Preface

The 13th Seminar on Algebra, Logic and Geometry in Informatics (ALGI) took place at Research Institute for Mathematical Sciences, Kyoto University in December 2002. ALGI started on 1995 as a series of seminars on applications of algebra, logic and geometry to informatics, and also applications of informatics to these areas of mathematics.

The meeting consisted of talks by 20 speakers, including invited tutorial talks by Satoko Itaya, Makoto Kikuchi, Misao Nagayama, Hitoshi Ohsaki, Atsushi Shimojima and Alex Simpson. Some talks were given at the joint sessions with the Computability on the Continuum Seminar (CC-Seminar) organized by Mariko Yasugi.

This volume contains 19 articles based on or related to the talks presented at the meeting. The organizers are grateful to the speakers/contributors as well as the participants, and acknowledge the support from Joint Research Service at RIMS.

ALGI-13 Organizers
Hitoshi Furusawa
Masahito Hasegawa
Yoshiki Kinoshita
Izumi Takeuti

代数、論理、幾何と情報科学
Algebra, Logic and Geometry in Informatics
短期共同研究報告集

2002 年 12 月 16 日〜20 日
研究代表者：産業技術総合研究所　木下 佳樹 (Yoshiki Kinoshita)

目次

Pointwise and Sequential Continuity in Constructive Analysis

Hajime Ishihara（石原 哉）
JAIST（北陸先端科学技術大学院大学）

We discuss various continuity properties, especially pointwise and sequential continuity, in Bishop's constructive mathematics; see [1, 2, 11] for Bishop's constructive mathematics and [3, 4, 5, 9] for various continuity properties. We say that a mapping f between metric spaces X and Y is *sequentially continuous* if $x_n \to x$ implies that $f(x_n) \to f(x)$; *pointwise continuous* if for each $x \in X$ and $\epsilon > 0$ there exists $\delta > 0$ such that $d(x, y) < \delta$ implies $d(f(x), f(y)) < \epsilon$ for all $y \in X$. We first show the following theorem.

Theorem 1 *The following are equivalent.*

1. *Every sequentially continuous mapping of a separable metric space into a metric space is pointwise continuous.*
2. *Every sequentially continuous mapping of a complete separable metric space into a metric space is pointwise continuous.*
3. BD-N. *Every countable pseudo-bounded subset of* $\mathbf{N}$ *is bounded.*

Here a subset A of $\mathbf{N}$ is said to be *pseudo-bounded* if for each sequence $\{a_n\}$ in A, $a_n < n$ for all sufficiently large n. Note that although BD-N holds in classical mathematics, intuitionistic mathematics and constructive recursive mathematics of Markov's school, a natural recursivisation of BD-N is independent of Heyting arithmetic [3, 5, 8, 10].

We also show that very important theorems in functional analysis – Banach's inverse mapping theorem, the open mapping theorem, the closed graph theorem, the Banach-Steinhaus theorem and the Hellinger-Toeplitz theorem – can be proved in Bishop's constructive mathematics for *sequentially continuous* linear mappings [6, 7]. However it has emerged that the theorems for *pointwise continuous* linear mappings are equivalent to BD-N [8].

参考文献

[1] Errett Bishop, *Foundations of Constructive Analysis*, McGraw-Hill, New York, 1967.

[2] Errett Bishop and Douglas Bridges, *Constructive Analysis*, Springer-Verlag, Heidelberg, 1985.

[3] Douglas Bridges, Hajime Ishihara, Peter Schuster and Luminiţa Vîţă, *Strong continuity implies uniform sequential continuity*, preprint, 2001.

[4] Hajime Ishihara, *Continuity and nondiscontinuity in constructive mathematics*, J. Symbolic Logic **56** (1991), 1349–1354.

[5] Hajime Ishihara, *Continuity properties in constructive mathematics*, J. Symbolic Logic **57** (1992), 557–565.

[6] Hajime Ishihara, *A constructive version of Banach's inverse mapping theorem*, New Zealand J. Math. **23** (1994), 35–43.

[7] Hajime Ishihara, *Sequential continuity of linear mappings in constructive mathematics*, J. UCS **3** (1997), 1250–1254.

[8] Hajime Ishihara, *Sequential continuity in constructive mathematics*, In: C.S. Calude, M.J. Dinneen and S. Sburlan eds., Combinatorics, Computability and Logic, Springer-Verlag, London, 2001, 5–12.

[9] Hajime Ishihara and Peter Schuster, *Some constructive uniform continuity theorem*, Q. J. Math. **53** (2002), 185–193.

[10] Hajime Ishihara and Satoru Yoshida, *A constructive look at the completeness of* $\mathcal{D}(\mathbf{R})$, J. Symbolic Logic **67** (2002), 1511–1519.

[11] A. S. Troelstra and D. van Dalen, *Constructivism in Mathematics*, Vol. 1 and 2, North-Holland, Amsterdam, 1988.

Effective Limit in Computable Analysis

竹内　泉
274-8510 千葉県船橋市三山 2 丁目 2–1 東邦大学理学部情報科学科
Takeuti Izumi
Information Science, Toho Univ., Miyama Hunabasi, Tiba, 274-8510 JAPAN
takeuti@is.sci.toho-u.ac.jp

1 Introduction

Most of the studies of computable analysis have been capturing the computability of structures over real numbers. However, there are started some studies on other general mathematical structures recently.

Hertling studied computability over algebraic structures [H], however he did not discuss topology. Schröder studied week limit spaces [S], however he did not discuss modulus of convergence. The author believe that the notions of topology and modulus of convergence are most important in the study of computable analysis. This work aims to combine their works, and to apply to the discussion on modulus of convergence.

In discussing modulus of convergence, it seems natural to do with uniform topology, which Yasugi and others studied [Y]. They discuss the computability under uniform topology with computability structure, while Hertling and Schröder did with representation.

There are several methods in studying computable analysis. One of them is representation and another is computability structure.

The notion of representation has been studied by Weihrauch [W] and others. It is a kind of a mechanical method. A representation r of a mathematical structure X is a surjective partial function of Σ^ω into X. A function $f : X \to X$ is computable if the is a computable partial function $\bar{f}$ of Σ^* into Σ^* such that $f \circ r = r \circ \bar{f}$. We will use $\mathbf{N} \to \mathbf{N}$ as Σ^ω, the domain of representation. It seems that representations have too much information which comes from the details of implementation of computation models.

The notion of computability structure is studied by Pour-El et al. [P]. It is a kind of an axiomatic method. A computability structure over a mathematical structure X is a subset of X^ω which satisfies the axioms of computability structure. A function f over X is called computable if f is locally uniformly continuous and preserves the computability structure over X. In order to assert that a computability structure is natural, the mathematical structure X has been requested to have a distance. Yasugi et al. attempt to replace the distance with effective uniform topology in the literature [Y].

This work aims at comparing these two ways, that by computability structure and that by representation. The author would like to show that the essential notions are equivalently defined by both ways.

There are several ways to define effective uniform topology. Yasugi et al. defined it with effective uniform neighbourhood system in the literature [Y]. This work defines it with effective limit, which has not been used in any other works. An effective limit is a partial function of arity ω over the underlying set, while an effective neighbourhood system is a family of subsets. It seems that a partial function is more familiar in computational theory than a family of subset is.

There are three similar words 'recursive', 'effective' and 'computable', all of which appear in this paper. There are only little difference in the senses of these words. In this paper, the word 'recursive' is used only for the notions concerned to recursive functions over natural numbers. The word 'computable' is used only for the notion of computability structure which defined by Pour-El et al. [P]. The word 'effective' is used for other mathematical structures, Thus, a function represented by a recursive Type-2 function is called a recursive function in this paper, although Weihrauch and others call it a computable function [W].

2 Uniform Topology

Notation 2.1 We write $\mathbf{N}$ for $\{1, 2, 3, ...\}$.

Definition 2.2 (Partial function) A function f is a partial function of X into Y iff f is a function of $X' \subset X$ into $Y' \subset Y$. This X' is called the *domain* of f and written as $dom(f)$. We write $f : X \to_p Y$ when f is a partial function of X into Y. For $x \in X$, we say that $f(x)$ is *defined* when $x \in dom(f)$, and $f(x)$ is *undefined* if not. When we write $f(x) = y$, we implicitly assert $x \in dom(f)$.

The *range* of $f : X \to_p Y$ is the set $\{f(x) \in Y \mid x \in dom(f)\}$ and written as $range(f)$. For $f : X \to_p Y$ and $g : Y \to_p Z$, the concatenation $g \circ f : X \to_p Z$ is defined as following: $dom(g \circ f) = \{x \in X \mid x \in dom(f), f(x) \in dom(g)\}$ and $g \circ f(x) = g(f(x))$ for $x \in dom(g \circ f)$.

For partial functions $f : X \to_p Y$ and $g : X' \to_p Y'$, the relation $f \subset_p g$ holds iff $dom(f) \subset dom(g)$ and $f(x) = g(x)$ for all $x \in dom(f)$. It may be the case that $X \neq X'$ or $Y \neq Y'$ although $f \subset_p g$. For partial functions f and g, it holds $f = g$ if $f \subset_p g$ and $g \subset_p f$.

Notation 2.3 We write $X \to Y$ for the function space $\{f \mid f : X \to Y\}$, and $X \to_p Y$ for the set of all the partial functions $\{f \mid f : X \to_p Y\}$. Thus, the notation $E \subset X \to Y$ does not mean $E : X \to_p Y$ but means $\forall f \in E.\ f : X \to Y$.

The operators $\to$ and $\to_p$ are right associative. Thus, the notation $X \to Y \to_p Z$ is an abbreviation of $X \to (Y \to_p Z)$.

Definition 2.4 (Modular Limit) Let X be a set. An partial function LM is a *modular limit* over X if there is a strictly increasing function $m : \mathbf{N} \to \mathbf{N}$ and the followings hold:

0. $LM : X^\omega \to_p X$

1. For each $x \in X$, $LM(x, x, x, ...) = x$
2. Let f be a function of $\mathbf{N} \to \mathbf{N}$ such that $f(n) \geq n$ for any $n \in \mathbf{N}$. If $(x_1, x_2, x_3, ...) \in dom(LM)$, then $(x_{f(1)}, x_{f(2)}, x_{f(3)}, ...) \in dom(LM)$ and $LM(x_{f(1)}, x_{f(2)}, x_{f(3)}, ...) = LM(x_1, x_2, x_3, ...)$
3. Let $\{x_{i,j}\}_{i,j}$ be a double sequence in X, that is, $x_{i,j} \in X$ for any $i, j \in \mathbf{N}$. Let $(y_1, y_2, y_3, ...)$ be a sequence in X. and z be a point in X. Suppose that $y_i = LM(x_{i,1}, x_{i,2}, x_{i,3}, ...)$ for each i.

   If $LM(y_1, y_2, y_3, ...) = z$, then $LM(x_{m(1),m(1)}, x_{m(2),m(2)}, x_{m(3),m(3)}, ...) = z$.
4. Let $\{x_{i,j}\}_{i,j}$ be a double sequence in X, that is, $x_{i,j} \in X$ for any $i, j \in \mathbf{N}$. Let $(y_1, y_2, y_3, ...)$ be a sequence in X. and z be a point in X. Suppose that $y_i = LM(x_{i,1}, x_{i,2}, x_{i,3}, ...)$ for each i.

   If $LM(x_{1,1}, x_{2,2}, x_{3,3}, ...) = z$, then $LM(y_{m(1)}, y_{m(2)}, y_{m(3)}, ...) = z$.

The function $m(\,)$, which appears in the conditions 3 and 4, is called a *modulus of diagonal convergence*.

Definition 2.5 (Effective Limit) A modular limit is called an *effective limit* when the modulus of diagonal convergence of it is a recursive function.

Remark 2.6 The conditions 3 and 4 in the definition of modular limit means that all the converging sequence under the modular limit are uniformly converging.

Definition 2.7 (Uniform Neighbourhood System) Let X be a set. The series of subsets $\{V_i(x)\}_{i \in \mathbf{N}, x \in X}$ is a *uniform neighbourhood system* iff it satisfies the followings:

1. $x \in V_i(x) \subset X$ for each $i \in \mathbf{N}$ and each $x \in X$.
2. $V_i(x) \supset V_j(x)$ for $i < j$
3. There is a function $m : \mathbf{N} \to \mathbf{N}$ such that if $y \in V_{m(i)}(x)$ then $x \in V_i(y)$
4. There is a function $m' : \mathbf{N} \to \mathbf{N}$ such that if $y \in V_{m'(i)}(x)$ then $V_{m'(i)}(y) \subset V_i(x)$

The functions m and m' in the conditions 3 and 4 are called the *moduli*.

Definition 2.8 (Effective Uniform Neighbourhood System) The uniform neighbourhood system is *effective* if both of the moduli are recursive functions.

Remark 2.9 When we replace the condition 4 of the definition above with:
$\forall x \in X. \forall i \in \mathbf{N}. \exists j \in \mathbf{N}. \forall y \in V_j(x). \exists k \in \mathbf{N}.\ V_k(y) \subset V_i(x)$
then, the conditions defines the ordinary notion of countable neighbourhood systems. If we exchange the order of quantifiers $\forall y$ and $\exists k$, then, this condition is equivalent to the condition 4 of the definition above.

Proposition 2.10 *Let X be a set and $\{V_i(x)\}_{i,x}$ be a uniform neighbourhood system over X. Then, X is a Hausdorff space iff $\bigcap_{i \in \mathbf{N}} V_i(x) = \{x\}$ for any $x \in X$.*

Remark 2.11 There are several well-known definitions of uniform topology which are equivalent to each other. One of them is the definition by uniform neighbourhood systems. The following propositions show that modulus limit also gives the equivalent definition of uniform topology.

Definition 2.12 (Closeness relation) Let LM be the modular limit of X. The relation $x \overset{n}{\to} y$ holds iff there is a sequence $(x_1, x_2, x_3, ...)$ such that $x = x_i$ and $LM(x_1, x_2, x_3, ...) = y$. We call this relation *closeness relation*.

Proposition 2.13 *The followings hold. The function $m(\ )$ is the modulus of diagonal convergence of the modular limit in the followings.*

1. *If $i \leq j$ and $x \overset{j}{\to} y$, then $x \overset{i}{\to} y$.*
2. *If $m(i) \leq j$ and $x \overset{j}{\to} y$, then $y \overset{i}{\to} x$.*
3. *If $m(i) \leq j$, $x \overset{j}{\to} y$ and $y \overset{j}{\to} z$, then $x \overset{i}{\to} z$.*
4. *If $m(m(i)) \leq j$ and $x \overset{j}{\to} z \overset{j}{\leftarrow} y$, then $x \overset{i}{\to} y$.*
5. *If it holds that $x_i \overset{m(i)}{\to} y$ for each i, then $LM(x_1, x_2, x_3, ...) = y$.*

Lemma 2.14 *Let X be a set and LM be a modular limit of modulus m over X. For each $i \in \mathbf{N}$ and each $x \in X$, a subet $V_i(x) \subset X$ is defined as $V_i(x) = \{y \mid y \overset{i}{\to} x\}$ Then, $\{V_i(x)\}_{i,x}$ is a uniform neighbourhood system of modulus m over X, and it makes X a Hausdorff space.*

Moreover, if LM is an effective limit, then $\{V_i(x)\}_{i,x}$ is an effective uniform neighbourhood system.

Lemma 2.15 *Let X be a set and $\{V_i(x)\}_{i,x}$ be a uniform neighbourhood system over X which makes X a Hausdorff space. A partial function $LM : X^\omega \to_p X$ is defined as follows:*

$$dom(LM) = \{(x_1, x_2, ...) \mid \exists y \in X.\ \forall i \in \mathbf{N}.\ x_i \in V_i(y)\}$$
$$LM(x_1, x_2, ...) = y \text{ iff } \forall i \in \mathbf{N}.\ x_i \in V_i(y)$$

Note that such x is uniquely determined, because X is a Hausdorff space by the neighbourhood system $\{V_i(x)\}_{i,x}$.

Then, LM is a modular limit of modulus m over X.

Moreover, if $\{V_i(x)\}_{i,x}$ is an effective uniform neighbourhood system, then LM is an effective limit.

Remark 2.16 The previous lemmata 2.14 and 2.15 mean that modular limit is corresponds to uniform neighbourhood system, and effective limit is corresponds to effective uniform neighbourhood system.

Example 2.17 Let $\mathbf{R}$ be the set of all the real numbers. Define a partial function $LimE : \mathbf{N}^\omega \to_p \mathbf{N}$ as:

$$dom(LimE) = \left\{ (x_1, x_2, x_3, ...) \;\middle|\; |x_i - x_j| < \frac{1}{i} + \frac{1}{j} \right\}$$
$$LimE(x_1, x_2, ...) = x \text{ iff } \lim x_i = x$$

Then, this partial function $LimE$ is an effective limit with the modulus of diagonal convergence $n \mapsto 2n$.

Example 2.18 For $i \in \mathbf{N}$ and $x \in \mathbf{R}$, put $V_i^{\mathbf{R}}(x) \subset \mathbf{R}$ as $y \in V_i^{\mathbf{R}}(x)$ iff $|y - x| < 1/i\}$. Then, $V_i^{\mathbf{R}}(x) = \{y \mid y \stackrel{n}{\to} x\}$ and $\{V_i^{\mathbf{R}}(x)\}_{i,x}$ is an effective uniform neighbourhood system.

Example 2.19 The functional space $\mathbf{N} \to \mathbf{N}$ is regarded as a topological space. Define a partial function $LM_{\mathbf{N},\mathbf{N}} : (\mathbf{N} \to \mathbf{N})^\omega \to_p \mathbf{N} \to \mathbf{N}$ as:

1. $(x_1, x_2, ...) \in dom(LM_{\mathbf{N},\mathbf{N}})$ iff for any $i \leq j < k$, $x_j(i) = x_k(i)$
2. For $(x_1, x_2, ...) \in dom(LM_{\mathbf{N},\mathbf{N}})$, $LM_{\mathbf{N},\mathbf{N}}(x_1, x_2, ...) = (i \mapsto x_i(i))$

The condition 1 means that x_j and x_k have the common initial segment of length j where $j < k$. Then, this $LM_{\mathbf{N},\mathbf{N}}$ is a modular limit with the identity function $i \mapsto i$ as the modulus of diagonal convergence, and it induces the ordinary topology of $\mathbf{N} \to \mathbf{N}$.

3 Algebraic Structure

Definition 3.1 (Algebraic structure) A sequence $X = (|X|, f_1, f_2, f_3, ..., R_1, R_2, R_3, ...)$ is an *algebraic structure* iff it consists of the underlying set $|X|$, partial functions $f_i : |X|^{arity(f_i)} \to_p |X|$, and subsets $R_i \subset |X|^{arity(R_i)}$. An algebraic structure X has finite partial functions or countably many infinite partial functions, and also X has finite relations or countably many infinite relations. For any f_i and any R_i, the arities $arity(f_i)$ and $arity(R_i)$ belong to $0 \cup \mathbf{N} \cup \{\omega\}$. The set of the functions and relations $\{f_1, f_2, f_3, ..., R_1, R_2, R_3, ...\}$ are called the signature of X.

We sometimes identify X to $|X|$, and simply write X for $|X|$.

We may define that an algebraic structure has many sorted, or it refers some other algebraic structures.

Definition 3.2 (Algebraic Structure With Uniform Topology) Let $X = (|X|, f_1, f_2, f_3, ..., R_1, R_2, R_3, ...)$ be an algebraic structure. Suppose that a partial function $f_i = LM$ belongs to the signature, and $f_i = LM$ satisfies the definition of modular limit over the domain $|X|$. Then, this X is an *algebraic structure with uniform topology* by the modular limit LM.

If the modular limit is effective, then it is called an *algebraic structure with effective uniform topology*

Example 3.3 we define an algebraic structure $\mathbf{R} = (|\mathbf{R}|, 0, 1, +, -, \times, /, LimE, <)$ as follows:

The set $|\mathbf{R}|$ is the set of all the real numbers. The constants 0 and 1 are the numbers 0 and 1 themselves. The functions $+$, $-$ and $\times$ are the ordinary

summation, subtraction and multiplication. The partial function / is the ordinary division. The partial function $LimE : \mathbf{R}^\omega \to_p \mathbf{R}$ is defined as in the example 2.17, which is a limit operation with modulus $1/n$. The relation $<$ is the ordinary inequality without equality.

Then, This $\mathbf{R}$ is an algebraic structure with effective uniform topology.

We sometimes identify $\mathbf{R}$ and $|\mathbf{R}|$.

4 Representation

Notation 4.1 We identify a function $f : \mathbf{N} \to \mathbf{N}$ to a infinite sequence $(f(0), f(1), f(2), ...) \in \mathbf{N}^\omega$. For a finite sequence $x \in \mathbf{N}^*$ and a finite or infinite sequence $y \in \mathbf{N}^* \cup \mathbf{N}^\omega$, we write $x \sqsubseteq y$ when x is an initial segment of y. Put $\phi_{\mathbf{N}^*} : \mathbf{N} \to \mathbf{N}^*$ as the standard enumeration of $\mathbf{N}^*$ and $\psi_{\mathbf{N}^*} : \mathbf{N}^* \to \mathbf{N}$ as its inverse.

Definition 4.2 (Recursive Type-2 Function) A partial function of $\mathbf{N}^\omega \to_p \mathbf{N}^\omega$ is called a *partial Type-2 function* or *partial functional*. A partial functional $F : \mathbf{N}^\omega \to_p \mathbf{N}^\omega$ is *recursive* iff there is a partial recursive function $f : \mathbf{N} \to_p \mathbf{N}$ which satisfies the following three conditions. We write $\hat{f}$ for $\phi_{\mathbf{N}^*} \circ f \circ \psi_{\mathbf{N}^*} : \mathbf{N}^* \to \mathbf{N}^*$. Then, the conditions are the following:
1. The function $\hat{f}$ is monotone with respect to $\sqsubseteq$, that is, for any $y \sqsubseteq z \in \mathbf{N}^*$, $\hat{f}(y) \sqsubseteq \hat{f}(z)$.
2. For each $x : \mathbf{N}^\omega$, $x \in dom(F)$ iff there are arbitrary long $\hat{f}(y)$ such that $y \sqsubseteq x$, that is, for every $n \in \mathbf{N}$, there exists $y \sqsubseteq x$ such that $\hat{f}(y)$ is longer than the length n.
3. For $x \in dom(F)$, $F(x)$ is the infinite sequence such that $\hat{f}(y) \sqsubseteq F(x)$ for any $y \sqsubseteq x$.

Remark 4.3 This definition is equivalent to the notion of recursive functions relative to another functions (which is given by Odifretti [O]). And also this is equivalent to the notion defined by Type-2 machines (by Weihrauch [W]).

Definition 4.4 (Representation) A partial function $r : \mathbf{N}^\omega \to_p X$ is a *representation* of X iff it is a surjection, that is, $range(r) = X$.

Definition 4.5 (Pairing) A *pairing function* $\langle -, - \rangle : \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ is defined as

$$\langle m, n \rangle = \frac{(m + n - 1)(m + n - 2)}{2} + m,$$

which is a standard bijection of $\mathbf{N} \times \mathbf{N} \to \mathbf{N}$.

Remark 4.6 With this paring function $\langle -, - \rangle$, we can regard a unary function as a binary function, such as a unary unction $(n \mapsto f(n)) : \mathbf{N} \to \mathbf{N}$ as a binary function $((m, n) \mapsto f(\langle m, n \rangle)) : \mathbf{N}^2 \to \mathbf{N}$. As inverse, we can regard a binary function as a unary function, such as a binary function $((m, n) \mapsto f(m, n)) : \mathbf{N}^2 \to \mathbf{N}$ as a unary function $(\langle m, n \rangle \mapsto f(m, n)) : \mathbf{N} \to \mathbf{N}$.

Definition 4.7 (Recursive function with respect to representations) Let X and Y be sets. Let r and r' be representations of X and of Y respectively. A partial function $f : X \to_p Y$ is *recursive* with respect to r and r' iff there is a Type-2 partial recursive functional $F : \mathbf{N}^\omega \to_p \mathbf{N}^\omega$ such that $f \circ r \subset_p r' \circ \bar{F}$. If $X = Y$ and $r = r'$, then we say simply that the f is recursive with respect to r.

Remark 4.8 We say that $f : X^2 \to_p X$ is recursive with respect to r iff it is recursive with respect to (r', r) where r' is defined as:

$r'(n \mapsto \langle w_1(n), w_2(n)\rangle) = (r(w_1), r(w_2))$ for any $w_1, w_2 \in \mathbf{N}^\omega$.

We say similarly for $f : X^3 \to_p X$, $f : X^4 \to_p X$ and so forth.

We say that $f : X^\omega \to_p X$ is recursive with respect to r iff it is recursive with respect to (r'', r) where r'' is defined as:

$r'(n \mapsto (i \mapsto w(\langle n, i\rangle)) = r(w)$ for any $w \in \mathbf{N}^\omega$.

Definition 4.9 (Reducibility) Let X be a set, and r be a representation of X. A partial function $f : \mathbf{N}^\omega \to_p X$ is *continuously reducible* to r if there is a partial continuous functional $F : \mathbf{N}^\omega \to \mathbf{N}^\omega$ such that $dom(f) = dom(F)$ and $f = r \circ F$. The partial function f is *reducible* to r if such F is recursive.

Definition 4.10 (Admissibility) Let X be a topological space. A representation $r : \mathbf{N}^\omega \to_p X$ is *admissible* if the following conditions hold:
1. (continuity) r is continuous.
2. (finality) Every partial continuous function $q : \mathbf{N}^\omega \to_p X$ is continuously reducible to r.

Remark 4.11 The notion of admissibility is just a topological notion, and does not include none of effectivity.

Lemma 4.12 *Let X be an algebraic structure with uniform topology, and LM be the modular limit of X, Let r be a representation of X. Suppose that LM is computable with respect to r. Then any continuous partial function $f : \mathbf{N}^\omega \to X$ r is continuously reducible to r.*

Theorem 4.13 *If a representation r of X is continuous, and the modular limit of X is recursive with respect to r, then r is admissible.*

5 Computability Structure

Remark 5.1 A binary partial recursive function $f : \mathbf{N} \times \mathbf{N} \to_p \mathbf{N}$ induces a total function of natural numbers into partial recursive functions $(m \mapsto (n \mapsto f(m, n))) : \mathbf{N} \to (\mathbf{N} \to_p \mathbf{N})$. When we say a recursive function $g : \mathbf{N} \to (\mathbf{N} \to_p \mathbf{N})$, we will denote that there is a binary partial recursive function $f : \mathbf{N} \times \mathbf{N} \to_p \mathbf{N}$ such that $g : m \mapsto g(m) = (n \mapsto f(m, n))$.

Definition 5.2 (Term) Let $\{f_1, f_2, ..., R_1, R_2, ...\}$ be a signature and $V = \{v_1, v_2, v_3, ...\}$ be a set of countably infinitely many variables. Then we define the set of formal expressions $Term_\alpha$ which is defined for each ordinal α as follows:

$$
\begin{aligned}
&Term_0 = V \\
&Term_{\alpha+1} = Term_\alpha \\
&\qquad \cup \{f_i\langle t_1, t_2, ..., t_{arity(f_i)}\rangle \mid arity(f_i) \in \{0\} \cup \mathbf{N}, t_j \in Term_\alpha\} \\
&\qquad \cup \{f_i\langle t_1, t_2, t_3, ...\rangle \mid arity(f_i) = \omega, t_j \in Term_\alpha\} \\
&Term_\alpha = \bigcup_{\beta < \alpha} Term_\beta \text{ for a limit ordinal } \alpha
\end{aligned}
$$

It is easy to seen that $Term_\alpha$ is saturated for uncountable ordinals α, because the arity of every function is countable. Thus $Term_\alpha$ is unique for any uncountable ordinal α. We define $Term$ as:

$Term = Term_\alpha$ for the least uncountable ordinal α.

Elements of $Term$ are called *terms*.

Definition 5.3 (Evaluation) Let X be a structure. We define a partial function $[\![-]\!] : Term \times (\mathbf{N} \to_p X) \to_p X$

– For a variable $v_i \in V = Term_0$ and a partial function $\rho : \mathbf{N} \to_p X$,
$[\![v_i]\!]_\rho$ is defined iff so is $\rho(i)$, and $[\![v_i]\!]_\rho = \rho(i)$.

– For a term $f_i\langle t_1, t_2, ...\rangle \in Term_{\alpha+1} - Term_\alpha$ and a partial function $\rho : \mathbf{N} \to_p X$,
$[\![f_i\langle t_1, t_2, ...\rangle]\!]_\rho$ is defined iff $f_i([\![t_1]\!]_\rho, [\![t_2]\!]_\rho, ...)$ is defined
and $[\![f_i\langle t_1, t_2, ...\rangle]\!]_\rho = f_i([\![t_1]\!]_\rho, [\![t_2]\!]_\rho, ...)$

Definition 5.4 (Standard representation of terms) We define a representation r_T of $Term$ with representations r_α of $Term_\alpha$ for any ordinals α.

Let $j_i : \mathbf{N} \to \mathbf{N}$ be a function such that $j_i(n) = 1 + \langle i, n\rangle$. Note that $\bigcup_{i \in \mathbf{N}} j_i(\mathbf{N}) = \{2, 3, 4, ...\}$ and all the summands are disjoint to each other.

The function r_0 of $V = Term_0$ is defined as follows.

For $x : \mathbf{N}^\omega$, $x \in dom(r_0)$ iff $x(1) = 1$

For $x \in dom(r_0)$, $r_0(x) = v_{x(2)}$

The representation $r_{\alpha+1}$ of $Term_{\alpha+1}$ is defined as follows.

For $x : \mathbf{N}^\omega$, $x \in dom(r_0)$ iff either

$x \in dom(r_\alpha)$

or,

$x(1) \neq 1$,
the signature has the function $f_{x(1)-1}$
and $x \circ j_i \in dom(r_\alpha)$ for each $i < arity(f_{x(1)-1})$

For $x \in dom(r_{\alpha+1})$,

if $x \in dom(r_\alpha)$ then $r_{\alpha+1}(x) = r_\alpha(x)$,
otherwise $r_{\alpha+1}(x) = f_{x(1)-1}\langle r_\alpha(x \circ j_1), r_\alpha(x \circ j_2), r_\alpha(x \circ j_3), ...\rangle$

The representation r_α of $Term_\alpha$ for a limit ordinal α is defined as follows.

$dom(r_\alpha) = \bigcup_{\beta < \alpha} dom(r_\beta)$

For $a \in dom(r_\alpha)$,

if $x \in dom(r_0)$ then $r_\alpha(x) = r_0(x) = v_{x(1)}$,
and if $x \in dom(r_{\beta+1}) - dom(r_\beta)$ then $r_\alpha(x) = r_{\beta+1}(x)$

This r_α is saturate as α is uncountable. We define r_T as

$r_T(x) = r_\alpha(x)$ for the least uncountable ordinal α.

Remark 5.5 It is obvious that this $r_\alpha : \mathbf{N}^\omega \to_p Term_\alpha$ is surjective. Thus, r_T is a surjection into $Term$.

Definition 5.6 (Computability Structure) Let X be an algebraic structure. A subset $S \subset \mathbf{N} \to X$ is a *computability structure* over X if S satisfies the followings:
1. (Permutation) For each $s \in S$ and each total recursive function $f : \mathbf{N} \to \mathbf{N}$, it holds that $s \circ f \in S$.
2. (Merging) For any $s, s' \in S$, there is $s'' \in S$ such that $s''(2n) = s(n)$ and $s''(2n+1) = s'(n)$.
3. (Effective sequence of terms) For each $s \in S$ and each recursive function $f : \mathbf{N} \to (\mathbf{N} \to_p \mathbf{N})$, if $f(m) \in dom(r_T)$ and $[\![r_T(f(m))]\!]_{(i \mapsto s(\langle n,i \rangle))}$ is defined for each $m, n \in \mathbf{N}$, then there is $s' \in S$ such that $s'(\langle m, n \rangle) = [\![r_T(f(m))]\!]_{(i \mapsto s(\langle n,i \rangle))}$

Example 5.7 Regard $\mathbf{N}^\omega$ as an algebraic structure with uniform topology $(\mathbf{N}^\omega, LM_{\mathbf{N},\mathbf{N}})$, as in Example 2.19. Put $S_{\mathbf{N}^\omega}$ as

$\{f : \mathbf{N} \to \mathbf{N}^\omega \mid f(i)(j) = g(i,j),\ g$ is a binary total recursive function$\}$.

Then, this $S_{\mathbf{N}^\omega}$ is a computability structure over $\mathbf{N}^\omega$. This $S_{\mathbf{N}^\omega}$ is the standard computability structure of it.

Definition 5.8 (Computable function) Let X and Y be algebraic structure with effective uniform topology, and LM^X and LM^Y be the effective limits of them, respectively. Let S_X and S_Y be computability structures of X and Y, respectively.

Then, a partial function $f : X \to Y$ is *computable* if there exists a total recursive function $m : \mathbf{N}^2 \to \mathbf{N}$ and the following five conditions hold:

1. There is $a \in S_X$ such that $s(\mathbf{N})$ is dense in $dom(f)$.
2. For each $s \in S_X$, there is a partial recursive function $i : \mathbf{N}^2 \to_p \mathbf{N}$ such that for each $n \in \mathbf{N}$, if $s(n) \in dom(f)$, then $LM^X(a(i(n,1)), a(i(n,2)), a(i(n,3)), ...) = s(n)$.
3. $f \circ a \in S_Y$
4. For every $(i_1, i_2, i_3, ...) \in \mathbf{N}^\omega$, if $(a(i_1), a(i_2), a(i_3), ...) = dom(LM^X)$, then there is a sequence $(j_1, j_2, j_3, ...) \in \mathbf{N}^\omega$ such that $m(i_{j_n}, n) \leq j_n$ for any $n \in \mathbf{N}$.
5. if $LM^X(a(i_1), a(i_2), a(i_3), ...) = x$, and $m(i_{j_n}, n) \leq j_n$ for any $n \in \mathbf{N}$, then $LM^Y(f(a(i_{j_1})), f(a(i_{j_2})), f(a(i_{j_3})), ...) = f(x)$.

Proposition 5.9 *Let $f : X \to_p Y$ and $g : Y \to_p Z$ be computable functions with respect to (S_X, S_Y) and (S_Y, S_Z), respectively. If $range(f) \subset dom(g)$, then $g \circ f$ is computable with respect to (S_X, S_Z).*

Notation 5.10 We abbreviate the following condition over a quadruple (X, LM, r, S) as the condition $(*)$:
– The first component X is an algebraic structures with effective uniform topology.
– The second LM is the effective limits of X.
– The third r is a representation of X such that all the partial functions of the signature of X are recursive with respect to r. Thus, S_r is a computable structures over X.
– As for the forth component, $S = S_r$.

Lemma 5.11 *Let (X, LM, r, S_r) be a quadruple which satisfies the condition $(*)$. Then, for each function $f : \mathbf{N}^\omega \to X$, if f is computable with respect to $(S_{\mathbf{N}^\omega}, S_r)$, then f is reducible to r.*

Remark 5.12 This proof follows the steps similar to those in the proof of Lemma 4.12.

Corollary 5.13 *A partial function $f : \mathbf{N}^\omega \to_p \mathbf{N}^\omega$ is recursive if it is computable with respect to $S_{\mathbf{N}^\omega}$.*

Lemma 5.14 *Let f be a partial function of $\mathbf{N}^\omega \to_p \mathbf{N}^\omega$. Suppose that f is recursive and there is $a \in S_{\mathbf{N}^\omega}$ such that $a(\mathbf{N})$ is dense in $dom(f)$. Then, f is computable with respect to $S_{\mathbf{N}^\omega}$.*

Corollary 5.15 *Let f be a partial function of $\mathbf{N}^\omega \to_p \mathbf{N}^\omega$. Suppose that there is $a \in S_{\mathbf{N}^\omega}$ such that $a(\mathbf{N})$ is dense in $dom(f)$. Then, f is recursive iff it is computable with respect to $S_{\mathbf{N}^\omega}$.*

Lemma 5.16 *Let (X, LM^X, r, S_X) be a quadruple which satisfies the condition $(*)$. Let Y be an algebraic structure with effective uniform topology, and LM^Y be the effective limit of it. Let S_Y be a computability structure over Y. Suppose that r is computable with respect to $(S_{\mathbf{N}^\omega}, S_X)$.*

Then, for any $f : X \to_p Y$, if $f \circ r$ is computable with respect to $(S_{\mathbf{N}^\omega}, S_Y)$, then f is computable with respect to (S_X, S_Y).

Theorem 5.17 (Main Theorem) *Let (X, LM^X, r_X, S_X) and (Y, LM^Y, r_Y, S_Y) be quadruples which satisfy the condition $(*)$. Suppose that r_X and r_Y are computable with respect to $(S_{\mathbf{N}^\omega}, S_X)$ and $(S_{\mathbf{N}^\omega}, S_Y)$, respectively.*

Let $f : X \to_p Y$ and $F : \mathbf{N}^\omega \to_p \mathbf{N}^\omega$ be partial functions which satisfy the following:

1. *$f \circ r_X = r_Y \circ F$*
2. *$range(F) \subset dom(R_Y)$*
3. *There is $a \in S_{\mathbf{N}^\omega}$ such that $a(\mathbf{N})$ is dense in $dom(F)$.*

Then, F is recursive iff f is computable with respect to (S_X, S_Y).

6 Computability in Yasugi's sense

Remark 6.1 There is another definition of computable functions, which is given by Yasugi et al. [Y]. Their definition is defined by using uniform neighbourhood system. We call the computability defined by their definition the computability in Yasugi's sense, or Y-computability.

Definition 6.2 (Computable function in Yasugi's sense) Let $X = (|X|, \{V_i^X(x)\}_{i,x}, S_X)$ be a triples which consists of an underlying space X, an effective uniform neighbourhood system $\{V_i^X(x)\}$, and a computability structure S_X. As is usual, we identify $|X|$ to X.

A partial function $f : X \to_p \mathbf{R}$ is *computable in Yasugi's sense*, or *Y-computable*, iff the following hold:

1. For $s \in S_X$, if $s(\mathbf{N}) \in dom(f)$, then $f \circ s \in S_{\mathbf{R}}$.
2. For any $s \in S_X$, there is a total recursive function $j_s : \mathbf{N} \to \mathbf{N}^\omega$ such that $f(V^X_{j_s(i,n)}(x_i)) \subset V^{\mathbf{R}}_n(f(x_i))$ for any i and n.
3. There are $e \in S_X$ and a total recursive function $j_e : \mathbf{N} \to \mathbf{N}^\omega$ such that
   (3.1) The range $e(\mathbf{N})$ is dense in X.
   (3.2) $f(V^X_{j_e(i,n)}(x_i)) \subset V^{\mathbf{R}}_n(f(x_i))$ for any i and n.
   (3.3) $\bigcup_{i \in \mathbf{N}} V^X_{j_e(i,n)}(x_i) = X$.

Lemma 6.3 *Let X be an algebraic structure with effective uniform topology, and LM be the effective limit of it. Put $\{V_i(x)\}$ be the effective uniform neighbourhood system defined as Lemma 2.14. Let S_X be a computable structure over X.*

If a function $f : X \to \mathbf{R}$ is computable, then it is Y-computable.

Lemma 6.4 *Let X be an algebraic structure with effective uniform topology, and LM be the effective limit of it. Put $\{V_i(x)\}$ be the effective uniform neighbourhood system defined as Lemma 2.14. Let S_X be a computable structure over X. Suppose the following set is recursively enumerable for each $s \in S_X$:*

$$\{(i, j, n, n') \in \mathbf{N}^4 \mid V_j(s(n')) \subset V_i(s(n))\}$$

If a function $f : X \to \mathbf{R}$ is Y-computable, then it is computable.

Corollary 6.5 *Under the same assumption as the previous lemma (6.4), a function $f : X \to \mathbf{R}$ is computable iff it is Y-computable.*

Remark 6.6 The algebraic structure $\mathbf{R}$ satisfies the assumption of the previous lemma (6.4). Therefore, for each function of $\mathbf{R} \to \mathbf{R}$, it is computable iff it is Y-computable.

Remark 6.7 This condition appears in the definition of computability by Yasugi et al.:

$$\bigcup_{i \in \mathbf{N}} V^X_{j_e(i,n)}(x_i) = X.$$

The function j_e corresponds to the modulus m in our definition.

In their definition, the condition asserts only the existence of i such that $x \in V^X j_e(i,n)(x_i)$ for x, but not the effectivity of such i. Our definition asserts the effectivity of such i, because our definition asserts the numerical inequality, which is recursively justified.

7 Conclusional remark

The essential definition of this work is Definition 5.8, which defines computable functions over algebraic structures with effective uniform topology. The main theorem (Theorem 5.17) says that the computability in our definition is equivalent to that in the definition by Weihrauch and others, with some suitable assumption. And also, Corollary 6.5 says that our computability is equivalent to that by Yasugi and others, with some suitable assumption.

The author think that this definition 5.8 has some conditions on the domain of the function, which seems inessential. The author would like to make this condition more natural as a future work.

