The Δ-calculus: Syntax and Types

Luigi Liquori, Claude Stolze

The Δ-calculus: Syntax and Types

Luigi Liquori **Claude Stolze** [1]
Université Côte d'Azur, Inria, France
[Luigi.Liquori,Claude.Stolze]@inria.fr

Abstract

We present the Δ-calculus, an explicitly typed λ-calculus with strong pairs, projections and explicit type coercions. The calculus can be parametrized with different intersection type theories $\mathcal{T}$, *e.g.* the Coppo-Dezani, the Coppo-Dezani-Sallé, the Coppo-Dezani-Venneri and the Barendregt-Coppo-Dezani ones, producing a family of Δ-calculi with related intersection typed systems. We prove the main properties like Church-Rosser, unicity of type, subject reduction, strong normalization, decidability of type checking and type reconstruction. We state the relationship between the intersection type assignment systems *à la* Curry and the corresponding intersection typed systems *à la* Church by means of an essence function translating an explicitly typed Δ-term into a pure λ-term one. We finally translate a Δ-term with type coercions into an equivalent one without them; the translation is proved to be coherent because its essence is the identity. The generic Δ-calculus can be parametrized to take into account other intersection type theories as the ones in the Barendregt *et al.* book.

Keywords Intersection types, Lambda calculus *à la* Church and *à la* Curry, Proof-functional logics

Acknowledgements We are grateful to Benjamin Pierce and Furio Honsell for the useful comments and remarks.

1 Introduction

Intersection type theories $\mathcal{T}$ were first introduced as a form of *ad hoc* polymorphism in (pure) λ-calculi *à la* Curry. The paper by Barendregt, Coppo, and Dezani [4] is a classic reference, while [5] is a definitive reference.
Intersection type assignment systems $\lambda_\cap^\mathcal{T}$ have been well-known in the literature for almost 40 years for many reasons: among them, characterization of strongly normalizing λ-terms [5], λ-models [1], automatic type inference [26], type inhabitation [43, 37], type unification [17]. As intersection had its classical development for type assignment systems, many papers tried to find an explicitly typed λ-calculus *à la* Church corresponding to the original intersection type assignment systems *à la* Curry. The programming language Forsythe, by Reynolds [38], is probably the first reference, while Pierce's Ph.D. thesis [33] combines also unions, intersections and bounded polymorphism. In [45] intersection types were used as a foundation for typed intermediate languages for optimizing compilers for higher-order polymorphic programming languages; implementations of typed programming language featuring intersection (and union) types can be found in SML-CIDRE [14] and in StardustML [18, 19].
Annotating pure λ-terms with intersection types is not simple: a classical example is the difficulty to decorate the bound variable of the explicitly typed polymorphic identity $\lambda x{:}?.x$ such that the type of the identity is $(\sigma \to \sigma) \cap (\tau \to \tau)$: previous attempts showed that the full power of the intersection type discipline can be easily lost.

[1] Work supported by the COST Action CA15123 EUTYPES "The European research network on types for programming and verification".

In this paper, we define and prove the main properties of the Δ-calculus, a generic intersection typed system for an explicitly typed λ-calculus *à la* Church enriched with strong pairs, denoted by $\langle \Delta_1 \,, \Delta_2 \rangle$, projections, denoted by $pr_i\, \Delta$, and type coercions, denoted by Δ^σ.
A *strong pair* $\langle \Delta_1 \,, \Delta_2 \rangle$ is a special kind of cartesian product such that the two parts of a pair satisfies a given property $\mathcal{R}$ on their "essence", that is $\wr\, \Delta_1 \wr\, \mathcal{R} \,\wr\, \Delta_2 \wr$.
An *essence* $\wr\, \Delta \,\wr$ of a Δ-term is a pure λ-term obtained by erasing type decorations, projections and choosing one of the two elements inside a strong pair. As examples,

$$\begin{aligned}
\wr\, \langle \lambda x{:}\sigma \cap \tau.pr_2\, x \,, \lambda x{:}\sigma \cap \tau.pr_1\, x \rangle \,\wr &\;=\; \lambda x.x \\
\wr\, \lambda x{:}(\sigma \to \tau) \cap \sigma.(pr_1\, x)(pr_2\, x) \,\wr &\;=\; \lambda x.x\, x \\
\wr\, \lambda x{:}\sigma \cap (\tau \cap \rho).\langle \langle pr_1\, x \,, pr_2\, pr_1\, x \rangle \,, pr_2\, pr_2\, x \rangle \,\wr &\;=\; \lambda x.x
\end{aligned}$$

and so on. Therefore, the essence of a Δ-term is its untyped skeleton: a strong pair $\langle \Delta_1 \,, \Delta_2 \rangle$ can be typechecked if and only if $\wr\, \Delta_1 \wr\, \mathcal{R} \,\wr\, \Delta_2 \wr$ is verified, otherwise the strong pair will be ill-typed. The essence also gives the exact mapping between a term and its typing *à la* Church and its corresponding term and type assignment *à la* Curry. Changing the parameters $\mathcal{T}$ and $\mathcal{R}$ results in defining a totally different intersection typed system. For the purpose of this paper, we study the four well-known intersection type theories $\mathcal{T}$, namely Coppo-Dezani $\mathcal{T}_{\mathrm{CD}}$ [11], Coppo-Dezani-Sallé $\mathcal{T}_{\mathrm{CDS}}$ [12], Coppo-Dezani-Venneri $\mathcal{T}_{\mathrm{CDV}}$ [13] and Barendregt-Coppo-Dezani $\mathcal{T}_{\mathrm{BCD}}$ [4]. We will inspect the above type theories using three equivalence relations $\mathcal{R}$ on pure λ-terms, namely $\equiv$, $=_\beta$ and $=_{\beta\eta}$.
The combination of the above $\mathcal{T}$ and $\mathcal{R}$ allows to define ten meaningful typed systems for the Δ-calculus that can be pictorially displayed in a "Δ-chair" (see Definition 9). Following the same style as in the Barendregt *et al.* book [5], the edges in the chair represent an inclusion relation over the set of derivable judgments.
A type coercion Δ^τ is a term of type τ whose type-decoration denotes an application of a subsumption rule to the term Δ of type σ such that $\sigma \leqslant_{\mathcal{T}} \tau$: if we omit type coercions, then we lose the uniqueness of type property.
Section 3 shows a number of typable examples in the systems presented in the Δ-chair: each example is provided with a corresponding type assignment derivation of its essence. Some historical examples of Pottinger [36], Hindley [23] and Ben-Yelles [6] are essentially re-decorated and inhabited (when possible) in the Δ-calculus. The aims of this section is both to make the reader comfortable with the different intersection typed systems, and to give a first intuition of the correspondence between Church-style and Curry-style calculi.
Section 4 proves the metatheory for all the systems in the Δ-chair: Church-Rosser, unicity of type, subject reduction, strong normalization, decidability of type checking and type reconstruction and studies the relations between intersection type assignment systems *à la* Curry and the corresponding intersection typed systems *à la* Church. Notions of soundness, completeness and isomorphism will relate type assignment and typed systems. We also show how to get rid of type coercions Δ^τ defining a translation function, denoted by $\|\_\|$, inspired by the one of Tannen *et al.* [42]: the intuition of the translation is that if Δ has type σ and $\sigma \leqslant_{\mathcal{T}} \tau$, then $\|\sigma \leqslant_{\mathcal{T}} \tau\|$ is a Δ-term of type $\sigma \to \tau$, $(\|\sigma \leqslant_{\mathcal{T}} \tau\|\ \|\Delta\|)$ has type τ and $\wr\, \|\sigma \leqslant_{\mathcal{T}} \tau\| \,\wr$ is the identity $\lambda x.x$.

1.1 λ-calculi with intersection types *à la* Church

Several calculi *à la* Church appeared in the literature: they capture the power of intersection types; we briefly review them.
The Forsythe programming language by Reynolds [38] annotates a λ-abstraction with types as in $\lambda x{:}\sigma_1|\cdots|\sigma_n.M$. However, we cannot type a typed term, whose type erasure is the combinator $\mathsf{K} \equiv \lambda x.\lambda y.x$, with the type $(\sigma \to \sigma \to \sigma) \cap (\tau \to \tau \to \tau)$.

Pierce [34] improves Forsythe by using a **for** construct to build *ad hoc* polymorphic typing, as in $\mathbf{for}\,\alpha \in \{\sigma,\tau\}.\lambda x{:}\alpha, \lambda y{:}\alpha.x$. However, we cannot type a typed term, whose type erasure is $\lambda x.\lambda y.\lambda z.(x\,y\,,x\,z)$, with the type
$((\sigma \to \rho) \cap (\tau \to \rho') \to \sigma \to \tau \to \rho \times \rho') \cap ((\sigma \to \sigma) \cap (\sigma \to \sigma) \to \sigma \to \sigma \to \sigma \times \sigma)$.
Freeman and Pfenning [20] introduced refinement types, that is types that allow *ad hoc* polymorphism for ML constructors. Intuitively, refinement types can be seen as subtypes of a standard type: the user first defines a type and then the refinement types of this type. The main motivation for these refinement types is to allow non-exhaustive pattern matching, which becomes exhaustive for a given refinement of the type of the argument. As an example, we can define a type `boolexp` for boolean expressions, with constructors `True`, `And`, `Not` and `Var`, and a refinement type `ground` for boolean expressions without variables, with the same constructors except `Var`: then, the constructor `True` has type $\texttt{boolexp}\cap\texttt{ground}$, the constructor `And` has type $(\texttt{boolexp} * \texttt{boolexp} \to \texttt{boolexp}) \cap (\texttt{ground} * \texttt{ground} \to \texttt{ground})$ and so on. However, intersection is meaningful only when using constructors.
Wells *et al.* [45] introduced λ^{CIL}, a typed intermediate λ-calculus for optimizing compilers for higher-order programming languages. The calculus features intersection, union and flow types, the latter being useful to optimize data representation. λ^{CIL} can faithfully encode an intersection type assignment derivation by introducing the concept of virtual tuple, *i.e.* a special kind of pair whose type erasure leads to exactly the same untyped λ-term. A parallel context and parallel substitution, similar to the notion of [27, 28], is defined to reduce expressions in parallel inside a virtual tuple. Subtyping is defined only on flow types and not on intersection types: this system can encode the $\lambda_\cap^{\mathrm{CD}}$ type assignment system.
Wells and Haak [46] introduced λ^{B}, a more compact typed calculus encoding of λ^{CIL}: in fact, by comparing Fig. 1 and Fig. 2 of [46] we can see that the set of typable terms with intersection types of λ^{CIL} and λ^{B} are the same. In that paper, virtual tuples are removed by introducing branching terms, typable with branching types, the latter representing intersection type schemes. Two operations on types and terms are defined, namely expand, expanding the branching shape of type annotations when a term is substituted into a new context, and select, to choose the correct branch in terms and types. As there are no virtual tuples, reductions do not need to be done in parallel. As in [45], the $\lambda_\cap^{\mathrm{CD}}$ type assignment system can be encoded.
Frisch *et al.* [21] designed a typed system with intersection, union, negation and recursive types. The authors inherit the usual problem of having a domain space $\mathcal{D}$ that contains all the terms and, at the same time, all the functions from $\mathcal{D}$ to $\mathcal{D}$. They prevent this by having an auxiliary domain space which is the disjoint union of $\mathcal{D}^2$ and $\mathcal{P}(\mathcal{D}^2)$. The authors interpret types as sets in a well-suited model where the set-inspired type constructs are interpreted as the corresponding to set-theoretical constructs. Moreover, the model manages higher-order functions in an elegant way. The subtyping relation is defined as a relation on the set-theoretical interpretation $[\![\_]\!]$ of the types. For instance, the problem $\sigma\cap\tau \leqslant \sigma$ will be interpreted as $[\![\sigma]\!]\cap[\![\tau]\!] \subseteq [\![\sigma]\!]$, where $\cap$ becomes the set intersection operator, and the decision program actually decides whether $([\![\sigma]\!] \cap [\![\tau]\!]) \cap \overline{[\![\sigma]\!]}$ is the empty set.
Bono *et al.* [7] introduced a relevant and strict parallel term constructor to build inhabitants of intersections and a simple call-by-value parallel reduction strategy. An infinite number of constants $c^{\sigma\Rightarrow\tau}$ is applied to typed variables x^σ such that $c^{\sigma\Rightarrow\tau}\,x^\sigma$ is upcasted to type τ. It also uses a local renaming typing rule, which changes type decoration in λ-abstractions, as well as coercions. Term synchronicity in the tuples is guaranteed by the typing rules. The calculus uses van Bakel's strict version [2] of the $\mathcal{T}_{\mathrm{CD}}$ intersection type theory.

1.2 Logics for intersection types

Proof-functional (or strong) logical connectives, introduced by Pottinger [36], take into account the shape of logical proofs, thus allowing for polymorphic features of proofs to be made explicit in formulæ. This differs from classical or intuitionistic connectives where the meaning of a compound formula is only dependent on the truth value or the provability of its subformulæ.

Pottinger was the first to consider the intersection $\cap$ as a proof-functional connective. He contrasted it to the intuitionistic connective $\wedge$ as follows: *"The intuitive meaning of $\cap$ can be explained by saying that to assert $A \cap B$ is to assert that one has a reason for asserting A which is also a reason for asserting B, while to assert $A \wedge B$ is to assert that one has a pair of reasons, the first of which is a reason for asserting A and the second of which is a reason for asserting B"*.

A simple example of a logical theorem involving intuitionistic conjunction which does not hold for proof-functional conjunction is $(A \supset A) \wedge (A \supset B \supset A)$. Otherwise there would exist a term which behaves both as I and as K. Later, Lopez-Escobar [30] and Mints [31] investigated extensively logics featuring both proof-functional and intuitionistic connectives especially in the context of realizability interpretations.

It is not immediate to extend the judgments-as-types Curry-Howard paradigm to logics supporting proof-functional connectives. These connectives need to compare the shapes of derivations and do not just take into account their provability, *i.e.* the inhabitation of the corresponding type.

There are many proposals to find a suitable logics to fit intersection types; among them we cite [44, 39, 32, 9, 7, 35], and previous papers by the authors [15, 29, 40].

1.3 Raising the Δ-calculus to a Δ-framework.

Our goal is to build a prototype of a theorem prover based on the Δ-calculus and proof-functional logic. Recently [25], we have extended a subset of the generic Δ-calculus with other proof-functional operators like union types, relevant arrow types, together with dependent types as in the Edinburgh Logical Framework [22]: a preliminary implementation of a type checker appeared in [40] by the authors. In a nutshell:

Strong disjunction is a proof-functional connective that can be interpreted as the union type $\cup$ [15, 40]: it contrasts with the intuitionistic connective $\vee$. As Pottinger did for intersection, we could say that asserting $(A \cup B) \supset C$ is to assert that one has a reason for $(A \cup B) \supset C$, which is also a reason to assert $A \supset C$ and $B \supset C$. A simple example of a logical theorem involving intuitionistic disjunction which does not hold for strong disjunction is $((A \supset B) \cup B) \supset A \supset B$. Otherwise there would exist a term which behaves both as I and as K.

Strong (relevant) implication is yet another proof-functional connective that was interpreted in [3] as a relevant arrow type $\to_r$. As explained in [3], it can be viewed as a special case of implication whose related function space is the simplest one, namely the one containing only the identity function. Because the operators $\supset$ and $\to_r$ differ, $A \to_r B \to_r A$ is not derivable.

Dependent types, as introduced in the Edinburgh Logical Framework [22] by Harper *et al.*, allows considering proofs as first-class citizens albeit differently with respect to proof-functional logics. The interaction of both dependent and proof-functional operators is intriguing: the former mentions proofs explicitly, while the latter mentions proofs implicitly. Their combination therefore opens up new possibilities of formal reasoning on proof-theoretic semantics.

