)) = t$ and the extensionality law, this is equal to f as required.

Hence, the identity (1) is established.