Acknowledgements
The author is grateful to Brattka, Kamo, Mori, Tsuiki and Yasugi for discussions and comments.

References

[H] Hertling, P.: A real number structure that is effectively categorical. *Math. Log. Quart.*, 45:147–182, 1999.

[O] Odifreddi, P. G.: *Classical Recursion Theory.* Studies in logic and the foundations of mathematics, vol. 125, North-Holand, 1989.

[P] Pour-El, M. B. & Rechards, J. I.: *Computability in Analysis and Physics.* Springer-Verlag, 1989.

[S] Schröder, M.: Admissible representation of limit spaces, *Computability and Complexity in Analysis '00*, LNCS 2064, pp273–295, Springer, 2001.

[W] Weihrauch, K.: *Computable Analysis*, Springer-verlag, 2000.

[Y] Yasugi, M., et al.: Some properties of the effective uniform topological space, *Computability and Complexity in Analysis '00*, LNCS 2064, pp336–356, Springer, 2001.

整合的ドメインの極限要素集合の次元について
Dimensions of limit-sets of coherent domains

立木秀樹
Hideki Tsuiki
京都大学総合人間学部
Department of Integrated Human Studies, Kyoto University

1 序

著者は，位相空間の代数的ドメインによる表現を考える中で，整合的 (coherent) ω-代数的ドメインの極限 (non-finite) 要素のなす集合 L(D) の次元が，その順序集合としての高さと一致することを示した [Tsu03]。そして，この結果を可算基を持たない一般の整合的代数的ドメインに拡張することは難しそうだと数研の会議の席で述べた。この結果のための主要な命題は，次のものだった。

命題 1 D が整合的 ω-代数的ドメインであり，$x, y \in D$ の時，
1) $x \uparrow y$ であることと，$e \uparrow f$ が 全ての $e \in K(x)$，$f \in K(y)$ で成立することは同値。
2) $d \in K(D)$ とすると， $\uparrow d$ の閉包は $\downarrow\uparrow d$ である。

記号については後述する。これが，可算基の存在を仮定しなくても成り立つかどうかが問題であった。しかし，この結果がより一般に，整合的連続ドメインについて成り立つことを，Birmingham 大学の Martín Escardó 氏に教えて頂いた。それにより，上記の [Tsu03] の結果の整合的代数的ドメインへの拡張も可能となる。

連続ドメインに拡張するというのは，私はあまり考えていなかった。それは，代数的でない連続ドメイン D では明らかに $L(D)$ は無限次元となって，私の目的には，代数的ドメインまでしか拡張が意味をなさないためである。整合性は，連続ドメインにおいて，位相的に定義される性質であるが，代数的ドメインでは，それが，順序的構造の性質として，簡単に定式化される。それで，もし証明できるのなら，代数的ドメインの順序的構造だけを見ながらでも証明できると考えていた。というか，私が整合性にたどり着いたのは，代数的ドメインの次元的な性質を証明するために必要な順序構造の性質としてであり，それがたまたま，Plotkin による 2/3-SFP 定理によって，整合性とし

て知られた位相的な特徴づけと一致していたということで，私の中では，順序構造の性質がまず第一であった。

Escardó 氏に教えて頂いた上記の命題の連続ドメインに対する証明は，Lawson Topology における簡単な位相的考察からでてくる。この位相的な証明を順序集合の具体的な言葉に翻訳できるかは，まだ考えていない。しかし，順序集合的な具体的手続きからの構成が簡単ではない事柄が位相空間論的に簡単に導かれるのは，驚きであった。このことから，ドメイン理論において，位相は，“ 順序を位相空間としても解釈できる” といった付随的なものではなく，より中心的な考察対象だということが分かる。

本論では，連続ドメイン上の Compactly Saturated Set や Lawson 位相などの概念の紹介も兼ねて，M. Escardó 氏に教えて頂いた上記の結果について紹介し，さらに，それが，整合的連続ドメインの極限要素のなす集合の次元について導く結果について述べる。ドメインの位相的な性質については，[AJ94] などを参照されたい。

2 代数的ドメイン

部分順序集合 $(D \leq)$ であり，最小元 $\perp$ が存在し，全ての有向集合が上限を持つものを，dcpo (directed complete partial order) という。有向集合 A の上限を $\sqcup A$ と書く。dcpo D の中で，$y \leq \sqcup A$ ならば，$x \leq a$ がある $a \in A$ に対してなりたつ時，$x \ll y$ と書いて，x は y を近似するという。x が x を近似するとき，x は有限という。$\downarrow x = \{y \in D \mid y < x\}$ とする。$\Downarrow x = \{y \in D \mid y \ll x\}$ とする。同様に，$\uparrow x$ と $\Uparrow x$ を定義する。$\{x, y\}$ が上界を持つとき，$x \uparrow y$ と書くことにする。

D の有限な要素のなす集合を $K(D)$, D の極限要素 (すなわち，有限でない要素) のなす集合を $L(D)$ と書くことにする。$\downarrow x \cap K(D)$ のことを，$K(x)$ と書き，x の近似のなす集合と呼ぶことにする。D の部分集合 B が，D の任意の元 x に対し，$B \cap \Downarrow x$ が有向集合となり $x = \sqcup(B \cap \Downarrow x)$ を成り立たせるとき，B のことを基底と呼ぶ。$K(D)$ が D の基底になるとき，すなわち，D の全ての元 x に対し，$K(x)$ が有向集合となり，$\sqcup K(x) = x$ が成り立つ時，D を代数的ドメインとよぶ。

それに対し，$K(D)$ は基底にはならないかもしれないが，より大きな集合をとれば基底となるとき，D のことを連続ドメインという。B が基底なら，B より大きな集合は基底となるので，これは，D 全体が基底となることと同値である。それは，言い換えると，D の全ての元 x に対し，$\Downarrow x$ が有向集合となり，$\sqcup \Downarrow x = x$ と同値である。

連続ドメイン D において，任意の $K(D)$ の有限部分集合 A に対し，A の上界のなす集合 $up(A)$ が有限個の極小元をもち（その集合を S とする）任意の $x \in up(A)$ に対し，ある $y \in S$ で $y \leq x$ であるものが存在するとき，

D は性質 M を持つという。これは，SFP (あるいは bifinite) domain の条件の中の最初の 2 つと一致している。後で述べるように，2/3-SFP 定理により，代数的ドメインでは，性質 M と整合性が同値であるため，[Tsu03] では，代数的ドメインでこの性質を持つものを整合的と呼んでいた。

3 上記命題の順序的証明

まず，[Tsu03] において行われた，命題 1 の順序的な証明を考える。

1) x, y をそれぞれ上限とする $K(D)$ の元の上昇鎖 $d_0 = \bot < d_1 < \ldots$, $e_0 = \bot < e_1 < \ldots$ をとる。仮定より，d_i と e_i は上界をもつ。それを，f_i とする。f_i が $f_0 < f_1 < \ldots$ を満たせば，その極限は，x と y の上界となり，目的が達せられるが，一般にそれは成り立たない。そこで，新たに $f_i \leq g_i, e_i \leq g_i$ を満たす上昇鎖 $g_0 < g_1 < \ldots$ であって，g_i より大きな f_j $(j > i)$ の集合 S_i が無限集合となる様なものを帰納的に構成する。$g_0 = \bot$ とする。g_i まで構成されたとする。S_i の元はすべて，$\{g_i, e_{i+1}, f_{i+1}\}$ の上界である。よって，性質 M により，$\{g_i, e_{i+1}, f_{i+1}\}$ の極小上界の集合は有限集合であり，S_i のそれぞれの要素は，この中のどれかの要素よりも大きい。S_i は無限集合より，その有限集合のある要素で，それより大きな S_i の要素が無限個存在するものがある。それを，g_{i+1} とおく。すると，g_{i+1} は条件を満たす。

2) x が $\uparrow d$ の閉包に入っているとはすなわち，x を含む全ての開集合が $\uparrow d$ と交わるということである。それはまた，$K(x)$ の全ての元 e に対して，$\uparrow e$ と $\uparrow d$ が交わること，すなわち，$e \uparrow d$ となることを意味する。よって，(1) より，$e \uparrow x$ と同値であり，これは，言い方を変えれば，$x \in \downarrow\uparrow d$ ということである。

この命題の 2 番を用いて，次の定理が証明される。詳細は [Tsu03] を見られたい。

定理 2 *D が整合的 ω-代数的ドメインの時，$L(D)$ の Scott 位相による位相空間の次元は，$L(D)$ の順序集合としての高さと等しい。*

ここで，位相空間の次元は，small inductive dimension のことである。

系 3 *可算な文字集合 Σ に対し，$\bot$ の出現を n 個まで許す無限文字列全体の集合 $\Sigma^{\omega}_{\bot,n}$ の次元は n である。*

これから，位相空間の $\Sigma^{\omega}_{\bot,n}$ の中での表現を考えたときに，n より次元の大きな空間は $\Sigma^{\omega}_{\bot,n}$ での表現ができないことがわかる。

4 Compact saturated set と Lawson 位相

この章では，[AJ94] に従い，整合的ドメインについて解説する。証明などは [AJ94, Smy92] などを参照されたい。

連続ドメイン D には，Scott 位相 (σ_D) という位相が入ることはよく知られている。$\Uparrow x$ $(x \in D)$ は開集合であり，これらがこの位相のベースとなっている。この位相を補う概念として，compact saturated set がある。compact saturated set は，Scott 位相によって，コンパクトであり，saturated（近傍の積集合となる）な集合である。連続ドメインにおいて， saturated な集合は上に閉じている $(\uparrow S = S)$ というのと同値であるので，これは，上に閉じたコンパクト集合ということである。compact saturated set に対して，その開近傍全体は， lattice σ_D 上の Scott 開フィルターとなる。さらに，逆もいえる。

定理 4 (Hofmann-Mislove Theorem) *D が連続ドメインの時，compact saturated set の集合と，lattice σ_D 上の Scott 開フィルターの集合と一対一の対応がある。*

連続ドメイン D で 2つの compact saturated set の積集合がコンパクトになる（よって，再び compact saturated set となる）時，D を整合的ドメインとよぶ。

命題 5 *連続ドメイン D が整合的であることと，全ての Scott 開集合 O, U_1, U_2 で $O \ll U_1$, $O \ll U_2$ なものに対し，$O \ll U_1 \cap U_2$ となることは同値である。*

D 上に入る Scott 位相より強い位相として，Lawson 位相がある。それは，Scott 開集合に加えて $D \setminus \uparrow x$ という形の集合も開集合として定義される位相である。Scott open が positive な情報を意味しているとすると，$D \setminus \uparrow x$ は，negative な情報も意味していることになる。特に，Lawson 位相は Hausdorff である。

命題 6 *連続ドメイン D が整合的であることと，D 上の Lawson 位相がコンパクトであることは同値である。*

次章で次の命題を用いる。

命題 7 *連続ドメインでは，Lawson コンパクトな集合に対し，その lower set は Scott-閉集合である ([AJ94] の Lemma 6.2.20)。*

さらに，次のことがいえる。

定理 8 (Plotkin の 2/3-SFP 定理) *代数的ドメインにおいては，性質 M をもつことと整合的であることは同値である。*

5　対応する命題の位相的証明

最初の命題に対応する，連続ドメインに関する命題は以下の様になる。

命題 9 *D が整合的連続ドメインの時,*
1) $x, y \in D$ に対し，$x \uparrow y$ であることと，$e \uparrow f$ が 全ての $e \ll x$，$f \ll y$ で成立することは同値。
2) $d \in K(D)$ に対し，$\uparrow d$ の閉包は $\downarrow\uparrow d$ である。

(1) の証明 ([Esc98])。compact saturated set $\uparrow x, \uparrow y$ を考える。$e \ll x$ に対し，$\Uparrow e$ 全体がベースとして構成するフィルターを F_x とする。$\cap F_x = \uparrow x$ となる。同様にフィルター F_y を定義する。

$$H = \{U \cap V \mid U \in F_x, V \in F_y\}.$$

とおく。H は，filter となる。さらに，$U_1 \ll U_2 \in F_x$, $V_1 \ll V_2 \in F_y$ に対し，$U_1 \cap V_1 \ll U_2, U_1 \cap V_1 \ll V_2$ であることから，D が整合的であることより，$U_1 \cap V_1 \ll U_2 \cap V_2 \in F_y$ が成り立つ。これはすなわち，このフィルターが Scott 開フィルターであることを意味している。よって，Hofmann-Mislove Theorem より，$\cap H$ は compact saturated set となる。このフィルタは開集合全体のなすフィルターではないので，$\cap H$ は空集合ではない。そして，その要素はどれも x および y よりも大きい。

(2) (1) を用いずに，直接示す。$\uparrow d$ は, D の retract であり，コンパクトの連続像より Lawson コンパクトである。そして，命題 7 により，その lower set は Scott-閉集合である。

6　次元の結果の拡張

さて，定理 2 について考える。命題 9 (2) の性質を満たす代数的ドメインのことを，ここでは，up-down-閉包なドメインと呼ぶことにする。up-down-閉包なドメインが全て定理 2 の結果を満たす訳ではない。次元は，それぞれの開基の要素の境界の次元が n-1 ということで帰納的に示されるので，境界がまた up-down-閉包なドメインとなる必要があるのだが，D が up-down-閉包なドメインであっても，それぞれの $\uparrow d, d \in K(D)$ の境界が up-down-閉包なドメインとなる保証はない（反例もつくれる）。しかし，整合的ドメインにおいて $\uparrow d$ の境界をとっても整合的ドメインとなるので，整合的ドメインにおいては，それぞれの $\uparrow d$ の境界をとっても up-down-閉包なドメインとなることが言える。よって，定理 2 が整合的代数的ドメイン一般でいえる。

定理 10 *D が整合的代数的ドメインの時，$L(D)$ の Scott 位相による位相空間の次元は，$L(D)$ の順序集合としての高さと等しい。*

代数的ドメインでない連続ドメインにおいては，$L(D)$ の順序集合としての高さは明らかに無限になり，その場合，$L(D)$ の次元も無限となるので，この結果は，D が代数的ドメインの時にも成り立つ。

系 11 任意濃度の文字集合 Σ に対しても，$\perp$の出現を n 個まで許す無限文字列全体の集合 $\Sigma^{\omega}_{\perp,n}$ の次元は n である。

謝辞 Alex Shimpson 氏，および，Martín Escardó 氏に，有益な議論ができたこと，様々なことを教えて頂いたことを感謝します。

References

[AJ94] Samson Abramsky and Achim Jung. Domain theory. In S. Abramsky, D. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science Volume 3*, pages 1–168. Oxford University Press, 1994.

[Esc98] Martin H. Escardó. Properly injective spaces and function spaces. *Topology and Its Applications*, 89(1–2):75–120, 1998.

[Smy92] M. B. Smyth. Topology. In S. Abramsky, D. M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science, volume 1*, pages 641–761. Clarendon Press, Oxford, 1992.

[Tsu03] Hideki Tsuiki. Compact metric spaces as minimal-limit sets in domains of bottomed sequences. *Mathematical Structures in Computer Science*, 2003. to appear.

メレオトポロジーと計算
Mereotopology and Computation

三好博之
Hiroyuki MIYOSHI
京都産業大学 理学部 計算機科学科
Dept of Computer Science
Kyoto Sangyo University

Extended Abstract for the 13th ALGI (Dec 16–19, 2002)

1 メレオロジーとメレオトポロジー

メレオトポロジーとは部分 - 全体関係に関する一階の理論であるメレオロジーを，ユークリッド空間が満たすようなある程度強いトポロジーの公理を満たすように拡張したものである．

形式的なメレオロジーには二つの起源があり，一つは 1920 年代から 30 年代にかけて Lesniewski によって行われた研究である [Si87]．彼の目的は集合論によらない唯名論的な数学の基礎付けであり，Prototheticと名づけた論理の上に Ontology および Mereology という理論を構築して，通常の数学が可能になるような体系を作り上げた[1]．彼の Mereology では部分 - 全体関係を表す述語に関する最小限の公理からなる体系により集合論的議論が可能な体系の構築を試みている．もう一つの起源は [LG40] から始まる個体計算（The Calculus of Individuals）である．彼らの動機は数学よりもむしろ哲学であるがやはり唯名論的立場に基づいている．Goodman はこの体系を発展させて芸術をはじめとする様々な分野に適用している [Go76]．しかしこの二つの元の体系は大まかには一階述語論理の理論であり，体系としては類似したものであった．

最近の注目すべきアプローチとしては B. Smith を中心としたグループによるものが挙げられる [Sm96]．彼らの研究は Husserl のフォーマルオントロジーを中心とした前期現象学や Gibson による生態学的心理学に動機付けられたものであり，部分 - 全体関係をトポロジカルな接続（connection）や接触（contact）といった関係と区別した上で，これら相互の関係を明確にすると

[1] Prototheticについては A. Church“ Introduction to Mathematical Logic ”(1944, 1956) に簡単な解説がある．

いうアプローチをとっている．このために考えられたのがメレオトポロジーである．これらの研究は，生態学，地政学，地図情報システム，知識工学，といった様々な分野に積極的に応用が試みられている．

2　メレオロジーの体系

次にメレオロジーの形式的体系について概説してみよう．ここでは [CV99] の形式化を取り上げる．メレオロジーは部分 - 全体関係を表す述語 P についての公理からなる．P を用いて述語 O, U, PP が定義される．

Ground Mereology (M) = (P.1)+(P.2)+(P.3)
(P.1) Pxx　(Reflexivity)
(P.2) Pxy ∧ Pyx → x=y　(Antisymmmetry)
(P.3) Pxy ∧ Pyz → Pxz　(Transitivity)

Oxy ≡ ∃ z(Pzx ∧ Pzy)　(Overlap)
Uxy ≡ ∃ z(Pxz ∧ Pyz)　(Underlap)
PPxy ≡ Pxy ∧ ¬ Pyx　(Proper Part)

等号を定義により導入したい場合は以下のようにする．

x=y ⟷ Pxy ∧ Pyx
(A.I) x=y → (ϕ x ⟷ ϕ y)　(Axiom of Identity)
(ϕ: any formula)

さらに公理を追加することにより様々なメレオロジーを構成する．

Minimal Mereology (MM) = (M)+(P.4)
(P.4) PPxy → ∃ z(Pzy ∧ ¬ Ozx)
真の部分 x があれば，x とオーバーラップしない別の部分もある
(Weak Supplimentation)

Extensional Mereology (EM) = (M)+(P.5)
(P.5) ¬ Pyx → ∃ z(Pzy ∧ ¬ Ozx)
x の部分でないものは x とオーバーラップしない部分を持つ．
(Strong Supplementation)

Extensional Mereology については次の外延性が成り立つ．

命題 1 (外延性)

(∃ zPPzx ∨ ∃ zPPzy) → (∀ z(PPzx ⟷ PPzy) → x=y)

さらに閉包性として和と積の公理を導入する．

Closure Mereology (CM) = (M)+(P.6)+(P.7)
(P.6) $Uxy \rightarrow \exists z \forall w(Owz \longleftrightarrow (Owx \vee Owy))$ (Sum)
(P.7) $Oxy \rightarrow \exists z \forall w(Pwz \longleftrightarrow (Pwx \wedge Pwy))$ (Product)

Minimal Closure Mereology (CMM) = (MM)+(P.6)+(P.7)
Extensional Closure Mereology (CEM) = (EM)+(P.6)+(P.7)

このとき外延性から一意性が得られるので和と積が定義できる.

$x \cup y \equiv \iota z \forall w((Owz \longleftrightarrow (Owx \vee Owy)))$ (Sum op)
$x \cap y \equiv \iota z \forall w((Pwz \longleftrightarrow (Pwx \wedge Pwy)))$ (Product op)
(ι : the description operator)

さらに以下の融合公理を導入する.

General Mereology (GM) = (M)+(P.8)
(P.8) $\exists x \phi \rightarrow \exists z \forall y(Oyz \longleftrightarrow \exists x(\phi \wedge Oyx))$ (Fusion)
(ϕ: any formula)
General Extensional Mereology (GEM) = (MM)+(P.8) = (EM)+(P.8)

GEM では以下の演算が定義できる.

$\sigma x \phi \equiv \iota z \forall y(Oyz \longleftrightarrow \exists x(\phi \wedge Oyx))$ (general sum)
$\pi x \phi \equiv \sigma z \forall x(\phi \rightarrow Pzx)$ (general product)
(ϕ: any formula)

このとき積と和の演算はσを用いて再定義することができる.

$x \cup y \equiv \sigma x(Pzx \vee Pzy)$
$x \cap y \equiv \sigma x(Pzx \wedge Pzy)$

3 メレオトオポロジーの体系

メレオロジーでトポロジカルな推論を行うためには, まず基本的な述語として C を導入する. Cxy は x は y とつながっていることを表す. そしてこれを P と関係付ける.

Ground Topology (T) = (C.1)+(C.2)
(C.1) Cxx (Reflexivity)
(C.2) $Cxy \rightarrow Cyx$ (Symmmetry)

$x \subseteq y \equiv \forall z(Czx \rightarrow Czy)$ (x is enclosed in y)

Ground Mereotopology (MT) = (M)+(T)+(C.3)
(C.3) $Pxy \rightarrow x \subseteq y$ (Monotonicity)

次に C や⊑を用いて開，閉などのトポロジカルな概念に対応する述語，および閉包 cl(x) やコンプリメント $(-)^c$ などの演算を定義する．そしてこれらが位相的閉包に関する標準的な Kuratowski 公理系の以下のような類似物を満足するように公理を追加することによりトポロジカルな推論を可能にする．

```
(K.1) P(x)(cl(x))                          (Inclusion)
(K.2) cl(cl(x)) = cl(x)                    (Idempotence)
(K.3) cl(x ∪ y) = cl(x) ∪ cl(y)            (Additivity)
```

4　計算的観点からの形式化

メレオロジーやメレオトポロジーでは，たとえば紛争中の国境といったような，幾何学的対象が決定されていないという事態そのものは取り扱うことができない．このような事態について考えるためには，それらを結果が定まるとは限らず（非停止性）未知の外部からの影響があり得る（開放性）ようなものとして表現する必要がある．このような性質は幾何学的対象を計算と見なすことによりある程度うまく取り扱えると考えられる．そこでここではメレオロジーをカルキュラスとして再構成することを提案したい．

こういったことを考えるにあたってはこのような事態を扱うときの形式化の役割についてよく考える必要がある．ここで必要とされる形式化とは，世界における現象の様々な（しかしすべてではない）側面を理解するためのものである．すると，論理と計算という二つの形式化は，その理解にどのような形で寄与しているのかが問題になる．通常，論理の内部ではその論理における証明の様々な側面は推論されない．例えば，停止しない証明過程や，証明の途中における予期せぬ意味論の変更，証明の途中結果に基づく論理の修正といったことは，論理としてはうまく扱うことができない．しかし計算という観点からはリアクティブな計算，外界とのインタラクション，計算リフレクション，といったことと関連付けることができる．こういったことから現象をカルキュラスとして形式化することは完全とはいえないまでもこういった側面を扱うための有力な手段であると考えられる．

カルキュラスの果たす役割については次の Wittgenstein の挙げた例がひとつの説明を与えてくれる．

> 誰かが我々に和を求める問題を，例えば |||||||||| + |||||||||| のように棒の表記で出して，我々が計算している間，我々が気づかないうちに棒を取り除いたり加えたりして面白がっている，と想像してみよう．彼は「でもその和は正しくない」と言い続け，我々は全部やり直し続けて毎回馬鹿にされるだろう．— 実際，厳密に言えば我々は計算の正しさの基準について何の概念も持っていない．([Wi69] IV 18, p.330)

これは実際の計算が行われるという場面で常にさらされる開放性を端的な形で表している．棒は予期せぬ人物がそっとそれらを加えたり取り去ったりすることを可能にしている．カルキュラスはこの棒の役割を演ずると考えることができる．

5　ソリッドモデルからのドメインの構成

それではそのカルキュラスの意味論には何が必要だろうか．ここで我々は意味論において二つの意味を扱うことになる．すなわち現象の異なる二つの側面に対応するメレオトポロジカルな意味と計算的な意味であり，これらは例えば Euclid 空間と Scott ドメインというように二つのトポロジーとして表現することが考えられる．そこでこれらを同時に取り扱える意味領域として，Edalat と Lieutier による位相空間からのソリッドドメインの構成を採用することにする．これは以下のように定義される．

定義 1 位相空間 X のソリッドドメイン $(SX, <)$ とは X の互いに素な部分集合の対 (A, B)（パーシャルソリッド）からなる集合に $(A_1, B_1) < (A_2, B_2) \Leftrightarrow A_1 \subset A_2 \wedge B_1 \subset B_2$ で定義される情報順序を入れたものである．このとき SX は有向完備半順序（*dcpo*）になる．また SX のサブドメイン SbX は $SbX = \{(A, B) \in SX \mid B^c \ \textit{is compact}\} \cup \{(\emptyset, \emptyset)\}$ に包含による順序を入れたものとして定義される．これは SX の部分 *dcpo* になる．

以下 SX に関連する事柄については証明なしに事実のみを述べる．詳しくは [EL02] を参照して欲しい．

6　メレオトポロジカルな対象のカルキュラス

ここで定義するカルキュラスは PCF (Programming language for Computable Functions) をベースとする．PCF とはドメインの論理である LCF (Logic for Computable Functions) をカルキュラスとして用いるために Plotkin により定義されたものであり [Pl77]，ドメイン理論に基づいた意味論を持っている．論理演算は真偽値関数として扱われる．そしてそこにメレオトポロジカルなデータ型とそれについての演算 $\sqcup, \sqcap, \sqsubseteq$ を加える．ここでは意味論的な制約により複雑になることを避けるため，有界なパーシャルソリッドのみを扱うことにする．(有界でない場合については後述する)．

```
Types
        T ::=   bool            (booleans)
          |     nat             (natural numbers)
          |     bsol            (bounded partial solids)
```

```
          |     T1 → T2          (function type)
Terms
        t ::=   x_T              (variables)
          |     c_T              (constants)
          |     λ t1:T.t2        (abstraction)
          |     (t1)(t2)         (application)
          |     undef_T          (undefined)
```

型付け規則は通常通りなので省略する.

```
Constant symbols
        tt : bool
        ff : bool
        0 : nat
        succ : nat → nat
        zero? : nat → bool
        cond_T : bool → T → T → T    (T: ground type)
        fix_T : (T →T) → T
        ⊔ : bsol → bsol → bsol
        ⊓ : bsol → bsol → bsol
        ⊑ : bsol → bsol → bool
```

実際にはカルキュラスを適用する問題領域に応じてさらにソリッド定数や定数関数を加えることを想定している.

7　パーシャルソリッドについての演算

SX ではパーシャルソリッドについての連続な演算や述語が定義できるので，これにより ⊔, ⊓, ⊑ に表示的意味論を与える．ここで ⊑ に対応する部分ソリッド関係は有界なパーシャルソリッドと（有界とは限らない）パーシャルソリッドの間にのみ定義される．この述語は X がハウスドルフ空間であれば連続となる．メレオトポロジカルな演算についての解釈は以下のようになる.

```
        [bool] = {bottom, true, false}   (flat domain)
        (A,B)[⊔](C,D) = (A ∪ C, B ∩ D)
        (A,B)[⊓](C,D) = (A ∩ C, B ∪ D)
        (A,B)[⊑](C,D) = true     if B ∪ C = X
                        false    if A ∩ D ≠ ∅
                        bottom   otherwise
```

また P（部分 - 全体関係）は ⊑ により定義される.

```
P: bsol → bsol → bool
P ≡ λ x. λ y.(x⊑y)
```

さらに別のメレオトポロジカルな記号を用いる際には，公理に従って定義と意味論を与えることになる．

なおここではメンバーシップ関係については考えなかった．これはそもそもメレオロジーが集合論的なメンバーシップ関係を原始的な述語としないからであるが，実用上はそれに類似した演算があると便利であることも多い．ソリッドドメインではこのような演算に意味を与えることもできる．よってさらに要素に相当する型とメンバーシップ記号を導入して意味を与えることが考えられる．ここでその要素の意味領域をどのようにとるかについては適用する問題との兼ね合いで考慮すべき点があるが，X として R^n を考える場合にはインターバルドメイン IR^n で十分であることが多い．これは $\{\prod_{1=i}^{n}[a_i, b_i] \subset R^n \mid a_i, b_i \in R,\, a_i \leq b_i\} \cup \{R^n\}$ に逆包含順序を入れたものである．このとき要素 $\prod_{i=1}^{n}[a_i, b_i]$ がすべての i について $a_i = b_i$ であれば X の一点に対応する．

8 今後の課題

ここでは有界なパーシャルソリッドに限って議論してきた．有界でないパーシャルソリッドを導入するにはそれに対応する型 `sol`，および部分型の関係 `bsol < sol` が必要となる．よってカルキュラスに何らかの形で多相型もしくは型強制を導入する必要がある．このようなカルキュラスはオブジェクト指向プログラミングとの関係において様々な研究がなされてきたのでその成果を利用することが考えられる [Ts94]．

また幾何学的情報の取り扱いがこれで十分かどうかという問題もある．例えば WWW のような世界から得られる情報は十分に構造化されておらず，部分的であり，矛盾を含んでいることもある．このような中から得られる幾何学的情報をどのように扱うのが適切であるかについてはまだ検討の余地が大きい．これは知識工学におけるフォーマルオントロジーの研究 [FOIS] や半構造化データの研究 [Ca00] との関連が考えられる．

さらに理論的な側面では，Edalat のドメインの構成のエッセンスについて考察するために Synthetic Domain Theory (SDT) を用いてソリッドドメインを公理化することが考えられる．このとき Reus と Streicher による SDT の構文的アプローチ [RS99] との関連は興味深い問題である．また幾何学的対象の別の数学的表現を考えるにあたって Heijmans による数学的モルフォロジー [He94] との関連も考察してみたい．そこでは幾何学的対象を確定してゆく過程を束構造の随伴関係により表現しており，計算とカテゴリー論的極限との類似を持ち込むことにより抽象的に計算として解釈することができると考えられる．

哲学的な応用としては，必ずしも人間に限定されない認識論であるアンドロイド認識論 [FGH95] とのかかわりが考えられる．またメレオロジーの研究から生態学やアフォーダンスとの関係も派生してくると考えられる [SV99]．そもそも本稿のような話題を取り上げる背景には，計算についての存在論および認識論的な考察がある．これについては哲学者との共同プロジェクト `http://philcomp.org/`を参照されたい．

参考文献

[FOIS] *FOIS — International Conference on Fromal Ontology in Information Systems* (`http://www.fois.org/`).

[CV99] R. Casati and A. C. Varzi, *Parts and Places. The Structures of Spatial Representation*, Cambridge, MA, and London: MIT Press [Bradford Books], 1999.

[Ca00] L. Cardelli, "Semistructured Computation", in: R. C. H. Connor and A. O. Mendelzon (Eds.): *Research Issues in Structured and Semistructured Database Programming*, Lecture Notes in Computer Science 1949. Springer, 2000. pp.1–16.

[EL02] A. Edalat and A. Lieutier, "Foundation of a computable solid modelling", *Theoretical Computer Science* 284(2): 319-345 (2002).

[FGH95] K. M. Ford, C. Glymour and P. J. Hayes (Eds.), *Android Epistemology*, AAAI Press, 1995.

[Go76] N. Goodman, *Languages of Art: An Approach to a Theory of Symbols*. Indianapolis and Cambridge: Hackett Publishing, 1976.

[He94] H. J. A. M. Heijmans, *Morphological Image Operators*, Academic Press, 1994.

[LG40] H. S. Leonard and N. Goodman, "The Calculus of Individuals and Its Uses", *Journal of Symbolic Logic*, 5: 45–55 (1940).

[Pl77] G. D. Plotkin, "LCF Considered as a Programming Language". *Theoretical Computer Science* 5(3): 225–255 (1977).

[RS99] B. Reus, T. Streicher, "General Synthetic Domain Theory — A Logical Approach", *Mathematical Structures in Computer Science* 9:177–223. Cambridge University Press, 1999.

[Si87] P. Simons, *Parts: A Study in Ontology*, Oxford University Press, 1987.

[Sm96] Barry Smith, "Mereotopology: A Theory of Parts and Boundaries", *Data and Knowledge Engineering*, Vol.20, 1996, pp.287–303.

[SV99] B. Smith and A. Varzi, "The Niche", *Noûs*, 33:2 (1999), 214–238.

[Ts94] H. Tsuiki, "A Normalizing Calculus with Overloading and Subtyping", *TACS* 1994: 273-295.

[Wi69] L. Wittgenstein, *Philosophische Grammatik*, Blackwell, Oxford, 1969. Edited by R. Rhees.

Structural Induction and the λ-Calculus

René Vestergaard*

School of Information Science
Japan Advanced Institute of Science and Technology

Abstract. We consider formal provability with structural induction and related proof principles in the λ-calculus presented with first-order abstract syntax over one-sorted variable names. As well as summarising and elaborating on earlier, formally verified proofs (in Isabelle/HOL) of the relative renaming-freeness of β-residual theory and β-confluence, we also present proofs of η-confluence, βη-confluence, the strong weakly-finite β-development (aka residual-completion) property, residual β-confluence, η-over-β-postponement, and notably β-standardisation. In the latter case, the known proofs fail in instructive ways. Interestingly, our uniform proof methodology, which has relevance beyond the λ-calculus, properly contains pen-and-paper proof practices in a precise sense. The proof methodology also makes precise what is the full algebraic proof burden of the considered results, which we, moreover, appear to be the first to resolve.

1 Introduction

The use of structural induction and related proof principles for simple syntax (i.e., first-order abstract syntax over one-sorted variable names) is a long-standing and widely-used practice in the programming-language theory community. Unfortunately, at a first, closer inspection it seems that the practice is not formally justifiable because of a need to avoid undue variable capture when performing substitution, thus breaking the syntactic equality underlying structural induction, etc.. Even more worrying is the fact that, in spite of substantial efforts in the mechanised theorem-proving community, no formal proof developments (prior to what we report on here) have been able to overcome the problems that are encountered with substitution and go on to successfully employ the proof principles in question. Indeed, and starting with de Bruijn [6], it has become an active research area to define, formalise, and automate alternative syntactic frameworks that, on the one hand, preserve as much of the inherent naturality of simple syntax as possible. At the same time, they are customised to provide suitable induction and recursion principles for any considered language [6–10, 12, 17, 21]. However, by changing the underlying syntactic framework, the algebraic meaning of, e.g., a diamond property also changes, which means that, e.g., confluence as proved and as defined no longer coincide, cf. Lemma 18 and [25].

In the recognition that the above is both unfortunate as far as the formal status of the existing informal literature is concerned and unsatisfactory from a mathematical perspective, we pursue the naive approach in this article (while incorporating the relevant aspects of [24, 25]). In particular, we show that it is, indeed, possible to base formal proofs on first-order abstract syntax over one-sorted variable names and hope to convince the reader that, while the technical gap between pen-and-paper and formal proofs is rather large, the conceptual gap is somewhat smaller. Furthermore, we hope that the comprehensive range of applications of the proof methodology that we present here will establish its wider relevance.

1.1 Syntax of the λ-Calculus

The λ-calculus is intended to capture the concept of a *function*. It does so, first of all, by providing syntax that can be used to express function application and definition:

$$e ::= x \mid e_1 e_2 \mid \lambda x.e$$

The above, informal syntax says that a λ-term, e, is defined inductively as either a variable name, as an application of one term to another, or as a λ-, or functional, abstraction of a variable name over a term. The variable names, x, are typically taken to be, or range over, words over the Latin alphabet. In Section 2, we will review the exact requirements to variable names in an abstract sense. Being based on a simple, inductive definition, λ-terms also come equipped with a range of primitive proof principles [1, 3].

Syntactic Equality As a λ-term, e, is finite and consists of variable names, the obvious variable-name equality, $=_{\mathcal{VN}}$, which exists at least in the case of words over the Latin alphabet, canonically extends to all λ-terms:

$$\frac{x =_{\mathcal{VN}} y}{x =_{\Lambda^{\mathrm{var}}} y} \qquad \frac{e_1 =_{\Lambda^{\mathrm{var}}} e_1' \;\; e_2 =_{\Lambda^{\mathrm{var}}} e_2'}{e_1 e_2 =_{\Lambda^{\mathrm{var}}} e_1' e_2'} \qquad \frac{x =_{\mathcal{VN}} y \;\; e =_{\Lambda^{\mathrm{var}}} e'}{\lambda x.e =_{\Lambda^{\mathrm{var}}} \lambda y.e'}$$

Structural Induction In order to prove properties about the set of λ-terms, we can proceed by means of *structural induction*, mimicking the inductive definition of the terms:

$$\frac{\forall x.P(x) \;\; \forall e_1, e_2.P(e_1) \wedge P(e_2) \Rightarrow P(e_1 e_2) \;\; \forall x, e.P(e) \Rightarrow P(\lambda x.e)}{\forall e.P(e)}$$

* vester@jaist.ac.jp, http://www.jaist.ac.jp/~vester

$$
\begin{aligned}
y[x := e]_{\mathrm{Cu}} &= \begin{cases} e & \text{if } x = y \\ y & \text{otherwise} \end{cases} \\
(e_1 e_2)[x := e]_{\mathrm{Cu}} &= e_1[x := e]_{\mathrm{Cu}} e_2[x := e]_{\mathrm{Cu}} \\
(\lambda y.e_0)[x := e]_{\mathrm{Cu}} &= \begin{cases} \lambda y.e_0 & \text{if } x = y \\ \lambda y.e_0[x := e]_{\mathrm{Cu}} & \text{if } x \neq y \ \wedge (y \notin \mathrm{FV}(e) \ \vee \ x \notin \mathrm{FV}(e_0)) \\ \lambda z.e_0[y := z]_{\mathrm{Cu}}[x := e]_{\mathrm{Cu}} & \text{o/w; } \textit{first } z \notin \{x\} \cup \mathrm{FV}(e) \cup \mathrm{FV}(e_0) \end{cases}
\end{aligned}
$$

Fig. 1. Curry-style capture-avoiding substitution

$$
\frac{}{x[x := e]_{\mathrm{Cu}} = e} \qquad \frac{x \neq y}{y[x := e]_{\mathrm{Cu}} = y} \qquad \frac{e_1[x := e]_{\mathrm{Cu}} = e_1' \quad e_2[x := e]_{\mathrm{Cu}} = e_2'}{(e_1 e_2)[x := e]_{\mathrm{Cu}} = e_1' e_2'}
$$

$$
\frac{}{(\lambda x.e_0)[x := e]_{\mathrm{Cu}} = \lambda x.e_0} \qquad \frac{x \neq y \quad (y \notin \mathrm{FV}(e) \vee x \notin \mathrm{FV}(e_0')) \quad e_0[x := e]_{\mathrm{Cu}} = e_0'}{(\lambda y.e_0)[x := e]_{\mathrm{Cu}} = \lambda y.e_0'}
$$

$$
\frac{x \neq y \ \ y \in \mathrm{FV}(e) \ \ x \in \mathrm{FV}(e_0) \ \ z = \mathrm{Fresh}((e_0 e)x) \ \ e_0[y := z]_{\mathrm{Cu}} = e_0' \ \ e_0'[x := e]_{\mathrm{Cu}} = e_0''}{(\lambda y.e_0)[x := e]_{\mathrm{Cu}} = \lambda z.e_0''}
$$

Fig. 2. Curry-style substitution (re-)defined inductively

Structural Case-Splitting As each syntax constructor of the λ-calculus is unique, we see that it is possible to case-split on terms — with E_i in some suitable meta-language:

$$
\begin{aligned}
\text{case } e \text{ of} \quad & x \Rightarrow E_1(x) \\
\mid \ & e_1 e_2 \Rightarrow E_2(e_1, e_2) \\
\mid \ & \lambda x.e_0 \Rightarrow E_3(x, e_0)
\end{aligned}
$$

Structural Recursion Based on case-splitting and well-foundedness of terms, we can even define functions on λ-terms by means of structural recursion, i.e., by making recursive calls only on the sub-terms of a given constructor:

$$
\begin{aligned}
f(x) &= E_1(x) \\
f(e_1 e_2) &= E_2(f(e_1), f(e_2)) \\
f(\lambda x.e) &= E_3(x, f(e))
\end{aligned}
$$

The above implies that f is well-defined: it is computable by virtue of well-foundedness of terms and total because the definition case-splits exhaustively on λ-terms. As an example application, we define the function that computes the *free variables* in a term, i.e., the variable names that do not occur inside a λ-abstraction of themselves.

Definition 1

$$
\begin{aligned}
\mathrm{FV}(y) &= \{y\} \\
\mathrm{FV}(e_1 e_2) &= \mathrm{FV}(e_1) \cup \mathrm{FV}(e_2) \\
\mathrm{FV}(\lambda y.e) &= \mathrm{FV}(e) \setminus \{y\}
\end{aligned}
$$

Proposition 2 $\mathrm{FV}(-)$ *is a total, computable function.*

1.2 Reduction and Substitution

In order to have λ-abstractions act as functions and not to have too many, e.g., identity functions, amongst other things, we are typically interested in the following relations that can be applied anywhere in a term — their precise form is due to Curry [4].