Minimal type theory $\leqslant_{\min}$

(refl)	$\sigma \leqslant \sigma$	(incl)	$\sigma \cap \tau \leqslant \sigma, \sigma \cap \tau \leqslant \tau$
(glb)	$\rho \leqslant \sigma, \rho \leqslant \tau \Rightarrow \rho \leqslant \sigma \cap \tau$	(trans)	$\sigma \leqslant \tau, \tau \leqslant \rho \Rightarrow \sigma \leqslant \rho$

Axiom schemes

$(\mathtt{U}_{top})$	$\sigma \leqslant \mathtt{U}$	$(\mathtt{U}_{\rightarrow})$	$\mathtt{U} \leqslant \sigma \rightarrow \mathtt{U}$
$(\rightarrow\cap)$	$(\sigma \rightarrow \tau) \cap (\sigma \rightarrow \rho) \leqslant \sigma \rightarrow (\tau \cap \rho)$		

Rule scheme

$(\rightarrow)$ $\quad \sigma_2 \leqslant \sigma_1, \tau_1 \leqslant \tau_2 \Rightarrow \sigma_1 \rightarrow \tau_1 \leqslant \sigma_2 \rightarrow \tau_2$

Figure 1 Minimal type theory $\leqslant_{\min}$, axioms and rule schemes (see Fig. 13.2 and 13.3 of [5])

$$\frac{x{:}\sigma \in B}{B \vdash_\cap^{\mathcal{T}} x : \sigma}\ (ax) \qquad \frac{B, x{:}\sigma \vdash_\cap^{\mathcal{T}} M : \tau}{B \vdash_\cap^{\mathcal{T}} \lambda x.M : \sigma \rightarrow \tau}\ (\rightarrow I)$$

$$\frac{B \vdash_\cap^{\mathcal{T}} M : \sigma \quad B \vdash_\cap^{\mathcal{T}} M : \tau}{B \vdash_\cap^{\mathcal{T}} M : \sigma \cap \tau}\ (\cap I) \qquad \frac{B \vdash_\cap^{\mathcal{T}} M : \sigma \rightarrow \tau \quad B \vdash_\cap^{\mathcal{T}} N : \sigma}{B \vdash_\cap^{\mathcal{T}} M\,N : \tau}\ (\rightarrow E)$$

$$\frac{B \vdash_\cap^{\mathcal{T}} M : \sigma \cap \tau}{B \vdash_\cap^{\mathcal{T}} M : \sigma}\ (\cap E_1) \qquad \frac{B \vdash_\cap^{\mathcal{T}} M : \sigma \cap \tau}{B \vdash_\cap^{\mathcal{T}} M : \tau}\ (\cap E_2)$$

$$\frac{\mathtt{U} \in \mathbb{A}}{B \vdash_\cap^{\mathcal{T}} M : \mathtt{U}}\ (top) \qquad \frac{B \vdash_\cap^{\mathcal{T}} M : \sigma \quad \sigma \leqslant_{\mathcal{T}} \tau}{B \vdash_\cap^{\mathcal{T}} M : \tau}\ (\leqslant_{\mathcal{T}})$$

Figure 2 Generic intersection type assignment system $\lambda_\cap^{\mathcal{T}}$ (see Figure 13.8 of [5])

2 Syntax, Reduction and Types

▶ **Definition 1** (Type atoms, type syntax, type theories and type assignment systems). We briefly review some basic definition from Subsection 13.1 of [5], in order to define type assignment systems. The set of atoms, intersection types, intersection type theories and intersection type assignment systems are defined as follows:

1. **(Atoms).** Let $\mathbb{A}$ denote a set of symbols which we will call type atoms, and let $\mathtt{U}$ be a special type atom denoting the universal type. In particular, we will use $\mathbb{A}_\infty = \{\mathtt{a}_i \mid i \in \mathbb{N}\}$ with $\mathtt{a}_i$ being different from $\mathtt{U}$ and $\mathbb{A}_\infty^{\mathtt{U}} = \mathbb{A}_\infty \cup \{\mathtt{U}\}$.
2. **(Syntax).** The syntax of intersection types, parametrized by $\mathbb{A}$, is: $\sigma ::= \mathbb{A} \mid \sigma \rightarrow \sigma \mid \sigma \cap \sigma$.
3. **(Intersection type theories $\mathcal{T}$).** An intersection type theory $\mathcal{T}$ is a set of sentences of the form $\sigma \leqslant \tau$ satisfying at least the axioms and rules of the minimal type theory $\leqslant_{\min}$ defined in Figure 1. The type theories $\mathcal{T}_{\mathrm{CD}}, \mathcal{T}_{\mathrm{CDV}}, \mathcal{T}_{\mathrm{CDS}}$, and $\mathcal{T}_{\mathrm{BCD}}$ are the smallest type theories over $\mathbb{A}$ satisfying the axioms and rules given in Figure 3. We write $\mathcal{T}_1 \sqsubseteq \mathcal{T}_2$ if, for all σ, τ such that $\sigma \leqslant_{\mathcal{T}_1} \tau$, we have that $\sigma \leqslant_{\mathcal{T}_2} \tau$. In particular $\mathcal{T}_{\mathrm{CD}} \sqsubseteq \mathcal{T}_{\mathrm{CDV}} \sqsubseteq \mathcal{T}_{\mathrm{BCD}}$ and $\mathcal{T}_{\mathrm{CD}} \sqsubseteq \mathcal{T}_{\mathrm{CDS}} \sqsubseteq \mathcal{T}_{\mathrm{BCD}}$. We will sometime note, for instance, BCD instead of $\mathcal{T}_{\mathrm{BCD}}$.
4. **(Intersection type assignment systems $\lambda_\cap^{\mathcal{T}}$).** We define in Figure 2[§] an infinite collection of type assignment systems parametrized by a set of atoms $\mathbb{A}$ and a type theory $\mathcal{T}$. We name four particular type assignment systems in the table below, which is an excerpt from Figure 13.4 of [5]. $B \vdash_\cap^{\mathcal{T}} M : \sigma$ denotes a derivable type assignment judgment in the type assignment system $\lambda_\cap^{\mathcal{T}}$. Type checking is not decidable for $\lambda_\cap^{\mathrm{CD}}$, $\lambda_\cap^{\mathrm{CDV}}$, $\lambda_\cap^{\mathrm{CDS}}$, and $\lambda_\cap^{\mathrm{BCD}}$.

[§] Although rules $(\cap E_i)$ are derivable with $\leqslant_{\min}$, we add them for clarity.

$\lambda_\cap^{\mathcal{T}}$	$\mathcal{T}$	$\mathbb{A}$	$\leqslant_{\min}$ plus	ref.
$\lambda_\cap^{\mathrm{CD}}$	$\mathcal{T}_{\mathrm{CD}}$	$\mathbb{A}_\infty$	$-$	[11]
$\lambda_\cap^{\mathrm{CDS}}$	$\mathcal{T}_{\mathrm{CDS}}$	$\mathbb{A}_\infty^{\mathtt{U}}$	$(\mathtt{U}_{top})$	[12]
$\lambda_\cap^{\mathrm{CDV}}$	$\mathcal{T}_{\mathrm{CDV}}$	$\mathbb{A}_\infty$	$(\rightarrow), (\rightarrow\cap)$	[13]
$\lambda_\cap^{\mathrm{BCD}}$	$\mathcal{T}_{\mathrm{BCD}}$	$\mathbb{A}_\infty^{\mathtt{U}}$	$(\rightarrow), (\rightarrow\cap), (\mathtt{U}_{top}), (\mathtt{U}\rightarrow)$	[4]

Figure 3 Type theories $\lambda_\cap^{\mathrm{CD}}$, $\lambda_\cap^{\mathrm{CDS}}$, $\lambda_\cap^{\mathrm{CDV}}$, and $\lambda_\cap^{\mathrm{BCD}}$. The ref. column refers to the original article these theories come from.

2.1 The Δ-calculi

Intersection type assignment systems and Δ-calculi have in common their type syntax and intersection type theories. The generic syntax of the Δ-calculus is defined as follows.

▶ **Definition 2** (Generic Δ-calculus syntax).

$$\Delta \quad ::= \quad u_\Delta \mid x \mid \lambda x{:}\sigma.\Delta \mid \Delta\,\Delta \mid \langle \Delta \,, \Delta \rangle \mid pr_i\,\Delta \mid \Delta^\sigma \qquad\qquad i \in \{1,2\}$$

u_Δ denotes an infinite set of constants, indexed with a particular untyped Δ-term. Δ^σ denotes an explicit coercion of Δ to type σ. The expression $\langle \Delta \,, \Delta \rangle$ denotes a pair that, following the Lopez-Escobar jargon [30], we call "strong pair" with respective projections pr_1 and pr_2. The essence function $\wr \_ \wr$ is an erasing function mapping typed Δ-terms into pure λ-terms. It is defined as follows.

▶ **Definition 3** (Essence function).

$$\begin{array}{rcl@{\qquad}rcl@{\qquad}rcl}
\wr x \wr & \stackrel{def}{=} & x & \wr \Delta^\sigma \wr & \stackrel{def}{=} & \wr \Delta \wr & \wr u_\Delta \wr & \stackrel{def}{=} & \wr \Delta \wr \\
\wr \lambda x{:}\sigma.\Delta \wr & \stackrel{def}{=} & \lambda x.\wr \Delta \wr & \wr \Delta_1\,\Delta_2 \wr & \stackrel{def}{=} & \wr \Delta_1 \wr \wr \Delta_2 \wr & & & \\
\wr \langle \Delta_1 \,, \Delta_2 \rangle \wr & \stackrel{def}{=} & \wr \Delta_1 \wr & \wr pr_i\,\Delta \wr & \stackrel{def}{=} & \wr \Delta \wr \quad i \in \{1,2\} & & &
\end{array}$$

One could argue that the choice of $\wr \langle \Delta_1 \,, \Delta_2 \rangle \wr \stackrel{def}{=} \wr \Delta_1 \wr$ is arbitrary and could have been replaced with $\wr \langle \Delta_1 \,, \Delta_2 \rangle \wr \stackrel{def}{=} \wr \Delta_2 \wr$. However, the typing rules will ensure that, if $\langle \Delta_1 \,, \Delta_2 \rangle$ is typable, then, for some suitable equivalence relation $\mathcal{R}$, we have that $\wr \Delta_1 \wr \;\mathcal{R}\; \wr \Delta_2 \wr$. Thus, strong pairs can be viewed as constrained cartesian products. The generic reduction semantics reduces terms of the Δ-calculus as follows.

▶ **Definition 4** (Generic reduction semantics). Syntactical equality is denoted by $\equiv$.

1. **(Substitution)** Substitution on Δ-terms is defined as usual, with the additional rules:

$$u_{\Delta_1}[\Delta_2/x] \stackrel{def}{=} u_{(\Delta_1[\Delta_2/x])} \qquad \text{and} \qquad \Delta_1^\sigma[\Delta_2/x] \stackrel{def}{=} (\Delta_1[\Delta_2/x])^\sigma$$

2. **(One-step reduction).** We define three notions of reduction:

$$\begin{array}{rcll}
(\lambda x{:}\sigma.\Delta_1)\,\Delta_2 & \longrightarrow & \Delta_1[\Delta_2/x] & (\beta) \\
pr_i\,\langle \Delta_1 \,, \Delta_2 \rangle & \longrightarrow & \Delta_i \quad i \in \{1,2\} & (pr_i) \\
\lambda x{:}\sigma.\Delta\,x & \longrightarrow & \Delta \quad x \notin \mathrm{FV}(\Delta) & (\eta)
\end{array}$$

Observe that $(\lambda x{:}\sigma.\Delta_1)^\sigma\,\Delta_2$ is not a redex, because the λ-abstraction is coerced. The contextual closure is defined as usual except for reductions inside the index of u_Δ that

$$\frac{\mathtt{U} \in \mathbb{A}}{B \vdash^{\mathcal{T}}_{\mathcal{R}} u_\Delta : \mathtt{U}}\ (top) \qquad \frac{x{:}\sigma \in B}{B \vdash^{\mathcal{T}}_{\mathcal{R}} x : \sigma}\ (ax) \qquad \frac{B, x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \tau}{B \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma.\Delta : \sigma \to \tau}\ (\to I)$$

$$\frac{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \sigma \quad B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \tau \quad \wr\Delta_1\wr \,\mathcal{R}\, \wr\Delta_2\wr}{B \vdash^{\mathcal{T}}_{\mathcal{R}} \langle \Delta_1 , \Delta_2 \rangle : \sigma \cap \tau}\ (\cap I) \qquad \frac{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \sigma \to \tau \quad B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \sigma}{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 \Delta_2 : \tau}\ (\to E)$$

$$\frac{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma \cap \tau}{B \vdash^{\mathcal{T}}_{\mathcal{R}} pr_1\, \Delta : \sigma}\ (\cap E_1) \qquad \frac{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma \cap \tau}{B \vdash^{\mathcal{T}}_{\mathcal{R}} pr_2\, \Delta : \tau}\ (\cap E_2) \qquad \frac{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma \quad \sigma \leqslant_{\mathcal{T}} \tau}{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta^{\tau} : \tau}\ (\leqslant_{\mathcal{T}})$$

Figure 4 Generic intersection typed system $\Delta^{\mathcal{T}}_{\mathcal{R}}$

are forbidden (even though substitutions are propagated). We write $\longrightarrow_{\beta pr_i}$ for the contextual closure of the (β) and (pr_i) notions of reduction, $\longrightarrow_{\eta}$ for the contextual closure of (η). We also define a synchronous contextual closure, which is like the usual contextual closure except for the strong pairs, as defined in point (3). Synchronous contextual closure of the notions of reduction generates the reduction relations $\longrightarrow^{\parallel}_{\beta pr_i}$ and $\longrightarrow^{\parallel}_{\eta}$.

3. **(Synchronous closure of $\longrightarrow^{\parallel}$).** Synchronous closure is defined on the strong pairs with the following constraint:

$$\frac{\Delta_1 \longrightarrow^{\parallel} \Delta'_1 \quad \Delta_2 \longrightarrow^{\parallel} \Delta'_2 \quad \wr\Delta'_1\wr \equiv \wr\Delta'_2\wr}{\langle \Delta_1 , \Delta_2 \rangle \longrightarrow^{\parallel} \langle \Delta'_1 , \Delta'_2 \rangle}\ (Clos^{\parallel})$$

Note that we reduce in the two components of the strong pair;

4. **(Multistep reduction).** We write $\longrightarrow\!\!\!\!\!\rightarrow_{\beta pr_i}$ (resp. $\longrightarrow\!\!\!\!\!\rightarrow^{\parallel}_{\beta pr_i}$) as the reflexive and transitive closure of $\longrightarrow_{\beta pr_i}$ (resp. $\longrightarrow^{\parallel}_{\beta pr_i}$);
5. **(Congruence).** We write $=_{\beta pr_i}$ as the symmetric, reflexive, transitive closure of $\longrightarrow\!\!\!\!\!\rightarrow_{\beta pr_i}$.

We mostly consider βpr_i-reductions, thus to ease the notation we omit the subscript in βpr_i-reductions.

The next definition introduces a notion of synchronization inside strong pairs.

▶ **Definition 5** (Synchronization)**.** A Δ-term is synchronous if and only if, for all its subterms of the shape $\langle \Delta_1 , \Delta_2 \rangle$, we have that $\wr\Delta_1\wr \equiv \wr\Delta_2\wr$.

It is easy to verify that $\longrightarrow^{\parallel}$ preserves synchronization, while it is not the case for $\longrightarrow$. The next definition introduces a generic intersection typed system for the Δ-calculus that is parametrizable by suitable equivalence relations on pure λ-terms $\mathcal{R}$ and type theories $\mathcal{T}$ as follows.

▶ **Definition 6** (Generic intersection typed system)**.** The generic intersection typed system is defined in Figure 4. We denote by $\Delta^{\mathcal{T}}_{\mathcal{R}}$ a particular typed system with the type theory $\mathcal{T}$ and under an equivalence relation $\mathcal{R}$ and by $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$ a corresponding typing judgment.

The typing rules are intuitive for a calculus *à la* Church except rules $(\cap I)$, (top) and $(\leqslant_{\mathcal{T}})$. The typing rule for a strong pair $(\cap I)$ is similar to the typing rule for a cartesian product, except for the side-condition $\wr\Delta_1\wr \,\mathcal{R}\, \wr\Delta_2\wr$, forcing the two parts of the strong pair to have essences compatible under $\mathcal{R}$, thus making a strong pair a special case of a cartesian pair. For instance, $\langle \lambda x{:}\sigma.\lambda y{:}\tau.x , \lambda x{:}\sigma.x \rangle$ is not typable in $\Delta^{\mathcal{T}}_{\equiv}$; $\langle (\lambda x{:}\sigma.x)\, y , y \rangle$ is not typable in $\Delta^{\mathcal{T}}_{\equiv}$ but it is in $\Delta^{\mathcal{T}}_{=_\beta}$; $\langle x , \lambda y{:}\sigma.((\lambda z{:}\tau.z)\, x)\, y \rangle$ is not typable in $\Delta^{\mathcal{T}}_{\equiv}$ nor $\Delta^{\mathcal{T}}_{=_\beta}$ but it is in $\Delta^{\mathcal{T}}_{=_{\beta\eta}}$. In the typing rule (top), the subscript Δ in u_Δ is not necessarily typable so $\wr u_\Delta \wr$ can easily be any arbitrary λ-term. The typing rule $(\leqslant_{\mathcal{T}})$ allows to change the type of a Δ-term from

σ to τ if $\sigma \leqslant_{\mathcal{T}} \tau$: the term in the conclusion must record this change with an explicit type coercion $\_^{\tau}$, producing the new term Δ^{τ}: explicit type coercions are important to keep the unicity of typing derivations.

The next definition introduces a partial order over equivalence relations on pure λ-terms and an inclusion over typed systems as follows.

▶ **Definition 7** ($\mathcal{R}$ and $\sqsubseteq$)**.** **1.** Let $\mathcal{R} \in \{\equiv, =_{\beta}, =_{\beta\eta}\}$. $\mathcal{R}_1 \sqsubseteq \mathcal{R}_2$ if, for all pure λ-terms M, N such that $M \; \mathcal{R}_1 \; N$, we have that $M \; \mathcal{R}_2 \; N$;

2. if $B \vdash^{\mathcal{T}_1}_{\mathcal{R}_1} \Delta : \sigma$ and $\Delta^{\mathcal{T}_1}_{\mathcal{R}_1} \sqsubseteq \Delta^{\mathcal{T}_2}_{\mathcal{R}_2}$, then $B \vdash^{\mathcal{T}_2}_{\mathcal{R}_2} \Delta : \sigma$.

▶ **Lemma 8.** **1.** $\Delta^{\text{CD}}_{\mathcal{R}} \sqsubseteq \Delta^{\text{CDS}}_{\mathcal{R}} \sqsubseteq \Delta^{\text{BCD}}_{\mathcal{R}}$ *and* $\Delta^{\text{CD}}_{\mathcal{R}} \sqsubseteq \Delta^{\text{CDV}}_{\mathcal{R}} \sqsubseteq \Delta^{\text{BCD}}_{\mathcal{R}}$*;*

2. $\Delta^{\mathcal{T}_1}_{\mathcal{R}_1} \sqsubseteq \Delta^{\mathcal{T}_2}_{\mathcal{R}_2}$ *if* $\mathcal{T}_1 \sqsubseteq \mathcal{T}_2$ *and* $\mathcal{R}_1 \sqsubseteq \mathcal{R}_2$*.*

2.2 The Δ-chair

The next definition classifies ten typed systems for the Δ-calculus: some of them already appeared (sometime with a different notation) in the literature by the present authors.

▶ **Definition 9** (Δ-chair)**.**

Ten typed systems $\Delta^{\mathcal{T}}_{\mathcal{R}}$ can be drawn pictorially in a Δ-chair, where the arrows represent an inclusion relation. $\Delta^{\text{CD}}_{\equiv}$ corresponds roughly to [27, 28] (in the expression $M@\Delta$, M is the essence of Δ) and in its intersection part to [40]; $\Delta^{\text{CDS}}_{\equiv}$ corresponds roughly in its intersection part to [16], $\Delta^{\text{BCD}}_{\equiv}$ corresponds in its intersection part to [29], $\Delta^{\text{CD}}_{=_{\beta\eta}}$ corresponds in its intersection part to [15]. The other typed systems are basically new. The main properties of these systems are:

1. All the $\Delta^{\mathcal{T}}_{\equiv}$ systems enjoys the synchronous subject reduction property, the other systems also enjoy ordinary subject reduction (Th. 23);
2. All the systems strongly normalize (Th. 26);
3. All the systems correspond to the to original type assignment systems except $\Delta^{\text{CD}}_{=_{\beta}}$, $\Delta^{\text{CDV}}_{=_{\beta}}$, $\Delta^{\text{CDV}}_{=_{\beta\eta}}$ and $\Delta^{\text{BCD}}_{=_{\beta\eta}}$ (Th. 28);
4. Type checking and type reconstruction are decidable for all the systems, except $\Delta^{\text{CDS}}_{=_{\beta}}$, $\Delta^{\text{BCD}}_{=_{\beta}}$, and $\Delta^{\text{BCD}}_{=_{\beta\eta}}$ (Th. 30).

$$\begin{tikzcd}
& \Delta^{\text{CDV}}_{=_{\beta\eta}} \arrow[rr] & & \Delta^{\text{BCD}}_{=_{\beta\eta}} \\
& \Delta^{\text{CDV}}_{=_{\beta}} \arrow[rr] \arrow[u] & & \Delta^{\text{BCD}}_{=_{\beta}} \arrow[u] \\
\Delta^{\text{CD}}_{=_{\beta}} \arrow[ur] \arrow[rr] & & \Delta^{\text{CDS}}_{=_{\beta}} \arrow[ur] & \\
& \Delta^{\text{CDV}}_{\equiv} \arrow[rr] \arrow[uu] & & \Delta^{\text{BCD}}_{\equiv} \arrow[uu] \\
\Delta^{\text{CD}}_{\equiv} \arrow[ur] \arrow[rr] \arrow[uu] & & \Delta^{\text{CDS}}_{\equiv} \arrow[ur] \arrow[uu] &
\end{tikzcd}$$

3 Examples

This section shows examples of typed derivations $\Delta^{\mathcal{T}}_{\mathcal{R}}$ and highlights the corresponding type assignment judgment in $\lambda^{\mathcal{T}}_{\cap}$ they correspond to, in the sense that we have a derivation $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$ and another derivation $B \vdash^{\mathcal{T}}_{\cap} \wr \Delta \wr : \sigma$. The correspondence between intersection typed systems $\Delta^{\mathcal{T}}_{\mathcal{R}}$ and intersection type assignment $\lambda^{\mathcal{T}}_{\cap}$ will be defined in Subsection 5.1.