1. $(\lambda x.e)e' \dashrightarrow_{\beta^{\mathrm{Cu}}} e[x := e']_{\mathrm{Cu}}$
2. $\lambda y.e[x := y]_{\mathrm{Cu}} \dashrightarrow_{\alpha^{\mathrm{Cu}}} \lambda x.e$, if $y \notin \mathrm{FV}(e)$

Our interest in 2., above is the equivalence relation it induces. We denote it by $==_\alpha$, cf. Appendix B, and we will eventually factor it out, as is standard.

Variable Capture In his seminal formalist presentation of the λ-calculus [4], Curry defines the above *substitution* operator, $-[- := -]_{\mathrm{Cu}}$, essentially as in Figure 1. The last clause is the interesting one. It renames the considered y into the first z that has not been used already.[1] Consider, for example, the substitution of x for z in the two terms $\lambda x.z$ and $\lambda y.z$. Both terms-as-functions discard their argument. If we simply replace the z in the terms with x, the latter would still discard its argument but the former would become the identity function and this discrepancy would lead to inconsistencies.

Well-Definedness Of formalist relevance, we remark that Curry-style substitution is not well-defined by construction as the definition does not employ structural

[1] While the notion "the first z" is trivially well-defined in the present case, the issue is a bit more subtle in a wider context, as we shall see in Section 2.

recursion. The offender is the last clause that applies $-[x := e]$ to a term, $e_0[y := z]$, which is not a subterm of $\lambda y.e_0$ in general. It can be observed that while $e_0[y := z]$ is not a sub-term of $\lambda y.e_0$, it will have the same size as e_0 and we can thus establish the well-formedness of $-[- := -]_{\mathrm{Cu}}$ by external means. Alternatively, we can introduce a more advanced, *parallel substitution* operator [22]. However, as we eventually will distance ourselves from the use of renaming in substitution, we will do neither but instead refer to Section 2.3 for an alternative derivation of Curry-style substitution.

Variable-Name Indeterminacy Having initially committed ourselves to using renaming in substitution, a range of problems are brought down on us. Hindley [11] observed, for example, that it becomes impossible to predict the variable name used for a given abstraction after reducing, thus putting, e.g., confluence out of reach:

$$(\lambda x.(\lambda y.\lambda x.xy)x)y \begin{array}{l} \xrightarrow{\beta^{\mathrm{Cu}}} (\lambda y.\lambda x.xy)y \xrightarrow{\beta^{\mathrm{Cu}}} \lambda x.xy \\ \xrightarrow[\beta^{\mathrm{Cu}}]{} (\lambda x.\lambda z.zx)y \xrightarrow[\beta^{\mathrm{Cu}}]{} \lambda z.zy \end{array}$$

In the lower branch, the innermost x-abstraction must be renamed to a z-abstraction, while the upper branch never encounters the variable-name clash. Hindley proceeded to define a β-relation on α-equivalence classes that overcomes the above indeterminacy by factoring it out:

$$\lfloor e \rfloor =^{\mathrm{def}} \{e' \mid e \texttt{==}_\alpha e'\}$$
$$\lfloor e_1 \rfloor \rightarrow_{\beta^{\mathrm{Hi}}} \lfloor e_2 \rfloor =^{\mathrm{def}} \exists e_1' \in \lfloor e_1 \rfloor, e_2' \in \lfloor e_2 \rfloor . e_1' \dashrightarrow_{\beta^{\mathrm{Cu}}} e_2'$$

No relevant proof principles are introduced by this and the approach can not be used in a formal setting as it stands.

Broken Induction Steps Instead of factoring out α-equivalence altogether, one could attempt to reason up to post-fixed name unification. Unfortunately, this would lead to a range of unusual situations as far as subsequent uses of abstract rewriting is concerned. An example is the following attempted adaptation of the well-known equivalence between confluence and the Church-Rosser property. Please refer to Appendix A for a precise definition of our diagram notation.

Non-Lemma 3

Proof (FAILS) By reflexive, transitive, symmetric induction in $=$.

Base, Reflexive, Symmetric Cases: Simple.

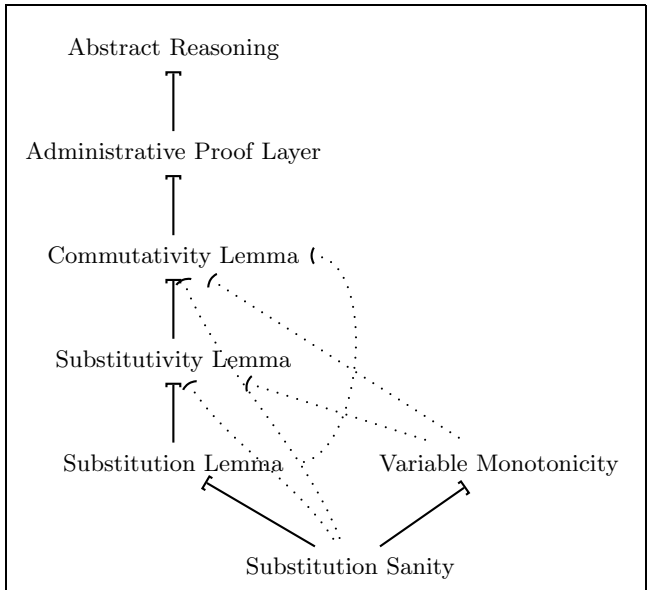
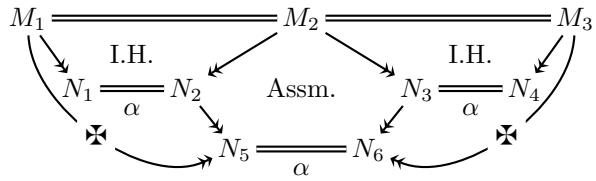


Fig. 3. The proof-layer hierarchy for primitive equational reasoning about the λ-calculus as simple syntax

Transitive Case: Breaks down.



Broken α-Equality in Sub-Terms Having failed in our attempts to control limited use of α-equivalence, one might think that the syntactic version of Hindley's approach, cf. Section 1.2, could work: that it is possible to state all properties about terms up to $\texttt{==}_\alpha$ rather than the primitive $=_{\Lambda^{\mathrm{var}}}$.

Lemma 4 (Simplified Substitution modulo α)

$$e_1 \texttt{==}_\alpha e_2 \wedge x \neq y_i \wedge y_1 \neq y_2$$
$$\Downarrow$$
$$e_1[x_1 := y_1]_{\mathrm{Cu}}[x_2 := y_2]_{\mathrm{Cu}} \texttt{==}_\alpha e_2[x_2 := y_2]_{\mathrm{Cu}}[x_1 := y_1]_{\mathrm{Cu}}$$

Proof (FAILS) By structural induction in e_1.

Most Cases: Trivial.

Last Abstraction Case (simplified): Breaks down.

$$\begin{array}{rl} & (\lambda y_1.e)[x_1 := y_1]_{\mathrm{Cu}}[x_2 := y_2] \\ = & \lambda z.e'[x_1 := y_1]_{\mathrm{Cu}}[x_2 := y_2]_{\mathrm{Cu}} \\ \texttt{==}^{\maltese}_{\alpha} & \lambda z.e'[x_2 := y_2]_{\mathrm{Cu}}[x_1 := y_1]_{\mathrm{Cu}} \\ = & (\lambda z.e')[x_2 := y_2]_{\mathrm{Cu}}[x_1 := y_1]_{\mathrm{Cu}} \end{array}$$

The problem above is that e and e' are *not* actually α-equivalent, even if $\lambda y_1.e$ and $\lambda z.e'$ are, and the $\texttt{==}_\alpha$-step can thus *not* be substantiated by the induction hypothesis. Consider, e.g., e as y_1 and e' as z. The above result is certainly correct but, unfortunately, not provable with the tools we have at our disposal at the moment.

1.3 This article

The results we are dealing with are mostly well-known and have been addressed in several contexts. Indeed, a number of truly beautiful and concise informal proofs exist; see, in particular, Takahashi [23], whom we owe a great debt. This article, therefore, spends little energy on those parts of the proofs and focuses instead on what it takes to formalise them. There are two key issues: (i) the syntactic properties that can actually be established up to $=_{\Lambda^{\mathrm{var}}}$ (as opposed to $==_\alpha$, which we have seen to be highly problematic) and (ii) how to generalise these to the algebraic properties we are seeking. The full typeset proofs (roughly 100 pages for the proofs alone) are available from our homepage.

In general, our proofs follow the structure that we present in Figure 3. It is based on nested inductions. The full-coloured arrows mean "is the key lemma for", while the others mean "is used to substantiate side-conditions on lemma applications". The first issue above, (i), is expressed in the addition of the "Variable Monotinicity" proof layer in Figure 3. The second issue, (ii), is entirely accounted for in the "Administrative Proof Layer" in Figure 3.

The proofs underpinning Sections 3 and 4.1 have been verified in full in Isabelle/HOL (at least in the case of one of the alternatives they present) [24, 25]. By the nature of Figure 3, this means that substantial parts of the other proofs essentially have been verified as well.

Apart from the various technical sections in the body of this paper, the appendix section contains an explanation of our diagram notation (Appendix A) and our other notation (Appendix B) as well as some well-known rewriting results that we use (Appendix C).

2 The λ^{var}-Calculus

Having seen that the standard presentations of the λ-calculus lead to formalist problems, we will now give an alternative presentation that overcomes them. The different presentations differ only in how they lend themselves to provability. Their equational properties are equivalent.

2.1 Formal Syntax

We use e's to range over the inductively built-up set of λ-terms. The variable names, $\mathcal{VN}$, are generic but must meet certain minimal requirements.

Definition 5 $\Lambda^{\mathrm{var}} ::= \mathcal{VN} \mid \Lambda^{\mathrm{var}}\Lambda^{\mathrm{var}} \mid \lambda\mathcal{VN}.\Lambda^{\mathrm{var}}$

Assertion 6 *$\mathcal{VN}$ is a single-sorted set of objects, aka* variable names.

Assertion 7 *$\mathcal{VN}$-equality, $=_{\mathcal{VN}}$, is decidable.*

$$\begin{aligned}
\mathrm{BV}(y) &= \emptyset \\
\mathrm{BV}(e_1 e_2) &= \mathrm{BV}(e_1) \cup \mathrm{BV}(e_2) \\
\mathrm{BV}(\lambda y.e) &= \{y\} \cup \mathrm{BV}(e)
\end{aligned}$$

$$\begin{aligned}
\mathrm{Capt}_x(y) &= \emptyset \\
\mathrm{Capt}_x(e_1 e_2) &= \mathrm{Capt}_x(e_1) \cup \mathrm{Capt}_x(e_2) \\
\mathrm{Capt}_x(\lambda y.e) &= \begin{cases} \{y\} \cup \mathrm{Capt}_x(e) & \text{if } x \in \mathrm{FV}(\lambda y.e) \\ \emptyset & \text{otherwise} \end{cases}
\end{aligned}$$

Fig. 4. Bound and capturing variable names

Assertion 8 *There exists a total, computable function,* $\mathrm{Fresh}(-) : \Lambda^{\mathrm{var}} \longrightarrow \mathcal{VN}$*, such that:*[2]

$$\mathrm{Fresh}(e) \notin \mathrm{FV}(e) \cup \mathrm{BV}(e)$$

The last assertion trivially implies that $\mathcal{VN}$ is infinite.[3]

We shall use x's, y's, and z's as meta-variables of $\mathcal{VN}$ and, by a slight abuse of notation, also as actual variable names in terms. We will suppress the $\mathcal{VN}$ suffix on variable-name equality and merely write, e.g., $x = y$.

2.2 Orthonormal Reduction

The key technicality to prevent implicit renaming is our use of a predicate, $\mathrm{Capt}_x(e_1) \cap \mathrm{FV}(e_2) = \emptyset$, cf. Figure 4, which guarantees that no capture takes place in the substitution: $e_1[x := e_2]$. It coincides with the notion of *not free for*.

Definition 9 (The λ^{var}-Calculus) *The terms of the λ^{var}-calculus are Λ^{var}, cf. Definition 5. The (indexed) α-, β-, and η-reduction relations of λ^{var}: $\overline{\dashrightarrow}_{i\alpha}$, $\dashrightarrow_\beta$, and $\dashrightarrow_\eta$ are given inductively in Figure 5. The plain α-relation is:*

$$e \dashrightarrow_\alpha e' \Leftrightarrow^{\mathrm{def}} \exists y.e \overset{y}{\dashrightarrow}_{i\alpha} e'$$

Unlike the situation with Curry-style substitution, we see that our notion of substitution is defined by structural recursion and, hence, is well-defined by construction.

Proposition 10 *$-[x := e]$ is a total, computable function.*

[2] For the definition of $\mathrm{BV}(-)$, see Figure 4.

[3] In the setting of Nominal Logic [19], the assertion also validates the axiom of choice, which is known to be provably inconsistent with the Fraenkel-Mostowski set theory that underpins Nominal Logic. Nominal Logic instead guarantees the existence of *some* fresh variable name, which by design can be *any* variable name except for a finite number. More work needs to be done to clarify the correspondence between simple syntax and syntax based on Nominal Logic.

$$y[x := e] = \begin{cases} e \text{ if } x = y \\ y \text{ otherwise} \end{cases}$$
$$(e_1 e_2)[x := e] = e_1[x := e] e_2[x := e]$$
$$(\lambda y.e_0)[x := e] = \begin{cases} \lambda y.e_0[x := e] \text{ if } x \neq y \wedge y \notin \mathrm{FV}(e) \\ \lambda y.e_0 \quad \text{otherwise} \end{cases}$$

$$\frac{y \notin \mathrm{Capt}_x(e) \cup \mathrm{FV}(e)}{\lambda x.e \xrightarrow{y}_{i\alpha} \lambda y.e[x := y]}\,(\alpha) \qquad \frac{e \xrightarrow{y}_{i\alpha} e'}{\lambda x.e \xrightarrow{y}_{i\alpha} \lambda x.e'} \qquad \frac{e_1 \xrightarrow{y}_{i\alpha} e_1'}{e_1 e_2 \xrightarrow{y}_{i\alpha} e_1' e_2} \qquad \frac{e_2 \xrightarrow{y}_{i\alpha} e_2'}{e_1 e_2 \xrightarrow{y}_{i\alpha} e_1 e_2'}$$

$$\frac{\mathrm{Capt}_x(e_1) \cap \mathrm{FV}(e_2) = \emptyset}{(\lambda x.e_1) e_2 \longrightarrow_\beta e_1[x := e_2]}\,(\beta) \qquad \frac{e \longrightarrow_\beta e'}{\lambda x.e \longrightarrow_\beta \lambda x.e'} \qquad \frac{e_1 \longrightarrow_\beta e_1'}{e_1 e_2 \longrightarrow_\beta e_1' e_2} \qquad \frac{e_2 \longrightarrow_\beta e_2'}{e_1 e_2 \longrightarrow_\beta e_1 e_2'}$$

$$\frac{x \notin \mathrm{FV}(e) = \emptyset}{\lambda x.e x \longrightarrow_\eta e}\,(\eta) \qquad \frac{e \longrightarrow_\eta e'}{\lambda x.e \longrightarrow_\eta \lambda x.e'} \qquad \frac{e_1 \longrightarrow_\eta e_1'}{e_1 e_2 \longrightarrow_\eta e_1' e_2} \qquad \frac{e_2 \longrightarrow_\eta e_2'}{e_1 e_2 \longrightarrow_\eta e_1 e_2'}$$

Fig. 5. Renaming-free substitution, $-[- := -]$, defined recursively, and α-, β-, η-reduction defined inductively over Λ^{var}

The β- and η-relations we have presented above do not incur any renaming that could have been performed in a stand-alone fashion by the α-relation, thus making them *orthogonal*. The *normality* part of our informal orthonormality principle is established by the following property, symmetry of $\longrightarrow_\alpha$, which implies that the α-relation itself is renaming-free.

Lemma 11

2.3 Curry's λ-Calculus Decomposed

In order to assure ourselves that the λ^{var}-calculus is indeed the right calculus and partly to test the usefulness of the associated primitive proof principles, we now show how to derive Curry's presentation from ours. First, we show that as far as our use of substitution is concerned, $-[- := -]$ coincides with $-[- := -]_{\mathrm{Cu}}$.

Proposition 12

$$\begin{array}{l} \mathrm{Capt}_x(e_a) \cap \mathrm{FV}(e) = \emptyset \\ \Downarrow \\ e_a[x := e] = e_a[x := e]_{\mathrm{Cu}} \end{array}$$

Proof A straightforward structural induction in e_a. □

What might not be obvious is that Curry-style substitution can be shown to decompose into the λ^{var}-calculus. In contrast to the structurally flawed Figure 1, Figure 2 introduces a primitively-defined, 4-ary relation that *is* Curry-style substitution, albeit with no claim of well-definedness.

Lemma 13

$$\begin{array}{l} e_a[x := e]_{\mathrm{Cu}} = e_a' \\ \Downarrow \\ \exists! e_b.e_a \twoheadrightarrow_\alpha e_b \ \wedge \ e_b[x := e] = e_a' \end{array}$$

Proof By rule induction in Curry-style substitution-as-a-relation, cf. Figure 2. Uniqueness of e_b is guaranteed by the functionality of Fresh$(-)$. □

We stress that the above property is not provable by structural induction in e_a and that it ensures that Curry-style substitution is, indeed, well-defined and functional.

Lemma 14 *For any x and e, $-[x := e]_{\mathrm{Cu}} = -$ is a total, computable function of the first, open argument onto the second, open argument.*

Lemma 13 also establishes the decomposition of Curry's calculus as a whole into the λ^{var}-calculus.

Lemma 15 $\longrightarrow_\alpha \ \subseteq \ (\longrightarrow_{\alpha^{\mathrm{Cu}}})^{-1} \ \subseteq \ \twoheadrightarrow_\alpha$

Lemma 16 $\longrightarrow_\beta \ \subseteq \ \longrightarrow_{\beta^{\mathrm{Cu}}} \ \subseteq \ \twoheadrightarrow_\alpha ; \longrightarrow_\beta$

2.4 The Real λ-Calculus

As suggested previously, the actual calculus we are interested in is the α-collapse of λ^{var}. Algebraically speaking, this means that we want to consider the following structure, cf. Hindley's presentation, Section 1.2.

Definition 17 (The Real λ-Calculus)

– $\Lambda =^{def} \Lambda^{\mathrm{var}}/{=\!=_\alpha}$

$$\begin{aligned}\mathrm{UB}(x) &= \mathrm{True}\\ \mathrm{UB}(e_1 e_2) &= \mathrm{UB}(e_1) \wedge \mathrm{UB}(e_2) \wedge (\mathrm{BV}(e_1) \cap \mathrm{BV}(e_2) = \emptyset)\\ \mathrm{UB}(\lambda x.e) &= \mathrm{UB}(e) \wedge x \notin \mathrm{BV}(e)\end{aligned}$$

Fig. 6. The *uniquely bound* Λ^{var}-predicate

$$\begin{aligned}&\_\ \ \lfloor - \rfloor :^{def} \Lambda^{\mathrm{var}} \longrightarrow \Lambda\\ &\qquad\qquad e \ \mapsto\ \{e' \mid e \mathrel{==}_\alpha e'\}\\ &-\ \lfloor e_1 \rfloor \longrightarrow_\beta \lfloor e_2 \rfloor \Leftrightarrow^{def} e_1 \mathrel{==}_\alpha ; \dashrightarrow_\beta ; \mathrel{==}_\alpha e_2\\ &-\ \lfloor e_1 \rfloor \longrightarrow_\eta \lfloor e_2 \rfloor \Leftrightarrow^{def} e_1 \mathrel{==}_\alpha ; \dashrightarrow_\eta ; \mathrel{==}_\alpha e_2\end{aligned}$$

It can be shown (without too much trouble) that Curry's, Hindley's, and our relations all are *pointwise* identical, cf. [25]. For now, we merely present the part of that result that pertains to the current set-up.

Lemma 18 *For* $\mathrm{X} \in \{\beta, \eta, \beta\eta\}$ *(any* X*, in fact), we have:*

$$\lfloor e \rfloor \longrightarrow\!\!\!\!\rightarrow_{\mathrm{X}} \lfloor e' \rfloor \Leftrightarrow e \dashrightarrow\!\!\!\!\rightarrow_{\alpha \mathrm{X}} e'$$

Proof By definition of the real relations and reflexive, transitive closure, we immediately see that

$$\lfloor e \rfloor \longrightarrow\!\!\!\!\rightarrow_{\mathrm{X}} \lfloor e' \rfloor \ \Leftrightarrow\ e\ (\mathrel{==}_\alpha ; \dashrightarrow_{\mathrm{X}} ; \mathrel{==}_\alpha)^\star\ e' \vee e \mathrel{==}_\alpha e'$$

The result thus follows directly from Lemma 11. □

3 Residual Theory

This section shows that residual theory, i.e., the exclusive contraction of pre-existing, or marked, redexes, provides a nice setting for quantifying the "computing power" of the renaming-free β-relation. We use t_i's as meta-variables over the marked terms and we allow ourselves to use Λ^{var}-concepts for the marked terms with only implicit coercions; in particular, we assume there is an $\alpha^{@}$-relation that can rename all (not just marked) abstractions.

Definition 19 (The Marked λ^{var}-Calculus)

$$\Lambda^{\mathrm{var}}_{@} ::= x \ \mid\ \Lambda^{\mathrm{var}}_{@} \Lambda^{\mathrm{var}}_{@} \ \mid\ \lambda\mathcal{V}\mathcal{N}.\Lambda^{\mathrm{var}}_{@} \ \mid\ (\lambda\mathcal{V}\mathcal{N}.\Lambda^{\mathrm{var}}_{@})\,@\,\Lambda^{\mathrm{var}}_{@}$$

$\dashrightarrow_{\beta^{@}}$ *is like* $\dashrightarrow_\beta$ *except only marked redexes,* $(\lambda x.t_1)\,@\,t_2$*, may be contracted (provided* $\mathrm{Capt}_x(t_1) \cap \mathrm{FV}(t_2) = \emptyset$*). We further define a* residual-completion *relation,* $\dashrightarrow\!\!\!\!\rightarrow_{\beta^{@}}$*, by induction over terms that attempts to contract all (marked) redexes in one step, starting from within.*[4]

[4] The relation corresponds closely to the parallel β-relation of Figure 7.

To address any inherent requirements for renaming in the λ-calculus, we introduce a formal notion called *Barendregt Conventional Form* (BCF),[5] which, as it turns out, provides a rational reconstruction of the usual (informal) Barendregt Variable Convention [2], cf. [25]. BCFs are terms where all variable names are different.

Definition 20 *Cf. Figures 4 and 6:*

$$\mathrm{BCF}(e) = \mathrm{UB}(e) \wedge (\mathrm{BV}(e) \cap \mathrm{FV}(e) = \emptyset)$$

As a first approximation to renaming-freeness, we note that it is a straightforward proof that BCFs residually completes, i.e., that all marked redexes in a BCF can be contracted from within without causing variable clashes.

Lemma 21 (BCF) $\bullet \dashrightarrow\!\!\!\!\rightarrow_{\beta^{@}} \circ$

We also show that the residual-completion relation is functional on the full β-residual theory of a term, i.e., that residual completion always catches up with itself.

Lemma 22

Proof The right-most conjunct follows from the left-most by a simple reflexive, transitive induction in which the latter constitutes the base case. The left-most conjunct follows by a rule induction in $\dashrightarrow\!\!\!\!\rightarrow_{\beta^{@}}$ for which it is paramount that redexes are enabled if $\mathrm{Capt}_x(-) \cap \mathrm{FV}(-) = \emptyset$ rather than only if $\mathrm{BV}(-) \cap \mathrm{FV}(-) = \emptyset$. Other than that, the proof is mostly straightforward, albeit big. □

The above property asserts that *when* residual completion exists, the considered divergence can be resolved as shown. The property allows us to prove that β-residual theory is renaming-free up to BCF-initiality, i.e., that no redexes are blocked by their side-condition.

Theorem 23 (BCF) $\bullet \dashrightarrow\!\!\!\!\rightarrow \bullet \dashrightarrow\!\!\!\!\rightarrow_{\beta^{@}} \circ$

Proof Consider a BCF and a $\dashrightarrow\!\!\!\!\rightarrow_{\beta^{@}}$-reduction of it. By Lemma 21, the considered BCF also residually completes and, by Lemma 22, the thus-created divergence can be resolved by a trailing residual completion. □

A subtle point of interest is that the above proof, in fact, shows that the β-residual theory of any term that residually-completes, i.e., is renaming-free if contracted from within, is renaming-free in general.

[5] The term was suggested to us by Randy Pollack.

$$\frac{}{x \multimap\!\!\!\!\mapsto_\beta x} \qquad \frac{e \multimap\!\!\!\!\mapsto_\beta e'}{\lambda x.e \multimap\!\!\!\!\mapsto_\beta \lambda x.e'} \qquad \frac{e_1 \multimap\!\!\!\!\mapsto_\beta e_1' \quad e_2 \multimap\!\!\!\!\mapsto_\beta e_2'}{e_1 e_2 \multimap\!\!\!\!\mapsto_\beta e_1' e_2'}$$

$$\frac{e_1 \multimap\!\!\!\!\mapsto_\beta e_1' \quad e_2 \multimap\!\!\!\!\mapsto_\beta e_2' \quad \mathrm{FV}(e_2') \cap \mathrm{Capt}_x(e_1') = \emptyset}{(\lambda x.e_1)e_2 \multimap\!\!\!\!\mapsto_\beta e_1'[x := e_2']}\ (\beta^{\shortparallel})$$

Fig. 7. The parallel β-relation for λ^{var}

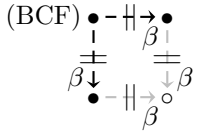
4 Confluence

The previous section establishes a rather large fragment of the λ^{var}-calculus as susceptible to primitive equational reasoning. This section summarises and elaborates on our formally verified efforts to bring this to bearing on β-confluence [25]. We also present proofs that apply the methodology to prove η- and $\beta\eta$-confluence.

4.1 β-Confluence

The $\dashrightarrow_\beta$-relation does not enjoy the diamond property because a redex that is contracted in one direction of a divergence can be duplicated (and erased) in the other direction by the substitution operator. As shown by Tait and Martin-Löf, the potential divergence "blow-up" does not materialise because it can be controlled by *parallel* reduction. Please refer to Figure 7 for the λ^{var}-version of this relation.

Lemma 24



Proof Rather than prove this property by an exhaustive case-splitting, thus resulting in a *minimally* resolving end-term, Takahashi observed that the considered diamond can be diagonalised by the relation that contracts all redexes in one step, i.e., by a *maximally* resolving end-term [23]. As we saw in Section 3 this is within reach of the structural proof principles of λ^{var}. □

The Full Proof Burden A real version of the parallel β-relation on syntax can be defined along the lines of Definition 17 (which, further to Lemma 21, turns out to be the *real* real parallel β-relation).

Definition 25 $\lfloor e_1 \rfloor \multimap\!\!\!\!\mapsto_\beta \lfloor e_2 \rfloor \Leftrightarrow^{def} e_1 \mathrel{==}_\alpha ; \multimap\!\!\!\!\mapsto_\beta ; \mathrel{==}_\alpha e_2$

In order to prove the diamond property for $\multimap\!\!\!\!\mapsto_\beta$, we need some measure of commutativity between α- and β-reduction.

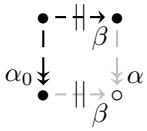
Fresh-Naming As the general α-/β-commutativity result is not provable, we introduce the following restricted α-relation, which only fresh-names.

Definition 26

$$e \dashrightarrow_{\alpha_0} e' \ \Leftrightarrow^{\mathrm{def}}\ \exists z.e \overset{z}{\dashrightarrow}_{i\alpha} e' \wedge z \notin \mathrm{FV}(e) \cup \mathrm{BV}(e)$$

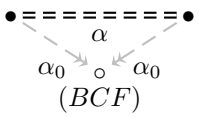
The fresh-naming α-relation can straightforwardly be proven to commute with the parallel (actually, any) β-relation with the proviso that the resolving α-steps are not necessarily fresh-naming (because of β-incurred term duplication).

Lemma 27



Similarly, the fresh-naming α-relation can be shown to resolve α-equivalence to a BCF (although the formal proof of this is surprisingly involved, cf. [25]).

Lemma 28



Applying Administration With these results in place, we can lift Lemma 24 to the real λ-calculus.

Lemma 29 $\diamond(\multimap\!\!\!\!\mapsto_\beta) \wedge \diamond(\dashrightarrow\!\!\!\!\twoheadrightarrow_\alpha ; \multimap\!\!\!\!\mapsto_\beta)$

Proof As for the left-most conjunct, see Figure 8 for the step by step resolution of the definitionally-given syntactic divergence. We trust the steps are self-evident and that it can be seen that a slight adaptation of the figure also proves the right-most conjunct. □

We are now in a position to establish β-confluence.

Theorem 30

$$\begin{aligned}\mathrm{Confl}(\longrightarrow_\beta) \wedge\ & \mathrm{Confl}(\dashrightarrow_{\alpha\beta}) \\ \wedge\ & \mathrm{Confl}(\dashrightarrow_{\alpha^{\mathrm{Cu}}\beta^{\mathrm{Cu}}}) \\ \wedge\ & \mathrm{Confl}(\longrightarrow_{\beta^{\mathrm{Hi}}})\end{aligned}$$

Proof The two top-most conjuncts are equivalent by Lemma 18. They can also be proved independently by applying the Diamond Tiling Lemma of Appendix C to the corresponding conjunct in Lemma 29. The third conjunct follows by Lemmas 15 and 16. The final conjunct follows in an analogous manner. □

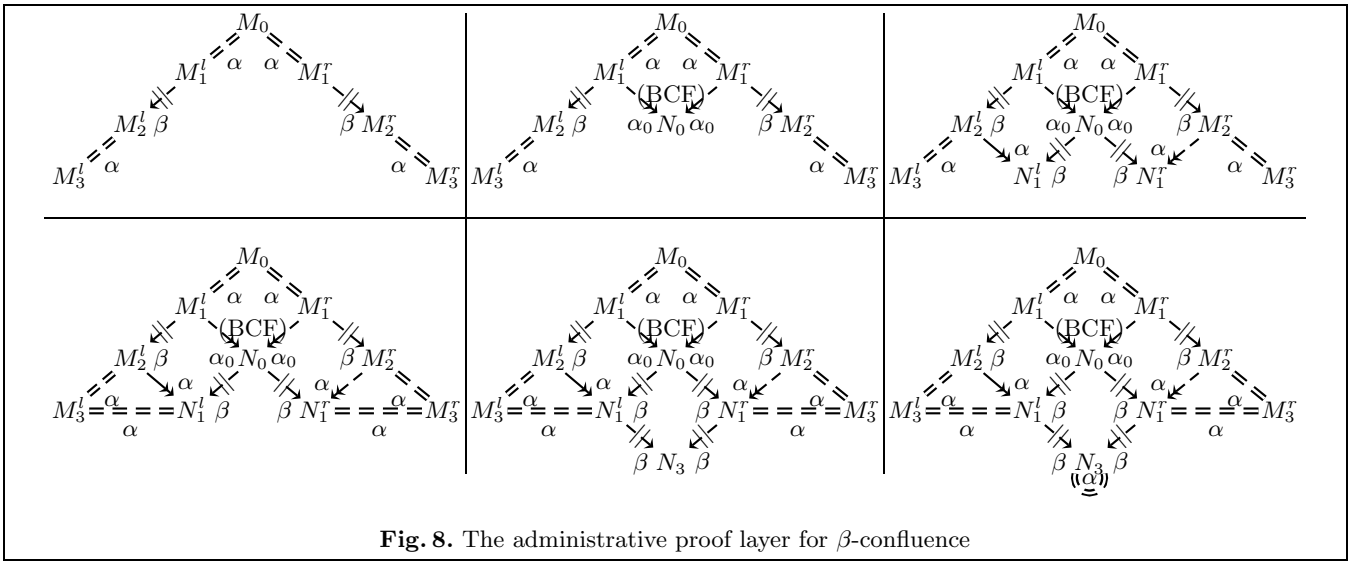


Fig. 8. The administrative proof layer for β-confluence

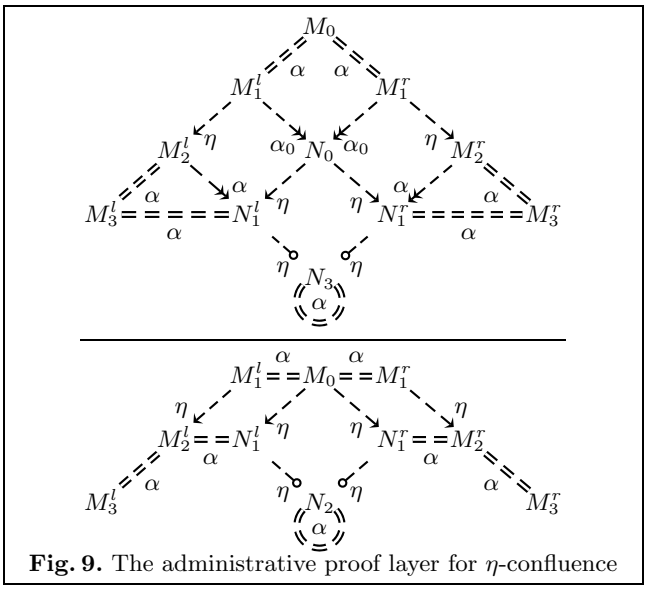
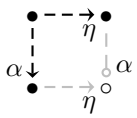


Fig. 9. The administrative proof layer for η-confluence

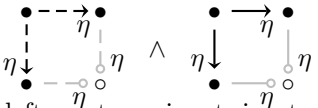
4.2 η-Confluence

Unlike the β-relation, η-reduction is natively renaming-free:

Lemma 31 (α/η Commutativity)



Lemma 32 (η Commutativity)



Proof The left-most conjunct is straightforwardly provable by structural means. The proof of the right-most property follows from the left-most as displayed in Figure 9. The top part of the figure is a proof by the *general* method; the lower part is an optimised version that takes advantage of η commuting with α, not just with α_0. □

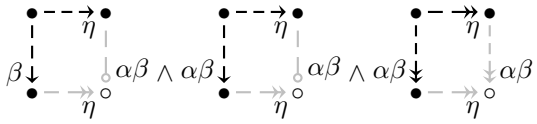
Theorem 33 $\mathrm{Confl}(\dashrightarrow_\eta) \wedge \mathrm{Confl}(\longrightarrow_\eta) \wedge \mathrm{Confl}(\dashrightarrow_{\alpha\eta})$

Proof The two left-most conjuncts can be established from the corresponding conjuncts in Lemma 32 by the Hindley-Rosen Lemma of Appendix C. The right-most conjunct can be established either by the Commuting Confluence Lemma of Appendix C applied to the left-most conjunct and generalisations of Lemmas 11 and 31 or, alternatively, it can be observed that the two right-most conjuncts are equivalent by Lemma 18. □

4.3 $\beta\eta$-Confluence

Since the η-relation is natively renaming-free and the β-relation relies on the α-relation, we must show that η-commutes with combined $\alpha\beta$-reduction in order to apply the Commuting Confluence Lemma of Appendix C.

Lemma 34



Proof The proof of the left-most conjunct is straightforward. The α-step in the resolution on the right is needed for the obvious divergence on $\lambda x.(\lambda y.e)x$, with $x \neq y$. The middle conjunct combines the left-most conjunct and Lemma 31. The right-most conjunct follows from the middle by the Hindley-Rosen Lemma of Appendix C. □

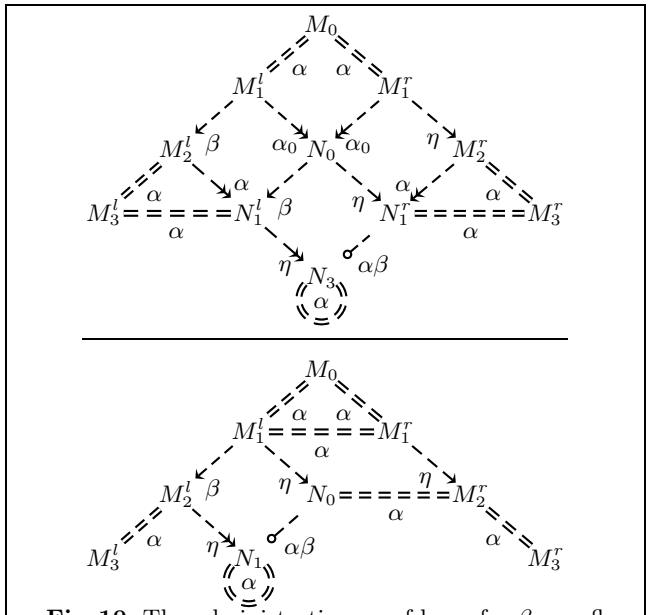
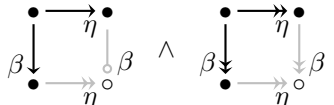


Fig. 10. The administrative proof layer for $\beta\eta$-confl

Lemma 35



Proof The left-most conjunct follows from the left-most conjunct of Lemma 34 as shown in Figure 10. The top part of the figure is by the *general* method; the lower part is an optimisation based on (full) $\alpha\eta$-commutativity, Lemma 31. The right-most conjunct follows by the Hindley-Rosen Lemma of Appendix C. □

Theorem 36 $\mathrm{Confl}(\longrightarrow_{\beta\eta}) \wedge \mathrm{Confl}(\dashrightarrow_{\alpha\beta\eta})$

Proof We first observe that the two conjuncts are equivalent by Lemma 18. They can also be proved independently by the Commuting Confluence Lemma of Appendix C applied to Theorems 30 and 33 as well as Lemma 35 and Lemma 34, respectively. □

5 Residual β-Confluence

We say that the reflexive, transitive closure of a residual relation is the associated *development* relation, a step of which is said to be *complete* if the target term does not contain a mark, $\mathrm{unMarked}(-)$. With this terminology in place, we define a weakened version of the strong finite development property.[6]

[6] The strong finite development property also requires that the residual relation is strongly normalising. It is typically used to prove (residual) confluence.

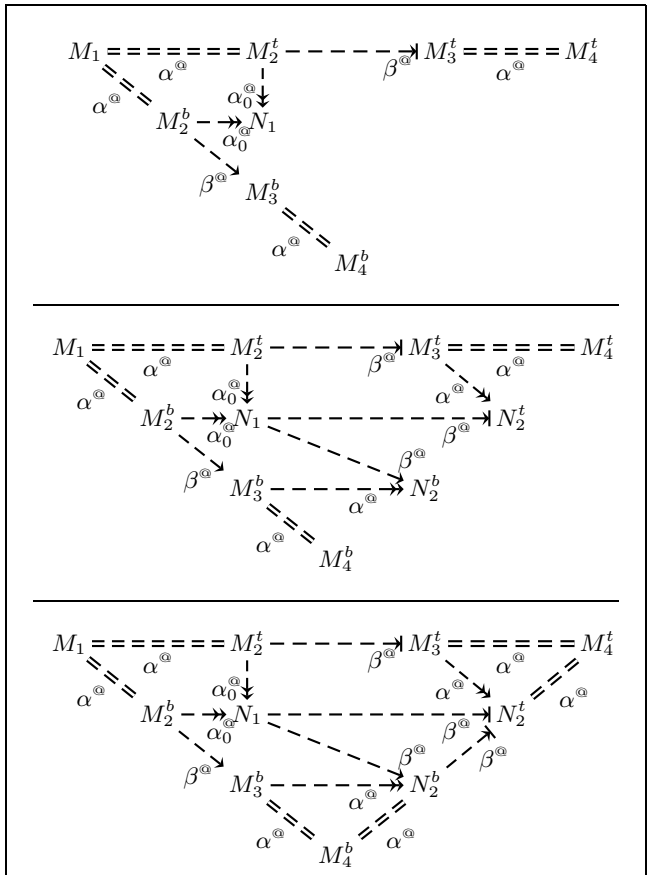


Fig. 11. The administrative proof layer for β-RCP

Definition 37 *Let* $\longrightarrow_{@}$ *be the residual relation of* $\longrightarrow$. *We say that* $\longrightarrow$ *enjoys the* strong weakly-finite development property, $\mathrm{SWFDP}(\longrightarrow)$, *if*

1. $t \longrightarrow\!\!\!\!\!\rightarrow_{@} t' \Rightarrow \exists t''.t' \longrightarrow\!\!\!\!\!\rightarrow_{@} t'' \wedge \mathrm{unMarked}(t'')$
   — *developments can be completed*
2. $t \longrightarrow\!\!\!\!\!\rightarrow_{@} t_i \wedge \mathrm{unMarked}(t_i) \wedge i \in \{1,2\} \;\Rightarrow\; t_1 = t_2$
   — *completions are unique*

To motivate the name of the property, we see that, indeed:

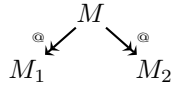
Proposition 38 $\mathrm{SWFDP}(\longrightarrow) \;\Rightarrow\; \mathrm{WN}(\longrightarrow_{@})$[7]

Proof By Definition 37, 1. and reflexivity of $\longrightarrow\!\!\!\!\!\rightarrow_{@}$. □

Surprisingly, perhaps, we have that already the SWFDP implies residual confluence.