▶ **Example 10** (Polymorphic identity)**.** In all of the intersection type assignment systems $\lambda^{\mathcal{T}}_{\cap}$ we can derive $\vdash^{\mathcal{T}}_{\cap} \lambda x.x : (\sigma \to \sigma) \cap (\tau \to \tau)$ A corresponding Δ-term is: $\langle \lambda x{:}\sigma.x \, , \lambda x{:}\tau.x \rangle$ that can be typed in all of the typed systems of the Δ-chair as follows

$$\dfrac{\dfrac{x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} x : \sigma}{\vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma.x : \sigma \to \sigma} \quad \dfrac{x{:}\tau \vdash^{\mathcal{T}}_{\mathcal{R}} x : \tau}{\vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\tau.x : \tau \to \tau} \quad \lambda x.x \; \mathcal{R} \; \lambda x.x}{\vdash^{\mathcal{T}}_{\mathcal{R}} \langle \lambda x{:}\sigma.x \, , \lambda x{:}\tau.x \rangle : (\sigma \to \sigma) \cap (\tau \to \tau)}$$

▶ **Example 11** (Auto application). In all of the intersection type assignment systems we can derive $\vdash^{\mathcal{T}}_{\cap} \lambda x.x\,x : ((\sigma \to \tau) \cap \sigma) \to \tau$ A corresponding Δ-term is: $\lambda x{:}(\sigma \to \tau) \cap \sigma.(pr_1\, x)(pr_2\, x)$ that can be typed in all of the typed systems of the Δ-chair as follows

$$\dfrac{\dfrac{\dfrac{x{:}(\sigma \to \tau) \cap \sigma \vdash^{\mathcal{T}}_{\mathcal{R}} x : (\sigma \to \tau) \cap \sigma}{x{:}(\sigma \to \tau) \cap \sigma \vdash^{\mathcal{T}}_{\mathcal{R}} pr_1\, x : \sigma \to \tau} \quad \dfrac{x{:}(\sigma \to \tau) \cap \sigma \vdash^{\mathcal{T}}_{\mathcal{R}} x : (\sigma \to \tau) \cap \sigma}{x{:}(\sigma \to \tau) \cap \sigma \vdash^{\mathcal{T}}_{\mathcal{R}} pr_2\, x : \sigma}}{x{:}(\sigma \to \tau) \cap \sigma \vdash^{\mathcal{T}}_{\mathcal{R}} (pr_1\, x)(pr_2\, x) : \tau}}{\vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}(\sigma \to \tau) \cap \sigma.(pr_1\, x)(pr_2\, x) : (\sigma \to \tau) \cap \sigma \to \tau}$$

▶ **Example 12** (Some examples in $\Delta^{\text{CDS}}_{\mathcal{R}}$). In $\lambda^{\text{CDS}}_{\cap}$ we can derive $\vdash^{\mathcal{T}_{\text{CDS}}}_{\cap} (\lambda x.\lambda y.x) : \sigma \to \mathtt{U} \to \sigma$, and using this type assignment, we can derive $z{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\cap} (\lambda x.\lambda y.x)\, z\, z : \sigma$. A corresponding Δ-term is: $(\lambda x{:}\sigma.\lambda y{:}\mathtt{U}.x)\, z\, z^{\mathtt{U}}$ that can be typed in $\Delta^{\text{CDS}}_{\mathcal{R}}$ as follows

$$\dfrac{\dfrac{\dfrac{\dfrac{z{:}\sigma, x{:}\sigma, y{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} x : \sigma}{z{:}\sigma, x{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \lambda y{:}\mathtt{U}.x : \mathtt{U} \to \sigma}}{z{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \lambda x{:}\sigma.\lambda y{:}\mathtt{U}.x : \sigma \to \mathtt{U} \to \sigma} \quad z{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} z : \sigma}{z{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} (\lambda x{:}\sigma.\lambda y{:}\mathtt{U}.x)\, z : \mathtt{U} \to \sigma} \quad \dfrac{z{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} z : \sigma \quad \sigma \leqslant_{\mathcal{T}_{\text{CDS}}} \mathtt{U}}{z{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} z^{\mathtt{U}} : \mathtt{U}}}{z{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} (\lambda x{:}\sigma.\lambda y{:}\mathtt{U}.x)\, z\, z^{\mathtt{U}} : \sigma}$$

As another example, we can also derive $\vdash^{\mathcal{T}_{\text{CDS}}}_{\cap} \lambda x.x : \sigma \to \sigma \cap \mathtt{U}$. A corresponding Δ-term is: $\lambda x{:}\sigma.\langle x\,, x^{\mathtt{U}}\rangle$ that can be typed in $\Delta^{\text{CDS}}_{\mathcal{R}}$ as follows

$$\dfrac{\dfrac{x{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} x : \sigma \quad \dfrac{x{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} x : \sigma \quad \sigma \leqslant_{\mathcal{T}_{\text{CDS}}} \mathtt{U}}{x{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} x^{\mathtt{U}} : \mathtt{U}} \quad x\ \mathcal{R}\ x}{x{:}\sigma \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \langle x\,, x^{\mathtt{U}}\rangle : \sigma \cap \mathtt{U}}}{\vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \lambda x{:}\sigma.\langle x\,, x^{\mathtt{U}}\rangle : \sigma \to \sigma \cap \mathtt{U}}$$

▶ **Example 13** (An example in $\Delta^{\text{CDV}}_{\mathcal{R}}$). In $\lambda^{\text{CDV}}_{\cap}$ we can prove the commutativity of intersection, *i.e.*$\vdash^{\mathcal{T}_{\text{CDV}}}_{\cap} \lambda x.x : \sigma{\cap}\tau \to \tau{\cap}\sigma$ A corresponding Δ-term is: $\langle \lambda x{:}\sigma \cap \tau.pr_2\, x, \lambda x{:}\sigma \cap \tau.pr_1\, x\rangle^{(\sigma\cap\tau)\to(\tau\cap\sigma)}$ that can be typed in $\Delta^{\text{CDV}}_{\mathcal{R}}$ as follows

$$\dfrac{\dfrac{\dfrac{\dfrac{x{:}\sigma \cap \tau \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} x : \sigma \cap \tau}{x{:}\sigma \cap \tau \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} pr_2\, x : \tau}}{\vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \lambda x{:}\sigma \cap \tau.pr_2\, x : (\sigma \cap \tau) \to \tau} \quad \dfrac{\dfrac{x{:}\sigma \cap \tau \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} x : \sigma \cap \tau}{x{:}\sigma \cap \tau \vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} pr_1\, x : \sigma}}{\vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \lambda x{:}\sigma \cap \tau.pr_1\, x : (\sigma \cap \tau) \to \sigma} \quad \lambda x.x\ \mathcal{R}\ \lambda x.x}{\vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \langle \lambda x{:}\sigma \cap \tau.pr_2\, x\,, \lambda x{:}\sigma \cap \tau.pr_1\, x\rangle : ((\sigma \cap \tau) \to \tau) \cap ((\sigma \cap \tau) \to \sigma)} \quad *}{\vdash^{\mathcal{T}_{\text{CDS}}}_{\mathcal{R}} \langle \lambda x{:}\sigma \cap \tau.pr_2\, x\,, \lambda x{:}\sigma \cap \tau.pr_1\, x\rangle^{(\sigma\cap\tau)\to(\tau\cap\sigma)} : (\sigma \cap \tau) \to (\tau \cap \sigma)}$$

where $*$ is $((\sigma \cap \tau) \to \tau) \cap ((\sigma \cap \tau) \to \sigma) \leqslant_{\mathcal{T}_{\text{CDV}}} (\sigma \cap \tau) \to (\tau \cap \sigma)$.

▶ **Example 14** (Another polymorphic identity in $\Delta^{\mathcal{T}}_{=_\beta}$). In all the $\Delta^{\mathcal{T}}_{=_\beta}$ you can type this Δ-term: $\langle \lambda x{:}\sigma.x\,, (\lambda x{:}\tau{\to}\tau.x)\, (\lambda x{:}\tau.x)\rangle$ The typing derivation is thus

$$\dfrac{\dfrac{x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} x : \sigma}{\vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.x : \sigma \to \sigma} \quad \dfrac{\dfrac{x{:}\tau \to \tau \vdash^{\mathcal{T}}_{=_\beta} x : \tau \to \tau}{\vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\tau{\to}\tau.x : (\tau \to \tau) \to (\tau \to \tau)} \quad \dfrac{x{:}\tau \vdash^{\mathcal{T}}_{=_\beta} x : \tau}{\vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\tau.x : \tau \to \tau}}{\vdash^{\mathcal{T}}_{=_\beta} (\lambda x{:}\tau{\to}\tau.x)\, (\lambda x{:}\tau.x) : \tau \to \tau} \quad \lambda x.x =_\beta (\lambda x.x)\, (\lambda x.x)}{\vdash^{\mathcal{T}}_{=_\beta} \langle \lambda x{:}\sigma.x\,, (\lambda x{:}\tau{\to}\tau.x)\, (\lambda x{:}\tau.x)\rangle : (\sigma \to \sigma) \cap (\tau \to \tau)}$$

▶ **Example 15** (Two examples in $\Delta^{\text{BCD}}_{\equiv}$ and $\Delta^{\text{BCD}}_{=_{\beta\eta}}$)**.** In $\lambda^{\text{BCD}}_{\cap}$ we can can type any term, including the non-terminating term $\Omega \stackrel{def}{=} (\lambda x.x\,x)\,(\lambda x.x\,x)$ More precisely, we have: $\vdash^{\mathcal{T}_{\text{BCD}}}_{\cap} \Omega : \mathtt{U}$ A corresponding Δ-term whose essence is Ω is: $(\lambda x{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,x)\,(\lambda x{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,x)^{\mathtt{U}}$ that can be typed in $\Delta^{\text{BCD}}_{\mathcal{R}}$ as follows

$$\dfrac{\dfrac{*}{\vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} \lambda x{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,x : \mathtt{U}\to\mathtt{U}} \quad \dfrac{\dfrac{*}{\vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} \lambda x{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,x : \mathtt{U}\to\mathtt{U}} \quad \mathtt{U}\to\mathtt{U} \leqslant_{\mathcal{T}_{\text{BCD}}} \mathtt{U}}{\vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} (\lambda x{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,x)^{\mathtt{U}} : \mathtt{U}}}{\vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} (\lambda x{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,x)\,(\lambda x{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,x)^{\mathtt{U}} : \mathtt{U}}$$

where $*$ is

$$\dfrac{\dfrac{x{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} x : \mathtt{U} \quad \mathtt{U} \leqslant_{\mathcal{T}_{\text{BCD}}} \mathtt{U}\to\mathtt{U}}{x{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} x^{\mathtt{U}\to\mathtt{U}} : \mathtt{U}\to\mathtt{U}} \quad x{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} x : \mathtt{U}}{x{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{\mathcal{R}} x^{\mathtt{U}\to\mathtt{U}}\,x : \mathtt{U}}$$

In $\lambda^{\text{BCD}}_{\cap}$ we can type $x{:}\mathtt{U}\to\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{\cap} x : (\mathtt{U}\to\mathtt{U}) \cap (\sigma\to\mathtt{U})$ A corresponding Δ-term whose essence is x is: $\langle x\,,\lambda y{:}\sigma.x\,y^{\mathtt{U}}\rangle$ that can be typed in $\Delta^{\text{BCD}}_{=_{\beta\eta}}$ as follows

$$\dfrac{x{:}\mathtt{U}\to\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} x : \mathtt{U}\to\mathtt{U} \quad \dfrac{\dfrac{x{:}\mathtt{U}\to\mathtt{U}, y{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} x : \mathtt{U}\to\mathtt{U} \quad \dfrac{x{:}\mathtt{U}\to\mathtt{U}, y{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} y : \sigma \quad \sigma \leqslant \mathtt{U}}{x{:}\mathtt{U}\to\mathtt{U}, y{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} y^{\mathtt{U}} : \mathtt{U}}}{x{:}\mathtt{U}\to\mathtt{U}, y{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} x\,y^{\mathtt{U}} : \mathtt{U}}}{x{:}\mathtt{U}\to\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} \lambda y{:}\sigma.x\,y^{\mathtt{U}} : \sigma\to\mathtt{U}} \quad x =_{\beta\eta} \lambda y.x\,y}{x{:}\mathtt{U}\to\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} \langle x\,,\lambda y{:}\sigma.x\,y^{\mathtt{U}}\rangle : (\mathtt{U}\to\mathtt{U}) \cap (\sigma\to\mathtt{U})}$$

Note that the $=_{\beta\eta}$ condition has an interesting loophole, as it is well-known that $\lambda^{\text{BCD}}_{\cap}$ does not enjoy $=_{\eta}$-conversion property. Theorem 51(1) will show that we can construct a Δ-term which does not correspond to any $\lambda^{\text{BCD}}_{\cap}$ derivation.

▶ **Example 16** (Pottinger)**.** The following examples can be typed in all the type theories of the Δ-chair (we also display in square brackets the corresponding pure λ-terms typable in $\lambda^{\mathcal{T}}_{\cap}$). These are encodings from the examples *à la* Curry given by Pottinger in [36].

$[\lambda x.\lambda y.x\,y] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}(\sigma\to\tau)\cap(\sigma\to\rho).\lambda y{:}\sigma.\langle (pr_1\,x)\,y)\,,(pr_2\,x)\,y\rangle : (\sigma\to\tau)\cap(\sigma\to\rho)\to\sigma\to\tau\cap\rho$

$[\lambda x.\lambda y.x\,y] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma\to\tau\cap\rho.\langle \lambda y{:}\sigma.pr_1\,(x\,y)\,,\lambda y{:}\sigma.pr_2\,(x\,y)\rangle : (\sigma\to\tau\cap\rho)\to(\sigma\to\tau)\cap(\sigma\to\rho)$

$[\lambda x.\lambda y.x\,y] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma\to\rho.\lambda y{:}\sigma\cap\tau.x\,(pr_1\,y) : (\sigma\to\rho)\to\sigma\cap\tau\to\rho$

$[\lambda x.\lambda y.x] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma\cap\tau.\lambda y{:}\sigma.pr_2\,x : \sigma\cap\tau\to\sigma\to\tau$

$[\lambda x.\lambda y.x\,y\,y] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma\to\tau\to\rho.\lambda y{:}\sigma\cap\tau.x\,(pr_1\,y)\,(pr_2\,y) : (\sigma\to\tau\to\rho)\to\sigma\cap\tau\to\rho$

$[\lambda x.x] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma\cap\tau.pr_1\,x : \sigma\cap\tau\to\sigma$

$[\lambda x.x] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma.\langle x\,,x\rangle : \sigma\to\sigma\cap\sigma$

$[\lambda x.x] \quad \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma\cap(\tau\cap\rho).\langle\langle pr_1\,x\,,pr_1\,pr_2\,x\rangle\,,pr_2\,pr_2\,x\rangle : \sigma\cap(\tau\cap\rho)\to(\sigma\cap\tau)\cap\rho$

In the same paper, Pottinger lists some types that cannot be inhabited by any intersection type assignment ($\nvdash^{\mathcal{T}}_{\cap}$) in an empty context, namely: $\sigma\to(\sigma\cap\tau)$ and $(\sigma\to\tau)\to(\sigma\to\rho)\to\sigma\to\tau\cap\rho$ and $((\sigma\cap\tau)\to\rho)\to\sigma\to\tau\to\rho$. It is not difficult to verify that the above types cannot be inhabited by any of the type systems of the Δ-chair because of the failure of the essence condition in the strong pair type rule.

▶ **Example 17** (Intersection is not the conjunction operator)**.** This counter-example is from the corresponding counter-example *à la* Curry given by Hindley [24] and Ben-Yelles [6]. The intersection type $(\sigma \to \sigma) \cap ((\sigma \to \tau \to \rho) \to (\sigma \to \tau) \to \sigma \to \rho)$ where the left part of the intersection corresponds to the type for the combinator I and the right part for the combinator S cannot be assigned to a pure λ-term. Analogously, the same intersection type cannot be assigned to any Δ-term.

3.1 On synchronization and subject reduction

For the typed systems $\Delta^{\mathcal{T}}_{\equiv}$, strong pairs have an intrinsic notion of synchronization: some redexes need to be reduced in a synchronous fashion unless we want to create meaningless Δ-terms that cannot be typed. Consider the Δ-term $\langle(\lambda x{:}\sigma.x)\, y\,, (\lambda x{:}\sigma.x)\, y\rangle$. If we use the $\longrightarrow$ reduction relation, then the following reduction paths are legal $\langle(\lambda x{:}\sigma.x)\, y\,, (\lambda x{:}\sigma.x)\, y\rangle \begin{array}{c} \nearrow^{\beta} \langle(\lambda x{:}\sigma.x)\, y\,, y\rangle \searrow_{\beta} \\ \searrow_{\beta} \langle y\,, (\lambda x{:}\sigma.x)\, y\rangle \nearrow^{\beta} \end{array} \langle y\,, y\rangle$. More precisely, the first and second redexes are rewritten asynchronously, thus they cannot be typed in any typed system $\Delta^{\mathcal{T}}_{\equiv}$, because we fail to check the left and the right part of the strong pair to be the same: the $\longrightarrow^{\parallel}$ reduction relation prevents this loophole and allows to type all redexes. In summary, $\longrightarrow^{\parallel}$ can be thought of as the natural reduction relation for the typed systems $\Delta^{\mathcal{T}}_{\equiv}$.

4 Metatheory of $\Delta^{\mathcal{T}}_{\mathcal{R}}$

4.1 General properties

Unless specified, all properties applies to the intersection typed systems $\Delta^{\mathcal{T}}_{\mathcal{R}}$. For lack of space all proofs are omitted: the interested reader can found more technical details in the Appendix. The Church-Rosser property is proved using the technique of Takahashi [41]. The parallel reduction semantics extends Definition 4 and it is inductively defined as follows.

▶ **Definition 18** (Parallel reduction semantics)**.**

$$
\begin{array}{rcllr}
x & \Longrightarrow & x & \textit{and} & u_\Delta \Longrightarrow u_\Delta \\
\Delta^\sigma & \Longrightarrow & (\Delta')^\sigma & \text{if } \Delta \Longrightarrow \Delta' & \\
\Delta_1\, \Delta_2 & \Longrightarrow & \Delta'_1\, \Delta'_2 & \text{if } \Delta_1 \Longrightarrow \Delta'_1 & \text{and } \Delta_2 \Longrightarrow \Delta'_2 \\
\lambda x{:}\sigma.\Delta & \Longrightarrow & \lambda x{:}\sigma.\Delta' & \text{if } \Delta \Longrightarrow \Delta' & \\
(\lambda x{:}\sigma.\Delta_1)\, \Delta_2 & \Longrightarrow & \Delta'_1[\Delta'_2/x] & \text{if } \Delta_1 \Longrightarrow \Delta'_1 & \text{and } \Delta_2 \Longrightarrow \Delta'_2 \\
\langle \Delta_1\,, \Delta_2\rangle & \Longrightarrow & \langle \Delta'_1\,, \Delta'_2\rangle & \text{if } \Delta_1 \Longrightarrow \Delta'_1 & \text{and } \Delta_2 \Longrightarrow \Delta'_2 \\
pr_i\, \Delta & \Longrightarrow & pr_i\, \Delta' & \text{if } \Delta \Longrightarrow \Delta' & \text{and } i \in \{1,2\} \\
pr_i\, \langle \Delta_1\,, \Delta_2\rangle & \Longrightarrow & \Delta'_i & \text{if } \Delta_i \Longrightarrow \Delta'_i & \text{and } i \in \{1,2\}
\end{array}
$$

Intuitively, $\Delta \Longrightarrow \Delta'$ means that Δ' is obtained from Δ by simultaneous contraction of some βpr_i-redexes possibly overlapping each other. Church-Rosser can be achieved by proving a stronger statement, namely $\Delta \Longrightarrow \Delta'$ implies $\Delta' \Longrightarrow \Delta^*$ where Δ^* is a Δ-term determined by Δ and independent from Δ'. The statement (1) is satisfied by the term Δ^* which is obtained from Δ by contracting all the redexes existing in Δ simultaneously.

▶ **Definition 19** (The map $\_^*$).

$$
\begin{array}{rcll}
x^* & \stackrel{def}{=} & x & u^*_\Delta \stackrel{def}{=} u_\Delta \\
(\Delta^\sigma)^* & \stackrel{def}{=} & (\Delta^*)^\sigma & \langle \Delta_1 \,, \Delta_2 \rangle^* \stackrel{def}{=} \langle \Delta^*_1 \,, \Delta^*_2 \rangle \\
(\lambda x{:}\sigma.\Delta)^* & \stackrel{def}{=} & \lambda x{:}\sigma.\Delta^* & ((\lambda x{:}\sigma.\Delta_1)\, \Delta_2)^* \stackrel{def}{=} \Delta^*_1[\Delta^*_2/x] \\
(\Delta_1\, \Delta_2)^* & \stackrel{def}{=} & \Delta^*_1\, \Delta^*_2 & \text{if } \Delta_1\, \Delta_2 \text{ is not a } \beta\text{-redex} \\
(\lambda x{:}\sigma.\Delta\, x)^* & \stackrel{def}{=} & \Delta^* & \text{if } x \notin \mathrm{fv}(\Delta) \\
(pr_i\, \Delta)^* & \stackrel{def}{=} & pr_i\, \Delta^* & \text{if } \Delta \text{ is not a strong pair}
\end{array}
$$

Now we have to prove the Church-Rosser property for the parallel reduction.

▶ **Lemma 20** (Confluence property for $\Longrightarrow$). *If $\Delta \Longrightarrow \Delta'$, then $\Delta' \Longrightarrow \Delta^*$.*

The Church-Rosser property follows.

▶ **Theorem 21** (Confluence). *If $\Delta_1 \longrightarrow\!\!\!\!\!\rightarrow \Delta_2$ and $\Delta_1 \longrightarrow\!\!\!\!\!\rightarrow \Delta_3$, then there exists Δ_4 such that $\Delta_2 \longrightarrow\!\!\!\!\!\rightarrow \Delta_4$ and $\Delta_3 \longrightarrow\!\!\!\!\!\rightarrow \Delta_4$.*

The next lemma says that all type derivations for Δ have an unique type.

▶ **Lemma 22** (Unicity of typing). *If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, then σ is unique.*

The next theorem states that all the $\Delta^{\mathcal{T}}_{\equiv}$ typed systems preserve synchronous βpr_i-reduction, and all the $\Delta^{\mathcal{T}}_{=_\beta}$ and $\Delta^{\mathcal{T}}_{=_{\beta\eta}}$ typed systems preserve βpr_i-reduction.

▶ **Theorem 23** (Subject reduction for βpr_i).

1. *If $B \vdash^{\mathcal{T}}_{\equiv} \Delta_1 : \sigma$ and $\Delta_1 \longrightarrow^{\parallel} \Delta_2$, then $B \vdash^{\mathcal{T}}_{\equiv} \Delta_2 : \sigma$;*
2. *for $\mathcal{R} \in \{=_\beta, =_{\beta\eta}\}$, if $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \sigma$ and $\Delta_1 \longrightarrow \Delta_2$, then $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \sigma$.*

The next theorem states that some of the typed systems on the back of the Δ-chair preserve η-reduction.

▶ **Theorem 24** (Subject reduction for η for $\mathcal{T}_{\mathrm{CDV}}, \mathcal{T}_{\mathrm{BCD}}$). *Let $\mathcal{T} \in \{\mathcal{T}_{\mathrm{CDS}}, \mathcal{T}_{\mathrm{BCD}}\}$. If $B \vdash^{\mathcal{T}}_{=_{\beta\eta}} \Delta_1 : \sigma$ and $\Delta_1 \longrightarrow_\eta \Delta_2$, then $B \vdash^{\mathcal{T}}_{=_{\beta\eta}} \Delta_2 : \sigma$.*

4.2 Strong normalization

The idea of the strong normalization proof is to embed typable terms of the Δ-calculus into Church-style terms of a target system, which is the simply-typed λ-calculus with pairs, in a structure-preserving way (and forgetting all the essence side-conditions). The translation is sufficiently faithful so as to preserve the number of reductions, and so strong normalization for the Δ-calculus follows from strong normalization for simply-typed λ-calculus with pairs. A similar technique has been used in [22] to prove the strong normalization property of LF and in [8] to prove the strong normalization property of a subset of $\lambda^{\mathrm{CD}}_{\cap}$.
The target system has one atomic type called $\circ$, a special constant term $u_\circ$ of type $\circ$ and an infinite number of constants c_σ of type σ for any type of the target system. We denote by $B \vdash_\times M : \sigma$ a typing judgment in the target system.

▶ **Definition 25** (Forgetful mapping).
- On intersection types.

$$|\mathsf{a}_i| \stackrel{def}{=} \circ \quad \forall \mathsf{a}_i \in \mathbb{A} \qquad \text{and} \qquad |\sigma \cap \tau| \stackrel{def}{=} |\sigma| \times |\tau| \qquad \text{and} \qquad |\sigma \to \tau| \stackrel{def}{=} |\sigma| \to |\tau|$$

$\Delta^{\mathcal{T}}_{\mathcal{R}}$	$\Delta^{\mathcal{T}}_{\mathcal{R}} \lhd \lambda^{\mathcal{T}}_{\cap}$	$\Delta^{\mathcal{T}}_{\mathcal{R}} \rhd \lambda^{\mathcal{T}}_{\cap}$
$\Delta^{\mathrm{CD}}_{\equiv}$	$\surd$	$\surd$
$\Delta^{\mathrm{CDV}}_{\equiv}$	$\surd$	$\surd$
$\Delta^{\mathrm{CDS}}_{\equiv}$	$\surd$	$\surd$
$\Delta^{\mathrm{BCD}}_{\equiv}$	$\surd$	$\surd$
$\Delta^{\mathrm{CD}}_{=_\beta}$	$\times$	$\surd$
$\Delta^{\mathrm{CDV}}_{=_\beta}$	$\times$	$\surd$
$\Delta^{\mathrm{CDS}}_{=_\beta}$	$\surd$	$\surd$
$\Delta^{\mathrm{BCD}}_{=_\beta}$	$\surd$	$\surd$
$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$	$\times$	$\surd$
$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$	$\times$	$\surd$

$\Delta^{\mathcal{T}}_{\mathcal{R}}$	TC/TR
$\Delta^{\mathrm{CD}}_{\equiv}$	$\surd$
$\Delta^{\mathrm{CDV}}_{\equiv}$	$\surd$
$\Delta^{\mathrm{CDS}}_{\equiv}$	$\surd$
$\Delta^{\mathrm{BCD}}_{\equiv}$	$\surd$
$\Delta^{\mathrm{CD}}_{=_\beta}$	$\surd$
$\Delta^{\mathrm{CDV}}_{=_\beta}$	$\surd$
$\Delta^{\mathrm{CDS}}_{=_\beta}$	$\times$
$\Delta^{\mathrm{BCD}}_{=_\beta}$	$\times$
$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$	$\surd$
$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$	$\times$

Source	Target
$\Delta^{\mathrm{CD}}_{\equiv}$	$\Delta^{\mathrm{CD}}_{=_\beta}$
$\Delta^{\mathrm{CDV}}_{\equiv}$	$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CDS}}_{\equiv}$	$\Delta^{\mathrm{CDS}}_{=_\beta}$
$\Delta^{\mathrm{BCD}}_{\equiv}$	$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CD}}_{=_\beta}$	$\Delta^{\mathrm{CD}}_{=_\beta}$
$\Delta^{\mathrm{CDV}}_{=_\beta}$	$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CDS}}_{=_\beta}$	$\Delta^{\mathrm{CDS}}_{=_\beta}$
$\Delta^{\mathrm{BCD}}_{=_\beta}$	$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$	$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$	$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$

Figure 5 On the left: Soundness, completeness, isomorphism. On the center: type checking/reconstruction. On the right: source and target languages of the translation

- On Δ-terms.

$$
\begin{array}{rclrcl}
|x|_B & \stackrel{def}{=} & x & |u_\Delta|_B & \stackrel{def}{=} & u_\circ \\
|\lambda x{:}\sigma.\Delta|_B & \stackrel{def}{=} & \lambda x.|\Delta|_{B,x:\sigma} & |\Delta_1\,\Delta_2|_B & \stackrel{def}{=} & |\Delta_1|_B\,|\Delta_2|_B \\
|\langle\Delta_1\,,\Delta_2\rangle|_B & \stackrel{def}{=} & (|\Delta_1|_B\,,|\Delta_2|_B) & |pr_i\,\Delta|_B & \stackrel{def}{=} & pr_i\,|\Delta|_B \\
|\Delta^\tau|_B & \stackrel{def}{=} & c_{|\sigma|\to|\tau|}\,|\Delta|_B \quad \text{if } B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma & & &
\end{array}
$$

- The map can be easily extended to basis B.

▶ **Theorem 26** (Strong normalization). *If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, then Δ is strongly normalizing.*

5 Typed systems *à la* Church *vs.* type assignment systems *à la* Curry

5.1 Relation between type assignment systems $\lambda^{\mathcal{T}}_{\cap}$ and typed systems $\Delta^{\mathcal{T}}_{\mathcal{R}}$

It is interesting to state some relations between type assignment systems *à la* Church and typed systems *à la* Curry. An interesting property is the one of isomorphism, namely the fact that whenever we assign a type σ to a pure λ-term M, the same type can be assigned to a Δ-term such that the essence of Δ is M. Conversely, for every assignment of σ to a Δ-term, a valid type assignment judgment of the same type for the essence of Δ can be derived. Soundness, completeness and isomorphism between intersection typed systems for the Δ-calculus and the corresponding intersection type assignment systems for the λ-calculus are defined as follows.

▶ **Definition 27** (Soundness, completeness and isomorphism). Let $\Delta^{\mathcal{T}}_{\mathcal{R}}$ and $\lambda^{\mathcal{T}}_{\cap}$.

1. (Soundness, $\Delta^{\mathcal{T}}_{\mathcal{R}} \lhd \lambda^{\mathcal{T}}_{\cap}$). $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$ implies $B \vdash^{\mathcal{T}}_{\cap} \wr\Delta\wr : \sigma$;
2. (Completeness, $\Delta^{\mathcal{T}}_{\mathcal{R}} \rhd \lambda^{\mathcal{T}}_{\cap}$). $B \vdash^{\mathcal{T}}_{\cap} M : \sigma$ implies there exists Δ such that $M \equiv \wr\Delta\wr$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$;
3. (Isomorphism, $\Delta^{\mathcal{T}}_{\mathcal{R}} \sim \lambda^{\mathcal{T}}_{\cap}$). $\Delta^{\mathcal{T}}_{\mathcal{R}} \rhd \lambda^{\mathcal{T}}_{\cap}$ and $\Delta^{\mathcal{T}}_{\mathcal{R}} \lhd \lambda^{\mathcal{T}}_{\cap}$.

The following properties and relations between typed and type assignment systems can be verified.

▶ **Theorem 28** (Soundness, completeness and isomorphism). *The following properties (left of Figure 5) between Δ-calculi and type assignment systems $\lambda_\cap^{\mathcal{T}}$ can be verified.*

The last theorem characterizes the class of strongly normalizing Δ-terms.

▶ **Theorem 29** (Characterization). *Every strongly normalizing λ-term can be type-annotated so as to be the essence of a typable Δ-term.*

We can finally state decidability of type checking (TC) and type reconstruction (TR).

▶ **Theorem 30** (Decidability of type checking and type reconstruction). *Figure 5 (in the center) list decidability of type checking and type reconstruction.*

5.2 Subtyping and explicit coercions

The typing rule $(\leqslant_{\mathcal{T}})$ in the general typed system introduces type coercions: once a type coercion is introduced, it cannot be eliminated, so *de facto* freezing a Δ-term inside an explicit coercion. Tannen *et al.* [42] showed a translation of a judgment derivation from a "Source" system with subtyping (Cardelli's Fun [10]) into an "equivalent" judgment derivation in a "Target" system without subtyping (Girard system F with records and recursion). In the same spirit, we present a translation that removes all explicit coercions. Intuitively, the translation proceeds as follows: every derivation ending with rule $(\leqslant_{\mathcal{T}})$ is translated into the following (coercion-free) derivation, *i.e.*

$$\frac{B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\sigma \leqslant_{\mathcal{T}} \tau\| : \sigma \to \tau \quad B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\Delta\|_B : \sigma}{B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\sigma \leqslant_{\mathcal{T}} \tau\| \; \|\Delta\|_B : \tau} \; (\to E)$$

where $\mathcal{R}'$ is a suitable relation such that $\mathcal{R} \sqsubseteq \mathcal{R}'$. Note that changing of the type theory is necessary to guarantee well-typedness in the translation of strong pairs. Summarizing, we provide a type preserving translation of a Δ-term into a coercion-free Δ-term such that $\wr\, \Delta \,\wr =_{\beta\eta} \wr\, \Delta' \,\wr$. The following example illustrates some trivial compilations of axioms and rule schemes of Figure 1.

▶ **Example 31** (Translation of axioms and rule schemes of Figure 1).

(refl) the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma} \rangle : \sigma \cap \sigma$ is translated to a coercion-free judgment $x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma.y)\, x \rangle : \sigma \cap \sigma$

(incl) the judgment $x{:}\sigma \cap \tau \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\tau} \rangle : (\sigma \cap \tau) \cap \tau$ is translated to a coercion-free judgment $x{:}\sigma \cap \tau \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma \cap \tau.pr_2\, y)\, x \rangle : (\sigma \cap \tau) \cap \tau$

(glb) the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma \cap \sigma} \rangle : \sigma \cap (\sigma \cap \sigma)$ is translated to a coercion-free judgment $x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma.\langle y \,, y \rangle)\, x \rangle : \sigma \cap (\sigma \cap \sigma)$

$(\mathtt{U}_{top})$ the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\mathtt{U}} \rangle : \sigma \cap \mathtt{U}$ is translated to a coercion-free judgment $x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma.u_y)\, x \rangle : \sigma \cap \mathtt{U}$

$(\mathtt{U}_\to)$ the judgment $x{:}\mathtt{U} \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma \to \mathtt{U}} \rangle : \mathtt{U} \cap (\sigma \to \mathtt{U})$ is translated to a coercion-free judgment $x{:}\mathtt{U} \vdash^{\mathcal{T}}_{=_{\beta\eta}} \langle x \,, (\lambda f{:}\mathtt{U}.\lambda y{:}\sigma.u_{(f\, y)})\, x \rangle : \mathtt{U} \cap (\sigma \to \mathtt{U})$

$(\to\cap)$ the judgment $x{:}(\sigma \to \tau) \cap (\sigma \to \rho) \vdash^{\mathcal{T}}_{\mathcal{R}} x^{\sigma \to \tau \cap \rho} : \sigma \to \tau \cap \rho$ is translated to a coercion-free judgment $x{:}(\sigma \to \tau) \cap (\sigma \to \rho) \vdash^{\mathcal{T}}_{=_{\beta\eta}} (\lambda f{:}(\sigma \to \tau) \cap (\sigma \to \rho).\lambda y{:}\sigma.\langle (pr_1\, f)\, y \,, (pr_2\, f)\, y \rangle)\, x : \sigma \to \tau \cap \rho$

$(\to)$ the judgment $x{:}\sigma \to \tau \cap \rho \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma \cap \rho \to \tau} \rangle : (\sigma \to \tau \cap \rho) \cap (\sigma \cap \rho \to \tau)$ is translated to a coercion-free judgment $x{:}\sigma \to \tau \cap \rho \vdash^{\mathcal{T}}_{=_{\beta\eta}} \langle x, (\lambda f{:}\sigma \to \tau \cap \rho.\lambda y{:}\sigma \cap \rho.pr_1\, (f\, (pr_1\, y)))\, x \rangle : (\sigma \to \tau \cap \rho) \cap (\sigma \cap \rho \to \tau)$

(trans) the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, (x^{\mathtt{U}})^{\sigma \to \mathtt{U}} \rangle : \sigma \cap (\sigma \to \mathtt{U})$ is translated to a coercion-free judgment $x{:}\sigma \vdash^{\mathcal{T}}_{=_{\beta\eta}} \langle x \,, (\lambda f{:}\mathtt{U}.\lambda y{:}\sigma.u_{(f\, y)})\, ((\lambda y{:}\sigma.u_y)\, x) \rangle : \sigma \cap (\sigma \to \mathtt{U})$

The next definition introduces two maps translating subtype judgments into explicit coercions functions and Δ-terms into coercion-free Δ-terms.