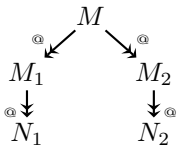
Lemma 39 $\mathrm{SWFDP}(\longrightarrow) \Rightarrow \mathrm{Confl}(\longrightarrow_{@})$

Proof Consider the following divergence:



[7] The predicate $\mathrm{WN}(-)$ stands for Weak Normalisation and means that all terms reduce to a normal form.

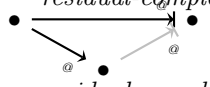
By Definition 37, 1., there exist N_1, N_2, such that unMarked(N_1), unMarked(N_2) and:



By transitivity of $\longrightarrow\!\!\!\!\!\rightarrow_{@}$ and Definition 37, 2., we see that, in fact, $N_1 = N_2$ and we are done. □

With direct reference to Section 3, we define the following property, which is fairly easily proven to be equivalent to the SWFDP.

Definition 40 *A relation, $\longrightarrow$, enjoys the residual-completion property,* RCP($\longrightarrow$)*, if there exists a residual-completion relation, $\longrightarrow\!\!\!\!\!\!\rightarrow_{@}$, such that:*

1. $\longrightarrow\!\!\!\!\!\!\rightarrow_{@} \subseteq \longrightarrow\!\!\!\!\!\rightarrow_{@}$
   — *residual-completion is a development*
2.  $\circ(\mathrm{NF}_{@})$
   — *residual-completion totally completes*
3. 
   — *residual-completion is residually co-final*

Lemma 41 RCP($\longrightarrow$) $\Leftrightarrow$ SWFDP($\longrightarrow$)

Our interest in the RCP is its constructive nature, in particular when the residual-completion relation is defined as a computable function the way we did in Section 3.

Lemma 42 RCP($\longrightarrow_{\beta}$) $\wedge$ SWFDP($\longrightarrow_{\beta}$)

Proof We prove the left-most conjunct. Clause 1. follows from the easily established fact that $\dashrightarrow\!\!\!\!\!\!\rightarrow_{\beta@} \subseteq \dashrightarrow\!\!\!\!\!\rightarrow_{\beta@}$. Clause 2 follows from Lemmas 21 and 28. Finally, Clause 3 is proved as shown in Figure 11. □

Theorem 43 Confl($\longrightarrow_{\beta@}$) $\wedge$ Confl($\dashrightarrow_{\alpha@\beta@}$)

We see that SN($\longrightarrow_{\beta@}$) (i.e., the difference between the SWFDP and the strong finite development property) is not needed for concluding confluence from a residual analysis of the β-relation, something which is in stark contrast to established opinion [2, p.283]. Strong finite development essentially implies confluence through Newman's Lemma, thus relying crucially on the (non-equational) SN-property for the residual relation. We think it a nice "purification" of the equational import of residual theory that an externally justified termination property is not needed for concluding confluence.

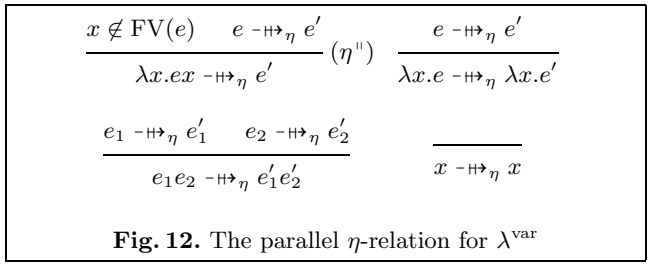


Fig. 12. The parallel η-relation for λ^{var}

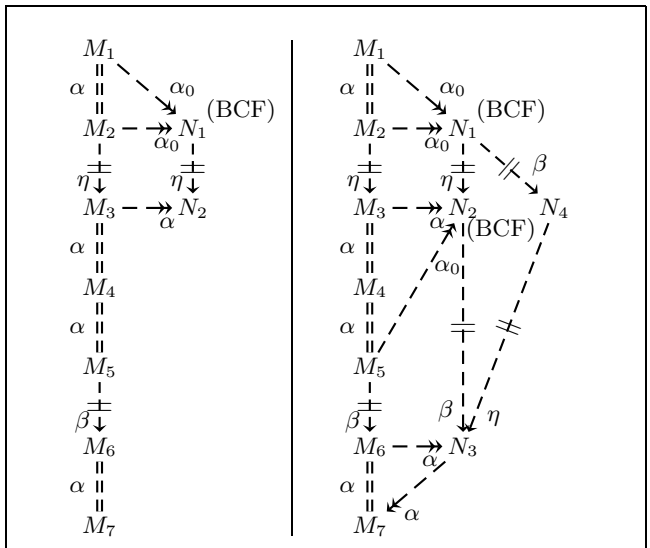


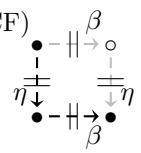
Fig. 13. The administrative proof layer for η-postponing

6 η-over-β-Postponement

As well as condensing Tait and Martin-Löf's use of parallel β-reduction for proving β-confluence, Takahashi [23] also shows how to adapt the parallel-reduction technology to other typical situations in the equational theory of the λ-calculus. One such situation is for proving η-over-β-postponement, cf. Figure 12. The proof presented by Takahashi [23] essentially goes through up to BCF-initiality as it stands, albeit not completely. Rather than focusing on the low-level technical details, this section merely shows the Administrative and Abstract proof layers of our formalisation of Takahashi's proof.

The notion of commutativity that we have considered so far is orthogonal in nature to that employed in the η-over-β-Postponement Theorem. Whereas the former can be described as *divergence commutativity*, this section focuses on *composition commutativity*.

Lemma 44 (BCF)



Proof The parallel η-relation is used to allow for the duplication of a η-redex by the β-contraction when the latter is performed first. The parallel β-relation, on the other hand, is used. e.g., for the following situation:

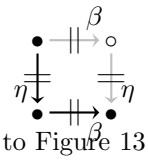
$$(\lambda x.(\lambda y.e_1)x)e_2 \dashrightarrow_\eta (\lambda y.e_1)e_2 \dashrightarrow_\beta e_1[y := e_2]$$

This reduction sequence commutes into a leading parallel β-step with a trailing η-step, which is in this case is reflexive:

$$(\lambda x.(\lambda y.e_1)x)e_2 \mathrel{-\!\!\!\mapsto}_\beta e_1[y := x][x := e_2]$$

BCF-initiality is used to enable the double (n-fold, in general) substitution in the commuted reduction sequence. □

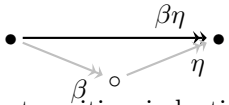
Lemma 45



Proof Please refer to Figure 13 for the details of the proof. A novel aspect of the proof is the existence of an α_0-step from M_5 to N_2. By construction, we know that the two terms are α-equivalent. A simple lemma shows that N_2 is a BCF because η-reduction preserves BCFs. The final result that is needed, i.e., that α_0-reduction can reach any BCF that is α-equivalent to the start term, can also be proved by structural means but it is not as straightforward as could be imagined. This is due to the need for the target BCF to be *any* BCF. □

With the one necessary technical lemma in place, we present the postponement theorem.

Theorem 46



Proof By reflexive, transitive induction in $\longrightarrow\!\!\!\!\!\rightarrow_{\beta\eta}$. The only interesting case is the transitive case, which follows in a manner akin to the Hindley-Rosen Lemma of Appendix C using Lemma 45. □

7 β-Standardisation

Standardisation is also a composition-commutativity result like postponement. It is a very powerful result that, informally speaking, says that any reduction sequence can be performed left-to-right. Standardisation implies results such as the left-most reduction lemma, etc., [2], and guarantees the existence of evaluation-order independent semantics [20].

This section addresses three different approaches to proving standardisation due to Mitschke [18], Plotkin [20], and David [5], respectively. The three approaches are fairy closely related, with Plotkin's proof bridging the other two, so to speak. Mitschke's and Plotkin's proofs both use semi-standardisation while David's and Plotkin's both can be described as *absorption* standardisation. In spite (actually *because*) of this, only Plotkin's approach is formally provable by the proof principles we are considering. We shall examine the failures of the other two proofs closely.

$$\frac{\mathrm{Capt}_x(e_1) \cap \mathrm{FV}(e_2) = \emptyset}{(\lambda x.e_1)e_2 \dashrightarrow_{\beta^{\mathrm{wh}}} e_1[x := e_2]}\,(\beta^{\mathrm{wh}}) \qquad \frac{e_1 \dashrightarrow_{\beta^{\mathrm{wh}}} e_1'}{e_1e_2 \dashrightarrow_{\beta^{\mathrm{wh}}} e_1'e_2}\,(@^{\mathrm{wh}})$$

Fig. 14. Weak-head β-reduction

$$\frac{e_1 \dashrightarrow_{\beta^{\mathrm{I}}} e_1'}{e_1e_2 \dashrightarrow_{\beta^{\mathrm{I}}} e_1'e_2}\,(@_1^{\mathrm{I}}) \qquad \frac{e_2 \dashrightarrow_\beta e_2'}{e_1e_2 \dashrightarrow_{\beta^{\mathrm{I}}} e_1e_2'}\,(@_2^{\mathrm{I}})$$

$$\frac{e \dashrightarrow_\beta e'}{\lambda x.e \dashrightarrow_{\beta^{\mathrm{I}}} \lambda x.e'}\,(\lambda^{\mathrm{I}})$$

$$\frac{}{x \mathrel{-\!\!\!\mapsto}_{\beta^{\mathrm{I}}} x}\,(\mathrm{Var}_{\shortparallel}^{\mathrm{I}}) \qquad \frac{e_1 \mathrel{-\!\!\!\mapsto}_{\beta^{\mathrm{I}}} e_1' \qquad e_2 \mathrel{-\!\!\!\mapsto}_\beta e_2'}{e_1e_2 \mathrel{-\!\!\!\mapsto}_{\beta^{\mathrm{I}}} e_1'e_2'}\,(@_{\shortparallel}^{\mathrm{I}})$$

$$\frac{e \mathrel{-\!\!\!\mapsto}_\beta e'}{\lambda x.e \mathrel{-\!\!\!\mapsto}_{\beta^{\mathrm{I}}} \lambda x.e'}\,(\lambda_{\shortparallel}^{\mathrm{I}})$$

Fig. 15. Inner and parallel inner β-reduction

7.1 Semi-Standardisation with Hereditary Recursion

In this section, we shall pursue a slight adaptation of Takahashi's adaptation [23] of Mitschke's proof [18]. Instead of head and a corresponding notion of inner reduction, we base the proof on weak-head reduction. This does not affect the formal status of the proof technique but does allow us to reuse the results of this section when pursuing Plotkin's approach. The main proof burden is to show that (weak-)head redexes can contracted before any inner redexes, so-called semi-standardisation.

Definition 47 *Weak-head β-reduction, $\dashrightarrow_{\beta^{\mathrm{wh}}}$, is defined in Figure 14. The corresponding (strong) inner, $\dashrightarrow_{\beta^{\mathrm{I}}}$, and parallel inner, $\mathrel{-\!\!\!\mapsto}_{\beta^{\mathrm{I}}}$, β-relations are defined in Figure 15.*

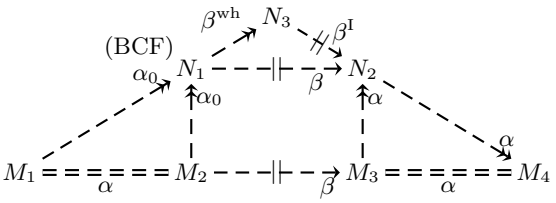


Fig. 16. The administrative proof layer for weak-head before inner parallel decomposition

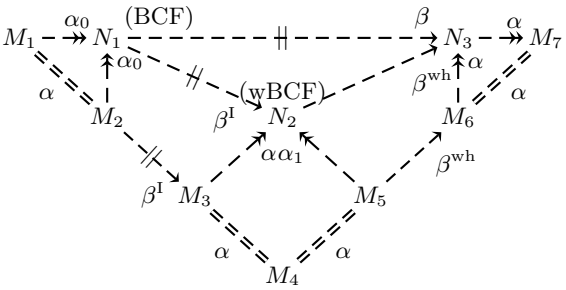
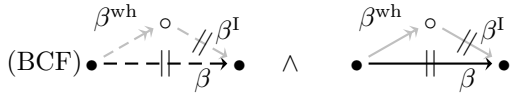


Fig. 17. The administrative proof layer for parallelization of weak-head after inner

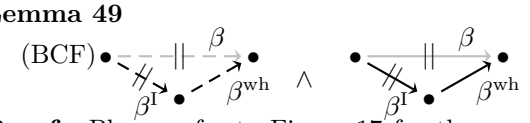
Lemma 48



Proof Please refer to Figure 16 for the proof of the right-most conjunct based on the left-most conjunct, wich, in turn, is proved by rule induction in $\text{-}\!\!\mapsto_{\beta}$. □

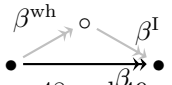
The use of BCF-initiality in the left-most conjunct above guarantees that weak-head redexes can be contracted without waiting for the contraction of an inner redex to eliminate a variable clash.

Lemma 49



Proof Please refer to Figure 17 for the proof of the right-most conjunct based on the left-most. We first note that the figure invokes the obvious adaptation of Lemma 27 to $\text{-}\!\!\mapsto_{\beta^{\mathrm{I}}}$. Although the proof as a whole is similar to that of η-over-β-postponement, cf. Lemma 45, we do not have that $\text{-}\!\!\mapsto_{\beta^{\mathrm{I}}}$ preserves BCFs, as is the case with $\text{-}\!\!\mapsto_{\eta}$. Instead, we can introduce a weakened notion of BCF, wBCF, that allows identical binders to occur in adjacent positions (but not nested and not coinciding with any free variables) and show that $\text{-}\!\!\mapsto_{\beta^{\mathrm{I}}}$ sends BCFs to wBCFs. In the same manner that α_0-reduction and the BCF-predicate correspond to each other, we can introduce an α_1-relation that corresponds to the wBCF-predicate. The α_1-relation is less well-behaved than the α_0-relation but we can, at least, show that it commutes with $\dashrightarrow_{\beta}$ (and thus $\dashrightarrow_{\beta^{\mathrm{wh}}}$), up to α-resolution. The left-most conjunct of the lemma, follows by rule induction in $\text{-}\!\!\mapsto_{\beta^{\mathrm{I}}}$. □

Lemma 50 (Semi-Standardisation)



Proof From Lemmas 48 and 49, cf. the obvious reflexive, transitive generalisation of Lemma 53 in the transitive case. □

At this point, the idea is to recursive over the $\circ$ in Lemma 50 and show that the sub-terms in which the outgoing $\longrightarrow\!\!\!\!\!\rightarrow_{\beta^{\mathrm{I}}}$-step are ordinary β-steps, themselves can be semi-standardised and so on. Unfortunately, the $\circ$ is quantified over α-equivalence classes, for which no recursion is possible and we are stuck.

7.2 Hereditary Weak-Head Standardisation

Plotkin [20] defines standardisation as the least contextually-closed relation on terms that enjoys left-absorptivity over weak-head reduction. The following presentation of the proof methodology owes a great debt to McKinna and Pollack [17]. The difference between their and our presentation is that we focus on provability with structural induction, etc., while they work with an alternative syntactic framework that is *derived* from first-order abstract syntax with *two*-sorted variable names. The proof requirements in their setting and in ours are substantially different as a result.

A First (Failing) Approach A first approach, which immediately fails, is to define Plotkin's relation directly on terms.

$$\frac{e \dashrightarrow\!\!\!\!\!\rightarrow_{\beta^{\mathrm{wh}}} e' \quad e' \succ\!\!\dashv_{\mathrm{P}} e''}{e \succ\!\!\dashv_{\mathrm{P}} e''} \qquad \frac{}{x \succ\!\!\dashv_{\mathrm{P}} x}$$

$$\frac{e_1 \succ\!\!\dashv_{\mathrm{P}} e_1' \quad e_2 \succ\!\!\dashv_{\mathrm{P}} e_2'}{e_1 e_2 \succ\!\!\dashv_{\mathrm{P}} e_1' e_2'} \qquad \frac{e \succ\!\!\dashv_{\mathrm{P}} e'}{\lambda x.e \succ\!\!\dashv_{\mathrm{P}} \lambda x.e'}$$

As standardisation pertains to *all* β-reductions (i.e., $\longrightarrow\!\!\!\!\!\rightarrow_{\beta}$, not just $\dashrightarrow_{\beta}$), the naive approach needs the full λ-calculus to be renaming-free, which it is not. The problem manifests itself in the required administrative proof layer for the standardisation property and its exact nature is of independent interest. The point is that, even if it is possible to prove the following key property (which, in fact, seems to be the case[8]), we cannot prove

[8] Coincidentally, it is interesting to note that the proof of the property can only be conducted by rule induction in $\succ\!\!\dashv_{\mathrm{P}}$ and not in $\dashrightarrow_{\beta}$.

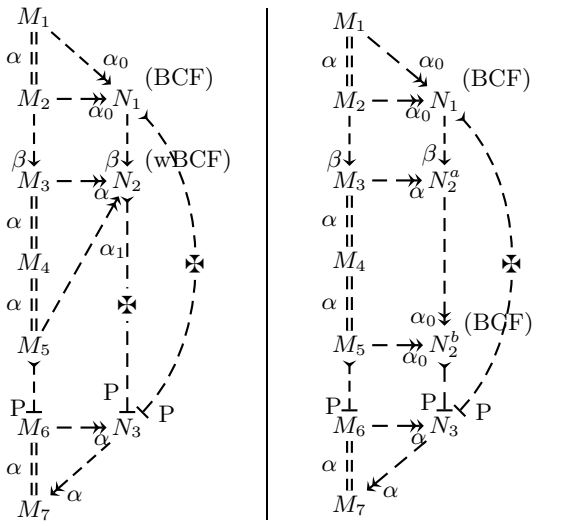


Fig. 18. Failed administrative proof layer for left-absorptivity of progression standardisation

full standardisation but at most standardisation of the renaming-free fragment of the λ^{var}-calculus.

Please refer to Figure 18 for the only two sensible approaches to the administrative proof layer for the following property, which is derived from the one above.

Non-Lemma 51

The left-most diagram in the figure attempts to align itself with Figure 13, which fails because $\succ\!\!\dashv_{\text{P}}$ only commutes with $\dashrightarrow_{\alpha_0}$. The right-most diagram adheres to this and fails because of the inserted $\dashrightarrow\!\!\!\!\twoheadrightarrow_{\alpha_0}$, which we cannot incorporate into the syntactic version of the property. It is even straightforward to come up with a counter-example.

$$(\lambda s.ss)(\lambda x.\lambda y.xy) \dashrightarrow_{\beta} (\lambda x.\lambda y.xy)(\lambda x.\lambda y.xy)$$

We can turn the end-term into an α-equivalent BCF, as it happens, which standardises:

$$(\lambda x_1.\lambda y_1.x_1y_1)(\lambda x_2.\lambda y_2.x_2y_2) \succ\!\!\dashv_{\text{P}} \lambda y_1.\lambda y_2.y_1y_2$$

As the end-term of this step uses the two y copies nested within each other, we see that the original start term does not standardise to it.

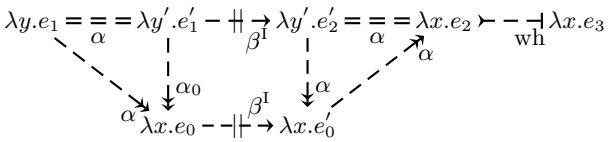


Fig. 19. The administrative proof layer for the $(\lambda_{\succ\!\dashv_{\text{wh}}})$-case of Lemma 54

Combining Term Structure and α-Collapsed Reduction In order to avoid these problems, we adapt the above definition slightly.

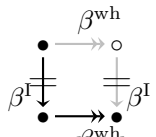
Definition 52

$$\frac{\lfloor e \rfloor \longrightarrow\!\!\!\!\!\twoheadrightarrow_{\beta\text{wh}} \lfloor e' \rfloor \quad e' \succ\!\!\dashv_{\text{wh}} e''}{e \succ\!\!\dashv_{\text{wh}} e''}\,(\text{wh}_{\text{pre}}) \qquad \frac{}{x \succ\!\!\dashv_{\text{wh}} x}\,(\text{V}_{\succ\!\dashv_{\text{wh}}})$$

$$\frac{e_1 \succ\!\!\dashv_{\text{wh}} e_1' \quad e_2 \succ\!\!\dashv_{\text{wh}} e_2'}{e_1e_2 \succ\!\!\dashv_{\text{wh}} e_1'e_2'}\,(@_{\succ\!\dashv_{\text{wh}}}) \qquad \frac{e \succ\!\!\dashv_{\text{wh}} e'}{\lambda x.e \succ\!\!\dashv_{\text{wh}} \lambda x.e'}\,(\lambda_{\succ\!\dashv_{\text{wh}}})$$

The definition mixes the advantages of being able to define relations inductively over terms with the use of reduction in the real λ-calculus to avoid issues of renaming. Note, however, that, further to the failed proof of Lemma 4, it is by no means obvious whether this mixture will lend itself to primitive structural reasoning. The proof-technical issue surfaces in the $(\text{V}_{\succ\!\dashv_{\text{wh}}})$-case of the proof of Lemma 54.

Lemma 53



Proof The property can be derived from Lemmas 48 and 49 based on a suitable adaptation of the Hindley-Rosen Lemma, cf. Appendix C. □

The key technical lemma in the present standardisation proof development is the following *absorption* property.

Lemma 54

$$\lfloor e_1 \rfloor \twoheadrightarrow\!\!\!\!\dashv_{\beta\text{I}} \lfloor e_2 \rfloor \wedge e_2 \succ\!\!\dashv_{\text{wh}} e_3 \;\Rightarrow\; e_1 \succ\!\!\dashv_{\text{wh}} e_3$$

Proof The proof is by rule induction in $\succ\!\!\dashv_{\text{wh}}$ and uses Lemma 53 before applying the I.H. and the definitional left-absorptivity over weak-head reduction when needed. As far as administration is concerned, the only interesting case is for abstraction.

Case $(\lambda_{\succ\!\dashv_{\text{wh}}})$**:** We are considering the following situation (although this takes some effort to substantiate).

$$\lambda y.e_1 =\!\!=_{\alpha} \lambda y'.e_1' \;\text{-}\!\!\twoheadrightarrow_{\beta\text{I}} \lambda y'.e_2' =\!\!=_{\alpha} \lambda x.e_2 \succ\!\!\dashv_{\text{wh}} \lambda x.e_3$$

By Definition 47 and the case, we have $e'_1 \mathrel{-\!\!\!\twoheadrightarrow_\beta} e'_2$ and $e_2 \succ\!\!\!\dashv_{\mathrm{wh}} e_3$. If $y' = x$, we can prove $e'_2 =\!=_\alpha e_2$, which means that we are considering $\lfloor e'_1 \rfloor \twoheadrightarrow_\beta \lfloor e'_2 \rfloor \succ\!\!\!\dashv_{\mathrm{wh}} e_3$, so to speak. From Lemma 48, we thus have: $\lfloor e'_1 \rfloor \longrightarrow\!\!\!\!\!\rightarrow_{\beta^{\mathrm{wh}}} \lfloor e''_1 \rfloor \twoheadrightarrow_{\beta^{\mathrm{I}}} \lfloor e'_2 \rfloor \succ\!\!\!\dashv_{\mathrm{wh}} e_3$. An application of the I.H. and an invocation of the $(\mathrm{wh}_{\mathrm{pre}})$-rule will then give us that $e'_1 \succ\!\!\!\dashv_{\mathrm{wh}} e_3$ and we have $\lambda x.e'_1 \succ\!\!\!\dashv_{\mathrm{wh}} \lambda x.e_3$ by the $(\lambda_{\succ\!\!\!\dashv_{\mathrm{wh}}})$-rule. A final (reflexive) application of the $(\mathrm{wh}_{\mathrm{pre}})$-rule thus finishes the case: $\lambda y.e_1 \succ\!\!\!\dashv_{\mathrm{wh}} \lambda x.e_3$. Unfortunately, we can not guarantee $y' = x$. Instead, Figure 19 shows how to overcome this using our general administrative proof-layer technology, cf. Figure 3. Based on the upper line, we first rewrite $\lambda y'.e'_1$ to (the BCF) $\lambda x.e_0$ (although it takes some effort to substantiate that this is possible). The commuting square involving $\lambda x.e'_0$ can then be constructed by the obvious adaptation of Lemma 27 and the diagram can finally be closed based on Lemma 11. To show that $\lambda y.e_1 \succ\!\!\!\dashv_{\mathrm{wh}}$-standardises to $\lambda x.e_3$, first apply the reasoning above to show that $\lambda x.e_0$ does and, then, use the $(\mathrm{wh}_{\mathrm{pre}})$-rule reflexively to show the result we are after.

Other Cases: Fairly straightforward. □

Theorem 55 $\lfloor e_1 \rfloor \longrightarrow\!\!\!\!\!\rightarrow_\beta \lfloor e_2 \rfloor \;\Rightarrow\; e_1 \succ\!\!\!\dashv_{\mathrm{wh}} e_2$

Proof By reflexive, left-transitive induction in $\longrightarrow\!\!\!\!\!\rightarrow_\beta$. The reflexive case is a straightforward structural induction. The left-transitive case follows by an I.H.-application followed by a case-split on the considered $\longrightarrow_\beta$-step into $\longrightarrow_{\beta^{\mathrm{wh}}}$ and $\longrightarrow_{\beta^{\mathrm{I}}}$ (seeing that we can show that the union of the latter two is the former). In case of $\longrightarrow_{\beta^{\mathrm{wh}}}$, we are done by definition of $\succ\!\!\!\dashv_{\mathrm{wh}}$. In case of $\longrightarrow_{\beta^{\mathrm{I}}}$, we are done by Lemma 54. □

7.3 (Failing) Progression Standardisation

An alternative proof development for standardisation was proposed by David [5] and pursued, more or less independently, in [13–15]. The idea is to define a standardisation relation directly by induction over terms (although this is only done implicitly in [5]): $\succ\!\!\!\dashv_{\mathrm{prg}}$, and to show that this relation right-absorbs the ordinary β-relation. In that sense, the proof development is the dual approach to what we considered in the previous section. Informally, the key technical point is to contract terms as follows, cf. [13, 15]:[9]

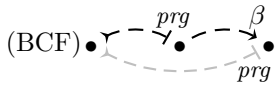
$$\frac{(..(e[x := e_0]e_1)..)e_k \succ\!\!\!\dashv_{\mathrm{prg}} e'}{(..((\lambda x.e)e_0)e_1..)e_k \succ\!\!\!\dashv_{\mathrm{prg}} e'}$$

[9] In order for the relation to make sense in the current setting, it is necessary to supply it with a finite axiomatisation, which can be done.

This ensures that contraction *progresses* from left-to-right while at the same time allowing newly created redexes to be contracted. Other rules allow redexes not to be contracted as the relation otherwise would be left-most reduction.

Right-Absorptivity As mentioned, the key technical lemma is purported to show right-absorptivity of $\succ\!\!\!\dashv_{\mathrm{prg}}$ over $\dashrightarrow_\beta$, which appears to be straightforward, at least in the case of the above contraction rule [5, 13–15].

Non-Lemma 56



Unfortunately, not even the BCF-initial version of the property is true. The following is a counter-example.

$$(\lambda s.ss)(\lambda x.(\lambda y.xy)z) \succ\!\!\!\dashv_{\mathrm{prg}} (\lambda y.(\lambda x.xz)y)z \dashrightarrow_\beta (\lambda y.yz)z$$

The problem in the counter-example is the last step of the standardisation, which amounts to the contraction of the redex involving the inner y-abstraction below.

$$(\lambda y.(\lambda x.(\lambda y.xy)z)y)z$$

As it happens, this is the point where the considered $\dashrightarrow_\beta$-step (i.e., the contraction of the redex involving the x-abstraction) must be inserted but that is not possible because of a clash with the inner y-abstraction.

Left-Absorptivity In sharp contrast with the above (and surprisingly, at first), it turns out that it is possible to prove left-absorptivity, as also seen at the beginning of Section 7.2.

The difference between right- and left-absorptivity is that the universal quantification over $\succ\!\!\!\dashv_{\mathrm{prg}}$ covers far fewer steps in the latter case than in the former. As we saw, this manifests itself when trying to prove standardisation for the real λ-calculus.

Non-Lemma 57

The counter-example at the beginning of Section 7.2 applies.

8 Conclusion

Standard, informal practice in the programming language theory community when using structural induction and related proof principles is to assume that variables clashes are not an issue (aka Barendregt's Variable Convention). We have shown this to be formally correct for a range of standard properties, possibly up to BCF-initiality, cf. Lemmas 21, 22, 24, 32, 34, 44, 48, 49, and 54. For the most part, we have been able to show that the undertaken proof burden resolution is formally incomplete in the sense that the formal proof burden can be met by the addition of a fairly simple administrative proof layer, cf. Figures 8, 9, 10, 11, 13, 16, and 17. The administrative proof layers mostly rely on the same additional lemmas, thus preventing a blow-up of proof obligations. We studied standardisation in some detail and found that only one out of three proof techniques appears to be amenable to the use of structural induction, etc..

A Commutative Diagrams

We use commutative diagrams in three different ways, which are distinguished in the way they write vertices.

A.1 Vertices as Terms

When written with terms as vertices, commutative diagrams simply describe reduction scenarios.

A.2 Vertices as M's, N's

We shall see next that commutative diagrams are used to express rewriting predicates such as:

> "For all terms, such that, ..., there exist terms, such that, ..."

In order to prove these results, we start by writing M's for the universally quantified terms and gradually introduce N's from supporting lemmas to eventually substantiate the existence claims. Please note that we use ✠ to signify "claimed" existences that are impossible.

A.3 Vertices as •'s, ∘'s

Formally, a commutative diagram of this nature is a set of vertices and a set of directed edges between pairs of vertices. Informally, the colour of a vertex (• vs ∘) denotes quantification modes over terms, universal and existential, respectively. A vertex may be guarded by a predicate. Edges are written as the relational symbol they pertain to and are either full-coloured (black) or half-coloured (gray). Informally, the colour indicates assumed and concluded relations, respectively. An edge connected to a ∘ must be half-coloured. A diagram must be type-correct on domains. A property is read off of a diagram thus:

1. write universal quantifications for all •'s
2. assume the full-coloured relations and the validation of any guard for a •
3. conclude the guarded existence of all ∘s and their relations

The following diagram and property are thus equivalent.

$$\begin{array}{ll} (P)\ \bullet \longrightarrow \bullet & e_1 \to e_2 \ \wedge\ e_1 \to e_3 \ \wedge\ P(e_1) \\ \quad\downarrow \quad\ \downarrow & \quad\Downarrow \\ \quad\bullet \longrightarrow \circ\ (Q) & \exists e_4 \,.\, e_2 \to e_4 \ \wedge\ e_3 \to e_4 \ \wedge\ Q(e_4) \end{array}$$

B Notation and Terminology

We say that a term *reduces* to another if the two are related by a *reduction* relation and we denote the relationship by an infix arrow between the two terms. The "direction" of the reduction should be thought of as being from-left-to-right. The sub-term of the left-hand side that a reduction step "acts upon" is called the *redex* of the reduction and it is said to be *contracted*.

- The converse of a relation, $\to$, is written $(\to)^{-1}$.
- Composition is:

$$a \to_1; \to_2 c \ \Leftrightarrow^{\text{def}}\ \exists b \,.\, a \to_1 b \ \wedge\ b \to_2 c$$

- Given two reduction relations $\to_1$ and $\to_2$, we have: $\to_{1,2} \ =^{\text{def}}\ \to_1 \cup \to_2$. If no confusion is possible, we omit the comma.
- The reflexive closure of a relation is:[10]

$$\frac{e_1 \to e_2}{e_1 \multimap e_2} \qquad \frac{}{e \multimap e}$$

- The reflexive, transitive closure is:

$$\frac{e_1 \to e_2}{e_1 \twoheadrightarrow e_2} \qquad \frac{}{e \twoheadrightarrow e} \qquad \frac{e_1 \twoheadrightarrow e_2 \qquad e_2 \twoheadrightarrow e_3}{e_1 \twoheadrightarrow e_3}$$

  We will also denote $\twoheadrightarrow$ by $(\to)^\star$.
- The reflexive, transitive, and symmetric closure is:

$$\frac{e_1 \to e_2}{e_1 = e_2} \qquad \frac{}{e = e} \qquad \frac{e_1 = e_2 \qquad e_2 = e_3}{e_1 = e_3} \qquad \frac{e_1 = e_2}{e_2 = e_1}$$

- The situation of a term reducing to two terms is called a *divergence*.

[10] This and the next two items are immediately associated with primitive induction principles. Equality, however, is only point-wise (or extensional), and no recursion principle is possible.

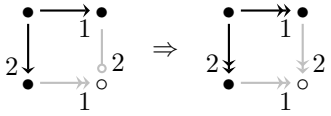
– Two diverging reductions, as defined above, are said to be *co-initial*.
– Dually, two reductions that share their end-term are said to be *co-final*.
– Co-initial reductions are *resolvable* if they compose with co-final reductions.
– A relation has the *diamond property*, $\diamond$, if any divergence can be resolved.
– A relation, $\rightarrow$, is *confluent*, Confl, if $\diamond(\twoheadrightarrow)$.

C Known Abstract Results

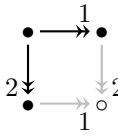
Diamond Tiling Lemma

$$(\exists \rightarrow_2 . \rightarrow_1 \subseteq \rightarrow_2 \subseteq \twoheadrightarrow_1 \wedge \diamond(\rightarrow_2)) \Rightarrow \diamond(\twoheadrightarrow_1)$$

Hindley-Rosen Lemma



Commuting Confluence Lemma



$\wedge$ Confl$(\rightarrow_1) \wedge$ Confl$(\rightarrow_2) \Rightarrow$ Confl$(\rightarrow_{1,2})$

References

1. Peter Aczel. An introduction to inductive definitions. In J. Barwise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, chapter C.7, pages 739–782. North-Holland, Amsterdam, 1977.
2. Henk Barendregt. *The Lambda Calculus — Its Syntax and Semantics (Revised Edition)*. North-Holland, 1984.
3. Rod Burstall. Proving properties of programs by structural induction. *The Computer Journal*, 12, 1967.
4. H. B. Curry and R. Feys. *Combinatory Logic*. North-Holland, Amsterdam, 1958.
5. René David. Une preuve simple de rèsultats classiques en λ calcul. *Comptes Rendus de l'Acadèmie des Sciences*, 320(11):1401–1406, 1995. Série I.
6. N.G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser Theorem. *Indag. Math.*, 34:381–392, 1972.
7. Joëlle Despeyroux and André Hirschowitz. Higher-order abstract syntax with induction in Coq. In Frank Pfenning, editor, *Proceedings of LPAR-5*, volume 822 of *LNAI*. Springer-Verlag, 1994.
8. Joëlle Despeyroux, Frank Pfenning, and Carsten Schürmann. Primitive recursion for higher-order abstract syntax. In Philippe De Groote and J. Roger Hindley, editors, *Proceedings of TLCA-3*, volume 1210 of *LNCS*. Springer-Verlag, 1997.
9. Marcelo Fiore, Gordon Plotkin, and Daniele Turi. Abstract syntax and variable binding. In Longo [16], pages 193–202.
10. Murdoch J. Gabbay and Andrew M. Pitts. A new approach to abstract syntax involving binders. In Longo [16], pages 214–224.
11. J. Roger Hindley. *The Church-Rosser Property and a Result in Combinatory Logic*. PhD thesis, University of Newcastle upon Tyne, 1964.
12. Martin Hofmann. Semantical analysis of higher-order abstract syntax. In Longo [16], pages 204–213.
13. Felix Joachimski and Ralph Matthes. Standardization and confluence for a lambda calculus with generalized applications. In Leo Bachmair, editor, *Proceedings of RTA-11*, volume 1833 of *LNCS*. Springer Verlag, 2000.
14. Ryo Kashima. On the standardization theorem for lambda-beta-eta-calculus. In *Proceedings of RPC'01*, 2001. Technical report, the Research Institute of Electrical Communication, Tohoku University, Japan.
15. Ralph Loader. Notes on simply typed lambda calculus. Technical report no. ECS-LFCS-98-381 of LFCS, University of Edinburgh, 1998.
16. Giuseppe Longo, editor. *Proceedings of LICS-14*. IEEE CS Press, 1999.
17. James McKinna and Randy Pollack. Some lambda calculus and type theory formalized. *Journal of Automated Reasoning*, 23(3–4), November 1999.
18. Gerd Mitschke. The standardization theorem for λ-calculus. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 25:29–31, 1979.
19. A. M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 200X. To appear; a preliminary version appears in TACS-4 2001, LNCS 2215.
20. Gordon D. Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science*, 1:125–159, 1975.
21. Carsten Schürmann. *Automating the Meta Theory of Deductive Systems*. PhD thesis, Carnegie Mellon University, 2000.
22. Allen Stoughton. Substitution revisited. *Theoretical Computer Science*, 59(3):317–325, August 1988.
23. Masako Takahashi. Parallel reductions in λ-calculus. *Information and Computation*, 118:120–127, 1995.
24. René Vestergaard and James Brotherston. The mechanisation of Barendregt-style equational proofs (the residual perspective). *Electronic Notes in Theoretical Computer Science*, 58(1), 2001. Invited version of MERLIN01 workshop paper.
25. René Vestergaard and James Brotherston. A formalised first-order confluence proof for the λ-calculus using one-sorted variable names. *Information and Computation*, 200X. To appear; special edition with selected papers from RTA01.

Nominal Sets, Equivariance Reasoning, and Variable Binding

Murdoch J. Gabbay mjg1003@cl.cam.ac.uk
Computer Laboratory, Cambridge University, UK

1 Equivariance reasoning

FM techniques are a methodology of thinking about syntax and in particular about syntax with binding. They take their name from the original FM (Fraenkel-Mostowski) theory of sets presented in my thesis [4]. For this document we use a different way of presenting FM techniques based on the idea of a **Nominal Set**, which is a set equipped with certain algebraic properties (just as a group, ring, or field, is a set equipped with certain properties), see Definition 1.2. First we need background machinery:

Definition 1.1. Fix a countably infinite **set of atoms** $\mathbb{A}$. Write typical elements $a, b, c, \ldots \in \mathbb{A}$. For $a, b \in \mathbb{A}$ write $(a\ b)$ for the function $\mathbb{A} \to \mathbb{A}$ such that $a \mapsto b$ and $b \mapsto a$ and $n \mapsto n$ for all other atoms $n \neq a, b$. This is a bijection with inverse itself, write $P_{\mathbb{A}}$ for the group generated by $(a\ b)$ for all $a, b \in \mathbb{A}$ under functional composition $\circ$. Write typical elements $\pi, \pi' \in P_{\mathbb{A}}$ and $\mathbf{Id} \in P_{\mathbb{A}}$ for the identity.

Definition 1.2. A **Nominal Set** is a pair $\langle X, \cdot \rangle$ of an underlying set X and **permutation action** $\cdot$ (written infix) of type $P_{\mathbb{A}} \times X \to X$ and satisfying the usual axioms, namely $\pi \cdot (\pi' \cdot x) = \pi \circ \pi' \cdot x$ and $\mathbf{Id} \cdot x = x$. The permutation action also satisfies a finiteness condition omitted here.[1]

The point is that finite labelled trees, and hence the standard model of syntax but also a natural model of *proofs* as trees, are Nominal Sets: the permutation action is given pointwise by the action on the labels. Thus for example natural numbers $\mathbb{N}$ satisfy a trivial permutation action given by $\pi \cdot n = n$ always. A datatype of trees for terms of the λ-calculus (using $\mathbb{A}$ as variable names)

$$\Lambda \cong \mathbb{A} + \Lambda \times \Lambda + \mathbb{A} \times \Lambda \tag{1}$$

is also a Nominal Set with the permutation action given pointwise by the action on the atoms labelling the tree. Furthermore we can represent a theory of α-equivalence on these terms as a subset of "well-formed" trees in the inductively defined set

$$T \cong \mathbb{A} + T \times T + \mathbb{A} \times T \times T, \tag{2}$$

namely those inductively constructed using the rules

$$a =_\alpha a \quad (\mathbf{Var}) \qquad \frac{t_1 =_\alpha t_1' \quad t_2 =_\alpha t_2'}{t_1 t_2 =_\alpha t_1' t_2'} \quad (\mathbf{App}) \qquad \frac{t\{c/a\} =_\alpha t'\{c/a'\}}{\lambda a t =_\alpha \lambda a' t'} \quad (\mathbf{Lam}) \tag{3}$$

where in (**Lam**) there is a side-condition that $c \notin \{a, a'\} \cup n(t) \cup n(t')$ (thus 'fresh' for the conclusion). The usefulness of this way of looking at syntax and properties of syntax as Nominal Sets begins with the following trivial theorem:

Theorem 1.3. *If a property (on trees) is defined (inductively) using predicates whose validity is invariant under permuting atoms, then the property is invariant under permuting atoms.*

Here "invariant under permuting..." means "given some valid instance of the property, a permutation π uniformly applied to its arguments yields another valid instance".