▶ **Definition 32** (Translations $\|-\|$ and $\|-\|_B$).

1. The minimal type theory $\leqslant_{\min}$ and the extra axioms and schemes are translated as follows.

$$
\begin{array}{lrcl}
(\text{refl}) & \|\sigma \leqslant_{\mathcal{T}} \sigma\| & \stackrel{def}{=} & \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.x : \sigma \to \sigma \\
(\text{incl}_1) & \|\sigma \cap \tau \leqslant_{\mathcal{T}} \sigma\| & \stackrel{def}{=} & \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma \cap \tau.pr_1\, x : \sigma \cap \tau \to \sigma \\
(\text{incl}_2) & \|\sigma \cap \tau \leqslant_{\mathcal{T}} \tau\| & \stackrel{def}{=} & \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma \cap \tau.pr_2\, x : \sigma \cap \tau \to \tau \\
(\text{glb}) & \left\| \dfrac{\rho \leqslant_{\mathcal{T}} \sigma \quad \rho \leqslant_{\mathcal{T}} \tau}{\rho \leqslant_{\mathcal{T}} \sigma \cap \tau} \right\| & \stackrel{def}{=} & \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\rho.\langle \|\rho \leqslant_{\mathcal{T}} \sigma\|\, x \,, \|\rho \leqslant_{\mathcal{T}} \tau\|\, x\rangle : \rho \to \sigma \cap \tau \\
(\text{trans}) & \left\| \dfrac{\sigma \leqslant_{\mathcal{T}} \tau \quad \tau \leqslant_{\mathcal{T}} \rho}{\sigma \leqslant_{\mathcal{T}} \rho} \right\| & \stackrel{def}{=} & \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.\, \|\tau \leqslant_{\mathcal{T}} \rho\|\ (\|\sigma \leqslant_{\mathcal{T}} \tau\|\ x) : \sigma \to \rho \\
(\mathtt{U}_{top}) & \|\sigma \leqslant_{\mathcal{T}} \mathtt{U}\| & \stackrel{def}{=} & \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.u_x : \sigma \to \mathtt{U} \\
(\mathtt{U}_{\to}) & \|\mathtt{U} \leqslant_{\mathcal{T}} \sigma \to \mathtt{U}\| & \stackrel{def}{=} & \vdash^{\mathcal{T}}_{=_{\beta\eta}} \lambda f{:}\mathtt{U}.\lambda x{:}\sigma.u_{(f\, x)} : \mathtt{U} \to (\sigma \to \mathtt{U})
\end{array}
$$

Let $\xi_1 \stackrel{def}{=} (\sigma \to \tau) \cap (\sigma \to \rho)$ and $\xi_2 \stackrel{def}{=} \sigma \to \tau \cap \rho$

$$
(\to\cap) \qquad \|\xi_1 \leqslant_{\mathcal{T}} \xi_2\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_{\beta\eta}} \lambda f{:}\xi_1.\lambda x{:}\sigma.\langle (pr_1\, f)\, x \,, (pr_2\, f)\, x\rangle : \xi_1 \to \xi_2
$$

Let $\xi_1 \stackrel{def}{=} \sigma_1 \to \tau_1$ and $\xi_2 \stackrel{def}{=} \sigma_2 \to \tau_2$

$$
(\to) \qquad \left\| \dfrac{\sigma_2 \leqslant_{\mathcal{T}} \sigma_1 \quad \tau_1 \leqslant_{\mathcal{T}} \tau_2}{\sigma_1 \to \tau_1 \leqslant_{\mathcal{T}} \sigma_2 \to \tau_2} \right\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_{\beta\eta}} \lambda f{:}\xi_1.\lambda x{:}\sigma_2.\, \|\tau_1 \leqslant_{\mathcal{T}} \tau_2\|\ (f\ (\|\sigma_2 \leqslant_{\mathcal{T}} \sigma_1\|\ x)) : \xi_1 \to \xi_2
$$

2. The translation $\|-\|_B$ is defined on Δ as follows.

$$
\begin{array}{rclrcll}
\|u_\Delta\|_B & \stackrel{def}{=} & u_{\|\Delta\|_B} & \|x\|_B & \stackrel{def}{=} & x & \\
\|\lambda x{:}\sigma.\Delta\|_B & \stackrel{def}{=} & \lambda x{:}\sigma.\, \|\Delta\|_{B,x:\sigma} & \|\Delta_1\, \Delta_2\|_B & \stackrel{def}{=} & \|\Delta_1\|_B\ \|\Delta_2\|_B & \\
\|\langle \Delta_1 \,, \Delta_2\rangle\|_B & \stackrel{def}{=} & \langle \|\Delta_1\|_B \,, \|\Delta_2\|_B\rangle & \|pr_i\, \Delta\|_B & \stackrel{def}{=} & pr_i\ \|\Delta\|_B & i \in \{1,2\} \\
\|\Delta^\tau\|_B & \stackrel{def}{=} & \|\sigma \leqslant_{\mathcal{T}} \tau\|\ \|\Delta\|_B & \multicolumn{4}{l}{\text{if } B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma.}
\end{array}
$$

By looking at the above translation functions we can see that if $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, then $\|\Delta\|_B$ is defined and it is coercion-free. The following lemma states that a coercion function is always typable in $\Delta^{\mathcal{T}}_{=_{\beta\eta}}$, that it is essentially the identity and that, without using the rule schemes $(\to\cap)$, $(\mathtt{U}_\to)$, and $(\to)$ the translation can even be derivable in $\Delta^{\mathcal{T}}_{=_\beta}$.

▶ **Lemma 33** (Essence of a coercion is an identity). **1.** *If $\sigma \leqslant_{\mathcal{T}} \tau$, then $\vdash^{\mathcal{T}}_{=_{\beta\eta}} \|\sigma \leqslant_{\mathcal{T}} \tau\| : \sigma \to \tau$ and $\wr \|\sigma \leqslant_{\mathcal{T}} \tau\| \wr =_{\beta\eta} \lambda x.x$;*

2. *If $\sigma \leqslant_{\mathcal{T}} \tau$ without using the rule schemes $(\to\cap)$, $(\mathtt{U}_\to)$, and $(\to)$, then $\vdash^{\mathcal{T}}_{=_\beta} \|\sigma \leqslant_{\mathcal{T}} \tau\| : \sigma \to \tau$ and $\wr \|\sigma \leqslant_{\mathcal{T}} \tau\| \wr =_\beta \lambda x.x$.*

We can now prove the coherence of the translation as follows.

▶ **Theorem 34** (Coherence). *If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, then $B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\Delta\|_B : \sigma$ and $\wr \|\Delta\|_B \wr\ \mathcal{R}' \wr \Delta \wr$, where $\Delta^{\mathcal{T}}_{\mathcal{R}}$ and $\Delta^{\mathcal{T}}_{\mathcal{R}'}$ are respectively the source and target intersection typed systems given in Figure 5 (right part).*

References

1 Fabio Alessi, Franco Barbanera, and Mariangiola Dezani-Ciancaglini. Intersection types and lambda models. *Theor. Comput. Sci.*, 355(2):108–126, 2006.

2 Steffen Van Bakel. Cut-elimination in the strict intersection type assignment system is strongly normalizing. *Notre Dame Journal of Formal Logic*, 45(1):35–63, 2004.

3 Franco Barbanera and Simone Martini. Proof-functional connectives and realizability. *Archive for Mathematical Logic*, 33:189–211, 1994.

4 Henk Barendregt, Mario Coppo, and Mariangiola Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *Journal of Symbolic Logic*, 48(4):931–940, 1983.

5 Henk P. Barendregt, Wil Dekkers, and Richard Statman. *Lambda calculus with types.* Cambridge University Press, 2013.

6 Choukri-Bey Ben-Yelles. *Type assignment in the lambda-calculus: syntax and semantics.* PhD thesis, University College of Swansea, 1979.

7 Viviana Bono, Betti Venneri, and Lorenzo Bettini. A typed lambda calculus with intersection types. *Theor. Comput. Sci.*, 398(1-3):95–113, 2008.

8 Antonio Bucciarelli, Adolfo Piperno, and Ivano Salvo. Intersection types and λ-definability. *Mathematical Structures in Computer Science*, 13(1):15–53, 2003.

9 Beatrice Capitani, Michele Loreti, and Betti Venneri. Hyperformulae, Parallel Deductions and Intersection Types. *BOTH, Electr. Notes Theor. Comput. Sci.*, 50(2):180–198, 2001.

10 Luca Cardelli and Peter Wegner. On understanding types, data abstraction, and polymorphism. *ACM Computing Surveys*, 17(4):471–523, December 1985.

11 Mario Coppo and Mariangiola Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, 1980.

12 Mario Coppo, Mariangiola Dezani-Ciancaglini, and Patrick Sallé. Functional characterization of some semantic equalities inside λ-calculus. In *International Colloquium on Automata, Languages, and Programming*, pages 133–146. Springer-Verlag, 1979.

13 Mario Coppo, Mariangiola Dezani-Ciancaglini, and Betti Venneri. Functional characters of solvable terms. *Zietschrift für Mathematische Logik und Grundlagen der Mathematik*, 27(2-6):45–58, 1981.

14 Rowan Davies. *Practical Refinement-Type Checking.* PhD thesis, Carnegie Mellon University, 2005. CMU-CS-05-110.

15 Daniel J. Dougherty, Ugo de'Liguoro, Luigi Liquori, and Claude Stolze. A realizability interpretation for intersection and union types. In *APLAS*, volume 10017 of *Lecture Notes in Computer Science*, pages 187–205. Springer-Verlag, 2016.

16 Daniel J. Dougherty and Luigi Liquori. Logic and computation in a lambda calculus with intersection and union types. In *LPAR*, volume 6355 of *Lecture Notes in Computer Science*, pages 173–191. Springer-Verlag, 2010.

17 Andrej Dudenhefner, Moritz Martens, and Jakob Rehof. The algebraic intersection type unification problem. *Logical Methods in Computer Science*, 13(3), 2017.

18 Joshua Dunfield. Refined typechecking with stardust. In *Proceedings of PLPV*, pages 21–32, 2007.

19 Joshua Dunfield. Elaborating intersection and union types. *J. Funct. Program.*, 24(2-3):133–165, 2014.

20 Timothy S. Freeman and Frank Pfenning. Refinement types for ML. In *Proceedings of PLDI*, pages 268–277, 1991.

21 Alain Frisch, Giuseppe Castagna, and Véronique Benzaken. Semantic subtyping: Dealing set-theoretically with function, union, intersection, and negation types. *Journal of the ACM (JACM)*, 55(4):19, 2008.

22 Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *J. ACM*, 40(1):143–184, 1993. URL: `http://doi.acm.org/10.1145/138027.138060`, `doi:10.1145/138027.138060`.

23 J. Roger Hindley. The simple semantics for Coppo-Dezani-Sallé types. In *International Symposium on Programming*, pages 212–226, 1982.

24 J. Roger Hindley. Coppo-Dezani types do not correspond to propositional logic. *Theor. Comput. Sci.*, 28:235–236, 1984.

25 Furio Honsell, Luigi Liquori, Claude Stolze, and Ivan Scagnetto. The delta-framework. In *Proc of FSTTCS*, pages 37:1–37:21, 2018.

26 Assaf J. Kfoury and Joe B. Wells. Principality and type inference for intersection types using expansion variables. *Theor. Comput. Sci.*, 311(1-3):1–70, 2004.

27 Luigi Liquori and Simona Ronchi Della Rocca. Towards an intersection typed system *à la* Church. *Electronic Notes in Theoretical Computer Science*, 136:43–56, 2005.

28 Luigi Liquori and Simona Ronchi Della Rocca. Intersection typed system *à la* Church. *Information and Computation*, 9(205):1371–1386, 2007.

29 Luigi Liquori and Claude Stolze. A decidable subtyping logic for intersection and union types. In *Proc of TTCS*, volume 10608 of *Lecture Notes in Computer Science*, pages 74–90. Springer-Verlag, 2017.

30 Edgar G. K. Lopez-Escobar. Proof functional connectives. In *Methods in Mathematical Logic*, volume 1130 of *Lecture Notes in Mathematics*, pages 208–221. Springer-Verlag, 1985.

31 Grigori Mints. The completeness of provable realizability. *Notre Dame Journal of Formal Logic*, 30(3):420–441, 1989.

32 Alexandre Miquel. The implicit calculus of constructions. In *TLCA*, pages 344–359, 2001.

33 Benjamin C. Pierce. *Programming with intersection types, union types, and bounded polymorphism.* PhD thesis, Technical Report CMU-CS-91-205. Carnegie Mellon University, 1991.

34 Benjamin C. Pierce. Programming with intersection types, union types, and polymorphism. Technical Report CMU-CS-91-106, Carnegie Mellon University, 1991.

35 Elaine Pimentel, Simona Ronchi Della Rocca, and Luca Roversi. Intersection types from a proof-theoretic perspective. *Fundam. Inform.*, 121(1-4):253–274, 2012.

36 Garrel Pottinger. A type assignment for the strongly normalizable λ-terms. In *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 561–577. Academic Press, 1980.

37 Jakob Rehof and Pawel Urzyczyn. The complexity of inhabitation with explicit intersection. In *Logic and Program Semantics - Essays Dedicated to Dexter Kozen on the Occasion of His 60th Birthday*, pages 256–270, 2012.

38 John C. Reynolds. Preliminary design of the programming language Forsythe. Report CMU–CS–88–159, Carnegie Mellon University, Pittsburgh, Pennsylvania, June 21, 1988.

39 Simona Ronchi Della Rocca and Luca Roversi. Intersection logic. In *CSL*, volume 2142 of *Lecture Notes in Computer Science*, pages 421–428. Springer-Verlag, 2001.

40 Claude Stolze, Luigi Liquori, Furio Honsell, and Ivan Scagnetto. Towards a Logical Framework with Intersection and Union Types. In *11th International Workshop on Logical Frameworks and Meta-languages, LFMTP*, pages 1 – 9, 2017.

41 Masako Takahashi. Parallel reductions in λ-calculus. *Information and computation*, 118(1):120–127, 1995.

42 Val Tannen, Thierry Coquand, Carl A. Gunter, and Andre Scedrov. Inheritance as implicit coercion. *Inf. Comput.*, 93(1):172–221, 1991.

43 Pawel Urzyczyn. The emptiness problem for intersection types. *J. Symb. Log.*, 64(3):1195–1215, 1999.

44 Betti Venneri. Intersection types as logical formulae. *J. Log. Comput.*, 4(2):109–124, 1994.

45 Joe B. Wells, Allyn Dimock, Robert Muller, and Franklyn Turbak. A calculus with polymorphic and polyvariant flow types. *J. Funct. Program.*, 12(3):183–227, 2002.

46 Joe B. Wells and Christian Haack. Branching types. In *ESOP*, volume 2305 of *Lecture Notes in Computer Science*, pages 115–132. Springer-Verlag, 2002.

A Metatheory of $\Delta_{\mathcal{R}}^{\mathcal{T}}$

A.1 General properties

Unless specified, all properties applies to the intersection typed systems $\Delta_{\mathcal{R}}^{\mathcal{T}}$.
The Church-Rosser property is proved using the technique of Takahashi [41]. The parallel reduction semantics extends Definition 4 and it is inductively defined as follows.

▶ **Definition 35** (Parallel reduction semantics).

$$
\begin{array}{rcllll}
x & \Longrightarrow & x & & \\
u_\Delta & \Longrightarrow & u_\Delta & & \\
\Delta^\sigma & \Longrightarrow & (\Delta')^\sigma & \text{if } \Delta \Longrightarrow \Delta' & \\
\Delta_1\,\Delta_2 & \Longrightarrow & \Delta'_1\,\Delta'_2 & \text{if } \Delta_1 \Longrightarrow \Delta'_1 & \text{and } \Delta_2 \Longrightarrow \Delta'_2 \\
\lambda x{:}\sigma.\Delta & \Longrightarrow & \lambda x{:}\sigma.\Delta' & \text{if } \Delta \Longrightarrow \Delta' & \\
(\lambda x{:}\sigma.\Delta_1)\,\Delta_2 & \Longrightarrow & \Delta'_1[\Delta'_2/x] & \text{if } \Delta_1 \Longrightarrow \Delta'_1 & \text{and } \Delta_2 \Longrightarrow \Delta'_2 \\
\langle \Delta_1 \,,\Delta_2 \rangle & \Longrightarrow & \langle \Delta'_1 \,,\Delta'_2 \rangle & \text{if } \Delta_1 \Longrightarrow \Delta'_1 & \text{and } \Delta_2 \Longrightarrow \Delta'_2 \\
pr_i\,\Delta & \Longrightarrow & pr_i\,\Delta' & \text{if } \Delta \Longrightarrow \Delta' & \text{and } i \in \{1,2\} \\
pr_i\,\langle \Delta_1 \,,\Delta_2 \rangle & \Longrightarrow & \Delta'_i & \text{if } \Delta_i \Longrightarrow \Delta'_i & \text{and } i \in \{1,2\}
\end{array}
$$

Intuitively, $\Delta \Longrightarrow \Delta'$ means that Δ' is obtained from Δ by simultaneous contraction of some βpr_i-redexes possibly overlapping each other. Church-Rosser can be achieved by proving a stronger statement, namely

$$\Delta \Longrightarrow \Delta' \quad \text{implies} \quad \Delta' \Longrightarrow \Delta^* \tag{1}$$

where Δ^* is a Δ-term determined by Δ and independent from Δ'. The statement (1) is satisfied by the term Δ^* which is obtained from Δ by contracting all the redexes existing in Δ simultaneously.

▶ **Definition 36** (The map $\_^*$).

$$
\begin{array}{rcll}
x^* & \stackrel{def}{=} & x & \\
u_\Delta^* & \stackrel{def}{=} & u_\Delta & \\
(\Delta^\sigma)^* & \stackrel{def}{=} & (\Delta^*)^\sigma & \\
\langle \Delta_1 \,, \Delta_2 \rangle^* & \stackrel{def}{=} & \langle \Delta_1^* \,, \Delta_2^* \rangle & \\
(\lambda x{:}\sigma.\Delta)^* & \stackrel{def}{=} & \lambda x{:}\sigma.\Delta^* & \\
(\Delta_1\, \Delta_2)^* & \stackrel{def}{=} & \Delta_1^*\, \Delta_2^* & \text{if } \Delta_1\, \Delta_2 \text{ is not a } \beta\text{-redex} \\
((\lambda x{:}\sigma.\Delta_1)\, \Delta_2)^* & \stackrel{def}{=} & \Delta_1^*[\Delta_2^*/x] & \\
(pr_i\, \Delta)^* & \stackrel{def}{=} & pr_i\, \Delta^* & \text{if } \Delta \text{ is not a strong pair} \\
(pr_i\, \langle \Delta_1 \,, \Delta_2 \rangle)^* & \stackrel{def}{=} & \Delta_i^* & i \in \{1, 2\}
\end{array}
$$

The next technical lemma will be useful in showing that Church-Rosser for $\longrightarrow\!\!\!\!\!\rightarrow$ can be inherited from Church-Rosser for $\Longrightarrow$.

▶ **Lemma 37.**
1. *If* $\Delta_1 \longrightarrow \Delta_1'$*, then* $\Delta_1 \Longrightarrow \Delta_1'$*;*
2. *if* $\Delta_1 \Longrightarrow \Delta_1'$*, then* $\Delta_1 \longrightarrow\!\!\!\!\!\rightarrow \Delta_1'$*;*
3. *if* $\Delta_1 \Longrightarrow \Delta_1'$ *and* $\Delta_2 \Longrightarrow \Delta_2'$*, then* $\Delta_1[\Delta_2/x] \Longrightarrow \Delta_1'[\Delta_2'/x]$*;*
4. $\Delta_1 \Longrightarrow \Delta_1^*$*.*

Proof. *(1)* can be proved by induction on the context of the redexes, while *(2)*, *(3)*, and *(4)* can be proved by induction on the structure of Δ_1. ◀

Now we have to prove the Church-Rosser property for the parallel reduction.

▶ **Lemma 38** (Confluence property for $\Longrightarrow$).
If $\Delta \Longrightarrow \Delta'$*, then* $\Delta' \Longrightarrow \Delta^*$*.*

Proof. By induction on the shape of Δ.
- if $\Delta \equiv x$, then $\Delta' \equiv x \Longrightarrow x \equiv \Delta^*$;
- if $\Delta \equiv u_\Delta$, then $\Delta' \equiv u_\Delta \Longrightarrow u_\Delta \equiv \Delta^*$;
- if $\Delta \equiv \Delta_1^\sigma$, then, for some Δ_1', we have that $\Delta_1 \Longrightarrow \Delta_1'$ and $\Delta' \equiv (\Delta_1')^\sigma$, therefore, by induction hypothesis, $\Delta' \Longrightarrow (\Delta_1^*)^\sigma \equiv \Delta^*$;
- if $\Delta \equiv \langle \Delta_1 \,, \Delta_2 \rangle$, then, for some Δ_1' and Δ_2', we have that $\Delta_1 \Longrightarrow \Delta_1'$, $\Delta_2 \Longrightarrow \Delta_2'$ and $\Delta' \equiv \langle \Delta_1' \,, \Delta_2' \rangle$. By induction hypothesis, $\Delta' \Longrightarrow \langle \Delta_1^* \,, \Delta_2^* \rangle \equiv \Delta^*$;
- if $\Delta \equiv \lambda x{:}\sigma.\Delta_1$, then, for some Δ_1', we have that $\Delta_1 \Longrightarrow \Delta_1'$ and $\Delta' \equiv \lambda x{:}\sigma.\Delta_1'$. By induction hypothesis, $\lambda x{:}\sigma.\Delta_1' \Longrightarrow \lambda x{:}\sigma.\Delta_1^* \equiv \Delta^*$;
- if $\Delta \equiv \Delta_1\, \Delta_2$ and Δ is not a β-redex, then, for some Δ_1' and Δ_2', we have that $\Delta_1 \Longrightarrow \Delta_1'$, $\Delta_2 \Longrightarrow \Delta_2'$ and $\Delta' \equiv \Delta_1'\, \Delta_2'$. By induction hypothesis, $\Delta' \Longrightarrow \Delta_1^*\, \Delta_2^* \equiv \Delta^*$;
- if $\Delta \equiv (\lambda x{:}\sigma.\Delta_1)\, \Delta_2$, then, for some Δ_1' and Δ_2', we have that $\Delta_1 \Longrightarrow \Delta_1'$, $\Delta_2 \Longrightarrow \Delta_2'$ and we have 2 subcases:
  - $\Delta' \equiv (\lambda x{:}\sigma.\Delta_1')\, \Delta_2'$: by induction hypothesis, $\Delta' \Longrightarrow \Delta_1^*[\Delta_2^*/x] \equiv \Delta^*$;
  - $\Delta' \equiv \Delta_1'[\Delta_2'/x]$: we also have $\Delta' \Longrightarrow \Delta_1^*[\Delta_2^*/x]$, thanks to point (3) of Lemma 37;
- if $\Delta \equiv pr_i\, \Delta_1$ and Δ_1 is not a strong pair, then, for some Δ_1', we have that $\Delta_1 \Longrightarrow \Delta_1'$ and $\Delta' \equiv pr_i\, \Delta_1'$, therefore, by induction hypothesis, $\Delta' \Longrightarrow pr_i\, \Delta_1^* \equiv \Delta^*$;

- if $\Delta \equiv pr_i \langle \Delta_1 \,, \Delta_2 \rangle$, then, for some Δ'_1 and Δ'_2, we have that $\Delta_1 \Longrightarrow \Delta'_1$, $\Delta_2 \Longrightarrow \Delta'_2$ and we have 2 subcases:
  - $\Delta' \equiv pr_i \langle \Delta'_1 \,, \Delta'_2 \rangle$: by induction hypothesis, $\Delta' \Longrightarrow \Delta^*_i \equiv \Delta^*$;
  - $\Delta' \equiv \Delta'_i$: we also have, by induction hypothesis, $\Delta' \Longrightarrow \Delta^*_i \equiv \Delta^*$.

◀

The Church-Rosser property follows.

▶ **Theorem 39** (Confluence).
If $\Delta_1 \longrightarrow\!\!\!\!\rightarrow \Delta_2$ and $\Delta_1 \longrightarrow\!\!\!\!\rightarrow \Delta_3$, then there exists Δ_4 such that $\Delta_2 \longrightarrow\!\!\!\!\rightarrow \Delta_4$ and $\Delta_3 \longrightarrow\!\!\!\!\rightarrow \Delta_4$.

Proof. Thanks to the first two points of Lemma 37, we know that $\longrightarrow\!\!\!\!\rightarrow$ is the transitive closure of $\Longrightarrow$, therefore we can deduce the confluence property of $\longrightarrow\!\!\!\!\rightarrow$ with the usual diagram chase, as suggested below.

$$\begin{tikzcd}
\Delta_{0,0} \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \Delta_{1,0} \arrow[r, Rightarrow] \arrow[d, Rightarrow, dashed] & \Delta_{2,0} \arrow[r, Rightarrow] \arrow[d, Rightarrow, dashed] & \Delta_{3,0} \arrow[d, Rightarrow, dashed] \\
\Delta_{0,1} \arrow[r, Rightarrow, dashed] \arrow[d, Rightarrow] & \Delta_{1,1} \arrow[r, Rightarrow, dashed] \arrow[d, Rightarrow, dashed] & \Delta_{2,1} \arrow[r, Rightarrow, dashed] \arrow[d, Rightarrow, dashed] & \Delta_{3,1} \arrow[d, Rightarrow, dashed] \\
\Delta_{0,2} \arrow[r, Rightarrow, dashed] & \Delta_{1,2} \arrow[r, Rightarrow, dashed] & \Delta_{2,2} \arrow[r, Rightarrow, dashed] & \Delta_{3,2}
\end{tikzcd}$$

◀

The next lemma says that all type derivations for Δ have an unique type.

▶ **Lemma 40** (Unicity of typing).
If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, then σ is unique.

Proof. By induction on the shape of Δ. ◀

The next lemma proves inversion properties on typable Δ-terms.

▶ **Lemma 41** (Generation).
1. *If $B \vdash^{\mathcal{T}}_{\mathcal{R}} x : \sigma$, then $x{:}\sigma \in B$;*
2. *if $B \vdash^{\mathcal{T}}_{\mathcal{R}} \lambda x{:}\sigma.\Delta : \rho$, then $\rho \equiv \sigma \to \tau$ for some τ and $B, x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \tau$;*
3. *if $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1\, \Delta_2 : \tau$, then there is σ such that $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \sigma \to \tau$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \sigma$;*
4. *if $B \vdash^{\mathcal{T}}_{\mathcal{R}} \langle \Delta_1 \,, \Delta_2 \rangle : \rho$, then there is σ, τ such that $\rho \equiv \sigma \cap \tau$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \sigma$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \tau$ and $\wr \Delta_1 \wr \mathcal{R} \wr \Delta_2 \wr$;*
5. *if $B \vdash^{\mathcal{T}}_{\mathcal{R}} pr_1\, \Delta : \sigma$, then there is τ such that $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma \cap \tau$;*
6. *if $B \vdash^{\mathcal{T}}_{\mathcal{R}} pr_2\, \Delta : \tau$, then there is σ such that $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma \cap \tau$;*
7. *if $B \vdash^{\mathcal{T}}_{\mathcal{R}} u_\Delta : \sigma$, then $\sigma \equiv \mathtt{U}$;*
8. *if $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta^\tau : \rho$, then $\rho \equiv \tau$ and there is σ such that $\sigma \leqslant_{\mathcal{T}} \tau$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$.*

Proof. The typing rules are uniquely syntax-directed, therefore we can immediately conclude. ◀

The next lemma says that all subterms of a typable Δ-term are typable too.

▶ **Lemma 42** (Subterms typability).
If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, and Δ' is a subterm of Δ, then there exists B' and τ such that $B' \supseteq B$ and $B' \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta' : \tau$.

Proof. By induction on the derivation of $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$. ◀

As expected, the weakening and strengthening properties on contexts are verified.

▶ **Lemma 43** (Free-variable properties).
1. *If* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, *and* $B' \supseteq B$, *then* $B' \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$*;*
2. *if* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, *then* $\mathrm{FV}(\Delta) \subseteq \mathit{Dom}(B)$*;*
3. *if* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, $B' \subseteq B$ *and* $\mathrm{FV}(\Delta) \subseteq \mathit{Dom}(B')$, *then* $B' \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$.

Proof. By induction on the derivation of $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$. ◀

The next lemma also says that essence is closed under substitution.

▶ **Lemma 44** (Substitution).
1. $\wr \Delta_1[\Delta_2/x] \wr \equiv \wr \Delta_1 \wr [\wr \Delta_2 \wr/x]$*;*
2. *If* $B, x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \tau$ *and* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \sigma$, *then* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1[\Delta_2/x] : \tau$.

Proof.
1. by induction on the shape of Δ_1;
2. by induction on the derivation. As an illustration, we show the case when the last applied rule is $(\cap I)$. Then we have that $B, x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle \Delta_1 , \Delta_1' \rangle : \tau \cap \tau'$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \sigma$; by induction hypothesis we have $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1[\Delta_2/x] : \tau$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1'[\Delta_2/x] : \tau'$. Moreover, thanks to point (1), we can show that $\wr \Delta_1[\Delta_2/x] \wr \, \mathcal{R} \, \wr \Delta_1'[\Delta_2/x] \wr$. As a consequence:

$$\frac{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1[\Delta_2/x] : \tau \quad B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1'[\Delta_2/x] : \tau' \quad \wr \Delta_1[\Delta_2/x] \wr \, \mathcal{R} \, \wr \Delta_1'[\Delta_2/x] \wr}{B \vdash^{\mathcal{T}}_{\mathcal{R}} \langle \Delta_1 , \Delta_1' \rangle [\Delta_2/x] : \tau \cap \tau'} \; (\cap I)$$

◀

In order to prove subject reduction, we need to prove that reducing Δ-terms preserve the side-condition $\wr \Delta_1 \wr \, \mathcal{R} \, \wr \Delta_2 \wr$ when typing the strong pair $\langle \Delta_1 , \Delta_2 \rangle$. We prove this in the following lemma.

▶ **Lemma 45** (Essence reduction).
1. *If* $B \vdash^{\mathcal{T}}_{\equiv} \Delta_1 : \sigma$ *and* $\Delta_1 \longrightarrow \Delta_2$, *then* $\wr \Delta_1 \wr =_\beta \wr \Delta_2 \wr$*;*
2. *for* $\mathcal{R} \in \{=_\beta, =_{\beta\eta}\}$, *if* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \sigma$ *and* $\Delta_1 \longrightarrow \Delta_2$, *then* $\wr \Delta_1 \wr \, \mathcal{R} \, \wr \Delta_2 \wr$*;*
3. *if* $B \vdash^{\mathcal{T}}_{=_{\beta\eta}} \Delta_1 : \sigma$ *and* $\Delta_1 \longrightarrow_\eta \Delta_2$, *then* $\wr \Delta_1 \wr =_\eta \wr \Delta_2 \wr$.

Proof. If Δ_1 is a redex, then we have three cases:
- if $\Delta_1 \equiv (\lambda x{:}\sigma.\Delta_1')\, \Delta_1''$ and Δ_2 is $\Delta_1'[\Delta_1''/x]$, then, thanks to Lemma 44(1) we have that $\wr \Delta_2 \wr \equiv \wr \Delta_1' \wr [\wr \Delta_1'' \wr/x]$, therefore $\wr \Delta_1 \wr =_\beta \wr \Delta_2 \wr$;
- if $\Delta_1 \equiv pr_i \, \langle \Delta_1' , \Delta_2' \rangle$ and Δ_2 is Δ_i', we know that Δ_1 is typable in $\Delta^{\mathcal{T}}_{\mathcal{R}}$, and thanks to Lemma 41(4), we have that $\wr \Delta_1' \wr \, \mathcal{R} \, \wr \Delta_2' \wr$. As a consequence, $\wr \Delta_1 \wr \, \mathcal{R} \, \wr \Delta_2 \wr$;
- if $\Delta_1 \equiv \lambda x{:}\sigma.\Delta' \, x$ with $x \notin \mathrm{FV}(\Delta')$, and Δ_2 is Δ', then $\wr \Delta_1 \wr =_\eta \wr \Delta_2 \wr$.

For the contextual closure, we have that $\Delta_1 \equiv \Delta[\Delta'/x]$, where $\Delta[\_]$ is a surrounding context and Δ' is a redex, and Δ_2 is $\Delta[\Delta''/x]$ where Δ'' is the contractum of Δ'. Then, by Lemma 42 we know that Δ' is typable and then we conclude by Lemma 44. ◀

The next theorem states that all the $\Delta^{\mathcal{T}}_{\equiv}$ typed systems preserve synchronous βpr_i-reduction, and all the $\Delta^{\mathcal{T}}_{=_\beta}$ and $\Delta^{\mathcal{T}}_{=_{\beta\eta}}$ typed systems preserve βpr_i-reduction.

▶ **Theorem 46** (Subject reduction for βpr_i).
1. *If* $B \vdash^{\mathcal{T}}_{\equiv} \Delta_1 : \sigma$ *and* $\Delta_1 \longrightarrow^{\parallel} \Delta_2$, *then* $B \vdash^{\mathcal{T}}_{\equiv} \Delta_2 : \sigma$*;*

2. *for* $\mathcal{R} \in \{=_\beta, =_{\beta\eta}\}$, *if* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_1 : \sigma$ *and* $\Delta_1 \longrightarrow \Delta_2$, *then* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta_2 : \sigma$.

Proof. If Δ_1 is a βpr_i-redex, then we proceed as usual using Lemmas 41 and 44. For the contextual closure, we proceed by induction on the derivation: we illustrate the most important case, namely $(\cap I)$ where we have to check that the essence condition is preserved. According to $\mathcal{R}$ we distinguish two cases:

1. (Case where $\mathcal{R}$ is $\equiv$). If $B \vdash^{\mathcal{T}}_{\equiv} \langle \Delta_1 \,, \Delta_2 \rangle : \sigma \cap \tau$ and $\langle \Delta_1 \,, \Delta_2 \rangle \longrightarrow^{\parallel} \langle \Delta'_1 \,, \Delta'_2 \rangle$, then $\wr \Delta'_1 \wr \equiv \wr \Delta'_2 \wr$ and, by induction hypothesis, $B \vdash^{\mathcal{T}}_{\equiv} \Delta'_1 : \sigma$ and $B \vdash^{\mathcal{T}}_{\equiv} \Delta'_2 : \tau$, therefore $B \vdash^{\mathcal{T}}_{\equiv} \langle \Delta'_1 \,, \Delta'_2 \rangle : \sigma \cap \tau$;
2. (Case where $\mathcal{R} \in \{=_\beta, =_{\beta\eta}\}$). If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \langle \Delta_1 \,, \Delta_2 \rangle : \sigma \cap \tau$ and $\langle \Delta_1 \,, \Delta_2 \rangle \longrightarrow \langle \Delta'_1 \,, \Delta'_2 \rangle$, then:
   - $\wr \Delta_1 \wr \, \mathcal{R} \, \wr \Delta_2 \wr$;
   - by Lemma 45 we have that $\wr \Delta'_1 \wr \, \mathcal{R} \, \wr \Delta_1 \wr$ and $\wr \Delta_2 \wr \, \mathcal{R} \, \wr \Delta'_2 \wr$;
   - by induction hypothesis we have that $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta'_1 : \sigma$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta'_2 : \tau$;

   therefore $\wr \Delta'_1 \wr \, \mathcal{R} \, \wr \Delta'_2 \wr$ and $B \vdash^{\mathcal{T}}_{\mathcal{R}} \langle \Delta'_1 \,, \Delta'_2 \rangle : \sigma \cap \tau$.