We have an example in the property of well-formedness of proof-trees of $=_\alpha$ given in (3). $\dfrac{}{a =_\alpha a}$ is a valid instance of (**Var**) and if we apply $(b\ a)$ to this we obtain $\dfrac{}{b =_\alpha b}$, which is also a valid instance of

[1] See [3, eq. 3], [5, eq. 4], and 'finite support' in [2, Def. 3.3].

(**Var**). The case of (**App**) is simple. A permutation applied to a valid instance of (**Lam**) is also a valid instance of (**Lam**) because $c \notin \{a, a'\} \cup n(t) \cup n(t')$ if and only if $\pi \cdot c \notin \{\pi \cdot a, \pi \cdot a'\} \cup n(\pi \cdot t) \cup n(\pi \cdot t')$.[2]

We now have a very concrete demonstration that proofs of $=_\alpha$ are invariant under permutation; we permute the atoms in the proof as a tree. We can take this further. Consider proving transitivity of $=_\alpha$ by induction on proof-trees. The induction predicate is (in words) "given a valid proof-tree Π concluding in $t =_\alpha t'$, for all valid proof-trees Π' concluding in $t' =_\alpha t''$, there exists a valid proof-tree Π'' concluding in $t =_\alpha t''$". This property is constructed using predicates invariant under permutations (validity of proofs of $=_\alpha$) and so is itself invariant under permutations. Thus from Theorem 1.3 we know if we have the inductive hypothesis of Π, we have it of $\pi \cdot \Pi$ for any permutation Π.

We proceed by induction on Π. The case of (**Lam**) for $t = \lambda a s$ and $t' = \lambda a' s'$ causes problems: we may assume Π proves $s[c/a] =_\alpha s'[c/a']$ and Π' proves $s'[c'/a'] =_\alpha s''[c'/a'']$ and we assume the inductive predicate for Π, but we do not know $c = c'$ so we cannot proceed. However, we *can* apply a permutation $(d\ c)$ to Π, and $(d\ c')$ to Π', for d chosen completely fresh. Now we have valid proofs $(d\ c) \cdot \Pi$ concluding in $s[d/a] =_\alpha s'[d/a']$ and $(d\ c') \cdot \Pi'$ concluding in $s'[d/a'] =_\alpha s''[d/a'']$, and also the inductive predicate for $(d\ c) \cdot \Pi$. We can now complete the proof of transitivity.

Just these ideas of permutations have already been adopted and put to use by other authors also in published work (see for example [6]).

2 Taking it further

There is an equivalence class of proofs concluding in $\lambda a t =_\alpha \lambda a' t'$, one for each fresh c; we can take it as an object in its own right. This is an instance of FM abstraction $[\mathbb{A}]X$ which exists for any Nominal Set X by virtue of the permutation action, which lets us rename atoms and construct an equivalence class in the general case (see [2, Section 5]). We can apply this to syntax as well as proofs:

$$\Lambda_\alpha \cong \mathbb{A} + \Lambda_\alpha \times \Lambda_\alpha + [\mathbb{A}]\Lambda_\alpha \tag{4}$$

is a datatype of λ-terms up to α-equivalence. An element of $[\mathbb{A}]\Lambda_\alpha$ is (concretely) an equivalence class of pairs $\langle a, t\rangle$ for $a \in \mathbb{A}$ a 'bound atom' fresh for the other atoms in the 'body' $t \in \Lambda_\alpha$.

There are various ways of taking this further; Nominal Sets form a category, the Schanuel Topos. Because it is a topos we can construct a general theory of abstractions and equivariance reasoning within it (this is FMCat in [5, Section 2], see also [2, p.21]). Nominal Sets are also a special case of a general theory of FM sets, see [4] and [2]. Nominal Sets can be axiomatised in first-order logic, see [7]. A team in Cambridge has developed FreshML, a programming language based on these principles in which we can program using abstractions and permutations, see [1]. Finally, I am currently implementing FM sets in Isabelle, see [4]. Further reading can be found in any of the references below, and my homepage `www.cl.cam.ac.uk/~mjg1003`.

References

[1] *FreshML homepage*, `http://www.freshml.org`.

[2] M. J. Gabbay and A. M. Pitts, *A new approach to abstract syntax with variable binding*, Formal Aspects of Computing **13** (2001), 341–363.

[3] Murdoch Gabbay, *FM for names and binding*, `cl.cam.ac.uk/~mjg1003/papers/talks/200212-jaist-talk.ps`.

[4] Murdoch J. Gabbay, *A theory of inductive definitions with alpha-equivalence*, Ph.D. thesis, Cambridge, UK, 2000.

[5] Murdoch J. Gabbay, *Theory and models of the π-calculus using fraenkel mostowski generalised*, Submitted to the commemorative volume for 35 years of Automath, F. Kamareddine (ed.)., September 2002.

[6] L. Cardelli, P. Gardner, G. Ghelli, *Manipulating trees with hiding*, FOSSACS 2003, to appear.

[7] A. M. Pitts, *Nominal logic, a first order theory of names and binding*, Information and Computation, To appear. (A preliminary version appeared in the *Proceedings of the 4th International Symposium on Theoretical Aspects of Computer Software* (TACS 2001), LNCS 2215, Springer-Verlag, 2001, pp 219–242.).

[2] This is not the case if we try to base the theorem on substitutions generated by $[b/a]$ instead of permutations generated by $(b\ a) = [b/a, a/b]$.

Equational Tree Automata: Towards Automated Verification of Network Protocols*

Hitoshi Ohsaki and Toshinori Takai

National Institute of Advanced Industrial Science and Technology (AIST)
Nakoji 3–11–46, Amagasaki 661–0974, Japan

{ohsaki,takai}@ni.aist.go.jp

Abstract. An extension of tree automata framework, called equational tree automata, is presented. This theory is useful to deal with unification modulo equational rewriting. In the manuscript, we demonstrate how equational tree automata can be applied to several realistic unification examples, e.g. including a security problem of network protocols.

1 Equational Tree Languages

Unification modulo equational theory is a central topic in automated reasoning. Tree automata are the powerful technique for handling unification modulo rewriting [2]. On the other hand, to model some network security problems like Diffie-Hellman key exchange algorithm, rewrite rules and equations (e.g. associativity and commutativity axioms) have to be separately dealt with in the underlying theory, but it causes the situation where the standard tree automata technique is useless. In our recent papers [5, 7], we have proposed an extension of tree automata, which is called equational tree automata. This framework subsumes Petri nets (Example 1). In a practical example, equational tree automata can be used to verify a security problem of Diffie-Hellman protocol (Example 2).

We start this section with basics of tree automata and the equational extension. A *tree automaton* (TA for short) $\mathcal{A}$ is defined by the 4-tuple $(\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{fin}, \Delta)$: each of those components is a signature $\mathcal{F}$ (a finite set of function symbols with fixed arities), a finite set $\mathcal{Q}$ of states (special constants with $\mathcal{F} \cap \mathcal{Q} = \varnothing$), a subset $\mathcal{Q}_{fin}$ of $\mathcal{Q}$ consisting of so-called *final states* and a finite set Δ of transition rules in the following form:

- $f(p_1, \ldots, p_n) \to t$

for some $f \in \mathcal{F}$ with $\mathsf{arity}(f) = n$ and $p_1, \ldots, p_n \in \mathcal{Q}$. The right-hand side t is a term consisting of f and state symbols. A function symbol f in the right-hand side must be the same as one in the left-hand side.

Each of $\mathcal{F}_{\mathrm{A}}$ and $\mathcal{F}_{\mathrm{C}}$ consists of some binary function symbols of the signature $\mathcal{F}$. The intersection of $\mathcal{F}_{\mathrm{A}}$ and $\mathcal{F}_{\mathrm{C}}$ is denoted by $\mathcal{F}_{\mathrm{AC}}$. A set of associativity axioms $f(f(x, y), z) \approx f(x, f(y, z))$ for all $f \in \mathcal{F}_{\mathrm{A}}$ is denoted by $\mathsf{A}(\mathcal{F}_{\mathrm{A}})$. Likewise, a set of commutativity axioms $f(x, y) \approx f(y, x)$ for all $f \in \mathcal{F}_{\mathrm{C}}$ is $\mathsf{C}(\mathcal{F}_{\mathrm{C}})$. The union of of $\mathsf{A}(\mathcal{F}_{\mathrm{AC}})$ and $\mathsf{C}(\mathcal{F}_{\mathrm{AC}})$ is represented by $\mathsf{AC}(\mathcal{F}_{\mathrm{AC}})$. If unnecessary to be explicit,

* This paper is a modified version of the authors' UNIF2002 paper [6].

we write A, C and AC, respectively. An *equational tree automaton* (ETA for short) $\mathcal{A}/\mathcal{E}$ is the combination of a TA $\mathcal{A}$ and a set $\mathcal{E}$ of equations over the same signature $\mathcal{F}$. An ETA $\mathcal{A}/\mathcal{E}$ is called

- *regular* if the right-hand side t is a single state q,
- *monotone* if the right-hand side t is a single state q or a term $f(q_1, \ldots, q_n)$

for every transition rule $f(p_1, \ldots, p_n) \to t$ in Δ. Equational tree automata defined in [4, 5, 7] are in the above monotone case.

A term t in $\mathcal{T}(\mathcal{F})$ is *accepted* by $\mathcal{A}/\mathcal{E}$ if $t \to^*_{\mathcal{A}/\mathcal{E}} q$ for some $q \in \mathcal{Q}_{fin}$. The set of terms accepted by $\mathcal{A}/\mathcal{E}$ is denoted by $\mathcal{L}(\mathcal{A}/\mathcal{E})$. A *tree language* (TL for short) L over $\mathcal{F}$ is some subset of $\mathcal{T}(\mathcal{F})$. A TL L is *$\mathcal{E}$-recognizable* if there exists $\mathcal{A}/\mathcal{E}$ such that $L = \mathcal{L}(\mathcal{A}/\mathcal{E})$. Similarly, L is called $\mathcal{E}$-monotone ($\mathcal{E}$-regular) if $\mathcal{A}/\mathcal{E}$ is monotone (regular). If L is $\mathcal{E}$-recognizable with $\mathcal{E} = \varnothing$, we say L is recognizable. Likewise, we say L is monotone (regular) if L is $\varnothing$-monotone ($\varnothing$-regular). We say $\mathcal{A}/\mathcal{E}$ is a C-TA (A-TA, AC-TA) if $\mathcal{E} = \mathsf{C}$ ($\mathcal{E} = \mathsf{A}$, $\mathcal{E} = \mathsf{AC}$, respectively).

Lemma 1. *Every C-recognizable tree language is regular.*

Proof. We suppose a tree language is recognizable with a C-TA $\mathcal{A}/\mathsf{C}$, where $\mathcal{A} = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{fin}, \Delta)$. Define $\mathcal{B} = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{fin}, \Delta')$ with $\Delta' = \{f(p_1, \ldots, p_n) \to q \mid f(q_1, \ldots, q_n) \to r \in \Delta$ such that $f(p_1, \ldots, p_n) \sim_{\mathsf{C}} f(q_1, \ldots, q_n)$ and $r \to^*_{\mathcal{A}/\mathsf{C}} q\}$. Then it can be proved that the regular TA $\mathcal{B}$ recognizes $\mathcal{L}(\mathcal{A}/\mathsf{C})$. □

Lemma 2. *The following language hierarchy holds if $\mathcal{E} = \mathsf{A}$:*

$$\mathcal{E}\text{-regular TL} \subsetneq \mathcal{E}\text{-monotone TL} \subsetneq \mathcal{E}\text{-recognizable TL}$$

However, the classes of regular TL and $\mathcal{E}$-recognizable TL are incomparable.

Proof. The first inclusion relation is proved in [7]. For the second inclusion, we suppose $\mathcal{F} = \mathcal{F}_0 \cup \{\mathsf{f}\}$ with $\mathcal{F}_{\mathrm{A}} = \{\mathsf{f}\}$. Here $\mathcal{F}_0$ denotes a set of constant symbols. Then, a (word) language W over $\mathcal{F}_0$ is context-sensitive if and only if an A-monotone TL is *maximal* for W. A TL L is called maximal for a language W if for all terms t in $\mathcal{T}(\mathcal{F})$, $\mathsf{leaf}(t) \in W$ if and only if $t \in L$. Similarly, it holds that a language W is recursively enumerable if and only if an A-recognizable TL is maximal for W. It is known that recursively enumerable languages strictly include context-sensitive languages. The difference of the classes of regular TL and $\mathcal{E}$-recognizable TL are proved by taking the TL $L_1 = \{\mathsf{f}(\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{a})\}$ under the assumption of $\mathcal{F}_{\mathrm{A}} = \{\mathsf{f}\}$. The TL L_1 is regular (as it is finite), but it not recognizable with A-TA, because an A-TA which accepts $\mathsf{f}(\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{a})$ also accepts $\mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, \mathsf{a}))$. On the other hand, we take the TL $L_2 = \{t \mid |t|_{\mathsf{a}} = |t|_{\mathsf{b}}\}$ over the signature $\mathcal{F} = \{\mathsf{f}, \mathsf{a}, \mathsf{b}\}$, where $\mathsf{arity}(\mathsf{f}) = 2$ and a, b are constant symbols. If $\mathcal{F}_{\mathrm{A}} = \{\mathsf{f}\}$ then L is A-regular (Lemma 8, [5]), but is not regular. □

Remark 1. We know the same hierarchy holds also for $\mathcal{E} = \mathsf{AC}$, except

$$\mathcal{E}\text{-monotone TL} \subsetneq \mathcal{E}\text{-recognizable TL}.$$

The above relation remains as an open question.

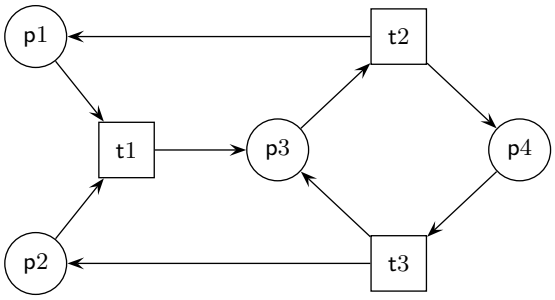


Fig. 1. A Petri net example: $\mathcal{P}$

2 AC-Tree Automata for Unification Problems

In this section we discuss the applications of equational tree automata, in particular AC-tree automata, for unification problems. Our examples rely on the following decidability result.

Theorem 1 (Reachable property problem). *Given a ground AC-TRS* $\mathcal{R}/\mathsf{AC}$ *and tree languages* L_1, L_2 *over* $\mathcal{F}$ *with* $\mathcal{F}_{\mathrm{AC}}$*. If* L_1 *and* L_2 *are* AC*-recognizable tree languages, it is decidable whether there exist some* s *in* L_1 *and* t *in* L_2 *such that* $s \to^*_{\mathcal{R}/\mathsf{AC}} t$*, i.e.* $(\to^*_{\mathcal{R}/\mathsf{AC}})[\,L_1\,] \cap L_2 \neq \varnothing$ *is a computable question.*

Proof. For a singleton $\mathcal{F}_{\mathrm{AC}}$, the proof proceeds in the way of Lemma 4 in [5]. To extend $\mathcal{F}_{\mathrm{AC}}$ by allowing to have arbitrary many AC-symbols, we apply the similar argument of Section 3 in [5]. □

Example 1. Petri nets are known to be a special class of ground AC-TRSs. A Petri net is a triple (P, T, W), where P is a finite set of places, T is a finite set of transitions and W is a weight-function $(P \times T) \cup (T \times P) \to \mathbb{N}$. For instance, the Petri net $\mathcal{P}$ illustrated in Fig. 1 has W with $W(\mathsf{p1}, \mathsf{t1}) = 1$, $W(\mathsf{p2}, \mathsf{t1}) = 1$, $W(\mathsf{p3}, \mathsf{t2}) = 1$, $W(\mathsf{p4}, \mathsf{t3}) = 1$, $W(\mathsf{t1}, \mathsf{p3}) = 1$, $W(\mathsf{t2}, \mathsf{p1}) = 1$, $W(\mathsf{t2}, \mathsf{p4}) = 1$, $W(\mathsf{t3}, \mathsf{p2}) = 1$, $W(\mathsf{t3}, \mathsf{p3}) = 1$. In the figure, places are denoted by circles, and transitions are squares. The value of W determines the weight of directed arcs between places and transitions. Then, the associated ground AC-TRS $(\mathcal{F}, \mathcal{R}/\mathsf{AC})$ is defined by $\mathcal{F} = \{+\} \cup \{\epsilon, \mathsf{p1}, \ldots, \mathsf{p4}\}$, $\mathcal{F}_{\mathrm{AC}} = \{+\}$ and $\mathcal{R} = \{\mathsf{p1}+\mathsf{p2} \to \mathsf{p3},\ \mathsf{p3} \to \mathsf{p1}+\mathsf{p4},\ \mathsf{p4} \to \mathsf{p2}+\mathsf{p3}\} \cup \{\epsilon + \mathsf{p}i \to \mathsf{p}i,\ \mathsf{p}i \to \mathsf{p}i + \epsilon \mid 1 \leqslant i \leqslant 4\}$. In this setting, a state of a Petri net (the number of *tokens* on each place) is encoded by a multiset of place symbols. The empty multiset is represented by ϵ.

Given two sets L_1, L_2 of states of $\mathcal{P}$. According to Theorem 1, it is decidable whether there exist states $m_1 \in L_1$ and $m_2 \in L_2$ such that $m_1 \to^*_{\mathcal{P}} m_2$, provided L_1, L_2 are leaf-languages of AC-recognizable tree languages over $\mathcal{F}$. The binary relation $\to^*_{\mathcal{P}}$ is the reflexive-transitive closure of one-step transition relation. This decidability property generalizes the result of Mayr [3].

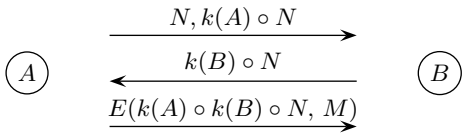


Fig. 2. Diffie-Hellman key exchange algorithm

Using the above property, for instance, we can solve *coverability* problem, which is a question of whether there exists m_3 such that $m_1 \to_{\mathcal{P}}^* m_3$ and $m_2 \subseteq m_3$. Actually, it is verified by solving the following question, which is decidable:

$$\exists\sigma?.\ \ t_1 \to_{\mathcal{R}/\mathsf{AC}}^* t_2 + x\sigma.$$

Here t_1, t_2 are terms over $\mathcal{F}$ such that $\mathsf{leaf}(t_1) = m_1$ and $\mathsf{leaf}(t_2) = m_2$.

Example 2. We consider a simple network protocol. The protocol illustrated in Fig. 2 is called Diffie-Hellman key exchange algorithm (e.g., Section 22.1, [8]). In the protocol, a principal A chooses a prime number N and sends to B together with an integer $k(A) \circ N$ that is generated with a random number $k(A)$. Here we suppose that nobody else can guess $k(A)$ from $k(A) \circ N$. Then B returns $k(B) \circ N$ to A. By assuming $\circ$ to be associative and commutative, $k(A) \circ k(B) \circ N$ can be used as a common secret key for A and B. It enables A to send only B a secret message M encrypted with this key. A security problem for this protocol is whether or not someone else can retrieve a secret message M by listening on the channel.

In term rewriting, the axiom of encryption and decryption and the property of keys are specified by the AC-rewrite system $\mathcal{R} = \{D(x, E(x, y)) \to y\}$ and $\mathsf{AC} = \{x \circ y \approx y \circ x,\ (x \circ y) \circ z \approx x \circ (y \circ z)\}$. On the other hand, a principal C wiretapping the channel can obtain N, $k(A) \circ N$, $k(B) \circ N$ and $E(k(A) \circ k(B) \circ N, M)$. Moreover, C is supposed to have personal data C, $k(C)$ and to be able to use function symbols $D, E, \circ$. So C's knowledge is the set L of terms constructible from these components. Then, the security problem is verified by solving the following unification problem:

$$\exists\sigma?.\ \ x\sigma \to_{\mathcal{R}/\mathsf{AC}}^* M \ \text{ for some } \ x\sigma \in L.$$

In this setting, $(\to_{\mathcal{R}/\mathsf{AC}}^*)[\,L\,]$ is an AC-monotone tree language. One should notice that in order to compute $(\to_{\mathcal{R}/\mathsf{AC}}^*)[\,L\,]$ by using a modified algorithm of Kaji *et al.* [2], *intersection-emptiness* problem for AC-monotone tree languages must be decidable. Obviously a membership problem $M \in (\to_{\mathcal{R}/\mathsf{AC}}^*)[\,L\,]$ is decidable.

Decidability results and closure properties for equational tree languages are summarized in Fig.3. In the figure, the check mark ✓ means "positive" and the cross × is "negative". The question mark ? means "open". If the same result holds in both regular and non-regular cases, it is represented by a single mark in a large column.

		C	A	AC
$\mathcal{L}(\mathring{A}/\mathcal{E}) = \varnothing$?	regular	✓	✓	✓
	non-regular		×	
$\mathcal{L}(\mathring{A}/\mathcal{E}) \subseteq \mathcal{L}(\mathcal{B}/\mathcal{E})$?	regular	✓	×	✓
	non-regular			?
$\mathcal{L}(\mathring{A}/\mathcal{E}) = \mathcal{T}(\mathcal{F})$?	regular	✓	×	✓
	non-regular			?
$\mathcal{L}(\mathring{A}/\mathcal{E}) \cap \mathcal{L}(\mathcal{B}/\mathcal{E}) = \varnothing$?	regular	✓	×	✓
	non-regular			

		C	A	AC
closed under $\cup$	regular	✓	✓	✓
	non-regular			
closed under $\cap$	regular	✓	×	✓
	non-regular		✓	
closed under $\overline{()}$	regular	✓	×	✓
	non-regular		✓	?

Fig. 3. Decidability results and closure properties

References

1. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi: *Tree Automata Techniques and Applications*, 2002. Draft available on `http://www.grappa.univ-lille3.fr/tata/`
2. Y. Kaji, T. Fujiwara and T. Kasami: *Solving a Unification Problem under Constrained Substitutions Using Tree Automata*, JSC 23, pp. 79–117, 1997.
3. E.W. Mayr: *An Algorithm for the General Petri Net Reachability Problem*, SIAM J. Comput. 13(3), pp. 441–460, 1984.
4. H. Ohsaki, H. Seki, and T. Takai: *Recognizing Boolean Closed A-Tree Languages with Membership Conditional Rewriting Mechanism*, Proc. of 14th RTA, 2003. To appear in LNCS.
5. H. Ohsaki and T. Takai: *Reachability and Closure Properties of Equational Tree Languages*, Proc. of 13th RTA, LNCS 2378, pp. 114-128, 2002.
6. H. Ohsaki and T. Takai: *A Tree Automata Theory for Unification Modulo Equational Rewriting*, Proc. of 16th UNIF, 2002.
7. H. Ohsaki: *Beyond Regularity: Equational Tree Automata for Associative and Commutative Theories*, Proc. of 15th CSL, LNCS 2142, pp. 539–553, 2001.
8. B. Schneier: *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Second Edition, John Wiley & Sons, 1996.

Towards a Convenient Category of Topological Domains

Alex Simpson*

LFCS, School of Informatics
University of Edinburgh, Scotland, UK

Abstract

We propose a category of topological spaces that promises to be convenient for the purposes of domain theory as a mathematical theory for modelling computation. Our notion of convenience presupposes the usual properties of domain theory, e.g. modelling the basic type constructors, fixed points, recursive types, etc. In addition, we seek to model parametric polymorphism, and also to provide a flexible toolkit for modelling computational effects as free algebras for algebraic theories. Our convenient category is obtained as an application of recent work on the remarkable closure conditions of the category of quotients of countably-based topological spaces. Its convenience is a consequence of a connection with realizability models.

1 Introduction

The title of this note deliberately echoes that of Steenrod's well-known paper: *A Convenient Category of Topological Spaces*, [46]. In his paper, Steenrod sets out to find a full subcategory of the category **Top** of all topological spaces that is "convenient" for the purposes of algebraic topology. One aspect of "convenience" that is particularly highlighted by Steenrod is cartesian closure, a property famously not enjoyed by **Top** itself. He argues at length that the category of compactly-generated Hausdorff spaces, which *is* cartesian closed, does provide all the "conveniences" needed for doing algebraic topology.

*Research supported by an EPSRC Advanced Research Fellowship and a visiting professorship at RIMS, Kyoto University.

Domain theory was originally developed by Dana Scott to meet the need for a mathematical theory capable of modelling a diversity of computational and programming language phenomena, see [15] for an overview. The goal of this note is to propose a category that is convenient for this purpose, where, by "convenience", we mean that the category should fulfill the general aim as amply as possible.

Such a notion of "convenience" is deliberately vague, and thus open to many interpretations. In Section 2, we formulate several specific demands on a "convenient" category, each elaborating an aspect of the general idea of "convenience". In the remainder of the note, we then develop a notion of domain that promises to meet all the requirements. The domains we settle on are certain topological spaces. Thus, once the programme of research outlined in this note has been completed, we expect to end up with a "convenient" category of "topological domains".

This note presents an early overview of ongoing research. No proofs are given. However, we distinguish clearly between results that have already been established and "conjectures" representing future work.

2 Requirements on a convenient category

In this section we expand upon the notion of "convenience" by placing five explicit requirements on a convenient category. We do not argue that the requirements we identify are the only possible ones, nor that they are essential for a category of domains to qualify as "convenient". Indeed, traditional domain theory has progressed a very long way without meeting all our requirements. The goal of this paper is rather to extend the achievements of traditional domain theory in new directions.

Our first requirement is based on the close connection between topology and computation, as developed, in particular, by Smyth, see [44] for a summary. A domain represents a "datatype" and must therefore have an underlying set of associated "data items". To each domain, it makes sense to associate the set of all "observable" (an abstraction of "semidecidable") properties over data items in the datatype. One can argue that the "logic of observable properties" should be closed under finite conjunctions and arbitrary disjunctions, i.e. that the extensions of the observable properties should form a topology on the underlying set, see [44, 48]. Thus a domain should, at least, be a topological space. Moreover, by allowing observations to make use of maps between domains, one can derive that every map between domains must be continuous. Furthermore, the notion of con-

tinuous function is itself an elegant mathematical abstraction of the notion of computable function, under which the finitary aspects of computability are modelled without recourse to any notion of algorithm. It is thus natural to impose the following requirement.

Requirement 1 (Topological category) The category of domains should be a full subcategory of the category **Top** of topological spaces and continuous functions.

The argument for Requirement 1 has been conceptual. However, an important benefit of the requirement is that domains lie in the realm of mainstream mathematical structures. For one thing, this means that we have a multitude of mathematical tools for manipulating domains. More importantly, the topological structure is indispensible for viewing familiar mathematical objects, such as the real numbers, metric spaces, etc., as embedded within domains. Such embeddings are essential for domains to be used to model computation over many forms of nondiscrete data, as in, e.g., [10, 9].

As a second requirement, we demand that the category of domains support all the standard constructions on domains, including the ability to model recursive definitions of both data and datatypes.

Requirement 2 (Basic structure) The category of domains should model (at least) the usual type constructors: cartesian product, $\times$; function space, $\to$; smash product, $\otimes$; strict function space, $\to_\perp$; coalesced sum, $\oplus$; and lifting, $(\cdot)_\perp$; see [1]. These should have the correct universal properties with respect to the categories of "strict" and "non-strict" maps between domains. The category should also model recursion and recursive types.

Requirement 2 is alone sufficient for modelling many forms of deterministic computation. Nevertheless, even in the realm of deterministic functional computation, recursive types are not sufficiently powerful for all applications. There are many additional forms of "type constructor" that one might also wish to model. In this note, we consider just one such feature, which is particularly important due to its power and its relationship with programming practice.

Requirement 3 (Parametric Polymorphism) The category of domains should model full second-order (parametric) polymorphism.

Here, it should, at least, be possible to incorporate parametricity using *relational parametricity* in the sense of Reynolds, [38, 29]. However, in the

view of the author, it is not settled that relational parametricity is the final story in giving a mathematical account of parametricity. It is possible that alternative accounts of parametric polymorphism may yet emerge.

It is also vital that the category of domains should model a rich variety of programming language features and computational behaviours, going beyond deterministic functional computation. The non-functional aspects of real-world computation can often be encapsulated as *computational effects*. As Plotkin and Power have argued, many such effects are modelled by free algebras for algebraic theories, see [36]. These include the familiar powerdomains used for modelling forms of nondeterminism, as well as many other *computational monads* [31]. Accordingly, we make the following requirement, whose importance was first recognised by Plotkin.

Requirement 4 (Computational effects) The category of domains should provide free algebras for a wide class of algebraic theories.

At this stage, we leave open the extent of the class of theories considered, but it should include, at the very least, all finitary equational algebraic theories.

Finally, we bring in an explicit connection with computability. It might be possible to satisfy the above requirements using topological spaces of large cardinality which have no possible computational significance. One would like a model in which all spaces have potential computational significance. One strong way of ensuring this is:

Requirement 5 (Effectivity) The notion of domain should have an effective counterpart, giving rise to a natural category of computable maps between effective domains, satisfying Requirements 2–4 above.

This requirement also has the direct benefit of establishing notions of computability that apply to all the constructions implicit in Requirements 2–4.

There are several conflicts between the above requirements and traditional domain theory, which we take to be the study of full subcategories of the category of directed-complete partial orders (dcpos) and continuous functions. By definition, all traditional categories of domains do satisfy Requirement 1. As we shall see, it is possible for them to satisfy many of the other requirements independently, but not all requirements in combination.

In order to satisfy Requirement 5, it seems necessary to restrict to full subcategories of *ω-continuous* dcpos, on which notions of computability can be defined using enumerations of the countable bases of such dcpos. The category of ω-continuous dcpos is not cartesian closed; so combining Requirements 5 and 2 requires, at least, finding cartesian-closed full subcate-

gories. Such subcategories have been classified by Jung, see [1] for a survey. There are many, and it is indeed possible to satisfy Requirements 2 and 5 in combination.

Requirement 3 is more problematic. Indeed, to the best of my knowledge, no existing category of domains has been exhibited as a model of *parametric* polymorphism. There do exist models of non-parametric polymorphism, based on dependent product operations on domains, see e.g. [8]. However, Jung has shown that there is an essential difficulty in finding domain-theoretic models of polymorphism that are closed under the convex powerdomain [26]. Thus there is a problem in combining Requirement 4 with (even a non-parametric version of) Requirement 3.

Requirement 4 is also in conflict with Requirements 2 and 5, as the following example, due to Plotkin (private communication), shows. In the category of all ω-continuous dcpos, free algebras are available for any finitary equational theory. However, no nontrivial cartesian-closed full subcategory of ω-continuous dcpos is closed under the formation of free commutative monoids. Thus, in traditional domain theory, it is impossible to simultaneously satisfy Requirements 2, 4 and 5. This difficulty led Plotkin to first pose the problem of finding a category satisfying Requirements 2, 4 and 5 in combination. His recent work with Power on computational effects [36] has highlighted the computational importance of this problem.

The probabilistic powerdomain [40, 24] gives another example of a possible conflict between Requirements 2, 4 and 5. Although the category of all ω-continuous dcpos is closed under the probabilistic powerdomain [23], Jung and Tix have cast doubt on whether any cartesian-closed full subcategory remains closed under it [27]. In this case, there is no definitive negative result. Nonetheless, the formidable technical difficulties in the way of combining ω-continuous dcpos, probabilistic powerdomain and function space raise the question of whether traditional domain theory provides the right setting for combining probabilistic computation with Requirements 2 and 5.

In contrast, it does seem possible to satisfy all of Requirements 2–5, using notions of domain that arise in *realizability models* [28]. The type constructors and recursive types are worked out in [28, 37]. The interpretation of parametric polymorphism is being worked out in detail by Birkedal and Rosolini [7]. An indication of how to address free algebras appeared in [33], although the details there are for the special case of powerdomains only. Furthermore, many realizability models have an intrinsic notion of computability built into them, rendering Requirement 5 superfluous.

One major drawback of realizability models, however, is that the intrinsic mathematical structure of the objects modelling types is intangible. Objects

are given as partial equivalence relations, possibly satisfying additional properties, over a partial combinatory algebra. In general, isomorphic objects may have very different partial equivalence relations underlying them. It is not at all straightforward to extract intrinsic properties of objects from their external presentations as partial equivalence relations.

What we would like instead is a notion of domain given by an explicit definition of the intrinsic mathematical structure involved. Requirement 1, whose many other benefits we have already discussed, is a strong constraint in this direction. It is this requirement that we take as our starting point for deriving our convenient category of domains.

3 Quotients of countably-based spaces

Requirement 1 demands that our category of domains be a full subcategory of **Top**. At the same time, Requirement 5 demands that an effective version of the category be available. Thus our domains must be topological spaces to which it is possible to associate a notion of effectivity. Such a notion of effectivity is known to be available for all countably-based (a.k.a. second-countable) topological spaces. For example, the topology of any such space A can be "presented" using a topological pre-embedding $\pi_A \colon A \longrightarrow \mathcal{P}\omega$,[1] where $\mathcal{P}\omega$ is the powerset of ω with the Scott topology. Given two such presentations $\pi_A \colon A \longrightarrow \mathcal{P}\omega$ and $\pi_B \colon B \longrightarrow \mathcal{P}\omega$, a continuous function $f \colon A \longrightarrow B$ is said to be *effective* if there is some computable $r \colon \mathcal{P}\omega \longrightarrow \mathcal{P}\omega$ (in the standard sense, see [43]) such that $r \circ \pi_A = \pi_B \circ f$.[2] For further discussion of related notions of effectivity, see e.g. [44, §5.1].

Restricting to countably-based spaces is unnecessarily constraining. It turns out to be possible to associate a notion of effectivity with a more general class of space. The spaces we consider are arbitrary quotient spaces of countably-based spaces, i.e. spaces X for which there exists a topological quotient $q \colon A \longrightarrow\!\!\!\!\!\rightarrow X$, where A is countably based. We call such spaces *qcb spaces*, and we write **QCB** for the full subcategory of **Top** consisting of such spaces. Quite unexpectedly, **QCB** has very good categorical structure.

[1] Being a *pre-embedding* means that, for every open $U \subseteq A$, there exists an open $V \subseteq B$ such that $U = \pi_A^{-1}(V)$. A pre-embedding is a topological embedding if and only if it is an injective function.

[2] In general, r only determines f when π_B is a topological embedding, i.e. when B is T_0. A reader who prefers effective maps to be determined by their computational component may adapt the discussion throughout the paper by assuming all spaces to be T_0.

Theorem 3.1 *The category* **QCB** *has all countable limits and colimits and is cartesian closed.*

For proofs of the theorem see [30, 41]. N.b., the countable colimits are inherited from **Top**, but limits are *not* inherited; both finite products and equalizers differ in **QCB** and **Top**.

Theorem 3.1 is a major reason for considering the category of all quotients of countably-based spaces rather than simply restricting to countably-based spaces themselves. The category of countably-based spaces does not have coequalizers, and, more importantly, it is not cartesian closed.

In the remainder of this section we give two other characterizations of qcb spaces. The first shows explicitly how qcb spaces can be provided with an associated notion of effectivity. The second gives a more intrinsic characterization of qcb spaces.

To give an account of effectivity, we introduce the following definitions, which are motivated by the straightforward Proposition 3.4 below.

Definition 3.2 An *ω-representation* of a topological space X is given by a topological quotient $q\colon A \longrightarrow\!\!\!\!\!\rightarrow X$, where A is a countably-based space.

Definition 3.3 An ω-representation $q\colon A \longrightarrow\!\!\!\!\!\rightarrow X$ is said to be *ω-projecting* if, for every countably-based B and continuous $f\colon B \longrightarrow X$, there exists a continuous $g\colon B \longrightarrow A$ such that $q \circ g = f$.

Proposition 3.4 *Suppose that $q\colon A \longrightarrow\!\!\!\!\!\rightarrow X$ and $r\colon B \longrightarrow\!\!\!\!\!\rightarrow Y$ are ω-projecting ω-representations. Then, for any continuous $f\colon X \longrightarrow Y$, there exists a continuous $g\colon A \longrightarrow B$ making the square below commute.*

$$\begin{array}{ccc} A & \xrightarrow{\;g\;} & B \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle r} \\ X & \xrightarrow[\;f\;]{} & Y \end{array}$$

Also, given arbitrary functions f, g making the square commute, if g is continuous then so is f.

Thus ω-projecting ω-representations determine the topological structure of the represented spaces to the extent that continuous maps between X and

Y are completely determined by continuous maps between the representing countably-based spaces A and B.

The above definitions relate to the desire of having an associated notion of effectivity as follows. The notion of an effective map between X and Y can be derived from the notion of effective map between A and B as countably-based spaces. Specifically, we stipulate that a continuous function $f\colon X \longrightarrow Y$ is *effective* if there exists an effective continuous $g\colon A \longrightarrow B$ making the diagram commute. The notion of effectivity thus depends upon presentations of A and B, and also upon q and r, but such dependency is unavoidable. Effectivity is always associated with the presentation of mathematical structure, rather than directly with the structure itself.

We have shown that a notion of effective map is available between those spaces for which there exists an ω-projecting ω-representation. Such spaces were first introduced in [30], where, using a result due to Schröder, it is proved that they coincide with qcb spaces.

Another result of Schröder's gives a more intrinsic characterization of qcb spaces. Recall that the relation of sequence convergence on a topological space X is defined by $(x_i) \to x$ if, for every open $U \ni x$, the sequence (x_i) is eventually in U (i.e. there exists n such that, for all $i \geq n$, $x_i \in U$). A subset $W \subseteq X$ is said to be *sequentially open* if $(x_i) \to x \in W$ implies that (x_i) is eventually in W. Trivially, every open subset is sequentially open. We say that X is *sequential* if every sequentially open subset is open. A *sequential pseudobase* of a topological space X is a family $\mathcal{B}$ of subsets of X satisfying: for every open $U \subseteq X$ and convergent $(x_i) \to x \in U$, there exists $B \in \mathcal{B}$ such that $x \in B \subseteq U$ and (x_i) is eventually in B. This notion is due to Schröder [42]. Note that a family of open sets is a sequential pseudobase if and only if it is a base for the topology.

We can now summarise the main characterizations of qcb spaces.

Theorem 3.5 *The following are equivalent for a topological space X.*

1. *X is a qcb space.*
2. *X has an ω-projecting ω-representation.*
3. *X is sequential and has a countable sequential pseudobase.*

The implication 2 implies 1 is trivial. Lawson has given a direct proof that 1 implies 3 [11]. Bauer gives the construction of an ω-projecting ω-representation from a countable sequential pseudobase for a sequential space [3]. Nevertheless, all the main ingredients of the theorem are, in a

slightly different context, due to Schröder, see [42, Theorem 13] and and [41, Theorem 3.2.4]. The latter result provides another interesting characterization of qcb spaces using a generalization of Weihrauch's notion of Baire space representation [49]. This leads to an alternative (but presumably equivalent) account of effectivity in terms of Type-2 Turing Machines.

There are many ways of understanding the cartesian-closed structure of **QCB**. As in [30], the structure of **QCB** is preserved by an embedding $\mathbf{QCB} \hookrightarrow \mathbf{Equ}$, where **Equ** is Scott's category of *equilogical spaces* [4].[3] This embedding allows one to understand limits in **QCB** in terms of limits in **Equ**, which are defined using limits in **Top**, but with the caveat that spaces are considered modulo an equivalence relation. Alternatively, the cartesian-closed structure of **QCB** can be understood via structure-preserving embeddings into many cartesian-closed coreflective hulls of **Top**. For example, there are such embeddings into the categories of sequential spaces [30], of (not necessarily Hausdorff) compactly-generated spaces, and of core-compactly-generated spaces [11].

The existence of so many structure-preserving embeddings persuades the author that **QCB** is an "inevitable" category, arising as a subcategory of any of the main approaches to reconciling topological continuity and cartesian closure.[4] People differ in whether they prefer to consider cartesian-closed subcategories of **Top**, but lose topological limits; or to consider cartesian-closed supercategories of **Top**, retaining topological limits, but going outside the familiar world of topology. In the author's view, these two alternatives are not in conflict. In either case, **QCB** lives as a full subcategory via a structure-preserving embedding. Moreover, all reasonable spaces lie inside **QCB**. Furthermore, the existence of many categories embedding **QCB** provides a range of alternative tools for understanding constructions in **QCB** (e.g. one can use the sequential function space, the compactly-generated function space, the function space in **Equ**, etc.).