◀

The next theorem states that some of the typed systems on the back of the Δ-chair preserve η-reduction.

▶ **Theorem 47** (Subject reduction for η for $\mathcal{T}_{\mathrm{CDV}}, \mathcal{T}_{\mathrm{BCD}}$).
Let $\mathcal{T} \in \{\mathcal{T}_{\mathrm{CDS}}, \mathcal{T}_{\mathrm{BCD}}\}$. *If* $B \vdash^{\mathcal{T}}_{=_{\beta\eta}} \Delta_1 : \sigma$ *and* $\Delta_1 \longrightarrow_\eta \Delta_2$, *then* $B \vdash^{\mathcal{T}}_{=_{\beta\eta}} \Delta_2 : \sigma$.

Proof. If Δ_1 is a η-redex, then we proceed as usual using Lemmas 41 and 43. For the contextual closure the proof proceeds exactly as in Theorem 46. ◀

▶ Remark (About subject expansion).
We know that some of the intersection type assignment systems *à la* Curry (*viz.* $\lambda^{\mathrm{BCD}}_\cap$ and $\lambda^{\mathrm{CDS}}_\cap$) satisfy the subject β-expansion property: one may ask whether this property can also be meaningful in typed systems *à la* Church. It is not surprising to see that the answer is negative because type-decorations of bound variables are hard-coded in the λ-abstraction and cannot be forgotten. As a trivial example of the failure of the subject-expansion in all the typed systems, consider the following reduction:

$$(\lambda x{:}\sigma.x)\,(\lambda x{:}\sigma.x) \longrightarrow (\lambda x{:}\sigma.x)$$

Obviously we can type $\vdash^{\mathcal{T}}_{\mathcal{R}} (\lambda x{:}\sigma.x) : \sigma \to \sigma$ but $\nvdash^{\mathcal{T}}_{\mathcal{R}} (\lambda x{:}\sigma.x)\,(\lambda x{:}\sigma.x) : \sigma \to \sigma$.

A.2 Strong normalization

The idea of the strong normalization proof is to embed typable terms of the Δ-calculus into Church-style terms of a target system, which is the simply-typed λ-calculus with pairs, in a structure-preserving way (and forgetting all the essence side-conditions). The translation is sufficiently faithful so as to preserve the number of reductions, and so strong normalization for the Δ-calculus follows from strong normalization for simply-typed λ-calculus with pairs. A similar technique has been used in [22] to prove the strong normalization property of LF and in [8] to prove the strong normalization property of a subset of $\lambda^{\mathrm{CD}}_\cap$.
The target system has one atomic type called $\circ$, a special constant term $u_\circ$ of type $\circ$ and an infinite number of constants c_σ of type σ for any type of the target system. We denote by $B \vdash_\times M : \sigma$ a typing judgment in the target system.

▶ **Definition 48** (Forgetful mapping).

- On intersection types.

$$
\begin{array}{rcl}
|\mathsf{a}_i| & \stackrel{def}{=} & \circ \quad \forall \mathsf{a}_i \in \mathbb{A} \\
|\sigma \cap \tau| & \stackrel{def}{=} & |\sigma| \times |\tau| \\
|\sigma \to \tau| & \stackrel{def}{=} & |\sigma| \to |\tau|
\end{array}
$$

  The map can be easily extended to basis B.

- On Δ-terms.

$$
\begin{array}{rcl}
|x|_B & \stackrel{def}{=} & x \\
|u_\Delta|_B & \stackrel{def}{=} & u_\circ \\
|\lambda x{:}\sigma.\Delta|_B & \stackrel{def}{=} & \lambda x.|\Delta|_{B,x:\sigma} \\
|\Delta_1\, \Delta_2|_B & \stackrel{def}{=} & |\Delta_1|_B\, |\Delta_2|_B \\
|\langle \Delta_1 \,, \Delta_2 \rangle|_B & \stackrel{def}{=} & (|\Delta_1|_B \,, |\Delta_2|_B) \\
|pr_i\, \Delta|_B & \stackrel{def}{=} & pr_i\, |\Delta|_B \\
|\Delta^\tau|_B & \stackrel{def}{=} & c_{|\sigma| \to |\tau|}\, |\Delta|_B \quad \text{if } B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma
\end{array}
$$

The following technical lemma states some properties of the forgetful function.

▶ **Lemma 49.**

1. *If* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$*, then* $|\Delta|_B$ *is defined, and, for all* $B' \supseteq B$*,* $|\Delta|_B \equiv |\Delta|_{B'}$*;*
2. $|\Delta_1[\Delta_2/x]|_B \equiv |\Delta_1|_B[|\Delta_2|_B/x]$*;*
3. *If* $\Delta_1 \longrightarrow \Delta_2$*, then* $|\Delta_1|_B \longrightarrow |\Delta_2|_B$*;*
4. *If* $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$ *then* $|B| \vdash_\times |\Delta|_B : |\sigma|$.

Proof.

1. by induction on the derivation;
2. by induction on Δ_1. The only interesting part is $\Delta_1 \equiv \lambda y{:}\sigma.\Delta_1'$: by induction hypothesis, we have that $|\Delta_1'[\Delta_2/x]|_{B,x:\sigma} \equiv |\Delta_1'|_{B,x:\sigma}[|\Delta_2|_{B,x:\sigma}/x]$. Therefore, we see that $|(\lambda y{:}\sigma.\Delta_1')[\Delta_2/x]|_B \equiv \lambda y{:}\sigma.|\Delta_1'[\Delta_2/x]|_{B,x:\sigma} \equiv \lambda y{:}\sigma.|\Delta_1'|_{B,x:\sigma}[|\Delta_2|_{B,x:\sigma}/x]$, but, from point (1), we know that $|\Delta_2|_{B,x:\sigma} \equiv |\Delta_2|_B$, and we conclude;
3. by induction on the context of the redex;
4. by induction on the derivation.

◀

Strong normalization follows easily from the above lemmas.

▶ **Theorem 50** (Strong normalization).
If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$*, then* Δ *is strongly normalizing.*

Proof. Using Lemma 49 and the strong normalization of the simply typed λ-calculus with cartesian pairs. ◀

B Typed systems *à la* Church *vs.* type assignment systems *à la* Curry

B.1 Relation between type assignment systems $\lambda_\cap^{\mathcal{T}}$ and typed systems $\Delta_{\mathcal{R}}^{\mathcal{T}}$

It is interesting to state some relations between type assignment systems *à la* Church and typed systems *à la* Curry. An interesting property is the one of isomorphism, namely the fact that whenever we assign a type σ to a pure λ-term M, the same type can be assigned to a Δ-term such that the essence of Δ is M. Conversely, for every assignment of σ to a Δ-term, a valid type assignment judgment of the same type for the essence of Δ can be derived.
Soundness, completeness and isomorphism between intersection typed systems for the Δ-calculus and the corresponding intersection type assignment systems for the λ-calculus are defined as follows.

▶ **Definition 51** (Soundness, completeness and isomorphism).
Let $\Delta_{\mathcal{R}}^{\mathcal{T}}$ and $\lambda_\cap^{\mathcal{T}}$.

1. (Soundness, $\Delta_{\mathcal{R}}^{\mathcal{T}} \lhd \lambda_\cap^{\mathcal{T}}$). $B \vdash_{\mathcal{R}}^{\mathcal{T}} \Delta : \sigma$ implies $B \vdash_\cap^{\mathcal{T}} \wr \Delta \wr : \sigma$;
2. (Completeness, $\Delta_{\mathcal{R}}^{\mathcal{T}} \rhd \lambda_\cap^{\mathcal{T}}$). $B \vdash_\cap^{\mathcal{T}} M : \sigma$ implies there exists Δ such that $M \equiv \wr \Delta \wr$ and $B \vdash_{\mathcal{R}}^{\mathcal{T}} \Delta : \sigma$;
3. (Isomorphism, $\Delta_{\mathcal{R}}^{\mathcal{T}} \sim \lambda_\cap^{\mathcal{T}}$). $\Delta_{\mathcal{R}}^{\mathcal{T}} \rhd \lambda_\cap^{\mathcal{T}}$ and $\Delta_{\mathcal{R}}^{\mathcal{T}} \lhd \lambda_\cap^{\mathcal{T}}$.

The following properties and relations between typed and type assignment systems can be verified.

▶ **Theorem 52** (Soundness, completeness and isomorphism).
The following properties between Δ-calculi and type assignment systems $\lambda_\cap^{\mathcal{T}}$ are verified.

$\Delta_{\mathcal{R}}^{\mathcal{T}}$	$\Delta_{\mathcal{R}}^{\mathcal{T}} \lhd \lambda_\cap^{\mathcal{T}}$	$\Delta_{\mathcal{R}}^{\mathcal{T}} \rhd \lambda_\cap^{\mathcal{T}}$
$\Delta_{\equiv}^{\mathrm{CD}}$	$\surd$	$\surd$
$\Delta_{\equiv}^{\mathrm{CDV}}$	$\surd$	$\surd$
$\Delta_{\equiv}^{\mathrm{CDS}}$	$\surd$	$\surd$
$\Delta_{\equiv}^{\mathrm{BCD}}$	$\surd$	$\surd$
$\Delta_{=_\beta}^{\mathrm{CD}}$	$\times$	$\surd$
$\Delta_{=_\beta}^{\mathrm{CDV}}$	$\times$	$\surd$
$\Delta_{=_\beta}^{\mathrm{CDS}}$	$\surd$	$\surd$
$\Delta_{=_\beta}^{\mathrm{BCD}}$	$\surd$	$\surd$
$\Delta_{=_{\beta\eta}}^{\mathrm{CDV}}$	$\times$	$\surd$
$\Delta_{=_{\beta\eta}}^{\mathrm{BCD}}$	$\times$	$\surd$

Proof.

($\lhd$) Soundness for $\Delta_{\equiv}^{\mathcal{T}}$. Let Δ be such that $B \vdash_{\equiv}^{\mathcal{T}} \Delta : \sigma$. We proceed by induction on the derivation. All cases proceed straightforwardly since all rules of the type and subtype system $\vdash_{\equiv}^{\mathcal{T}}$ correspond exactly to the rules of the same name in the corresponding type assignment system $\vdash_\cap^{\mathcal{T}}$ and in the same type theory $\mathcal{T}$. Therefore $M \equiv \wr \Delta \wr$ can be easily be defined and derived with type σ.

Soundness for $\Delta_{=_\beta}^{\{\mathrm{CDS},\mathrm{BCD}\}}$. Let $\mathcal{T} \in \{\mathcal{T}_{\mathrm{CDS}}, \mathcal{T}_{\mathrm{BCD}}\}$. We know, thanks to [5] (Figure 14.2), that the following rule is admissible for $\lambda^{\mathcal{T}}_{\cap}$:

$$\dfrac{B \vdash^{\mathcal{T}}_{\cap} M : \sigma \quad B \vdash^{\mathcal{T}}_{\cap} N : \tau \quad M =_{\beta} N}{B \vdash^{\mathcal{T}}_{\cap} M : \sigma \cap \tau}\,(\cap I)_{\text{adm}}$$

Then the proof proceeds by induction on the derivation of $B \vdash^{\mathcal{T}}_{=_\beta} \Delta : \sigma$. The most important case is when the last used rule is $(\cap I)$: by induction we get $B \vdash^{\mathcal{T}}_{\cap} \wr \Delta_1 \wr : \sigma$, and $B \vdash^{\mathcal{T}}_{\cap} \wr \Delta_2 \wr : \tau$, and $\wr \Delta_1 \wr =_{\beta} \wr \Delta_2 \wr$, and, by the essence definition, $\wr \langle \Delta_1 \,, \Delta_2 \rangle \wr =_{\beta} \wr \Delta_1 \wr$. Apply rule $(\cap I)_{\text{adm}}$ and conclude with $B \vdash^{\mathcal{T}}_{\cap} \wr \Delta_1 \wr : \sigma \cap \tau$.

($\not\Leftarrow$) Loss of soundness for $\Delta^{\text{CD}}_{=_\beta}$ and $\Delta^{\text{CDV}}_{=_\beta}$. Let $\mathcal{T} \in \{\mathcal{T}_{\text{CD}}, \mathcal{T}_{\text{CDV}}\}$. Let $\mathsf{S} \stackrel{\mathit{def}}{=} \lambda x.\lambda y.\lambda z.x\,z\,(y\,z)$ and $\mathsf{K} \stackrel{\mathit{def}}{=} \lambda x.\lambda y.x$. Let $\Delta \stackrel{\mathit{def}}{=} (\lambda x{:}(\sigma \to \tau \to \rho) \to ((\sigma \to \tau) \to \sigma \to \rho) \to \sigma \to \tau \to \rho.\lambda y{:}(\sigma \to \tau \to \rho) \to (\sigma \to \tau) \to \sigma \to \rho.\lambda z{:}\sigma \to \tau \to \rho.x\,z\,(y\,z))(\lambda x{:}\sigma \to \tau \to \rho.\lambda y{:}(\sigma \to \tau) \to \sigma \to \rho.x)(\lambda x{:}\sigma \to \tau \to \rho.\lambda y{:}\sigma \to \tau.\lambda z{:}\sigma.x\,z\,(y\,z))$. Δ is a simply-typed term of type $(\sigma \to \tau \to \rho) \to (\sigma \to \tau \to \rho)$, and its essence is $\wr \Delta \wr \equiv \mathsf{S}\,\mathsf{K}\,\mathsf{S}$. Consider the following counter-example:

$$\dfrac{\dfrac{\vdots}{\vdash^{\mathcal{T}}_{=_\beta} \Delta : (\sigma \to \tau \to \rho) \to (\sigma \to \tau \to \rho)} \quad \dfrac{x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} x : \sigma}{\vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.x : \sigma \to \sigma} \quad \mathsf{S}\,\mathsf{K}\,\mathsf{S} =_\beta \lambda x.x}{\dfrac{\vdash^{\mathcal{T}}_{=_\beta} \langle \Delta \,, \lambda x{:}\sigma.x \rangle : ((\sigma \to \tau \to \rho) \to (\sigma \to \tau \to \rho)) \cap (\sigma \to \sigma)}{\vdash^{\mathcal{T}}_{=_\beta} \mathit{pr}_2\, \langle \Delta \,, \lambda x{:}\sigma.x \rangle : \sigma \to \sigma}}$$

The essence of $\mathit{pr}_2\, \langle \Delta \,, \lambda x{:}\sigma.x \rangle$ is $\mathsf{S}\,\mathsf{K}\,\mathsf{S}$, but, if σ is an atomic type:

$$\nvdash^{\mathcal{T}}_{\cap} \mathsf{S}\,\mathsf{K}\,\mathsf{S} : \sigma \to \sigma$$

Loss of soundness in $\Delta^{\text{CDV}}_{=_{\beta\eta}}$ is proved via the following counterexample, where $B \stackrel{\mathit{def}}{=} \{x{:}(\sigma \to \tau) \cap \rho\}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{B, y{:}\sigma \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} x : (\sigma \to \tau) \cap \rho}{B, y{:}\sigma \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} \mathit{pr}_1\, x : \sigma \to \tau} \quad B, y{:}\sigma \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} y : \sigma}{B, y{:}\sigma \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} (\mathit{pr}_1\, x)\, y : \tau}}{B \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} \lambda y{:}\sigma.(\mathit{pr}_1\, x)\, y : \sigma \to \tau} \quad \dfrac{B \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} x : (\sigma \to \tau) \cap \rho}{B \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} \mathit{pr}_2\, x : \rho} \quad \lambda y.x\,y =_{\beta\eta} x}{B \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} \langle \lambda y{:}\sigma.(\mathit{pr}_1\, x)\, y \,, \mathit{pr}_2\, x \rangle : (\sigma \to \tau) \cap \rho}}{B \vdash^{\mathcal{T}_{\text{CDV}}}_{=_{\beta\eta}} \mathit{pr}_2\, \langle \lambda y{:}\sigma.(\mathit{pr}_1\, x)\, y \,, \mathit{pr}_2\, x \rangle : \rho}$$

The essence of $\mathit{pr}_2\, \langle \lambda y{:}\sigma.(\mathit{pr}_1\, x)\, y \,, \mathit{pr}_2\, x \rangle$ is $\lambda y.x\,y$, but, if ρ is an atomic type:

$$x{:}(\sigma \to \tau) \cap \rho \nvdash^{\mathcal{T}_{\text{CDV}}}_{\cap} \lambda y.x\,y : \rho$$

Loss of soundness in $\Delta^{\text{BCD}}_{=_{\beta\eta}}$ is proved via the following counterexample:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x{:}\sigma, y{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} x : \sigma \quad \sigma \leqslant_{\mathcal{T}} \mathtt{U} \to \mathtt{U}}{x{:}\sigma, y{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} x^{\mathtt{U} \to \mathtt{U}} : \mathtt{U} \to \mathtt{U}} \quad x{:}\sigma, y{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} y : \mathtt{U}}{x{:}\sigma, y{:}\mathtt{U} \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} x^{\mathtt{U} \to \mathtt{U}}\, y : \mathtt{U}}}{x{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} \lambda y{:}\mathtt{U}.x^{\mathtt{U} \to \mathtt{U}}\, y : \mathtt{U} \to \mathtt{U}} \quad x{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} x : \sigma \quad \lambda y.x\,y =_{\beta\eta} x}{x{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} \langle \lambda y{:}\mathtt{U}.x^{\mathtt{U} \to \mathtt{U}}\, y \,, x \rangle : (\mathtt{U} \to \mathtt{U}) \cap \sigma}}{x{:}\sigma \vdash^{\mathcal{T}_{\text{BCD}}}_{=_{\beta\eta}} \mathit{pr}_2\, \langle \lambda y{:}\mathtt{U}.x^{\mathtt{U} \to \mathtt{U}}\, y \,, x \rangle : \sigma}$$

The essence of $pr_2\,\langle \lambda y{:}\mathtt{U}.x^{\mathtt{U}\to\mathtt{U}}\,y\;,x\rangle$ is $\lambda y.x\,y$, but, if σ is an atomic type (different than $\mathtt{U}$):

$$x{:}\sigma \nvdash_{\cap}^{\mathcal{T}_{\mathrm{BCD}}} \lambda y.x\,y : \sigma$$

($\triangleright$) Let M be such that $B \vdash_{\cap}^{\mathcal{T}} M : \sigma$ for a given B. We proceed by induction on the derivation. All cases proceed straightforwardly since all rules of the type and subtype assignment system $\vdash_{\cap}^{\mathcal{T}}$ correspond exactly to the rules of the same name in the corresponding typed system $\vdash_{\mathcal{R}}^{\mathcal{T}}$ and in the same type theory $\mathcal{T}$. Therefore a Δ-term can be easily be constructed and derived with type σ; ◀

The last theorem characterizes the class of strongly normalizing Δ-terms.