On the other hand, in spite of so many available tools, aspects of the cartesian closure of **QCB** remain hard to understand. For example, consider the "type hierarches" over $\mathbb{N}$ (with the discrete topology) and $\mathbb{R}$ (with the Euclidean topology) given by $\mathbb{N}, \mathbb{N}^{\mathbb{N}}, \mathbb{N}^{\mathbb{N}^{\mathbb{N}}}, \ldots$ and $\mathbb{R}, \mathbb{R}^{\mathbb{R}}, \mathbb{R}^{\mathbb{R}^{\mathbb{R}}}, \ldots$. Here, $\mathbb{N}^{\mathbb{N}^{\mathbb{N}}}$ and $\mathbb{R}^{\mathbb{R}^{\mathbb{R}}}$ are examples of non-countably-based spaces that are nonetheless qcb spaces. It is easily shown that these spaces are Hausdorff. Also, $\mathbb{N}^{\mathbb{N}^{\mathbb{N}}}$ is totally disconnected. As the following open questions illustrate, other basic

[3] Because we are not requiring qcb spaces to be T_0, the T_0 condition in the definition of equilogical space must be omitted.

[4] It remains to be checked that there is a structure-preserving embedding of **QCB** into Hyland's category of "filter spaces" [18]. The author strongly expects this to be the case.

properties of their topologies remain, however, tantalisingly elusive.

Question 3.6 Are the spaces $\mathbb{N}^{\mathbb{N}^{\mathbb{N}}}$ and $\mathbb{R}^{\mathbb{R}^{\mathbb{R}}}$ regular?

Question 3.7 Is $\mathbb{N}^{\mathbb{N}^{\mathbb{N}}}$ zero dimensional (i.e. does it have a base of clopen subsets)?

Question 3.7 was posed in [5], where an application of a positive answer to the question is given.

4 Topological (pre)domains

Quotients of countably-based spaces form, apparently, the largest class of topological spaces to which it is possible to associate a notion of effectivity. The category **QCB** thus addresses Requirements 1 and 5. Moreover, Theorem 3.1 shows that the category has surprisingly rich categorical structure. Nonetheless, it does not satisfy Requirements 2–4. In this section, we address Requirement 2, by cutting down to a full subcategory of **QCB**.

It is easiest to address Requirement 2 by identifying, in the first place, a category of *predomains* within **QCB**. In traditional domain theory, predomains are distinguished from domains by not being required to have least element in the partial order. (Thus predomains are dcpos, and domains are *pointed* dcpos.) This relaxation allows, for example, the category of predomains to have countable coproducts. Although endomorphisms on predomains need not have fixed points, the familiar cartesian-closed category of domains, which does have a fixed-point operator, is recovered simply as the full sucategory of those predomains that do have least element. The fixed-point operator exists because partial orders are required to be directed complete and because continuous functions (with respect to the Scott topology) preserve directed suprema. In fact, as is well known, the weaker properties of ω-completeness and ω-continuity suffice.

In traditional domain theory, the topology (the Scott topology) is derived from the partial order. To define our notion of predomain, we also work with order-theoretic properties, but we take the topology as primary and the order as derived. Recall that the *specialization order* $\sqsubseteq$ on a topological space X is defined by $x \sqsubseteq y$ if, for all open $U \subseteq X$, $x \in U$ implies $y \in U$. In general $\sqsubseteq$ is a preorder on X. The space X is said to be T_0 if $\sqsubseteq$ is a partial order. We can now give a definition of topological predomain.

Definition 4.1 (Topological predomain) A *topological predomain* is a topological space X satisfying the following properties:

1. X is a qcb space;

2. $\sqsubseteq$ is an ω-complete partial order (in particular, X is T_0); and

3. every open $U \subseteq X$ is inaccessible by ω-lubs (i.e., for any ascending sequence $x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \ldots$, if $\bigsqcup_i x_i \in U$ then $x_i \in U$ for some i).

Apart from being phrased with respect to ω-lubs rather than directed lubs, this definition recalls the notion of *monotone convergence space* [14].

Definition 4.2 A topological space X is a *monotone convergence space* if: the specialization order on X is a dcpo (in particular, X is T_0), and every open subset of X is Scott-open with respect to the order.

Monotone convergence spaces include: all T_1 spaces, all sober spaces, and all dcpos with the Scott topology.

Proposition 4.3 *A qcb space is a topological predomain if and only if it is a monotone convergence space.*

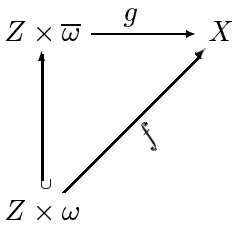
Thus, because of the restriction to qcb spaces, it makes no difference whether topological predomains are defined using ω-lubs or using directed lubs.

Next, we give a useful category-theoretic characterization of topological predomains. We write ω for the set of natural numbers under the Alexandroff topology on the their usual linear ordering. We write $\overline{\omega}$ for $\omega \cup \{\infty\}$, where $\infty = \bigsqcup_i i$, with the Scott topology.

Proposition 4.4 *In the category* **QCB** *the following are equivalent.*

1. *X is a topological predomain.*

2. *For all qcb spaces Z and maps $f\colon Z \times \omega \longrightarrow X$, there exists a unique $g\colon Z \times \overline{\omega} \longrightarrow X$ such that the diagram below commutes:*



We write **TP** for the category of topological predomains and continuous functions.

Corollary 4.5 **TP** *is an exponential ideal of* **QCB**.

Explicitly, this means that: (i) for topological predomains X and Y, the product $X \times Y$ in **QCB** is a topological predomain; and (ii) for any qcb space X and topological predomain Y, the function space Y^X in **QCB** is a topological predomain. Thus, in particular, **TP** is cartesian closed and the embedding $\mathbf{TP} \hookrightarrow \mathbf{QCB}$ preserves the cartesian-closed structure. It is also easy to see, directly from the definition of topological predomain, that **TP** is closed under countable coproducts in **QCB** (and hence in **Top**). The existence of more intricate (co)limits is a consequence of the result below.

Proposition 4.6 (Schröder) *The category of monotone convergence spaces is a full reflective subcategory of* **Top**, *and the reflection functor cuts down to a refection* $R\colon \mathbf{QCB} \to \mathbf{TP}$.

Corollary 4.7 **TP** *has all countable limits and colimits.*

Limits in **TP** are inherited from **QCB**, but colimits are constructed using the reflection. So neither limits nor colimits in **TP** are inherited, in general, from **Top**. (This is a fact of life, not a problem.)

We now briefly discuss how Requirement 2 has been met. First, another definition is needed.

Definition 4.8 (Topological domain) A *topological domain* is a topological predomain for which $\sqsubseteq$ has a least element.

The various type constructors listed in Requirement 2 can all be defined on topological domains entirely in the expected way. Their universal properties are also as expected. The important categories here are: **TD**, the category of continuous functions between topological domains; and $\mathbf{TD}_\perp$, its subcategory of strict maps (those preserving the least element). The category **TD** is cartesian closed (it is an exponential ideal of **TP**), with a least-fixed-point operator; the category $\mathbf{TD}_\perp$ is symmetric monoidal closed, with respect to $\otimes$ and $\to_\perp$, and has $\oplus$ as coproduct. Moreover, the inclusion of $\mathbf{TD}_\perp$ in **TD** has a left adjoint, given by $(\cdot)_\perp$, with the adjunction giving rise to a model of intuitionistic linear type theory. Furthermore, $\mathbf{TD}_\perp$ is also characterized as the Eilenberg-Moore category of the monad given by $(\cdot)_\perp$ on **TP**.[5] Finally, the standard technology for recursive domain equations, [45],

[5] As in ordinary domain theory, $\mathbf{TD}_\perp$ is also equivalent to the Kleisli category of the lifting monad.

applies directly to $\mathbf{TD}_\perp$, which can be shown to be algebraically compact in the sense of Freyd [12, 13].

We end this section by relating **TP** and **TD** to traditional categories of domains. Firstly, note that **TP** contains very many topological spaces that are not normally included in categories of predomains; for example: n-dimensional Euclidean space $\mathbb{R}^n$, and also $\mathbb{R}^\omega$; Baire space, $\mathbb{N}^\omega$; Cantor space 2^ω; etc. Also, by Corollary 4.5, the type hierarchies $\mathbb{N}, \mathbb{N}^\mathbb{N}, \mathbb{N}^{\mathbb{N}^\mathbb{N}}, \ldots$ and $\mathbb{R}, \mathbb{R}^\mathbb{R}, \mathbb{R}^{\mathbb{R}^\mathbb{R}}, \ldots$ are in **TP**.

Obviously, **TP** contains every dcpo whose Scott topology has a countable base. In particular, every ω-continuous dcpo is contained in **TP**. Actually, it can be shown that a continuous dcpo is contained in **TP** if and only if it is ω-continuous. Thus the extra generality of allowing non-countably-based spaces does not manifest itself with continuous dcpos.[6] As **TP** is cartesian closed, it should be interesting to examine function spaces over the notorious examples of ω-algebraic dcpos that are not contained within any of the cartesian-closed categories of ω-continuous cpos, see e.g. [1]. We have not yet performed the required calculations, but we strongly expect that such function spaces in **TP** go outside the category of dcpos, i.e. that they result in spaces whose topology is not the Scott topology.

On the other hand, well-behaved categories of dcpos do turn out to live faithfully inside **TD** (and so also inside **TP**). Let $\omega\mathbf{S}$ be the category of continuous functions between ω-continuous Scott domains (i.e. bounded-complete pointed ω-continuous dcpos).[7]

Proposition 4.9 *$\omega\mathbf{S}$ is an exponential ideal of* **QCB** *(hence also of* **TP**, *and of* **TD**).

Conjecture 4.10 *$\omega\mathbf{S}$ is the largest full subcategory of pointed ω-continuous dcpos that forms an exponential ideal of* **TD**.

Being an exponential ideal is a very strong requirement. A weaker requirement is merely to ask for the existing cartesian-closed structure of a category of domains to be preserved. Jung has identified the largest cartesian-closed full subcategory, $\omega\mathbf{FS}$, of the category of pointed ω-continuous dcpos [25].

Conjecture 4.11 *The embedding $\omega\mathbf{FS} \hookrightarrow \mathbf{TD}$ preserves the cartesian-closed structure.*

[6] This is a special case of a more general result: a core-compact space is qcb if and only if it is countably based, see [11].

[7] One reason for requiring Scott domains to be pointed is that the category of all bounded-complete ω-continuous dcpos is not cartesian closed.

5 Topological predomains and realizability

Proposition 4.9 and, if true, Conjecture 4.11 show that, to a large extent, traditional domain theory lives faithfully inside **TD**. The benefit of the much richer world provided by **TD** (and also **TP**) is that Requirements 3 and 4 can also, apparently, be addressed. To appreciate this, we now give an alternative characterization of the category **TP** using the techniques of realizability semantics.

In this section, we assume the reader has some acquaintance with realizability models, as presented in [28]. We only consider models built over Scott's combinatory algebra $\mathcal{P}\omega$ [43]. We write $\mathbf{Asm}(\mathcal{P}\omega)$ for the associated category of *assemblies* and $\mathbf{Mod}(\mathcal{P}\omega)$ for its full subcategory of *modest sets*. Using the terminology of [28], consider the *divergence* $D = \{\emptyset\}$. As in [28, §4], the divergence determines a *lifting functor* L and *dominance* Σ. Using the lifting functor L, we identify the notion of a *complete* object of $\mathbf{Asm}(\mathcal{P}\omega)$, see [28, §5]. Using the dominance Σ, we define the notion of an *extensional* object of $\mathbf{Asm}(\mathcal{P}\omega)$, see [28, §10]. Let $\mathbf{CE}(\mathcal{P}\omega)$ be the full subcategory of complete extensional objects of $\mathbf{Asm}(\mathcal{P}\omega)$. As in [28], $\mathbf{CE}(\mathcal{P}\omega)$ is in fact a full subcategory of $\mathbf{Mod}(\mathcal{P}\omega)$. Moreover, for very general reasons, $\mathbf{CE}(\mathcal{P}\omega)$ is a well-behaved category of predomains in $\mathbf{Mod}(\mathcal{P}\omega)$; see, for example, [32] (where extensional objects are called *regular* Σ*-posets*).

In fact, $\mathbf{CE}(\mathcal{P}\omega)$ is much more that a well-behaved category of predomains. It is a complete internal full subcategory of $\mathbf{Asm}(\mathcal{P}\omega)$ in the sense explained in [19, 21]. This implies many properties: $\mathbf{CE}(\mathcal{P}\omega)$ is a full reflective subcategory of $\mathbf{Asm}(\mathcal{P}\omega)$; as an internal category, $\mathbf{CE}(\mathcal{P}\omega)$ is cocomplete; $\mathbf{CE}(\mathcal{P}\omega)$ is a model of the polymorphic λ-calculus; and, by [39, 7], this model can be refined to yield a model of parametric polymorphism. Finally, following the approach of Phoa and Taylor [33], it should be possible to use the internal completeness of $\mathbf{CE}(\mathcal{P}\omega)$ to construct free algebras for (at least finitary) algebraic theories. Although the details of this construction need further work, there seems to be no fundamental obstacle to it succeeding. In summary, the category $\mathbf{CE}(\mathcal{P}\omega)$ promises to satisfy Requirements 3 and 4.

At this point, the reader may be wondering what all this has to do with topological (pre)domains. The connection is provided by the result below.

Theorem 5.1 *The categories* **TP** *and* $\mathbf{CE}(\mathcal{P}\omega)$ *are equivalent.*

This theorem is a consequence of properties of the embedding of **QCB** into the category of countably-based equilogical spaces given in [30], using the fact that the latter category is equivalent to $\mathbf{Asm}(\mathcal{P}\omega)$ [5]. Several further

ingredients are needed in the proof, including the characterization of **TP** given by Proposition 4.4.

6 Summary and further work

We have introduced a category **TP** of *topological predomains*, with a subcategory **TD** of *topological domains*, that together promise to meet our 5 requirements for "convenience". In the definition, Requirements 1 and 5 were explicitly taken account of by the use of qcb spaces, as in Section 3. Requirement 2 was established directly in terms of the the topological definition of the categories **TP** and **TD**. On the other hand, our arguments for Requirements 3 and 4 were indirect, making use of a connection with realizability models. (It should be repeated that the argument for Requirement 4 is not yet rigorous.)

As was emphasised in Section 2, one of the major benefits of working with a category of topological spaces is that one is working with (at least to some extent) familiar mathematical structures. We thus view it as very desirable to establish Requirements 3 and 4 in direct topological terms. It should not be necessary to take a detour through a realizability model in order to understand the interpretation of polymorphic types and computational effects.

In the case of Requirement 3, our results indicate the existence of a purely topological model of parametric polymorphism. In fact, by constructing the model over $\mathbf{TD}_\perp$, it should be possible to obtain a concrete model combining parametric polymorphism, intuitionistic linear type theory and fixed points. As has been argued by Plotkin [34, 35], such a combination of features is immensely powerful. The existence of realizability models of this setting was first outlined by Plotkin, and is being worked out in detail (and in much greater generality) by Birkedal and Rosolini [7]. A syntactic model, based on a term calculus quotiented by operational equivalence, has been presented by Bierman, Pitts and Russo [6]. However, to the best of my knowledge, the only existing model defined in terms of concrete mathematical structures is due to R. Hasegawa, using his bicategorical theory of *twiners* [16]. Our work demonstrates that it is also possible to obtain a purely topological model of linear parametricity and fixed points. It is to be hoped that a direct topological interpretation of parametric polymorphism in the model will prove forthcoming.

For Requirement 4, the presence of arbitrary countable colimits in **TD** suggests that free algebras should be available for algebraic theories that

are, in a suitable sense, countably presented. For defining free algebras for computational effects, countable presentations should be of similar usefulness to the *countable enriched Lawvere theories* of Plotkin and Power [36], although the details will necessarily be different as **TD** is not locally countably presentable.[8] It is plausible that, in **TD**, it will also be possible to allow operations of uncountable arity, as long as the "arity" is given by a countably-based space, possibly even an arbitrary qcb space. Precise conditions on the algebraic theories should emerge when Requirement 4 is worked out in detail.

In addition to providing the general construction needed to verify Requirement 4, it will also be of interest to look at specific examples of computationally interesting effects; for example, probabilistic choice, which is traditionally modelled by the probabilistic powerdomain [40, 24]. As has already been reported, in the context of traditional domain theory, there are (possibly insurmountable) difficulties in combining the probabilistic powerdomain with any cartesian-closed category of continuous dcpos [27]. However, it is well established that the probabilistic powerdomain lives naturally in wider categories of topological spaces than dcpos, see e.g. [17, 2]. Our categories **TP** and **TD** will allow such a wider topological notion of probabilistic powerdomain to be combined with the usual domain-theoretic constructions (as summarized in Requirement 2) and also with polymorphism. At the time of writing, we have an explicit definition of a probabilistic powerdomain in **TP**. However, much remains to be done to relate it to the established definitions for ω-continuous dcpos (and wider classes of spaces). Also, we would like to characterize the probabilistic powerdomain as a free algebra, as in [23].

Another very interesting topic for future research is to combine Requirements 3 and 4 in the stronger sense of extending the notion of parametric polymorphism to incorporate parametricity for the operators associated with computational effects.

In this note, we have used effectivity as a motivating factor in the identification of qcb spaces and in the subsequent definition of topological (pre)domains. However, Requirement 5 has only been dealt with in a cursory manner. In particular, we have glossed over one important issue. The simple account of effectivity for **QCB** given in Section 3 is, in itself, insufficient as an account of effectivity in **TP**. The problem is that an effective version of Proposition 4.4 is required in order for domain-theoretic constructions, such as fixed points, to exist in the category of effective maps. In order to

[8] It is almost certainly impossible for any locally presentable category to satisfy either of Requirements 3 and 5.

implement this concretely, additional effective information is needed in the "presentation" of a topological predomain. Obtaining a palatable explicit description of this effective structure is one possible goal for future research. Moreover, it should also be established that all the constructions needed in fulfilling Requirements 2–4 behave well with respect to effective maps.

There is, however, an alternative more conceptual approach to effectivity in **TP**. This is, simply, to extract the notion of effectivity and its properties as a consequence of all constructions being formalizable in the internal logic of the realizability category $\mathbf{Asm}(\mathcal{P}\omega)$, or rather in its associated subcategory of effective maps. This approach should, of course, be equivalent to the external approach discussed above.

Ultimately, it will thus be beneficial to have both the topological account of **TD** and the alternative realizability account worked out in detail. Moreover, the realizability account suggests other perspectives. Theorem 5.1 gives one example of a situation in which the category of complete extensional objects in a realizability category turns out to have a very elegant concrete description (as the category **TP**). This is consistent with the established observation that, in many different realizability models, the complete extensional objects form the category of predomains with the most natural external decription, see the introduction to [32] for other examples. Nevertheless, several other categories of predomains are available in realizability models, see [28, 32] for overviews. It would be particularly interesting to obtain a concrete description of the category $\mathbf{Rep}(\mathcal{P}\omega)$ of *replete objects* in $\mathbf{Asm}(\mathcal{P}\omega)$, as defined by Hyland and Taylor [20, 47], which is nicely characterized as the smallest full reflective exponential ideal of $\mathbf{Asm}(\mathcal{P}\omega)$ containing the object Σ. Considered as a category of topological spaces, $\mathbf{Rep}(\mathcal{P}\omega)$ forms a category of predomains contained in **TP**. This containment is strict. For example, the space of natural numbers with the topology of cofinite subsets is a topological predomain that is not replete. This space is a well-known example of a T_1 space that is not sober. Indeed, the very definition of repleteness suggests a connection between repleteness and sobriety. In an attempt to make this connection rigorous, recall that the category of sober topological spaces is a full reflective subcategory of **Top**, see e.g. [22]. The following question, which should be compared with Proposition 4.6, seems nontrivial.

Question 6.1 Is the sober reflection of a qcb space also a qcb space?

If (and only if) the answer to this question is positive, then it holds that the category $\mathbf{Rep}(\mathcal{P}\omega)$ is equivalent to the category of sober qcb spaces.

Such an equivalence would provide a very elegant concrete description of the category $\mathbf{Rep}(\mathcal{P}\omega)$. This would be interesting as, at the time of writing, no concrete description of the category of replete objects in a realizability model has ever been established.

Acknowledgements

The material in this note was first presented at the Dagstuhl workshop, *Mathematical Structures for Computable Topology and Geometry*, in May 2002, and subsequently in an invited talk at the ALGI workshop, RIMS, Kyoto University, December 2002. I am grateful to the organisers for invitations to attend these events. In particular, I thank Masahito Hasegawa for inviting me to RIMS and for encouraging me to write this note for the ALGI proceedings. I have benefited from discussions with Martín Escardó, Reinhold Heckmann, Jimmie Lawson, Gordon Plotkin and Matthias Schröder. Paul Taylor's diagram macros were used.

References

[1] S. Abramsky and A. Jung. Domain theory. In *Handbook of Logic in Computer Science, Vol. 3*, pages 1–168. OUP, 1994.

[2] M. Alvarez-Manilla. Extension of valuations on locally compact sober spaces. *Topology Appl.*, 124:397–433, 2002.

[3] A. Bauer. A relationship between equilogical spaces and type two effectivity. *Electronic Notes in Theoretical Computer Science*, 45, 2001.

[4] A. Bauer, L. Birkedal, and D.S. Scott. Equilogical spaces. *Theoretical Computer Science*, to appear.

[5] A. Bauer, M.H. Escardó, and A.K. Simpson. Comparing functional paradigms for exact real-number computation. In *Proceedings ICALP 2002*, pages 488–500. Springer LNCS 2380, 2002.

[6] G.M. Bierman, A.M. Pitts, and C.V. Russo. Operational properties of Lily, a polymorphic linear lambda calculus with recursion. *Electronic Notes in Theoretical Computer Science*, 41, 2000.

[7] L. Birkedal and G. Rosolini. A parametric domain-theoretic model for the polymorphic lambda calculus. In preparation, 2003.

[8] Th. Coquand, C.A. Gunter, and G. Winskel. Domain theoretic models of polymorphism. *Information and Computation*, 81:123–167, 1989.

[9] A. Edalat. Domains for computation in mathematics, physics and exact real arithmetic. *Bulletin of Symbolic Logic*, 3:401–452, 1997.

[10] M.H. Escardó. PCF extended with real numbers. *Theoretical Computer Science*, 162:79–115, 1996.

[11] M.H. Escardó, J.D. Lawson, and A.K. Simpson. Core compactly generated spaces. In preparation, 2003.

[12] P.J. Freyd. Algebraically complete categories. In *Category Theory, Proceedings Como 1990*, pages 95–104. Springer LNM 1488, 1991.

[13] P.J. Freyd. Remarks on algebraically compact categories. In *Applications of Categories in Computer Science*, pages 95–106. LMS Lecture Notes 177, CUP, 1992.

[14] G. Gierz, K.H. Hofmann, K. Keimel, J.D. Lawson, M.W. Mislove, and D.S. Scott. *A Compendium of Continuous Lattices*. Springer-Verlag, 1980.

[15] C.A. Gunter and D.S. Scott. Semantic domains. In *Handbook of Theoretical Computer Science, Vol. B*, pages 633–674. Elsevier, 1990.

[16] R. Hasegawa. The theory of twiners and linear parametricity. In preparation, 2003.

[17] R. Heckmann. Spaces of valuations. *Annals of the New York Academy of Sciences*, 806:174–200, 1996.

[18] J.M.E. Hyland. Filter spaces and continuous functionals. *Annals of Mathematical Logic*, 16:101–143, 1979.

[19] J.M.E. Hyland. A small complete category. *Annals of Pure and Applied Logic*, 40:135–165, 1988.

[20] J.M.E. Hyland. First steps in synthetic domain theory. In *Category Theory, Proceedings Como 1990*, pages 131–156. Springer LNM 1488, 1991.

[21] J.M.E. Hyland, E.P. Robinson, and G. Rosolini. The discrete objects in the effective topos. *Proc. Lond. Math. Sci.*, 60:1–36, 1990.

[22] P.T. Johnstone. *Stone Spaces*. CUP, 1982.

[23] C. Jones. *Probabilistic Non-determinism*. PhD thesis, University of Edinburgh, 1990.

[24] C. Jones and G.D. Plotkin. A probabilistic powerdomain of evaluations. In *Proceedings 4th IEEE Symposium on Logic in Computer Science*, pages 186–195, 1989.

[25] A. Jung. The classification of continuous domains. In *Proceedings 5th IEEE Symposium on Logic in Computer Science*, pages 35–40, 1990.

[26] A. Jung. The dependent product construction in various categories of domains. *Theoretical Computer Science*, 79:359–363, 1991.

[27] A. Jung and R. Tix. The troublesome probabilistic powerdomain. *Electronic Notes in Theoretical Computer Science*, 13, 1998.

[28] J.R. Longley and A.K. Simpson. A uniform account of domain theory in realizability models. *Math. Struct. in Comp. Sci.*, 7:469–505, 1997.

[29] Q. Ma and J.C. Reynolds. Types, abstraction and parametric polymorphism, part 2. In *Mathematical Foundations of Programming Semantics*, pages 1–40. Springer, 1992.

[30] M. Menni and A.K. Simpson. Topological and limit space subcategories of countably-based equilogical spaces. *Math. Struct. in Comp. Sci.*, 12, 2002.

[31] E. Moggi. Notions of computation and monads. *Information and Computation*, 93:55–92, 1991.

[32] J. van Oosten and A.K. Simpson. Axioms and (counter)examples in synthetic domain theory. *Annals of Pure and Applied Logic*, 104:233–278, 2000.

[33] W.K.-S. Phoa and P. Taylor. The synthetic Plotkin powerdomain. Unpublished manuscript, 1990.

[34] G.D. Plotkin. Second-order type theory and recursion. Talk at the Scott Fest, 1993.

[35] G.D. Plotkin. Type theory and recursion. Invited talk at *8th IEEE Symposium on Logic in Computer Science*, 1993.

[36] G.D. Plotkin and A.J. Power. Computational effects and operations: an overview. *Electronic Notes in Theoretical Computer Science*, to appear. Special Issue for Prodeedings Domains VI.

[37] B. Reus and Th. Streicher. General synthetic domain theory — a logical approach. *Math. Struct. in Comp. Sci.*, 9:177–223, 1999.

[38] J.C. Reynolds. Types, abstraction and parametric polymorphism. In *Information Processing '83*, pages 513–523. North Holland, 1983.

[39] E.P. Robinson and G. Rosolini. Reflexive graphs and parametric polymorphism. In *Proceedings 9th IEEE Symposium on Logic in Computer Science*, pages 364–371, 1994.

[40] N. Saheb Djahromi. CPO's of measures for nondeterminism. *Theoretical Computer Science*, 12:19–37, 1980.

[41] M. Schröder. *Admissible Representations for Continuous Computations.* PhD thesis, FernUniversität, Hagen, 2002.

[42] M. Schröder. Extended admissibility. *Theoretical Computer Science*, 284:519–538, 2002.

[43] D.S. Scott. Data types as lattices. *SIAM J. Comput.*, 5:522–587, 1976.

[44] M.B. Smyth. Topology. In *Handbook of Logic in Computer Science, Vol. 1*, pages 641–761. OUP, 1992.

[45] M.B. Smyth and G.D. Plotkin. The category-theoretic solution of recursive domain equations. *SIAM J. Comput.*, 11:761–783, 1982.

[46] N.E. Steenrod. A convenient category of topological spaces. *Michigan Math. J.*, 14:133–152, 1967.

[47] P. Taylor. The fixed point property in synthetic domain theory. In *Proceedings 6th IEEE Symposium on Logic in Computer Science*, pages 152–160, 1991.

[48] S. Vickers. *Topology via Logic.* CUP, 1989.

[49] K. Weihrauch. *Computable Analysis.* Springer, 2000.

Channel Theory as a Philosophical Experiment (Abstract)

Atsushi Shimojima　下嶋篤
Japan Advanced Institute of Science and Technology　北陸先端科学技術大学院大学

In this short tutorial, I will introduce basic concepts of channel theory (Barwise and Seligman 1997), with an emphasis on philosophical motivations behind them. Specifically, we will see how the theory has been designed to give a general, naturalistic theory of information and thereby that of "representations", where the term broadly refers to symbols, pictures, and physical models in everyday life, as well as such abstract objects as mathematical models.

References

Jon Barwise and Jerry Seligman, *Information Flow: the Logic of Distributed Systems*, Cambridge University Press: Cambridge, 1997.

Ideas in Logic and Computer-Science related to Ludics

Misao Nagayama (永山 操)
Tokyo Woman's Christian Univ. (東京女子大学文理学部)
misao@twcu.ac.jp

Abstract

This note reports author's current thoughts and observations in logic and computer-science related to ludics.

1 Motivations.

When we work in proof theory, our basic objects are proofs and formulas. A proof consists of a series of inference rules: In order to successfully accumulate inference rules up to the axioms, which is supposed to the goal, we normally develop a heuristics. It is to become a proof-search algorithm. Such algorithms are often embedded into meta-mathematical proofs for the completeness theorems. We can name some proof-search algorithms, for example, resolution and unification for propositional and first-order logic, Skolem's theorem and Herbrand's theorem for first-order logic, and focalization for linear logic.

We would like to investigate such proof-search algorithms in terms of conputational relevance. We like to develop some formal system in which these algorithms themselves are to be investigated meta-mathematically. We expect such a formal system to provide

- a more general and abstract notion extending traditional notions in logic, and
- a concretion of proof-search algorithms.

We could see λ-calculus as such a formal system. Emphasized in a phrase "terms as programs", λ-terms are not only an abstraction of mathematical functions, but also concrete programs we know how to evaluate them. In summary, the λ-terms are both

- a more general and abstract notion of mathematical functions, and
- a concretion of programs.

We hope for a similar formal system not for mathematical functions but for proof-search algorithms. Based on this idea we study ludics introduced by Girard ([Gi]). Our topics are organized as follows:

- Section 2: Inference rules.
- Section 3: Axioms.
- Section 4: Variables.
- Section 5: Proofs.
- Section 6: Formulas.
- Section 7: Normalization.

2 Inference rules.

We started our discussion without mentioning any particular formal system. In order to analyze proofs further, we choose one specific formal system, namely Sequent calculus. Sequent calculus is equipped with an axiom scheme of simple form $A \vdash A$, and inference rules: the structural rules, and the left introduction and the right introduction rules for the logical connectives. We simply refer the right introduction rule for a logical connective as its inference rule.

Sequent calculus is preferred because of its internal completeness, namely the cut-elimination theorem. Thanks to the theorem, the search space for proofs becomes much smaller. An application of the last inference rule introduces the outermost logical connective. A bottom-up proof-search in a cut-free system boils down to find the last rule. One step to find the last rule can be thought as a partial function which provides an inference rule for a given ordered sequence of inference rules up to the current point. In fact, this idea leads us to the notion of innocent functions in game semantics ([HO],[FH]). We discuss their work in the last section.

In linear logic, the logical connectives $\&$, $\mathcal{P}$ and $\forall$ are negative, while $\oplus$, $\otimes$ and $\exists$ are positive. Let us call an inference rule for a negative connective and for a positive one by a negative rule and by positive one, respectively. The notion of the polarity arises in the following aspects:

- The law of distributivity,
- the reversibility and determinism,
- focalization and the priority in proof-search algorithms.

The law of distributivity holds:

$$A \otimes (B \oplus C) \sim (A \otimes B) \oplus (A \otimes C)$$
$$\exists x(A \otimes B(x)) \sim A \otimes \exists x B(x)$$
$$A \mathcal{P} (B \& C) \sim (A \mathcal{P} B) \& (A \mathcal{P} C)$$
$$\forall x(A \mathcal{P} B(x)) \sim A \mathcal{P} \forall B(x).$$

In linear logic, the negative rules are deterministic and reversible: For case of &, the inference rule is:

$$\frac{\vdash A, \Gamma \quad \vdash B, \Gamma}{\vdash A \& B, \Gamma} \ (\&).$$

Conversely, we can prove each premise from the conclusion. In the following, premise $\vdash A, \Gamma$ is derived.

$$\frac{\vdash A \& B, \Gamma \quad \dfrac{\vdash A, A^{\perp}}{\vdash A, A^{\perp} \oplus B^{\perp}} \ (\oplus)}{\vdash A, \Gamma} \ (Cut).$$

For case of $\mathcal{P}$, the inference rule is:

$$\frac{\vdash A, B, \Gamma}{\vdash A \mathcal{P} B, \Gamma} \ (\mathcal{P}).$$

Similarly, we can prove the premise from the conclusion:

$$\frac{\vdash A \mathcal{P} B, \Gamma \quad \dfrac{\vdash A^{\perp}, A \quad \vdash B^{\perp}, B}{\vdash A, B, A^{\perp} \otimes B^{\perp}} \ (\otimes)}{\vdash A, B, \Gamma} \ (Cut).$$

For case of $\forall$,

$$\frac{\vdash A[y/x], \Gamma}{\vdash \forall x A, \Gamma} \ (\forall),$$

where variable y is not free in Γ. We can prove the premise from the conclusion.

$$\frac{\vdash \forall x A, \Gamma \quad \dfrac{\vdash A[y/x]^{\perp}, A[y/x]}{\vdash \exists x A^{\perp}, A[y/x]} \ (\exists)}{\vdash A[y/x], \Gamma} \ (\&).$$

On the other hand, the positive rules are non-deterministic and irreversible.

The focalization allows us to treat a cluster of negative rules, and a cluster of positive ones, as *synthetic* connectives. We give a higher priority to the negative rules than to the positive ones. Thus we have the following proof search algorithm:

- Choose one of the formulas in the conclusion and a negative rule: Apply the rule and decompose the formula. Notice that this is deterministic, and the other formulas in the context remain the same.
- Continue choosing a negative rule, until it is no longer available in the conclusion.
- Choose one of the formulas in the conclusion and a positive rule: Apply the rule and decompose the formula. Notice that this is non-deterministic, the context may split in a non-recoverable manner (in the case of $\otimes$), or one of the sub-formulas may be lost (in the case of $\oplus$).

As for the negativity of the universal quantifier $\forall$, let us consider a provable formula in first-order classical logic

$$\exists x \forall y (P(x) \to P(y)).$$

We build its cut-free proof in classical sequent calculus in a bottom-up way. Then we soon realize that we need to defer our irreversible choice to avoid making a crucial error, instead, to apply the Contraction rule. Thus we eliminate $\exists$ in one of the two formulas in the conclusion. Then we are left with two formulas, whose outermost quantifiers are $\forall$ and $\exists$, respectively:

$$\dfrac{\dfrac{\vdash \forall y (P(x) \to P(y)), \exists x \forall y (P(x) \to P(y))}{\vdash \exists x \forall y (P(x) \to P(y)), \exists x \forall y (P(x) \to P(y))}\,(R\exists)}{\vdash \exists x \forall y (P(x) \to P(y))}\,(Cont.)$$

Which quantifier should we eliminate first, $\forall$ or $\exists$? Our proof-search algorithm above tells us that we should choose a negative connective when possible, which is $\forall$, in this case. Thus we further proceed the derivation:

$$\dfrac{\dfrac{\dfrac{\vdash P(x) \to P(z), \exists x \forall y (P(x) \to P(y))}{\vdash \forall y (P(x) \to P(y)), \exists x \forall y (P(x) \to P(y))}\,(R\forall)}{\vdash \exists x \forall y (P(x) \to P(y)), \exists x \forall y (P(x) \to P(y))}\,(R\exists)}{\vdash \exists x \forall y (P(x) \to P(y))}\,(Cont.)$$

Introducing a fresh variable z in the top inference rule is very tricky; there seems no particular reason to do this at this point. We think that this is an example of variables used as an anonymous representative. We think that Sequent calculus does not have a capacity to describe how to introduce a fresh variable and to use it effectively.

After this crucial step, we need to eliminate $\exists$ in the left formula using the variable

z, which is again a little tricky:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash P(x) \to P(z), \forall y(P(z) \to P(y))}{\vdash P(x) \to P(z), \exists x \forall y(P(x) \to P(y))}\,(R\exists)}{\vdash \forall y(P(x) \to P(y)), \exists x \forall y(P(x) \to P(y))}\,(R\forall)}{\vdash \exists x \forall y(P(x) \to P(y)), \exists x \forall y(P(x) \to P(y))}\,(R\exists)}{\vdash \exists x \forall y(P(x) \to P(y))}\,(Cont.)$$

The last step is matching so that a pair of $P(z)$ should later be used to supply an axiom. Matching is completed by the positive rule for $\exists$. This is an irreversible and non-deterministic step, leading us to the successful end of the proof. Moreover, we can interpret this step in a game semantic manner: The initial negative rule (Opponent) offers a function $P(z)$, or options $P(z)$ (in the sense that different z's provide various options), and the next positive rule (Player) responds to it by supplying $P(z)$ by a suitable variable. Here the variable z is chosen. Finally, we conclude our proof as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(z) \vdash P(z)}{P(x), P(z) \vdash P(z), P(y)}\,(Weak.)}{P(x) \vdash P(z), P(z) \to P(y)}\,(R\to)}{\vdash P(x) \to P(z), P(z) \to P(y)}\,(R\to)}{\vdash P(x) \to P(z), \forall y(P(z) \to P(y))}\,(R\forall)}{\vdash P(x) \to P(z), \exists x \forall y(P(x) \to P(y))}\,(R\exists)}{\vdash \forall y(P(x) \to P(y)), \exists x \forall y(P(x) \to P(y))}\,(R\forall)}{\vdash \exists x \forall y(P(x) \to P(y)), \exists x \forall y(P(x) \to P(y))}\,(R\exists)}{\vdash \exists x \forall y(P(x) \to P(y))}\,(Cont.)$$

It will be interesting to investigate further a computational aspect of the polarity in proof-search algorithms. In ludics, the inference rules are introduced not based on the left or right, but based on the polarity.

- Daimon:
$$\dfrac{}{\vdash \Lambda}\,\dagger$$
- Positive rule (one premise for each $i \in I, \Lambda_i$'s pairwise disjoint and included in Λ):
$$\dfrac{\cdots \xi i \vdash \Lambda_i \cdots}{\vdash \xi, \Lambda}\,(+, \xi, I)$$
- Negative rule (one premise for each $J \in \mathcal{N}$, all Λ_J's included in Λ):
$$\dfrac{\cdots \vdash \xi * J, \Lambda_J, \cdots}{\xi \vdash \Lambda}\,(-, \xi, \mathcal{N})$$

The basic objects in ludics are *designs*. Syntactically, designs are trees built from the rules above.

3 Axioms.

We observe the following two distinctive roles, syntactical and semantical, played by axioms normally in any proof system:

- Syntactical: To denote the terminals or initials of accumulated sequences of inference rules, and
- Semantical: To denote concepts taken for granted, such as valid formulas, typically the identity $A \vdash A$.

The first property leads us to two positive designs interpreted as terminals in ludics:

- *Daimon* $\mathfrak{Dai}$, consisting of † itself,

$$\frac{}{\vdash \Lambda}\ \dagger,$$

  which means convergence, while

- the *partial design* Ω, the empty set,

$$\frac{}{\vdash \Lambda}\ \Omega,$$

  which means divergence.

Besides being designs themselves, each daimon and Ω can be used, just as an axiom, as a rule to end a tree of inference rules at top of the tree.

The daimon † and the partial design Ω are unique maximal and minimal designs with respect to the observational order for the designs [Gi].(See also [C](p.15) for its formal definition.)

Besides such an algebraic treatment, Curien emphasizes a similarity between recoverable errors in computer science and daimon in ludics: Errors and daimon help us to terminate computation and to interactively explore the behavior of programs or proofs. Moreover, computation is stream-like, or demand-driven. This view helps us to better understand the observational order $\preceq$ in Girard's notation, in such a way that for designs $\mathfrak{D}_1$ and $\mathfrak{D}_2$, $\mathfrak{D}_1 \preceq \mathfrak{D}_2$ holds iff $\mathfrak{D}_2$ is more likely to terminate or converge in computation than $\mathfrak{D}_1$ is. Furthermore, the observational order allows us to analyze a structural difference between the two designs it orders. For example, a greater design $\mathfrak{D}_2$ can be obtained from $\mathfrak{D}_1$ by as follows; we replace either

- an Ω in $\mathfrak{D}_1$ with a tree $\neq \Omega$, or
- a sub-tree in $\mathfrak{D}_1$ by a $\dagger$,

to obtain $\mathfrak{D}_2$.

However, replacing "axiom-like" objects freely by something else seems against the second role of the axioms; namely, axioms denote something valid.

As for the gap we just mentioned, let us examine an example in semantics of Intutionistic predicate logic, which is taken from a text by Ono ([O]).