▶ **Theorem 53** (Characterization).
Every strongly normalizing λ-term can be type-annotated so as to be the essence of a typable Δ-term.

Proof. We know that every strongly normalizing λ-term M is typable in $\lambda_{\cap}^{\mathcal{T}}$. By Theorem 52 we have that $\Delta_{\mathcal{R}}^{\mathcal{T}} \triangleright \lambda_{\cap}^{\mathcal{T}}$, therefore there exists some typable Δ, such that $M \equiv \wr\,\Delta\,\wr$. ◀

We can finally state decidability of type checking (TC) and type reconstruction (TR).

▶ **Theorem 54** (Decidability of type checking and type reconstruction).

$\Delta_{\mathcal{R}}^{\mathcal{T}}$	TC/TR
$\Delta_{\equiv}^{\mathrm{CD}}$	$\surd$
$\Delta_{\equiv}^{\mathrm{CDV}}$	$\surd$
$\Delta_{\equiv}^{\mathrm{CDS}}$	$\surd$
$\Delta_{\equiv}^{\mathrm{BCD}}$	$\surd$
$\Delta_{=_\beta}^{\mathrm{CD}}$	$\surd$
$\Delta_{=_\beta}^{\mathrm{CDV}}$	$\surd$
$\Delta_{=_\beta}^{\mathrm{CDS}}$	$\times$
$\Delta_{=_\beta}^{\mathrm{BCD}}$	$\times$
$\Delta_{=_{\beta\eta}}^{\mathrm{CDV}}$	$\surd$
$\Delta_{=_{\beta\eta}}^{\mathrm{BCD}}$	$\times$

Proof. Both type checking and type reconstruction can be proved by induction on the structure of Δ, using the decidability of $\mathcal{T}_{\mathrm{BCD}}$ proved by Hindley [23] (see also [29]). By Theorem 52, the essences of all the Δ-terms, which are typable in $\Delta_{=_\beta}^{\mathrm{CD}}$, $\Delta_{=_\beta}^{\mathrm{CDV}}$, or $\Delta_{=_{\beta\eta}}^{\mathrm{CDV}}$, are typable in $\lambda_{\cap}^{\mathrm{CD}}$ or $\lambda_{\cap}^{\mathrm{CDV}}$, therefore they are strongly normalizing. As a consequence, the side-condition $\wr\,\Delta_1\,\wr\;\mathcal{R}\;\wr\,\Delta_2\,\wr$ is decidable for $\Delta_{=_\beta}^{\mathrm{CD}}$, $\Delta_{=_\beta}^{\mathrm{CDV}}$, and $\Delta_{=_{\beta\eta}}^{\mathrm{CDV}}$ and so type reconstruction and type checking are decidable too.
Type reconstruction and type checking are not decidable in $\Delta_{=_\beta}^{\mathrm{CDS}}$, $\Delta_{=_\beta}^{\mathrm{BCD}}$, and $\Delta_{=_{\beta\eta}}^{\mathrm{BCD}}$, because $\langle u_{\Delta_1}\;,u_{\Delta_2}\rangle$ is typable if and only if $\wr\,\Delta_1\,\wr =_\beta \wr\,\Delta_2\,\wr$ (resp. $\wr\,\Delta_1\,\wr =_{\beta\eta} \wr\,\Delta_2\,\wr$). However, $\wr\,\Delta_1\,\wr$ and $\wr\,\Delta_2\,\wr$ are arbitrary pure λ-terms, and both β-equality and βη-equality are undecidable. ◀

B.2 Subtyping and explicit coercions

The typing rule ($\leqslant_{\mathcal{T}}$) in the general typed system introduces type coercions: once a type coercion is introduced, it cannot be eliminated, so *de facto* freezing a Δ-term inside an explicit coercion. Tannen *et al.* [42] showed a translation of a judgment derivation from a "Source" system with subtyping (Cardelli's Fun [10]) into an "equivalent" judgment derivation in a "Target" system without subtyping (Girard system F with records and recursion). In the same spirit, we present a translation that removes all explicit coercions. Intuitively, the translation proceeds as follows: every derivation ending with rule

$$\frac{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma \quad \sigma \leqslant_{\mathcal{T}} \tau}{B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta^{\tau} : \tau} \ (\leqslant_{\mathcal{T}})$$

is translated into the following (coercion-free) derivation

$$\frac{B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\sigma \leqslant_{\mathcal{T}} \tau\| : \sigma \to \tau \quad B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\Delta\|_B : \sigma}{B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\sigma \leqslant_{\mathcal{T}} \tau\| \ \|\Delta\|_B : \tau} \ (\to E)$$

where $\mathcal{R}'$ is a suitable relation such that $\mathcal{R} \sqsubseteq \mathcal{R}'$. Note that changing of the type theory is necessary to guarantee well-typedness in the translation of strong pairs. Summarizing, we provide a type preserving translation of a Δ-term into a coercion-free Δ-term such that $\wr \Delta \wr =_{\beta\eta} \wr \Delta' \wr$.
The following example illustrates some trivial compilations of axioms and rule schemes of Figure 1.

▶ **Example 55** (Translation of axioms and rule schemes of Figure 1)**.**

(refl) the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma} \rangle : \sigma \cap \sigma$ is translated to a coercion-free judgment

$$x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma.y)\, x \rangle : \sigma \cap \sigma$$

(incl) the judgment $x{:}\sigma \cap \tau \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\tau} \rangle : (\sigma \cap \tau) \cap \tau$ is translated to a coercion-free judgment

$$x{:}\sigma \cap \tau \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma \cap \tau.pr_2\, y)\, x \rangle : (\sigma \cap \tau) \cap \tau$$

(glb) the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma \cap \sigma} \rangle : \sigma \cap (\sigma \cap \sigma)$ is translated to a coercion-free judgment

$$x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma.\langle y \,, y \rangle)\, x \rangle : \sigma \cap (\sigma \cap \sigma)$$

($\mathtt{U}_{top}$) the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\mathtt{U}} \rangle : \sigma \cap \mathtt{U}$ is translated to a coercion-free judgment

$$x{:}\sigma \vdash^{\mathcal{T}}_{=_\beta} \langle x \,, (\lambda y{:}\sigma.u_y)\, x \rangle : \sigma \cap \mathtt{U}$$

($\mathtt{U}_{\to}$) the judgment $x{:}\mathtt{U} \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma \to \mathtt{U}} \rangle : \mathtt{U} \cap (\sigma \to \mathtt{U})$ is translated to a coercion-free judgment

$$x{:}\mathtt{U} \vdash^{\mathcal{T}}_{=_{\beta\eta}} \langle x \,, (\lambda f{:}\mathtt{U}.\lambda y{:}\sigma.u_{(f\, y)})\, x \rangle : \mathtt{U} \cap (\sigma \to \mathtt{U})$$

($\to\cap$) the judgment $x{:}(\sigma \to \tau) \cap (\sigma \to \rho) \vdash^{\mathcal{T}}_{\mathcal{R}} x^{\sigma \to \tau \cap \rho} : \sigma \to \tau \cap \rho$ is translated to a coercion-free judgment

$$x{:}(\sigma \to \tau) \cap (\sigma \to \rho) \vdash^{\mathcal{T}}_{=_{\beta\eta}} (\lambda f{:}(\sigma \to \tau) \cap (\sigma \to \rho).\lambda y{:}\sigma.\langle (pr_1\, f)\, y \,, (pr_2\, f)\, y \rangle)\, x : \sigma \to \tau \cap \rho$$

($\to$) the judgment $x{:}\sigma \to \tau \cap \rho \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x \,, x^{\sigma \cap \rho \to \tau} \rangle : (\sigma \to \tau \cap \rho) \cap (\sigma \cap \rho \to \tau)$ is translated to a coercion-free judgment

$$x{:}\sigma \to \tau \cap \rho \vdash^{\mathcal{T}}_{=_{\beta\eta}} \langle x, (\lambda f{:}\sigma \to \tau \cap \rho.\lambda y{:}\sigma \cap \rho.pr_1\, (f\, (pr_1\, y)))\, x \rangle : (\sigma \to \tau \cap \rho) \cap (\sigma \cap \rho \to \tau)$$

(trans) the judgment $x{:}\sigma \vdash^{\mathcal{T}}_{\mathcal{R}} \langle x\,, (x^{\mathtt{U}})^{\sigma\to\mathtt{U}}\rangle : \sigma \cap (\sigma \to \mathtt{U})$ is translated to a coercion-free judgment

$$x{:}\sigma \vdash^{\mathcal{T}}_{=_{\beta\eta}} \langle x\,, (\lambda f{:}\mathtt{U}.\lambda y{:}\sigma.u_{(f\,y)})\,((\lambda y{:}\sigma.u_y)\,x)\rangle : \sigma \cap (\sigma \to \mathtt{U})$$

The next definition introduces two maps translating subtype judgments into explicit coercions functions and Δ-terms into coercion-free Δ-terms.

▶ **Definition 56** (Translations $\|-\|$ and $\|-\|_B$).

1. The minimal type theory $\leqslant_{\min}$ and the extra axioms and schemes are translated as follows.

(refl) $$\|\sigma \leqslant_{\mathcal{T}} \sigma\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.x : \sigma \to \sigma$$

(incl$_1$) $$\|\sigma \cap \tau \leqslant_{\mathcal{T}} \sigma\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma \cap \tau.pr_1\,x : \sigma \cap \tau \to \sigma$$

(incl$_2$) $$\|\sigma \cap \tau \leqslant_{\mathcal{T}} \tau\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma \cap \tau.pr_2\,x : \sigma \cap \tau \to \tau$$

(glb) $$\left\|\frac{\rho \leqslant_{\mathcal{T}} \sigma \quad \rho \leqslant_{\mathcal{T}} \tau}{\rho \leqslant_{\mathcal{T}} \sigma \cap \tau}\right\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\rho.\langle \|\rho \leqslant_{\mathcal{T}} \sigma\|\,x\,, \|\rho \leqslant_{\mathcal{T}} \tau\|\,x\rangle : \rho \to \sigma \cap \tau$$

(trans) $$\left\|\frac{\sigma \leqslant_{\mathcal{T}} \tau \quad \tau \leqslant_{\mathcal{T}} \rho}{\sigma \leqslant_{\mathcal{T}} \rho}\right\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.\,\|\tau \leqslant_{\mathcal{T}} \rho\|\;(\|\sigma \leqslant_{\mathcal{T}} \tau\|\;x) : \sigma \to \rho$$

($\mathtt{U}_{top}$) $$\|\sigma \leqslant_{\mathcal{T}} \mathtt{U}\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\sigma.u_x : \sigma \to \mathtt{U}$$

($\mathtt{U}_\to$) $$\|\mathtt{U} \leqslant_{\mathcal{T}} \sigma \to \mathtt{U}\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_{\beta\eta}} \lambda f{:}\mathtt{U}.\lambda x{:}\sigma.u_{(f\,x)} : \mathtt{U} \to (\sigma \to \mathtt{U})$$

Let $\xi_1 \stackrel{def}{=} (\sigma \to \tau) \cap (\sigma \to \rho)$ and $\xi_2 \stackrel{def}{=} \sigma \to \tau \cap \rho$

($\to\cap$) $$\|\xi_1 \leqslant_{\mathcal{T}} \xi_2\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_{\beta\eta}} \lambda f{:}\xi_1.\lambda x{:}\sigma.\langle (pr_1\,f)\,x\,, (pr_2\,f)\,x\rangle : \xi_1 \to \xi_2$$

Let $\xi_1 \stackrel{def}{=} \sigma_1 \to \tau_1$ and $\xi_2 \stackrel{def}{=} \sigma_2 \to \tau_2$

($\to$) $$\left\|\frac{\sigma_2 \leqslant_{\mathcal{T}} \sigma_1 \quad \tau_1 \leqslant_{\mathcal{T}} \tau_2}{\sigma_1 \to \tau_1 \leqslant_{\mathcal{T}} \sigma_2 \to \tau_2}\right\| \stackrel{def}{=} \vdash^{\mathcal{T}}_{=_{\beta\eta}} \lambda f{:}\xi_1.\lambda x{:}\sigma_2.\,\|\tau_1 \leqslant_{\mathcal{T}} \tau_2\|\;(f\;(\|\sigma_2 \leqslant_{\mathcal{T}} \sigma_1\|\;x)) : \xi_1 \to \xi_2$$

2. The translation $\|-\|_B$ is defined on Δ as follows.

$$\begin{array}{rcll}
\|u_\Delta\|_B & \stackrel{def}{=} & u_{\|\Delta\|_B} & \\
\|x\|_B & \stackrel{def}{=} & x & \\
\|\lambda x{:}\sigma.\Delta\|_B & \stackrel{def}{=} & \lambda x{:}\sigma.\,\|\Delta\|_{B,x:\sigma} & \\
\|\Delta_1\,\Delta_2\|_B & \stackrel{def}{=} & \|\Delta_1\|_B\;\|\Delta_2\|_B & \\
\|\langle \Delta_1\,, \Delta_2\rangle\|_B & \stackrel{def}{=} & \langle \|\Delta_1\|_B\,, \|\Delta_2\|_B\rangle & \\
\|pr_i\,\Delta\|_B & \stackrel{def}{=} & pr_i\;\|\Delta\|_B & i \in \{1,2\} \\
\|\Delta^\tau\|_B & \stackrel{def}{=} & \|\sigma \leqslant_{\mathcal{T}} \tau\|\;\|\Delta\|_B & \text{if } B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma.
\end{array}$$

By looking at the above translation functions we can see that if $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, then $\|\Delta\|_B$ is defined and it is coercion-free.

The following lemma states that a coercion function is always typable in $\Delta^{\mathcal{T}}_{=_{\beta\eta}}$, that it is essentially the identity and that, without using the rule schemes ($\to\cap$), ($\mathtt{U}_\to$), and ($\to$) the translation can even be derivable in $\Delta^{\mathcal{T}}_{=_\beta}$.

Source	Target
$\Delta^{\mathrm{CD}}_{\equiv}$	$\Delta^{\mathrm{CD}}_{=_\beta}$
$\Delta^{\mathrm{CDV}}_{\equiv}$	$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CDS}}_{\equiv}$	$\Delta^{\mathrm{CDS}}_{=_\beta}$
$\Delta^{\mathrm{BCD}}_{\equiv}$	$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CD}}_{=_\beta}$	$\Delta^{\mathrm{CD}}_{=_\beta}$
$\Delta^{\mathrm{CDV}}_{=_\beta}$	$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CDS}}_{=_\beta}$	$\Delta^{\mathrm{CDS}}_{=_\beta}$
$\Delta^{\mathrm{BCD}}_{=_\beta}$	$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$	$\Delta^{\mathrm{CDV}}_{=_{\beta\eta}}$
$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$	$\Delta^{\mathrm{BCD}}_{=_{\beta\eta}}$

Figure 6 On the left: source systems. On the right: target systems without the $(\leqslant_{\mathcal{T}})$ rule.

▶ **Lemma 57** (Essence of a coercion is an identity).

1. *If $\sigma \leqslant_{\mathcal{T}} \tau$, then $\vdash^{\mathcal{T}}_{=_{\beta\eta}} \|\sigma \leqslant_{\mathcal{T}} \tau\| : \sigma \to \tau$ and $\wr \|\sigma \leqslant_{\mathcal{T}} \tau\| \wr =_{\beta\eta} \lambda x.x$;*
2. *If $\sigma \leqslant_{\mathcal{T}} \tau$ without using the rule schemes $(\to\cap)$, $(\mathtt{U}_\to)$, and $(\to)$, then $\vdash^{\mathcal{T}}_{=_\beta} \|\sigma \leqslant_{\mathcal{T}} \tau\| : \sigma \to \tau$ and $\wr \|\sigma \leqslant_{\mathcal{T}} \tau\| \wr =_\beta \lambda x.x$.*

Proof. The proofs proceed in both parts by induction on the derivation of $\sigma \leqslant_{\mathcal{T}} \tau$. For instance, in case of (glb), we can verify that $\vdash^{\mathcal{T}}_{=_\beta} \lambda x{:}\rho.\langle \|\rho \leqslant_{\mathcal{T}} \sigma\|\, x\,, \|\rho \leqslant_{\mathcal{T}} \tau\|\, x\rangle : \rho \to \sigma \cap \tau$ using the induction hypotheses that $\|\rho \leqslant_{\mathcal{T}} \sigma\|$ (resp. $\|\rho \leqslant_{\mathcal{T}} \tau\|$) has type $\rho \to \sigma$ (resp. $\rho \to \tau$) and has an essence convertible to $\lambda x.x$. ◀

We can now prove the coherence of the translation as follows.

▶ **Theorem 58** (Coherence).
If $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, then $B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\Delta\|_B : \sigma$ and $\wr \|\Delta\|_B \wr\ \mathcal{R}'\ \wr \Delta \wr$, where $\Delta^{\mathcal{T}}_{\mathcal{R}}$ and $\Delta^{\mathcal{T}}_{\mathcal{R}'}$ are respectively the source and target intersection typed systems given in Figure 6.

Proof. By induction on the derivation. We illustrate the most important case, namely when the last type rule is $(\leqslant_{\mathcal{T}})$. In this case $\|\Delta^\tau\|_B$ is translated to $\|\sigma \leqslant_{\mathcal{T}} \tau\|\ \|\Delta\|_B$. By induction hypothesis we have that $B \vdash^{\mathcal{T}}_{\mathcal{R}} \Delta : \sigma$, and by Lemma 57 we have that $B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\sigma \leqslant_{\mathcal{T}} \tau\| : \sigma \to \tau$; therefore $B \vdash^{\mathcal{T}}_{\mathcal{R}'} \|\Delta^\tau\|_B : \tau$. Moreover, we know that $\wr \|\sigma \leqslant_{\mathcal{T}} \tau\| \wr\ \mathcal{R}'\ \lambda x.x$, and this gives $\wr \|\Delta^\tau\|_B \wr\ \mathcal{R}'\ \wr \|\Delta\|_B \wr$. Again by induction hypothesis we have that $\wr \|\Delta\|_B \wr\ \mathcal{R}'\ \wr \Delta \wr$, and this gives the thesis $\wr \|\Delta^\tau\|_B \wr\ \mathcal{R}'\ \wr \Delta^\tau \wr$. ◀