Let $(M, \leq, U, \models)$ be a Kripke model satisfying the following conditions:

1. $(M, \leq)$ is an ordered set,
2. $U : M \longrightarrow \mathcal{P}(W)$ is a map from M to the power set of some set W, satisfying the two conditions;
   (a) for any $a \in M$, $U(a) \neq \emptyset$ holds,
   (b) for any $a, b \in M$, $a \leq b$ implies $U(a) \subseteq U(b)$.
3. $\models$ is a binary relation between an element $a \in M$ and a closed formula A such that
   (a) $a \models P(\underline{u}_1, \cdots, \underline{u}_m) \Longleftrightarrow (\underline{u}_1, \cdots, \underline{u}_m) \in P^{I(a)}$,
   (b) $a \models A \wedge B \Longleftrightarrow a \models A$ and $a \models B$,
   (c) $a \models A \vee B \Longleftrightarrow a \models A$ or $a \models B$,
   (d) $a \models A \rightarrow B \Longleftrightarrow \forall b(a \leq b \Rightarrow (b \not\models A \text{ or } b \models B))$,
   (e) $a \models \neg A \Longleftrightarrow \forall b(a \leq b \Rightarrow b \not\models A)$,
   (f) $a \models \forall x A \Longleftrightarrow \forall b \forall u(a \leq b \Rightarrow (u \in U(b) \Rightarrow b \models A[\underline{u}/x]))$,
   (g) $a \models \exists x A \Longleftrightarrow \exists u(u \in U(b) \wedge a \models A[\underline{u}/x])$,

   where $I(a)$, for each $a \in M$, of the interpretation of the predicate symbols I is defined so that (1) $P^{I(a)} \subseteq U(a)^m$ and (2) $a \leq b$ implies $P^{I(a)} \subseteq P^{I(b)}$; and $\underline{u}$ stands for the name of u.

A formula A is valid in the frame $(M, \leq, U)$ iff it is true in the Kripke model $(M, \leq, U, \models)$ for any valuation $\models$.

Let M be a set satisfying the condition that for any $a \in M$, there exists a maximal element $a^*(\geq a)$ in M. Then formula

$$\forall x \neg\neg P(x) \rightarrow \neg\neg \forall x P(x)$$

is valid in the frame $(M, \leq, U)$. Let us demonstrate how to prove the validity of this formula. Firstly,

$$(1)\ \ b \models \forall x \neg\neg P(x)$$

is equivalent to

$$(1)'\ \ \forall c \forall u \forall d \exists e (b \leq c \Rightarrow (u \in U(c) \Rightarrow (c \leq d \Rightarrow (d \leq e \wedge e \models P[\underline{u}/x])))).$$

And

$$(2)\ \ b \models \neg\neg \forall P(x)$$

is equivalent to

$$(2)'\ \ \forall c \exists d \forall e \forall u (b \leq c \Rightarrow (c \leq d \wedge (d \leq e \Rightarrow (u \in U(c) \Rightarrow e \models P[\underline{u}/x])))).$$

We assume condition (1) and imply condition (2). The condition (2)' is of prenex form whose quantifiers are $\forall c \exists d \forall e \forall u$, and for any c instantiating the first quantifier $\forall c$, the second existential quantifier is ought to be instantiated by c^*. Thanks to the maximality of c^*, the next $\forall$ is trivially instantiated only by c^*. Thus the condition (2)' boils down to

$$(2)''\ \forall u (u \in U(c^*) \Rightarrow c^* \models P[\underline{u}/x]).$$

On the other hand, the condition (1)' is of prenex form whose quantifiers are $\forall c \forall u \forall d \exists e$, and the first quantifier $\forall c$ is ought to be instantiated by c^*. Again, thanks to the maximality of c^*, the rest of the quantifiers are all trivially instantiated only by c^*. Therefore the condition (1)' boils down to

$$(1)''\ \forall u (u \in U(c^*) \Rightarrow c^* \models P[\underline{u}/x]).$$

Clearly (1)" and (2)" are identical, and we have shown the claim.

Notice that the role played by the maximal element c^* reminds us the role of daimon. It is maximal, and it stops further inquiries for instantiating quantifiers in a non-trivial way. Moreover, instantiating a universal quantifier in (1) and an existential quantifier in (2) syntactically corresponds to (L $\forall$) and (R $\exists$), respectively in Sequent calculus: The polarities of these rule are both positive, as same as the polarity of daimon. It will be interesting to study further proof-search algorithms to check the validity of formulas in the Kripke frame, and to investigate its computational relevance.

Finally, the second role of the axiom leads us to a design called *Fax*, intended as the identity, or rather its infinite η-expansion in ludics. It plays a role of a function mapping one formula to the other isomorphic one. Fax $\mathfrak{Fax}_{\xi,\xi'}$ is the following design:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\ \mathfrak{Fax}_{\xi' * \mathfrak{i}, \xi * \mathfrak{i}} \\ \cdots \xi' * i \vdash \xi * i \cdots\end{array}}{\cdots \vdash \xi', \xi * I \cdots}\,(\xi', I)}{\xi \vdash \xi'}\,(\xi, \mathcal{P}_f(\mathcal{N}))$$

See Faggian et als. [FFDQ] for further discussions on Fax.

4 Variables.

In [AC] by Amadio and Curien, it is explained that the categorical interpretation of λ-calculus in CCC can be seen as a way of compiling a language with variables into a languages without them. The slogan there is that variables are replaced by projections. In other words, rather than giving a symbolic reference in the form of variable, one provides a path for accessing a certain information in the context. This is the starting point to define an abstract machine, which provides a high-level description of data-structures and algorithms used to efficiently reduce λ-terms.

In this section, we discuss a well-known partial algorithm to check the validity of a formula based on Skolem's and Herbrand's theorems. We think that the combination of two theorems provides us a usage of variables in logic, which seems to share the same perspective mentioned in the slogan above. As for the formulations of the theorems, we shall again count on the text by Ono ([O]).

In first-order predicate logic, Skolem's theorem provides us, for a given formula, a prenex existential formula called *Skolem normal form*: It is valid iff so is the original one, provided that we consider the former validity in the models for an extended language with new constants and function symbols. A formula

$$\forall x \exists y \exists z \forall w P(x, y, z, w)$$

has a Skolem normal form

$$\exists y \exists z P(c, y, z, f(y, z)),$$

In the Skolem normal form, the universal quantifiers in the original formula disappear, and the quantified variables x, w are substituted for the terms made of newly introduced a 0-ary constant c and a binary function symbol f, respectively. The parameters of each c and f are determined by the variables existentially quantified, between the one universally quantified at work and the another one, universally quantified on the left, closest to the one at work.

Let $\mathfrak{L}$ a fixed language containing at least one constant symbols. Let *Herbrand universe*, denoted by $H_{\mathfrak{L}}$, be the collection of variable-free terms of $\mathfrak{L}$. *Herbrand structure* $< H_{\mathfrak{L}}, J >$ for language $\mathfrak{L}$ is a structure such that for each variable-free term t in $\mathfrak{L}$ is, via J, interpreted by term t in $H_{\mathfrak{L}}$ itself. Notice that the former t is to be interpreted is a syntactical object, while the latter t in $H_{\mathfrak{L}}$ is a semantical one. Thus the notion of Herbrand structures introduces objects both syntactical and semantical.

Let A be a quantifier-free predicate formula

$$P(s) \to (Q(s, t) \vee P(t)),$$

where P, Q are predicate symbols unary and binary, respectively, and s, t are terms without variables.

Atomic predicates $P(s)$ and $Q(s,t)$, containing different predicate symbols P an Q respectively, are evaluated independently. In other words, each P and Q refers, just as a pointer, independently to a value in the Herbrand structure. On the other hand, if s, t are different terms, say, containing distinct function symbols, then they are interpreted by distinct elements in the Herbrand universe. Therefore, predicates $P(s)$ and $P(t)$ are evaluated independently. In other words, each s, t refers via its function symbols playing their role as a pointer, independently to a value in the Herbrand structure.

The observation above reminds us the view presented earlier in this section, that is, replacing a symbolic reference in the form of variable, one provides a path for accessing a certain information in the context. In our discussions, we replaced symbolic references in the form of universally quantified variable by a path or a pointer played by function symbols, thanks to the Skolem normalization, referring mutually independently to the values in the Herbrand universe, which is the context in our setting.

The proposition below introduces a decidable algorithm to check the validity of any quantifier-free formula A with no free variables.

Proposition 4.1 *Let A be a quantifier-free formula with no free variable in $\mathfrak{L}$. Then A is valid iff A is true in any Herbrand structure for $\mathfrak{L}$. Moreover, A is valid iff the propositional formula $\pi(A)$ is tautology; where $\pi(A)$ is obtained from A by replacing mutually distinct atomic predicates by corresponding mutually distinct propositional variables.*

5 Proofs.

A proof is built as a tree of inference rules. Normally, a proof-search algorithm provides us how to accumulate inference rules successfully up to the axioms, which is the goal.

Faggian and Hyland describe in [FH] that designs, the basic objects in ludics, are both:

- an abstraction of formal proofs, and
- a concretion of their semantical interpretation.

These two aspects of designs agree with our motivations presented in the first section: That is, we are looking for:

- a more general and abstract notion of formal logic, and
- a concretion of proof-search algorithms.

Let us remember Herbrand's theorem: The theorem allows us to check the validity of closed existential prenex formulas by means of a partial algorithm.

Theorem 5.1 (Herbrand) *Let $\exists x_0 \cdots x_n B$ be a closed existential prenex formula in language $\mathcal{L}$, where formula B contains no quantifier symbol. Then $\exists x_0 \cdots x_n B$ is valid iff there exists a natural number m (≥ 1) and terms $t_{i1}, \cdots, t_{in}$ $(i = 1, \cdots m)$ of Herbrand universe such that*

$$B[t_{11}/x_1, \cdots, t_{1n}/x_n] \vee \cdots \vee B[t_{m1}/x_1, \cdots, t_{mn}/x_n]$$

is true in any Herbrand structure for $\mathcal{L}$.

The notions of the structures, the universe and the validity in the theorem definitely suggest that the theorem be semantical. However, based on the theorem, we obtain a partial algorithm to check the validity of a closed formula: There is where, we believe, semantics and proof-search algorithms meet. A good algorithm is coupled with a good data structure. In our view, a good data structure in the partial algorithm here is a growing list of tuples of variable-free terms in the Herbrand universe: The list starts as the empty set; and the list is added tuples of terms mentioned in the theorem in the course of the partial algorithm up to the list:

x_1	$\cdots$	x_n
t_{11}	$\cdots$	t_{1n}
t_{21}	$\cdots$	t_{2n}
$\cdots$	$\cdots$	$\cdots$
t_{m1}	$\cdots$	t_{mn}

When the list arrives at this stage, the given formula $\exists x_0 \cdots x_n B$ is shown to be valid. It will be interesting to investigate how the above data structure is maintained by the partial algorithm, and if and how it is related to Sequential algorithms due to Berry and Curien ([AC]).

Thus we have started with a proof as a tree of inference rules, and proposed a "proof-search algorithms as semantics" view. In the next section, we shall discuss another aspect of a proof, namely its syntax in Sequent calculus.

6 Formulas.

A proof in Sequent calculus is a tree of sequents; and a sequent contains formulas. One role of formulas in a proof is to denote the location. However, a formula does not simply denote a location, but also denote a proof description, by which we can tell, how to merge two disjoint proofs into one, preserving its correctness. For example, a pair of cut-formulas indicate us how to merge two disjoint proofs via normalization.

The designs are supposed to be an abstraction of formal proofs. and the role of formulas as a proof description in fact is played by designs, which itself is the counterpart of proofs in ludics. In other words, the conceptual distinction between proofs and

formulas are less clear in ludics than in the typed λ-calculus, where emphasized are "Formulas as types" and "Proofs as terms" in the Curry-Howard isomorphism.

We shall study the notion of types in ludics by means of filter models, that is, the syntax of intersection types. (Chapter 3.3, p. 54. [AC]).

Definition 6.1 (ets, p. 55, [AC]) *Let $(D, \bullet)$ be an applicative structure. Consider the following operation on subsets of D:*

$$A \to B = \{d \in D : \forall e \in A\ (d \bullet e \in B)\}.$$

A subset of $\mathcal{P}(D)$ is called an extended type structure (ets for short) if it is closed under finite set theoretical intersections and under the operation $\to$ just defined.

An *extended abstract type structure (eats for short)* is given by a preorder $(S, \leq)$, called the carrier, whose elements are often called types: For the definition, refer to Definition 3.3.1 [AC]. The following lemma provides us a means to obtain an eats from an ets defined above.

Lemma 6.2 (Lemma 3.3.7, p. 55, [AC]) *An ets, ordered by inclusion is an eats.*

A filter of an inf-semi-lattice S can be defined in a standard way. (See Definition 3.3.8, p. 56, [AC], for instance.) The filter domain of an eats S is the set $\mathcal{F}(S)$ of filters of S, ordered by inclusion.

Lemma 6.3 (Lemma 3.3.10, p. 56, [AC]) *If S is an eats, then $\mathcal{F}(S)$ is a complete algebraic lattice.*

In ludics, we have an applicative structure defined by the collection of designs denoted by T, equipped with binary operation $Plays(\mathfrak{D}; \mathfrak{E})$. As a subset of $\mathcal{P}(T)$, we take *behaviors*, where a behavior is a set of designs equal to its biorthogonal. As for discussions on the behaviors, we follow [Gi].

The behaviors of the same base is closed under the set theoretical intersection.

Definition 6.4 (Inter, [Gi]) *Let $\mathbf{G}_k$ be a family of behaviors of the same base. Then we define $\bigcap_k \mathbf{G}_k$ as the intersection of the $\mathbf{G}_k$.*

The connective $\bigcap$ is strictly commutative and associative.

Theorem 6.5 (Theorem 14, [Gi]) *Let $\mathfrak{F}$ and $\mathfrak{A}$ be negative and positive designs, respectively. There there exists a unique design $(\mathfrak{F})\mathfrak{A}$, such that for any positive design $\mathfrak{B}$, the following holds.*

$$\ll \mathfrak{F} \mid \mathfrak{A} \odot \mathfrak{B} \gg = \ll (\mathfrak{F})\mathfrak{A} \mid \mathfrak{B} \gg$$

In the theorem above, $\mathfrak{A} \odot \mathfrak{B}$ is the commutative tensor product only defined for positive designs (refer to Definition 31). The binary operation $\multimap$ for ets in ludics is given by a negative behavior $\mathbf{G} \multimap \mathbf{H}$, which is defined as $\mathbf{G} \bowtie \mathbf{H}$ (Definition 34), for two alien behaviors $\mathbf{G}$ positive and $\mathbf{H}$ negative. (As for alienation, see Definition 39.)

$$\mathbf{G} \multimap \mathbf{H} = \{\mathfrak{A} \odot \mathfrak{B}; \mathfrak{A} \in \mathbf{G}, \mathfrak{B} \in \mathbf{H}^{\perp}\}^{\perp}$$

Proposition 6.6 (Proposition 12, [Gi]) *Let behaviors* $\mathbf{G}$ *and* $\mathbf{H}$ *be positive and negative, respectively; then*

$$\mathfrak{F} \in \mathbf{G} \multimap \mathbf{H} \Leftrightarrow \forall \mathfrak{A}(\mathfrak{A} \in \mathbf{G} \Rightarrow (\mathfrak{F})\mathfrak{A} \in \mathbf{H}).$$

Furthermore,

$$\mathfrak{F} \in \mathbf{G} \multimap \mathbf{H} \Leftrightarrow \forall \mathfrak{A}(\mathfrak{A} \in \mathbf{H}^{\perp} \Rightarrow (\mathfrak{F})\mathfrak{A} \in \mathbf{G}^{\perp}).$$

It will be interesting to investigate further an eats structure of the behaviors $\mathbf{BV}$. We discuss a connection between filters of the behaviors and *incarnations*.

Let $\mathfrak{D}$ a fixed design in $\mathbf{G}$. Then the set of designs in $\mathbf{G}$ included in $\mathfrak{D}$ is a non-empty family: The intersection of the family, called the *incarnation*, also belongs to $\mathbf{G}$, due to Closure theorem (Theorem 7).

Definition 6.7 (Incarnation, [Gi]) *Let* $\mathbf{G}$ *and* $\mathfrak{D}$ *be a behavior and its design, respectively. The incarnation* $\mid \mathfrak{D} \mid_{\mathbf{G}}$ *is defined as follows:*

$$\mid \mathfrak{D} \mid_{\mathbf{G}} = \bigcap \{\mathfrak{D}'; \mathfrak{D}' \subset \mathfrak{D} \text{ and } \mathfrak{D}' \in \mathbf{G}\}.$$

The incarnation belongs to $\mathbf{G}$.

Now the following proposition is immediate.

Proposition 6.8 (A filter by incarnation.) *A collection* $x_{\mathfrak{D}}$ *of behaviours*

$$x_{\mathfrak{D}} = \{\mathbf{G}; \mid \mathfrak{D} \mid_{\mathbf{G}} \subset \mathfrak{D}\}$$

forms a filter.

Proof. In order to show that $x_{\mathfrak{D}}$ is a filter, we check the following two conditions:

- $\mathbf{G}, \mathbf{H} \in x_{\mathfrak{D}} \Rightarrow \mathbf{G} \cap \mathbf{H} \in x_{\mathfrak{D}}$,
- $\mathbf{G} \in x_{\mathfrak{D}}$ and $\mathbf{G} \subset \mathbf{H} \Rightarrow \mathbf{H} \in x_{\mathfrak{D}}$

The first condition is due to Theorem 9 on the intersection and incarnation: We obtain

$$| \mathfrak{D} |_{\mathbf{G}\cap\mathbf{H}} = | \mathfrak{D} |_{\mathbf{G}} \cup | \mathfrak{D} |_{\mathbf{H}} .$$

The second condition is due to the contravariance easily shown from the definition of the incarnation, i.e.,

$$\mathbf{G} \subset \mathbf{H} \Rightarrow | \mathfrak{D} |_{\mathbf{H}} \subset | \mathfrak{D} |_{\mathbf{G}} .$$

■

We conjecture that the filters $\mathfrak{F}(\mathbf{BV})$ on behaviors is a complete algebraic lattice. We note that there are maps between $\mathfrak{F}(\mathbf{BV})$ and the collection $\mathbf{T}$ of designs,

$$x : \mathbf{T} \longrightarrow \mathfrak{F}(\mathbf{BV})$$

defined as $x(\mathfrak{D}) = x_{\mathfrak{D}}$, and

$$\mathfrak{D} : \mathfrak{F}(\mathbf{BV}) \longrightarrow \mathbf{T}$$

defined by

$$\mathfrak{D}(x) = \bigcup_{\mathbf{G}\in x} | \mathfrak{D} |_{\mathbf{G}} .$$

It will be interesting to further investigate these maps, a complete algebraic lattice structure in $\mathfrak{F}(\mathbf{BV})$, and its connection to one of the main results, stated in Theorem 10, of the additives in ludics:

$$| \mathbf{G}\&\mathbf{H} | = | \mathbf{G} | \times | \mathbf{H} |,$$

where $| \mathbf{G} |$ is the collection of designs $\mathfrak{D}$ in $\mathbf{G}$, satisfying $\mathfrak{D} = | \mathfrak{D} |_{\mathbf{G}}$.

7 Normalization.

Faggian and Hyland investigate ludics by means of HON game semantics in [FH], which we follow closely in this section. Here we develop syntax of designs as an abstraction of formal proofs. In particular, we pay attention to the following three roles of the formulas.

- A location, relative and absolute.
- An initial sequent.
- A class of proofs characterized by normalization.

We have corresponding notions in ludics, respectively:

- An address.

- A base.
- A behavior.

An *action* is a pair (ξ, I) of an address ξ, called a *focus* and I a finite set of natural numbers. An action corresponds to the application of an inference rule. A *base* is a sequent of addresses corresponding to the initial sequent of the derivation, or the conclusion of the proof.

Thanks to the focalization, it suffices to consider only sequents of form $\Xi \vdash \Lambda$, Ξ consisting at most one element and finite Λ.

A design is given by a base and a tree of actions. A tree of actions can be thought as a set of Sequent calculus branches [FH]. The precise definition is given by a set of mutually *coherent chronicles*, where our intentions are

- Chronicles: A formal branch in a focalized sequent calculus derivation.
- Coherence: A condition to let a set of chronicles all belong to the same proof.

Definition 7.1 (Chronicles, [FH]) *A chronicle of base $\Xi \vdash \Lambda$ is a sequence of actions $< \kappa_0, \kappa_1, \cdots, \kappa_n >$ such that:*

- *Alternation.*
- *Positive and Negative focuses.*
- *Destruction of focuses.*
- *Daimon. Daimon can only appear as the last action.*

Definition 7.2 (Coherence, [FH]) *The chronicles $\mathfrak{c}_1, \mathfrak{c}_2$ are coherent when*

- *Comparability. Either one extends the other, or they first differ on negative actions,*
- *Propagation. If $\mathfrak{c}_1, \mathfrak{c}_2$ first differ on κ_1, κ_2 with distinct focuses, then all ulterior focuses are distinct.*

Definition 7.3 (Designs, [Gi])) *A design $\mathcal{D}$ of base $\Xi \vdash \Lambda$ is a set of chronicles of base $\Xi \vdash \Lambda$ such that:*

- *Arborescence. $\mathfrak{D}$ is closed under restriction.*
- *Coherence. The chronicles of $\mathfrak{D}$ are pairwise coherent.*
- *Positivity. If $\mathfrak{c} \in \mathfrak{D}$ has no extension in $\mathfrak{D}$, then its last action is positive.*

- *Totality. If the base is positive, then $\mathfrak{D}$ is non-empty.*

The actions and interactions of designs correspond to moves and plays in the game semantic setting. For the simplicity, the bases of designs $\mathfrak{D}$ and $\mathfrak{E}$ are $\vdash<>$ and $<>\vdash$, respectively.

Definition 7.4 (Linear positions, [FH]) *A sequence of actions s is a linear position or a play, it it satisfies the following conditions:*

- *Alternation. Parity alternates.*
- *Justification. Each action is either initial or is justified by an earlier action.*
- *Linearity. Any address appears at most once.*
- *Daimon. Daimon can only appear as the last action.*

The notions of chronicles and linear positions are very close.

Definition 7.5 (Plays (binary), [FH]) *We define a play $\mathcal{P}$ denoted as $Plays(\mathfrak{D};\mathfrak{E})$ as follows:*

$$\begin{array}{l}\epsilon \in \mathcal{P}\\ p \in \mathcal{P} \text{ is a } P \text{ to play position and} \lceil p \rceil^P \kappa \in \mathfrak{D},\ \text{then } p\kappa \in \mathcal{P}\\ p \in \mathcal{P} \text{ is an } O \text{ to play position and} \lceil p \rceil^O \kappa \in \mathfrak{E},\ \text{then } p\kappa \in \mathcal{P}\end{array}$$

We note that $Plays(\mathfrak{D};\mathfrak{E})$ is totally ordered by the initial segment. Thus we denote the sup of $Plays(\mathfrak{D};\mathfrak{E})$ by $[\mathfrak{D} \rightleftharpoons \mathfrak{E}]$, which is possibly infinite.

Definition 7.6 (Dispute, Convergence, [FH]) *A sequence of actions $[\mathfrak{D} \rightleftharpoons \mathfrak{E}]$ is called a* dispute*, if it is finite and terminated with a Daimon. The normalization between designs $\mathfrak{D}$ and $\mathfrak{E}$ converges, if $[\mathfrak{D} \rightleftharpoons \mathfrak{E}]$ is a dispute, and diverges, otherwise.*

Definition 7.7 (Orthogonality, [FH]) *A design $\mathfrak{D}$ is called* orthogonal *to $\mathfrak{E}$, when the normalization between them converges. The design $\mathfrak{E}$ is called a counter-design of $\mathfrak{D}$.*

A *legal position* is a linear position satisfying the *visibility* condition (Def.7, [FH]).

Proposition 7.8 (Chronicles, Fact 2, Prop.1 [FH]) *Let p be a linear position in $Plays(\mathfrak{D};\mathfrak{E})$. Then,*

- *for any $q \sqsubseteq r^P \sqsubseteq p$, $\lceil q \rceil^P$ is a chronicle of $\mathfrak{D}$, and similarly,*
- *for any $q \sqsubseteq r^O \sqsubseteq p$, $\lceil q \rceil^O$ is a chronicle of $\mathfrak{E}$.*

- *Moreover p is a legal position.*

Due the last statement in the proposition above, any dispute is a legal position. Thus the chronicles in ludics correspond to the linear positions in game semantics; and the plays or disputes correspond to the legal positions.

Thus from a sequence of actions p in $Plays(\mathfrak{D};\mathfrak{E})$, the chronicles of $\mathfrak{D}$ and $\mathfrak{E}$ are obtained by view operations $\lceil\rceil^P, \lceil\rceil^O$, respectively. Conversely, for a given finite legal position p on the universal arena, we can extract a design and a counter-design such that the dispute between them is p and minimal among such pairs of designs. (Proposition 2, [FH]).

Definition 7.9 (Plays (unary), [FH]) *For a given design $\mathfrak{D}$, we define a unary Plays as:*

$$Plays(\mathfrak{D}) = \bigcup\{Plays(\mathfrak{D};\mathfrak{E}); \mathfrak{E} \textit{is a counter-design}\}$$

The following proposition explains $Plays(\mathfrak{D})$ in terms of $Plays(\mathfrak{D};\mathfrak{E})$ and $\mathfrak{D}$.

Proposition 7.10 (Designs, Facts 3,4,5, [FH]) *Let $\mathfrak{D}, \mathfrak{E}$ be a pair of a design and a counter-design. Then,*

- $Plays(\mathfrak{D}) \cap Plays(\mathfrak{E}) = Plays(\mathfrak{D};\mathfrak{E})$,
- $\mathfrak{D} \subseteq Plays(\mathfrak{D})$,
- $Plays(\mathfrak{D}) = \{p : p$ *is a legal position; and for all* $q(\sqsubseteq r^+ \sqsubseteq p)$ *satisfies* $\lceil q\rceil \in \mathfrak{D}\}$.

An abstract notion characterizing $Plays(\mathfrak{D})$ is the notion of *strategies* (Def. 9, [FH]). In particular, $Plays(\mathfrak{D})$ is an *innocent* strategy (Def. 10, [FH]).

Definition 7.11 (Views, Def. 11, [FH]) *Let S be an X-strategy. We define*

$$Views(S) = \{\lceil q\rceil^X : q \sqsubseteq r^+, r \in S\}.$$

The following proposition summarizes the relationship between $Views$ and $Plays$.

Proposition 7.12 (Fact 11, [FH]) *Let $\mathfrak{D}$ be a design, and S be an innocent strategy satisfying propagation. Then,*

- $View(S)$ *is a design, and*

$$Plays(Views(S)) = S,$$

- $Plays(\mathfrak{D})$ *is the smallest innocent strategy containing $\mathfrak{D}$, and*

$$Views(Plays(\mathfrak{D})) = \mathfrak{D}.$$

As we mentioned earlier, $Plays(\mathfrak{D})$ is an innocent strategy. Hence, we apply construction $Views$ to $Plays(\mathfrak{D})$, and we obtain

$$Views(Plays(\mathfrak{D})) = \{\lceil q \rceil : q \sqsubseteq r^+, r \in Plays(\mathfrak{D})\}.$$

The proposition above implies

$$Views(Plays(\mathfrak{D})) = \mathfrak{D}.$$

Due to the proposition above, we have thus obtained the following characterization of designs in terms of plays.

Proposition 7.13 (Designs, Prop. 3, [FH])

$$\mathfrak{D} = \{\lceil q \rceil : q \sqsubseteq r^+, r \in Plays(\mathfrak{D})\}.$$

Let us remind the idea presented by Amadio and Curien (p.43, [AC]), that the meaning of a term should be the collection of properties it satisfies in a suitable logic. In fact, the filter models of λ-calculus is based on this idea.

In this note, we have presented two characterizations of designs as collections, the one based on chronicles, and the other based on plays. A design thought as a term in ludics, its characterization as a collection, can be taken as a meaning defined by the properties it satisfies in a suitable logic, in the above sense.

- Which characterization is better, the one based on chronicles, or the one on plays.

Faggian and Hyland prefer the characterization based on plays. The plays are an interactive notion: So are the behaviors.

- What is the relationship between the chronicles in the plays and the behaviors?
- Does the notion of chronicles correspond to the notion of types in the filter models in some way?

These questions remind us the discussions in the previous section.

- What is the relationship between the designs and the filters of behaviors?

According to the theory of HON game semantics, the collection of innocent strategies ordered by inclusion is a dI-domain; a consistently complete, algebraic CPO which is coprime algebraic and satisfies axiom: Every compact element dominates only finitely many elements. (See [AC], [HO]). It will be interesting to investigate the duality for algebraic dcpo's in the collection of designs.

References

[AC] Amadio, R.M. and Curien, P.-L. *Domains and Lambda-Calculi*, Cambridge Tracts in Theoretical Computer Science 46.

[C] Curien, P.-L. *Introduction to ludics*, available from http://www.pps.jussieu.fr/~curien/.

[FH] Faggian, C. and Hyland, M. *Designs, disputes, and strategies*, CSL'02. Lecture Notes in Computer Science, Springer, Berlin, available from http://www.dpmms.cam.ac.uk/~cf245/pub.html.

[FFDQ] Faggian, C., Fleury-Donnadieu, M.-R. and Quatrini, M. *An introduction to uniformity in Ludics*, available from http://www.dpmms.cam.ac.uk/~cf245/pub.html.

[HO] Hyland, J.M.E. and Ong, C.-H.L. *On Full Abstraction for PCF: I, II, and III*, Information and Computation 163 pp.285-408 (2000).

[Gi] Girard, J.-Y. *Locus Solum*, Mathematical Structure in Computer Science 11 pp.301-506 (2001).

[O] 小野 寛晰. **情報科学における論理**, 情報数学セミナー, 日本評論社 (1994).

関係データベースにおける従属性検証システムの実装

本多 和正*

2003 年 2 月 24 日

概要

W.MacCaull [1] らによる関係データベースにおける従属性検証系は健全且つ完全であるが、適用可能な規則が複数存在するなどの理由により計算機上に実装するには不向きである。また、単に検証を目的とするのであれば特に必要のない規則も存在する。そこで、本研究では規則を計算機上での実装に向くように書き換えを行った。本論文では書き換えられた検証系もまた健全且つ完全であり、停止性を満たすことを示す。

1 準備

この節では本論文で用いる用語や数学構造等について述べる。

1.1 関係代数

本論文で用いる関係代数の体系は、pointwise で homogenous なものである。

集合 X 上の二項関係の集合を $(X \rightharpoonup X)$ と書く。二項関係 $\alpha, \beta \in (X \rightharpoonup X)$ に対し、二項演算 $\sqcup, \sqcap, \cdot$、単項演算 $^-$ を次のように定義する。

$$\begin{aligned}
(x,y) \in (\alpha \sqcup \beta) &\overset{def}{\iff} (x,y) \in \alpha \vee (x,y) \in \beta, \\
(x,y) \in (\alpha \sqcap \beta) &\overset{def}{\iff} (x,y) \in \alpha \wedge (x,y) \in \beta, \\
(x,y) \in (\alpha \cdot \beta) &\overset{def}{\iff} \bigvee_{u \in X}((x,u) \in \alpha \wedge (u,y) \in \beta), \\
(x,y) \in \alpha^- &\overset{def}{\iff} (x,y) \notin \alpha.
\end{aligned}$$

なお、演算子 $\cdot$ は通常省略する。全関係を ∇ と書く $(\forall x, y \in X.(x,y) \in \nabla)$。また、これらの定義より $\alpha \sqcup \alpha^- = \nabla$ が成り立つ。

1.2 context と相関関係

Definition 1.1. **集合 X, A と、各 $a \in A$ 及び 各 $x \in X$ に対し集合 $V(a)$, $f(x,a) \subseteq V(a)$ が与えられたとき、その対 $\mathcal{S} = \langle X, A, \{V(a) : a \in A\}, f \rangle$ を情報系 *(information system)* と呼ぶ。このとき $X, A, V(a)$ の要素をそれぞれ、対象 *(object)*、属性 *(attribute)*、属性値 *(attribute value)* と呼ぶ。**

また、任意の $a \in A$ に対し $V(a) = \{0, 1\}$ であるような情報系を *context* と呼ぶ。対象 1_A を $\forall a \in A. f(1_A, a) = 1$ と定義し、1_A を持つような *context* を *context with 1* と呼ぶ。

* Kazumasa HONDA, DOI, Kyushu University 九州大学大学院 システム情報科学府 情報理学専攻

Definition 1.2. 集合 A の部分集合から集合 X 上の二項関係への写像 $R : \wp(A) \to (X \rightharpoonup X)$ を *(A* から X 上への*)* フレーム *(frame)* と呼ぶ。フレーム R が以下の条件 *(S1),(S2)* を満たすとき *standard*、条件 *(S1),(S2),(S3)* を満たすとき *strong*、条件 *(S1),(S2'),(S3)* を満たすとき *semistrong* であるという。

(S1) 任意の $P \subseteq A$ に対し、$R(P)$ は同値関係、
(S2) 任意の $P, Q \subseteq A$ に対し、$R(P) \sqcap R(Q) = R(P \cup Q)$、
(S2') 任意の $P, Q \subseteq A$ に対し、$R(P) \sqcap R(Q) \sqsubseteq R(P \cup Q)$、
(S3) $R(\emptyset) = \nabla$。

Definition 1.3. 情報系において、フレーム $R^i : \wp(A) \to (X \rightharpoonup X)$ を $(x, y) \in R^i(P) \overset{def}{\iff} \forall a \in P. f(x, a) = f(y, a)$ で定義し、$R^i(P)$ を識別不能関係 *(indiscernibility relation)* と呼ぶ。

Lemma 1.4. 情報系において、フレーム R^i は *strong* である。

Definition 1.5. *context with 1* において、フレーム $R^{as} : \wp(A) \to (X \rightharpoonup X)$ を $(x, y) \in R^{as}(P) \overset{def}{\iff} (\forall a \in P. f(x, a) = 1 \iff \forall a \in P. f(y, a) = 1)$ と定義し、$R^{as}(P)$ を相関関係 *(association relation)* と呼ぶ。

Proposition 1.6. $a \in A, P, Q \subseteq A$ に対し次が成り立つ。

- $(x, 1_A) \in R^{as}(\{a\}) \iff f(x, a) = 1.$
- $(x, 1_A) \in R^{as}(P) \iff \forall a \in P.(x, 1_A) \in R^{as}(\{a\}).$
- $(x, 1_A) \in R^{as}(P \cup Q) \iff (x, 1_A) \in R^{as}(P) \wedge (x, 1_A) \in R^{as}(Q).$
- $(x, y) \in R^{as}(\{a\}) \iff (f(x, a) = 1 \wedge f(y, a) = 1) \vee (f(x, a) \neq 1 \wedge f(y, a) \neq 1).$
- $(x, y) \in R^{as}(P) \iff ((x, 1_A) \in R^{as}(P) \wedge (y, 1_A) \in R^{as}(P)) \vee ((x, 1_A) \in R^{as}(P)^- \wedge (y, 1_A) \in R^{as}(P)^-).$

Lemma 1.7. *context with 1* において、フレーム R^{as} は *semistrong* である。

Definition 1.8. *context with 1* において、任意の $P, Q \subseteq A$ に対し $R^{as}(P) = R^{as}(P \cup Q)$ が成り立つとき、P から Q への相関規則 $P \Rightarrow Q$ があるという。

1.3 情報構造

Definition 1.9. 集合 X、*(*有限*)* 集合 Par、Par から X 上へのフレーム $R : \wp(Par) \to (X \rightharpoonup X)$ の対 $\mathcal{K} = \langle X, Par, R \rangle$ を情報構造 *(information frame)* と呼ぶ。

フレーム R が *standard*、*strong*、*semistrong* のとき、情報構造は *standard*、*strong*、*semistrong* であるという。

以上の定義及び補題から、任意の情報系 $\mathcal{S}$ に対し $\langle X, A, R^i \rangle$ は strong な情報構造であり、任意の context with 1 に対して $\langle X, A, R^{as} \rangle$ が semistrong な情報構造であることがいえる。では、逆に semistrong な情報構造から context with 1 が作れるだろうか？それに答えるのが次の表現定理である。

Theorem 1.10 (Information representability of contexts with 1). $1_A \in X$ であるような *semistrong* な情報構造 $\mathcal{K} = \langle X, Par, R, 1_A \rangle$ が次の公理 *(As1),(As2)* を満たすなら、$R^{as}(P) = R(P)$ を満たす *context with 1* を構成できる。

(As1) $\forall P \subseteq Par.((x, y) \in R(P) \iff ((x, 1_A) \in R(P) \wedge (y, 1_A) \in R(P)) \vee ((x, 1_A) \in R(P)^- \wedge (y, 1_A) \in R(P)^-))$、

(As2) $\forall a \in Par.\forall P \subseteq Par.((x, 1_A) \in R(\{a\})^- \implies (x, 1_A) \in R(P \cup \{a\})^-)$。

以降、*(As1),(As2)* を満たす *semistrong* な情報構造 $\langle X, Par, R, 1_A\rangle$ のクラスを Ω^{as} と表記する。

この定理の証明の準備として以下の補題を証明する。

Lemma 1.11. $\langle X, Par, R, 1_A\rangle \in \Omega^{as}$ **において、** $\forall P \subseteq Par, \forall x \in X$ **に対し、**

$$(x, 1_A) \in R(P) \iff \forall a \in P.(x, 1_A) \in R(\{a\}).$$

Proof. $(As2)$ を同値変形して、$(As2')$ $\forall a \in Par.\forall P \subseteq Par.((x, 1_A) \in R(P \cup \{a\}) \implies (x, 1_A) \in R(\{a\}))$ とすると、$\forall a \in P$ に対し、$(x, 1_A) \in R(P) = R(P \cup \{a\}) \overset{(As2')}{\implies} (x, 1_A) \in R(\{a\})$ となり、更に semistrong 性 $(S2')$ により

$$\begin{array}{lll} (x, 1_A) \in R(P) & \implies \bigwedge_{a\in P}(x, 1_A) \in R(\{a\}) & (As2') \\ & \implies (x, 1_A) \in R(\bigcup_{a\in P}\{a\}) = R(P), & (S2') \end{array}$$

つまり、$(x, 1_A) \in R(P) \iff \bigwedge_{a\in P}(x, 1_A) \in R(\{a\}) \iff \forall a \in P.(x, 1_A) \in R(\{a\})$。 □

Corollary 1.12. $\langle X, Par, R, 1_A\rangle \in \Omega^{as}$ **において、** $\forall P, Q \subseteq Par, \forall x \in X$ **に対し、**

$$(x, 1_A) \in R(P \cup Q) \iff (x, 1_A) \in R(P) \wedge (x, 1_A) \in R(Q).$$

Proof.

$$\begin{array}{ll} (x, 1_A) \in R(P \cup Q) & \iff \bigwedge_{a\in P\cup Q}(x, 1_A) \in R(\{a\}) \\ & \iff \bigwedge_{a\in P}(x, 1_A) \in R(\{a\}) \wedge \bigwedge_{a\in Q}(x, 1_A) \in R(\{a\}) \\ & \iff (x, 1_A) \in R(P) \wedge (x, 1_A) \in R(Q). \end{array}$$

□

Proof of Thm.1.10. まず $A = Par$ とし、$f : X \times A \to \{0, 1\}$ を次のように定義する。

$$f(x, a) = 1 \overset{def}{\iff} (x, 1_A) \in R(\{a\}).$$

$\mathcal{K}$ の semistrong 性 $(S1)$ より、任意の $a \in A$ に対し $R(\{a\})$ は同値関係。よって $(1_A, 1_A) \in R(\{a\})$、つまり $\forall a \in A.f(1_A, a) = 1$。故に対 $\mathcal{S} = \langle X, A, \{0, 1\}, f, 1_A\rangle$ は context with 1 である。

あとは $R = R^{as}$ を示せばよいわけだが、Lemma 1.11 と f の定義、Prop.1.6 により、

$$\begin{array}{lll} (x, 1_A) \in R(P) & \iff \forall a \in P.(x, 1_A) \in R(\{a\}) & (Lemma1.11) \\ & \iff \forall a \in P.f(x, a) = 1 & (f \text{ の定義}) \\ & \iff \forall a \in P.(x, 1_A) \in R^{as}(\{a\}) & (Prop.1.6) \\ & \iff (x, 1_A) \in R^{as}(P). & (Prop.1.6) \end{array}$$

この結果と Prop.1.6、$(As1)$ により

$$\begin{array}{lll} (x, y) \in R^{as}(P) & \iff ((x, 1_A) \in R^{as}(P) \wedge (y, 1_A) \in R^{as}(P)) & (Prop.1.6) \\ & \quad \vee((x, 1_A) \in R^{as}(P)^- \wedge (y, 1_A) \in R^{as}(P)^-) & \\ & \iff ((x, 1_A) \in R(P) \wedge (y, 1_A) \in R(P)) & \\ & \quad \vee((x, 1_A) \in R(P)^- \wedge (y, 1_A) \in R(P)^-) & \\ & \iff (x, y) \in R(P). & (As1) \end{array}$$

□

1.4 LIS 式と RLIS 式

Definition 1.13. **原始パラメータの有限集合 $\mathcal{A}$ に対し、*LIS (language for information system)* 式の集合 $\mathcal{L}$ を次の *BNF* で定義する。**

$$\begin{aligned} C &::= \mathcal{A} \mid C\ C, \\ \mathcal{L} &::= \ ! \mid C \mid \neg\mathcal{L} \mid \mathcal{L} \cup \mathcal{L} \mid \mathcal{L} \cap \mathcal{L} \mid \mathcal{L} \to \mathcal{L} \mid \mathcal{L}; \mathcal{L}. \end{aligned}$$

また、$\mathcal{L}$ と *2* 個以上の変数記号の *(*有限*)* 集合 $\mathcal{V}$ に対し、$\mathcal{R} = \{xFy \mid F \in \mathcal{L}, x, y \in \mathcal{V}\}$ を *RLIS* 式の集合と定義する。

なお、$\mathcal{A}$ から構成されたことを強調する場合にはそれぞれ $C_{\mathcal{A}}, \mathcal{L}_{\mathcal{A}}, \mathcal{R}_{\mathcal{A}}$ と表記する。

Definition 1.14. **情報構造 $\mathcal{K} = \langle X, Par, R\rangle$ と原始意味 $m : \mathcal{A} \to Par$ が与えられたときに、その対 $\mathcal{M} = \langle \mathcal{K}, \mathcal{A}, m\rangle$ を $\mathcal{K}$ 上のモデルと呼び、意味関数 $\mathcal{M}[] : \mathcal{L}_{\mathcal{A}} \to (X \rightharpoonup X)$ を次のように定義する。**

$$m' : C_{\mathcal{A}} \to \wp(Par) \overset{def}{\iff} \begin{cases} m'(P) = \{m(P)\}, & P \in \mathcal{A}, \\ m'(PQ) = m'(P) \cup m'(Q), & P, Q \in C_{\mathcal{A}}. \end{cases}$$

$$\mathcal{M}[] : \mathcal{L}_{\mathcal{A}} \to (X \rightharpoonup X) \overset{def}{\iff} \begin{cases} \mathcal{M}[F] = R(m'(F)), & F \in C_{\mathcal{A}}, \\ \mathcal{M}[!] = R(\emptyset), & \\ \mathcal{M}[\neg F] = \mathcal{M}[F]^{-}, & F \in \mathcal{L}_{\mathcal{A}}, \\ \mathcal{M}[F \cup G] = \mathcal{M}[F] \sqcup \mathcal{M}[G], & F, G \in \mathcal{L}_{\mathcal{A}}, \\ \mathcal{M}[F \cap G] = \mathcal{M}[F] \sqcap \mathcal{M}[G], & F, G \in \mathcal{L}_{\mathcal{A}}, \\ \mathcal{M}[F \to G] = \mathcal{M}[\neg F] \sqcup \mathcal{M}[G], & F, G \in \mathcal{L}_{\mathcal{A}}, \\ \mathcal{M}[F; G] = \mathcal{M}[F]\mathcal{M}[G], & F, G \in \mathcal{L}_{\mathcal{A}}. \end{cases}$$

モデル $\mathcal{M} = \langle \mathcal{K}, \mathcal{A}, m\rangle$ において *LIS* 式 $F \in \mathcal{L}_{\mathcal{A}}$ が $\mathcal{M}[F] = \nabla$ を満たすとき、F は $\mathcal{M}$ で充足可能といい、任意の原始意味 $m : \mathcal{A} \to Par$ に対しモデル $\langle \mathcal{K}, \mathcal{A}, m\rangle$ で充足可能のときに F は $\mathcal{K}$ において妥当であるという。

また、$\mathcal{M}$ に対し $v : \mathcal{V} \to X$ を $\mathcal{M}$ 上の評価関数と呼ぶ。*RLIS* 式 $xFy \in \mathcal{R}_{\mathcal{A}}$ が $(v(x), v(y)) \in \mathcal{M}[F]$ を満たすとき v で充足可能といい、xFy が任意のモデル $\langle \mathcal{K}, \mathcal{A}, \forall m\rangle$ 上の任意の評価関数で充足可能のときに $\mathcal{K}$ において xFy は妥当であるという。さらに、情報構造のクラス Ω に属する任意の情報構造で妥当であるとき、Ω で妥当であるという。

$\mathcal{K}$ と $\mathcal{M}$、$x \in \mathcal{V}$ に対し、$\{v(x) \mid v : \mathcal{M}$ 上の評価関数 $\} = X$ であるので次が成り立つ。

Theorem 1.15. **情報構造 $\mathcal{K}$ において、*LIS* 式 F が妥当であることと *RLIS* 式 xFy $(x \neq y)$ が妥当であることは同値。**

Definition 1.16. ***LIS* 式のマクロとして *D* 式 *(*の集合 $\mathcal{D}$*)* 及びその解釈 $[] : \mathcal{D} \to \mathcal{L}$ を以下のように定義する。**

$$\begin{aligned} \mathcal{D}_0 &::= \ C \Rightarrow C, \\ \mathcal{D}_1 &::= \ \mathcal{D}_0 \mid \mathcal{D}_1 \cap \mathcal{D}_1, \\ \mathcal{D} &::= \ \mathcal{D}_1 \mid \mathcal{D}_1 \supset \mathcal{D}_0. \end{aligned}$$

$$\begin{aligned} [P \Rightarrow Q] &= \ (P \to PQ) \cap (PQ \to P), \\ [F \cap G] &= \ [F] \cap [G], \\ [F \supset G] &= \ (!; \neg[F]; !) \cup [G]. \end{aligned}$$

***D* 式の解釈後の *LIS* 式を *LIS-D* 式と呼び、その集合 $[\mathcal{D}]$ を $\mathcal{L}_D$ と書く。同様に *RLIS-D* 式、$\mathcal{R}_D$ を定義する。**

以降、D 式と LIS-D 式を同一視して議論する。

2 証明系

本研究で提案する証明系は LIS-D 式が情報構造のクラス Ω^{as} で妥当であることの検証を目的としている。
証明系が処理する言語は RLIS 式の列を拡張した次のような言語 $\mathcal{S}$ である。

$$\mathcal{S} = \{\top\} \cup (\mathcal{R} \cup \{\langle x(F;G)y, U\rangle \mid x(F;G)y \in \mathcal{R},\ U \subseteq \mathcal{V}\})^*.$$

モデル $\mathcal{M}$ と評価関数 v に対し、$\langle \mathcal{M}, v\rangle[\![\,]\!] : \mathcal{S} \to \textbf{Boolean}$ を次のように定義する。

$$\begin{aligned}
\langle \mathcal{M}, v\rangle[\![\top]\!] &= \textbf{true},\\
\langle \mathcal{M}, v\rangle[\![\varepsilon]\!] &= \textbf{false},\\
\langle \mathcal{M}, v\rangle[\![xFy]\!] &= (v(x), v(y)) \in \mathcal{M}[F],\\
\langle \mathcal{M}, v\rangle[\![\langle x(F;G)y, U\rangle]\!] &= \textstyle\bigvee_{u\in\mathcal{V}-U}(\langle \mathcal{M}, v\rangle[\![xFu]\!] \wedge \langle \mathcal{M}, v\rangle[\![uGy]\!])\\
\langle \mathcal{M}, v\rangle[\![\Delta, \Gamma]\!] &= \langle \mathcal{M}, v\rangle[\![\Delta]\!] \vee \langle \mathcal{M}, v\rangle[\![\Gamma]\!].
\end{aligned}$$

このとき $F, G \in \mathcal{L}$ に対し以下が成り立つ。

- $\forall \mathcal{M}, \forall v.\langle \mathcal{M}, v\rangle[\![xFy]\!] = \textbf{true} \iff xFy$ が妥当。
- $\langle \mathcal{M}, v\rangle[\![x!y]\!] = (v(x), v(y)) \in \mathcal{M}[!] = (v(x), v(y)) \in \nabla = \textbf{true} = \langle \mathcal{M}, v\rangle[\![\top]\!]$.
- $\langle \mathcal{M}, v\rangle[\![xFy, x\neg Fy]\!] = ((v(x), v(y)) \in \mathcal{M}[F]) \vee ((v(x), v(y)) \in \mathcal{M}[F]^-) = \textbf{true} = \langle \mathcal{M}, v\rangle[\![\top]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x(F \cup G)y]\!] = \langle \mathcal{M}, v\rangle[\![xFy, xGy]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x\neg(F \cup G)y]\!] = \langle \mathcal{M}, v\rangle[\![x\neg Fy]\!] \wedge \langle \mathcal{M}, v\rangle[\![x\neg Gy]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x(F \cap G)y]\!] = \langle \mathcal{M}, v\rangle[\![xFy]\!] \wedge \langle \mathcal{M}, v\rangle[\![xGy]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x\neg(F \cap G)y]\!] = \langle \mathcal{M}, v\rangle[\![x\neg Fy, x\neg Gy]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x(F \to G)y]\!] = \langle \mathcal{M}, v\rangle[\![x\neg Fy, xGy]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x\neg(F \to G)y]\!] = \langle \mathcal{M}, v\rangle[\![xFy]\!] \wedge \langle \mathcal{M}, v\rangle[\![x\neg Gy]\!]$.
- $\langle \mathcal{M}, v\rangle[\![\langle x(F;G)y, \mathcal{V}\rangle]\!] = \textbf{false} = \langle \mathcal{M}, v\rangle[\![\varepsilon]\!]$.
- $$\begin{aligned}
  \langle \mathcal{M}, v\rangle[\![x(F;G)y]\!] &= (v(x), v(y)) \in \mathcal{M}[F;G]\\
  &= (v(x), v(y)) \in \mathcal{M}[F]\mathcal{M}[G]\\
  &= \textstyle\bigvee_{z\in X}((v(x), z) \in \mathcal{M}[F] \wedge (z, v(y)) \in \mathcal{M}[G])\\
  &\geq \textstyle\bigvee_{u\in\mathcal{V}}((v(x), v(u)) \in \mathcal{M}[F] \wedge (v(u), v(y)) \in \mathcal{M}[G])\\
  &= \textstyle\bigvee_{u\in\mathcal{V}}(\langle \mathcal{M}, v\rangle[\![xFu]\!] \wedge \langle \mathcal{M}, v\rangle[\![uGy]\!])\\
  &= \langle \mathcal{M}, v\rangle[\![\langle x(F;G)y, \emptyset\rangle]\!].
  \end{aligned}$$
  なお、v が全射であれば等号が成り立つ。

また、Ω^{as} においてはさらに以下が成り立つ。ただし、$P, Q \in C$、$1 \in \mathcal{V}$ とし、任意の評価関数 v に対し $v(1) = 1_A$ とする。つまり、$\mathcal{V}$ は 3 点以上の有限集合である。

- $\langle \mathcal{M}, v\rangle[\![xPx]\!] = (v(x), v(x)) \in \mathcal{M}[P] = (v(x), v(x)) \in R(P) = \textbf{true} = \langle \mathcal{M}, v\rangle[\![\top]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x\neg Px]\!] = (v(x), v(x)) \in \mathcal{M}[P]^- = (v(x), v(x)) \in R(P)^- = \textbf{false} = \langle \mathcal{M}, v\rangle[\![\varepsilon]\!]$.

- $\langle \mathcal{M}, v\rangle[\![xPy]\!] = (v(x), v(y)) \in R(P)$
  $= ((v(x), 1_A) \in R(P) \wedge (v(y), 1_A) \in R(P))$
  $\vee((v(x), 1_A) \in R(P)^- \wedge (v(y), 1_A) \in R(P)^-)$
  $= ((v(x), v(1)) \in R(P) \wedge (v(y), v(1)) \in R(P))$
  $\vee((v(x), v(1)) \in R(P)^- \wedge (v(y), v(1)) \in R(P)^-)$
  $= (\langle \mathcal{M}, v\rangle[\![xP1]\!] \wedge \langle \mathcal{M}, v\rangle[\![yP1]\!]) \vee (\langle \mathcal{M}, v\rangle[\![x\neg P1]\!] \wedge \langle \mathcal{M}, v\rangle[\![y\neg P1]\!])$
  $= (\langle \mathcal{M}, v\rangle[\![xP1]\!] \vee \langle \mathcal{M}, v\rangle[\![y\neg P1]\!]) \wedge (\langle \mathcal{M}, v\rangle[\![x\neg P1]\!] \vee \langle \mathcal{M}, v\rangle[\![yP1]\!])$
  $= \langle \mathcal{M}, v\rangle[\![xP1, y\neg P1]\!] \wedge \langle \mathcal{M}, v\rangle[\![x\neg P1, yP1]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x\neg Py]\!] = \langle \mathcal{M}, v\rangle[\![x\neg P1, y\neg P1]\!] \wedge \langle \mathcal{M}, v\rangle[\![xP1, yP1]\!]$.
- $\langle \mathcal{M}, v\rangle[\![xPQ1]\!] = \langle \mathcal{M}, v\rangle[\![xP1]\!] \wedge \langle \mathcal{M}, v\rangle[\![xQ1]\!]$.
- $\langle \mathcal{M}, v\rangle[\![x\neg PQ1]\!] = \langle \mathcal{M}, v\rangle[\![x\neg P1, x\neg Q1]\!]$.

これらの式は $\mathcal{M}, v$ の取り方に依らないことは明らかである。そこで $\langle \mathcal{M}, v\rangle$ を省略して表記することで次の定理を得る。

Theorem 2.1. *$F, G \in \mathcal{L}$, $P, Q \in C$, $\Delta, \Gamma \in \mathcal{S}$, $x, y \in \mathcal{V}$* **に対して以下の等式たちが成り立ち、それらを用いて** *$[\![xFy]\!] \geq [\![\top]\!]$* **となるならば** *$xFy$* **は妥当である。**

- $[\![\Delta, \Gamma]\!] = [\![\Delta]\!] \vee [\![\Gamma]\!]$.
- $[\![x!y]\!] = [\![\top]\!]$.
- $[\![xFy, x\neg Fy]\!] = [\![\top]\!]$.
- $[\![x(F \cup G)y]\!] = [\![xFy, xGy]\!]$.
- $[\![x\neg(F \cup G)y]\!] = [\![x\neg Fy]\!] \wedge [\![x\neg Gy]\!]$.
- $[\![x(F \cap G)y]\!] = [\![xFy]\!] \wedge [\![xGy]\!]$.
- $[\![x\neg(F \cap G)y]\!] = [\![x\neg Fy, x\neg Gy]\!]$.
- $[\![x(F \to G)y]\!] = [\![x\neg Fy, xGy]\!]$.
- $[\![x\neg(F \to G)y]\!] = [\![xFy]\!] \wedge [\![x\neg Gy]\!]$.
- $[\![x(F; G)y]\!] \geq [\![\langle x(F; G)y, \emptyset\rangle]\!]$.
- $[\![\langle x(F; G)y, \mathcal{V}\rangle]\!] = [\![\varepsilon]\!]$.
- $[\![\langle x(F; G)y, U\rangle]\!] = \bigvee_{u \in \mathcal{V}-U}([\![xFu]\!] \wedge [\![uGy]\!])$
- $[\![xPx]\!] = [\![\top]\!]$.
- $[\![x\neg Px]\!] = [\![\varepsilon]\!]$.
- $[\![xPy]\!] = [\![xP1, y\neg P1]\!] \wedge [\![x\neg P1, yP1]\!]$.
- $[\![x\neg Py]\!] = [\![x\neg P1, y\neg P1]\!] \wedge [\![xP1, yP1]\!]$.
- $[\![xPQ1]\!] = [\![xP1]\!] \wedge [\![xQ1]\!]$.
- $[\![x\neg PQ1]\!] = [\![x\neg P1, x\neg Q1]\!]$.

Proof. (不) 等式たちの証明については既に述べたので省略。

$[\![xFy]\!] \geq [\![\top]\!]$ とは省略なしに書くと、任意の $\mathcal{M}, v$ に対し $\langle \mathcal{M}, v\rangle[\![xFy]\!] \geq \langle \mathcal{M}, v\rangle[\![\top]\!] =$ **true** が成り立つことである。そして、**true** の最大性から実際には $\langle \mathcal{M}, v\rangle[\![xFy]\!] =$ **true** となる。また、既に述べたように、任意の $\mathcal{M}, v$ に対し $\langle \mathcal{M}, v\rangle[\![xFy]\!] =$ **true** が成り立つことは xFy が妥当であることと同値であったので、$[\![xFy]\!] \geq [\![\top]\!]$ ならば xFy は妥当である。 □

定理 2.1 を用いると RLIS-D 式は次のように分解される。

- $F \supset G \in \mathcal{D}$ に対して、
  $[\![x[F \supset G]y]\!] = [\![x((!; \neg[F]; !) \cup [G])y]\!] = [\![x(!; \neg[F]; !)y, x[G]y]\!]$
  $\geq \bigvee_{a \in \mathcal{V}}([\![x!a, x[G]y]\!] \wedge [\![a(\neg[F]; !)y, x[G]y]\!])$
  $= \bigvee_{a \in \mathcal{V}}[\![a(\neg[F]; !)y, x[G]y]\!]$
  $\geq \bigvee_{a,b \in \mathcal{V}}([\![a\neg[F]b, x[G]y]\!] \wedge [\![b!y, x[G]y]\!])$
  $= \bigvee_{a,b \in \mathcal{V}}[\![a\neg[F]b, x[G]y]\!]$.

  ここで、$\mathcal{V}$ は有限集合であるので、$[\![a\neg[F]b, x[G]y]\!]$ たちはは有限個 ($|\mathcal{V}|^2$ 個) である。

- $F \cap G \in \mathcal{D}_1$ に対して、$[\![x[F \cap G]y]\!] = [\![x[F]y]\!] \wedge [\![x[G]y]\!]$.
- $F \cap G \in \mathcal{D}_1$ に対して、$[\![x\neg[F \cap G]y]\!] = [\![x\neg[F]y, x\neg[G]y]\!]$.
- $P \Rightarrow Q \in \mathcal{D}_0$ に対して、

$$\begin{aligned}
[\![x[P \Rightarrow Q]y]\!] &= [\![x((P \to PQ) \cap (PQ \to P))y]\!] \\
&= [\![x(P \to PQ)y]\!] \wedge [\![x(PQ \to P)y]\!] \\
&= [\![x\neg Py, xPQy]\!] \wedge [\![x\neg PQy, xPy]\!] \\
&= [\![x\neg Py, xPQy]\!] \wedge [\![x\neg PQy, xPy]\!] \\
&= \ [\![x\neg P1, y\neg P1, xPQy]\!] \wedge [\![xP1, yP1, xPQy]\!] \\
&\quad \wedge [\![x\neg PQ1, y\neg PQ1, xPy]\!] \wedge [\![xPQ1, yPQ1, xPy]\!] \\
&= \cdots \\
&= \ [\![x\neg P1, y\neg P1, yQ1]\!] \wedge [\![xP1, yP1, y\neg Q1]\!] \\
&\quad \wedge [\![xP1, y\neg P1, yQ1]\!] \wedge [\![x\neg P1, yP1, xQ1]\!] \\
&\quad \wedge [\![xP1, y\neg P1, xQ1, yQ1]\!] \wedge [\![x\neg P1, yP1, xQ1, yQ1]\!].
\end{aligned}$$

- $P \Rightarrow Q \in \mathcal{D}_0$ に対して、

$$\begin{aligned}
[\![x\neg[P \Rightarrow Q]y]\!] &= [\![x\neg((P \to PQ) \cap (PQ \to P))y]\!] \\
&= [\![x\neg(P \to PQ)y, x\neg(PQ \to P)y]\!] \\
&= [\![xPy, x\neg(PQ \to P)y]\!] \wedge [\![x\neg PQy, x\neg(PQ \to P)y]\!] \\
&= \ [\![xPy, xPQy]\!] \wedge [\![xPy, x\neg Py]\!] \\
&\quad \wedge [\![x\neg PQy, xPQy]\!] \wedge [\![x\neg PQy, x\neg Py]\!] \\
&= [\![xPy, xPQy]\!] \wedge [\![x\neg PQy, x\neg Py]\!] \\
&= \cdots \\
&= \ [\![xP1, y\neg P1, y\neg Q1]\!] \wedge [\![x\neg P1, yP1, x\neg Q1]\!] \\
&\quad \wedge [\![x\neg P1, y\neg P1, x\neg Q1, y\neg Q1]\!] \wedge [\![x\neg P1, y\neg P1, xQ1, yQ1]\!] \\
&\quad \wedge [\![xP1, yP1]\!].
\end{aligned}$$

以上から $[\![\top]\!], [\![xP1, y\neg Q1, \cdots]\!] (P, Q \in \mathcal{A})$ たちの論理積が下界となることがわかる。

また、この分解は RLIS-D 式と $\mathcal{V}$ の有限性から有限回で終了する。よって、この分解による LIS-D 式 F の妥当性判定は健全であり決定可能である。

そして、さらに次が成り立つ。

Theorem 2.2 (Completeness). *RLIS-D* **式** xFy **が** $[\![xFy]\!] \geq [\![\top]\!]$ **でない場合、**$\langle \mathcal{M}, v_0 \rangle [\![xFy]\!] =$ **false となるような** $\mathcal{M}_0, v_0$ **が存在する。**

Proof. $X = \mathcal{V}, 1_A = 1, Par = \mathcal{A}$ とし、$v_0 = id_{\mathcal{V}}, m = id_{\mathcal{A}}$ とする。v_0 の全射性から、Ω^{as} の情報構造 $\mathcal{K} = \langle X, Par, R \rangle$ たちに対し、モデル $\mathcal{M} = \langle \mathcal{K}, m \rangle$ は $\langle \mathcal{M}, v_0 \rangle [\![xFy]\!] = \bigwedge_{i \in I} \langle \mathcal{M}, v_0 \rangle [\![\Gamma_i]\!]$ を満たす。よって、ある $j \in I$ に対し、$\langle \mathcal{M}, v_0 \rangle [\![\Gamma_j]\!] =$ **false** となるような $\mathcal{M}$、正確には R_0 が存在すれば $\langle \mathcal{M}, v_0 \rangle [\![xFy]\!] =$ **false** となる。そこで、そのような R_0 が作れることを示す。

既に述べたように、Γ_j は $xP1, y\neg Q1, \cdots (P, Q \in \mathcal{A})$ の形をしているので、Γ_j に $xP1$ が含まれていれば $(x, 1) \notin R_0(\{P\})$ とし、$y\neg Q1$ が含まれていれば $(y, 1) \in R_0(\{Q\})$ としてやれば R_0 は $\langle \mathcal{M}, v_0 \rangle [\![\Gamma_j]\!] =$ **false** を満たす。しかし、このままでは R_0 は Ω_{as} のフレームではないので、さらに (As1),(As2),semistrong 性の閉包をとってやるとよい。なお、Γ_j には $xP1$ や $y\neg Q1$ といった本体が $\mathcal{A}$ もしくはそれに $\neg$ が付いたもので右端が 1 となる *RLIS* 式しか含まれないのでこの閉包は矛盾なく行うことができる。 □

Definition 2.3. **言語 $\mathcal{S}$ の構成要素 $\top \neq \forall\Gamma \in \mathcal{S}, \forall F,G \in \mathcal{L}, \forall P,Q \in C, \forall x,y \in \mathcal{V}$ に対し、 $\dfrac{\text{分解後の列}}{\text{列}}$ (規則名) という形式で以下の分解規則を定め、その集合をもって証明系 $\mathfrak{P}^{as}$ とする。表記の都合上、列の並びは考慮しない** *(i.e. $C,B,A,\cdots = A,B,C,\cdots$)*。

$$\frac{xFy,\Gamma \quad xGy,\Gamma}{x(F\cap G)y,\Gamma}\ (\cap) \qquad \frac{x\neg Fy,x\neg Gy,\Gamma}{x\neg(F\cap G)y,\Gamma}\ (\neg\cap) \qquad \frac{xFy,xGy,\Gamma}{x(F\cup G)y,\Gamma}\ (\cup) \qquad \frac{x\neg Fy,\Gamma \quad x\neg Gy,\Gamma}{x\neg(F\cup G)y,\Gamma}\ (\neg\cup)$$

$$\frac{x\neg Fy,xGy,\Gamma}{x(F\to G)y,\Gamma}\ (\to) \qquad \frac{xFy,\Gamma \quad x\neg Gy,\Gamma}{x\neg(F\to G)y,\Gamma}\ (\neg\to) \qquad \frac{\top}{x!y,\Gamma}\ (!) \qquad \frac{\top}{xFy,x\neg Fy,\Gamma}\ (FND)$$

$$\frac{xFu,\Gamma,\langle x(F;G)y,\{u\}\rangle \quad uGy,\Gamma,\langle x(F;G)y,\{u\}\rangle}{x(F;G)y,\Gamma}\ (;)\ u\in\mathcal{V} \qquad \frac{\Gamma}{\langle x(F;G)y,\mathcal{V}\rangle,\Gamma}\ (;)$$

$$\frac{xFu,\Gamma,\langle x(F;G)y,U\cup\{u\}\rangle \quad uGy,\Gamma,\langle x(F;G)y,U\cup\{u\}\rangle}{\langle x(F;G)y,U\rangle,\Gamma \quad (\text{if } U\neq\mathcal{V})}\ (;)\ u\in\mathcal{V}-U$$

$$\frac{\top}{xPx,\Gamma}\ (ref) \qquad \frac{\Gamma}{x\neg Px,\Gamma}\ (\neg ref)$$

$$\frac{xP1,y\neg P1,\Gamma \quad x\neg P1,yP1,\Gamma}{xPy,\Gamma \quad (\text{if } y\neq 1)}\ (as1) \qquad \frac{x\neg P1,y\neg P1,\Gamma \quad xP1,yP1,\Gamma}{x\neg Py,\Gamma \quad (\text{if } y\neq 1)}\ (\neg as1)$$

$$\frac{xP1,\Gamma \quad xQ1,\Gamma}{xPQ1,\Gamma}\ (as2) \qquad \frac{x\neg P1,x\neg Q1,\Gamma}{x\neg PQ1,\Gamma}\ (\neg as2)$$

$\mathfrak{P}^{as}$ の規則が適用できない列を閉列と呼び、列の長さが 1 以下の場合特に閉式と呼ぶ。逆に規則が適用可能な列・式を開列・開式と呼ぶ。定義から各 $P\in\mathcal{A}$、各 $x\in\mathcal{V}-\{1\}$ に対し $xP1$ 及び $x\neg P1$、$\top$、ε は閉式である。

また、規則の定義から次のことが成り立つ。

Theorem 2.4 (分解の一意性). **$\mathcal{V}$ の要素の並びを固定し、規則 $(;)$ において $\mathcal{V}$ や $\mathcal{V}-U$ の先頭要素を u としてとると仮定すると、任意の開式には適用可能な規則がただ一つのみ存在する。**

また、定理 2.1・定理 2.2 により $\mathfrak{P}^{as}$ は停止性と健全性、及び完全性を満たす。

Theorem 2.5 (停止性と健全性、完全性). **任意の *RLIS-D* 式 xFy は $\mathfrak{P}^{as}$ により、有限ステップで閉式の列たちに分解される また、全ての閉列が $\top$ であるなら xFy は Ω^{as} において妥当であり、そうでない場合は xFy が妥当とならないような Ω^{as} の情報構造が存在する。**

ここまでの議論において変数記号の集合 $\mathcal{V}$ は 3 点以上の有限集合とだけ仮定していたが、実装する際には適当な個数に限定する必要がある。実験結果から言うと少なくとも Armstrong の公式 (D 式で書き表すと $PQ\Rightarrow Q$、$(P\Rightarrow Q)\supset(PS\Rightarrow QS)$、$((P\Rightarrow Q)\cap(Q\Rightarrow S))\supset(P\Rightarrow S)$) の妥当性証明には $\mathcal{V}$ は 3 点集合で十分である。しかし、一般の D 式に対しても $\mathcal{V}$ を 3 点集合としても良いかどうかについては議論の余地がある。

参考文献

[1] W.MacCaull, A Proof System for Dependencies for Information Relations, Fundamenta Informaticae, 42, pp1-27, 2000

Demonic orders and quasi-totality in Dedekind categories

Yasuo Kawahara (河原 康雄)
Department of Informatics, Kyushu University
(九州大学システム情報科学研究院)

Hitomi Okuma (大隈 ひとみ)
Department of Informatics, Kyushu University
(九州大学システム情報科学府)

Abstract

This paper presents a proof of the associativity of demonic composition of relations in Dedekind categories and shows that the demonic composition is monotonic with respect to two demonic orderings on relations, which are defined by quasi-total relations, respectively.

1 Introduction

Relation algebras [8] are suitable for describing semantics of relational programming [4]. In particular demonic composition [2, 9, 1, 5, 10] and demonic orderings will be useful for designing nondeterministic programs [3, 10, 11]. For concrete relations R and S, the demonic composition $R \odot S$ relates elements x with elements y exactly if x is related with y by the usual relational composition RS and the image of x under R may not lie outside the domain of S (which should never be confused with the categorical concept of source of morphism):

$$(x, y) \in R \odot S \Leftrightarrow [\forall z : (x, z) \in R \Rightarrow z \in \mathrm{dom}(S)] \wedge (x, y) \in RS.$$

In this paper the demonic composition in Dedekind categories [6, 7] will be defined (without using complement operator). The proofs of associative law of demonic compositions are given earlier in [2, 9, 1, 5], here we give a proof using properties of Dedekind compositions. Moreover we study two demonic orderings of relations originally introduced by Desharnais et al. [5] and Xu et al. [10] and show several fundamental properties of them in Dedekind categories. In Section 2, we first review the definition of Dedekind categories. Then we introduce the demonic composition in a Dedekind category, and show some of its properties. In Section 3, we define quasi-totality of relations and give the definition of two refinement orderings, and provide existence conditions of the supremum and values of supremum and infimum of a set of relations with respect to both refinement orderings, respectively. Finally we prove the monotonicity of the demonic composition on these orderings.

2 Demonic Compositions

We will generalize demonic compositions into Dedekind categories and give a proof of associativity of the demonic compositions using properties of Dedekind compositions.

We first review the definition of a Dedekind category, a kind of relation category (following Olivier and Serrato, 1980) which is our general framework.

Throughout this paper, a morphism α from an object A into an object B in a Dedekind category (which will be defined below) will be called a relation, and denoted by a half arrow $\alpha : A \rightharpoonup B$. The

composite of a relation $\alpha : A \rightharpoonup B$ followed by a relation $\beta : B \rightharpoonup C$ will be written as $\alpha\beta : A \rightharpoonup C$. We denote the identity relation on an object A by id_A. The composition operator will bind stronger than all other binary operators.

Definition 2.1 A *Dedekind category* $\mathcal{D}$ is a category satisfying the following:
D1. [Complete Heyting Algebra] For all pairs of objects X and Y the hom-set $\mathcal{D}(X, Y)$ consisting of all relations of X into Y is a complete Heyting algebra with the least relation 0_{XY} and the greatest relation ∇_{XY}. Its algebraic structure will be denoted by

$$\mathcal{D}(X, Y) = (\mathcal{D}(X, Y), \sqsubseteq, \sqcup, \sqcap, 0_{XY}, \nabla_{XY}).$$

That is, (a) $\sqsubseteq$ is a partial order on $\mathcal{D}(X, Y)$, (b) $\forall \alpha \in \mathcal{D}(X, Y) :: 0_{XY} \sqsubseteq \alpha \sqsubseteq \nabla_{XY}$, (c) $\sqcup_{\lambda\in\Lambda}\alpha_\lambda \sqsubseteq \alpha$ iff $\alpha_\lambda \sqsubseteq \alpha$ for all $\lambda \in \Lambda$, (d) $\alpha \sqsubseteq \sqcap_{\lambda\in\Lambda}\alpha_\lambda$ iff $\alpha \sqsubseteq \alpha_\lambda$ for all $\lambda \in \Lambda$, and (e) $\alpha\sqcap(\sqcup_{\lambda\in\Lambda}\alpha_\lambda) = \sqcup_{\lambda\in\Lambda}(\alpha\sqcap\alpha_\lambda)$.
D2. [Converse] There is given a converse operation $^\sharp : \mathcal{D}(X, Y) \to \mathcal{D}(Y, X)$. That is, for all relations $\alpha, \alpha' : X \rightharpoonup Y$, $\beta : Y \rightharpoonup Z$, the following laws hold:
(a) $(\alpha\beta)^\sharp = \beta^\sharp\alpha^\sharp$, (b) $(\alpha^\sharp)^\sharp = \alpha$, (c) If $\alpha \sqsubseteq \alpha'$, then $\alpha^\sharp \sqsubseteq \alpha'^\sharp$.
D3. [Dedekind Formula] For all relations $\alpha : X \rightharpoonup Y$, $\beta : Y \rightharpoonup Z$ and $\gamma : X \rightharpoonup Z$ the Dedekind formula $\alpha\beta \sqcap \gamma \sqsubseteq \alpha(\beta \sqcap \alpha^\sharp\gamma)$ holds.
D4. [Residue] For all relations $\beta : Y \rightharpoonup Z$ and $\gamma : X \rightharpoonup Z$ the residue (or division, weakest precondition) $\gamma \div \beta : X \rightharpoonup Y$ is a relation such that $\alpha\beta \sqsubseteq \gamma$ if and only if $\alpha \sqsubseteq \gamma \div \beta$ for all morphisms $\alpha : X \rightharpoonup Y$. □

If all relations in a Dedekind category have complements, then the Dedekind category is called a Schröder category. It is well known that in a Schröder category the Dedekind formula is equivalent to an equivalence

$$\alpha\beta \sqsubseteq \gamma \;\Leftrightarrow\; \alpha^\sharp\gamma^- \sqsubseteq \beta^- \;\Leftrightarrow\; \gamma^-\beta^\sharp \sqsubseteq \alpha^-$$

which is called *Schröder rule*. A relation $f : X \rightharpoonup Y$ such that $f^\sharp f \sqsubseteq \mathrm{id}_Y$ (*univalent*) and $\mathrm{id}_X \sqsubseteq ff^\sharp$ (*total*) is called a *function* and may be introduced as $f : X \to Y$. A Dedekind category $\mathcal{D}$ is called *uniform* if $\nabla_{XY}\nabla_{YZ} = \nabla_{XZ}$ holds for all objects X, Y and Z in $\mathcal{D}$.

Before we define the demonic composition of relations in a Dedekind category, we consider the *Dedekind composition* $\alpha \ominus \beta$ defined by $\alpha^\sharp\gamma \sqsubseteq \beta$ iff $\gamma \sqsubseteq \alpha \ominus \beta$ for relations $\gamma : X \rightharpoonup Z$. It is easy to see that $\alpha \ominus \beta = (\beta^\sharp \div \alpha)^\sharp$.

The *demonic composition* in a Dedekind category $\mathcal{D}$ is defined by

$$\alpha \odot \beta = \alpha\beta \sqcap (\alpha \ominus \beta\nabla_{ZZ})$$

for relations $\alpha : X \rightharpoonup Y$ and $\beta : Y \rightharpoonup Z$. In Schröder categories it is clear that the demonic composition $\alpha \odot \beta$ can be rewritten to

$$\alpha \odot \beta = \alpha\beta \sqcap (\alpha(\beta\nabla_{ZZ})^-)^-.$$

The proofs of associativity of demonic composition using properties relate to complement were given in [2, 5]. Desharnais et al. [5] also give a proof of associativity by embedding a demonic semilattice in a relation algebra.

Proposition 2.2 *Let $\alpha : X \rightharpoonup Y$ and $\beta : Y \rightharpoonup Z$ be relations in a Dedekind category $\mathcal{D}$. If α is univalent or β is total, then $\alpha \odot \beta = \alpha\beta$. In particular,* $\mathrm{id}_X \odot \alpha = \alpha \odot \mathrm{id}_X = \alpha$.

Proof. First note that $\alpha \odot \beta = \alpha\beta$ iff $\alpha\beta \sqsubseteq \alpha \ominus \beta\nabla_{ZZ}$ iff $\alpha^\sharp\alpha\beta \sqsubseteq \beta\nabla_{ZZ}$. When α is univalent, $\alpha^\sharp\alpha\beta \sqsubseteq \beta \sqsubseteq \beta\nabla_{ZZ}$. Next assume β is total. Then $\nabla_{YZ} \sqsubseteq \beta\beta^\sharp\nabla_{YZ} \sqsubseteq \beta\nabla_{ZZ}$, and so $\alpha^\sharp\alpha\beta \sqsubseteq \beta\nabla_{ZZ}$. Consequently the last claim is clear from the fact that id_X is univalent and total. □

The *domain* relation $\mathsf{dom}\ \alpha : X \rightharpoonup X$ and the *range* (codomain) relation $\mathsf{ran}\ \alpha : Y \rightharpoonup Y$ of $\alpha : X \rightharpoonup Y$ are defined by $\mathsf{dom}\ \alpha = \alpha\alpha^\sharp \sqcap \mathrm{id}_X$ and $\mathsf{ran}\ \alpha = \alpha^\sharp\alpha \sqcap \mathrm{id}_Y$, respectively.

We have the following properties relate to the domain and range relations.

Proposition 2.3 *Let $\alpha : X \rightharpoonup Y$, $\beta : Y \rightharpoonup Z$ and $\gamma : X \rightharpoonup Z$ be relations in a Dedekind category $\mathcal{D}$. Then the following hold:*

(a) $\alpha(\mathit{ran}\ \alpha) = \alpha$ *and* $(\mathit{dom}\ \alpha)\alpha = \alpha$.

(b) $\mathit{ran}\ \gamma \sqsubseteq \mathit{ran}\ \beta \Leftrightarrow \gamma \sqsubseteq \nabla_{XY}\beta$ *and* $\mathit{dom}\ \gamma \sqsubseteq \mathit{dom}\ \alpha \Leftrightarrow \gamma \sqsubseteq \alpha\nabla_{YZ}$.

(c) $\alpha \odot \beta = (\mathit{dom}\ \gamma)\alpha\beta$ *where* $\gamma = \alpha \ominus (\beta\nabla_{ZZ})$.

(d) *If* $u \sqsubseteq \mathrm{id}_X$ *and* $\nabla_{XZ}\nabla_{ZX} = \nabla_{XX}$, *then* $u \sqsubseteq \mathit{dom}\ \gamma$ *iff* $\mathit{ran}\ (u\alpha) \sqsubseteq \mathit{dom}\ \beta$ *where* $\gamma = \alpha \ominus (\beta\nabla_{ZZ})$.

Proof. (a) It is clear from

$$\begin{array}{lcll} \alpha(\mathsf{ran}\ \alpha) & \sqsubseteq & \alpha & \{\ \mathsf{ran}\ \alpha \sqsubseteq \mathrm{id}_Y\ \} \\ & = & \alpha \sqcap \alpha\,\mathrm{id}_Y & \\ & \sqsubseteq & \alpha(\alpha^\sharp\alpha \sqcap \mathrm{id}_Y) & \{\ \text{Dedekind formula}\ \} \\ & = & \alpha(\mathsf{ran}\ \alpha). & \end{array}$$

(b) Assume that $\mathsf{ran}\ \gamma \sqsubseteq \mathsf{ran}\ \beta$. Then

$$\begin{array}{lcll} \gamma & = & \gamma(\mathsf{ran}\ \gamma) & \{\ (a)\ \} \\ & \sqsubseteq & \gamma(\mathsf{ran}\ \beta) & \{\ \text{assumption}\ \} \\ & \sqsubseteq & \gamma\beta^\sharp\beta & \{\ \mathsf{ran}\ \beta = \beta^\sharp\beta \sqcap \mathrm{id}_Z \sqsubseteq \beta^\sharp\beta\ \} \\ & \sqsubseteq & \nabla_{XY}\beta. & \{\ \gamma\beta^\sharp \sqsubseteq \nabla_{XY}\ \} \end{array}$$

Conversely assume that $\gamma \sqsubseteq \nabla_{XY}\beta$. Then

$$\begin{array}{lcll} \mathsf{ran}\ \gamma & = & \gamma^\sharp\gamma \sqcap \mathrm{id}_Z & \{\ \text{definition of range}\ \} \\ & \sqsubseteq & \gamma^\sharp\nabla_{XY}\beta \sqcap \mathrm{id}_Z & \{\ \text{assumption}\ \} \\ & = & (\gamma^\sharp\nabla_{XY} \sqcap \mathrm{id}_Z\beta^\sharp)\beta \sqcap \mathrm{id}_Z & \{\ \text{Dedekind Formula}\ \} \\ & \sqsubseteq & \beta^\sharp\beta \sqcap \mathrm{id}_Z & \\ & = & \mathsf{ran}\ \beta. & \{\ \text{definition of range}\ \} \end{array}$$

(c) Set $\gamma = \alpha \ominus \beta\nabla_{ZZ}$. First we show that $\gamma = (\mathsf{dom}\ \gamma)\nabla_{XZ}$. We have

$$\begin{array}{lcll} \gamma & = & (\mathsf{dom}\ \gamma)\gamma & \{\ (a)\ \} \\ & \sqsubseteq & (\mathsf{dom}\ \gamma)\nabla_{XZ} & \{\ \gamma \sqsubseteq \nabla_{XZ}\ \} \\ & = & (\gamma\gamma^\sharp \sqcap \mathrm{id}_X)\nabla_{XZ} & \{\ \text{definition of domain}\ \} \\ & \sqsubseteq & \gamma\nabla_{ZZ} & \{\ \gamma^\sharp\nabla_{XZ} \sqsubseteq \nabla_{ZZ}\ \} \\ & = & \gamma. & \{\ \text{Proposition A.3(e)}\ \} \end{array}$$

Hence $\gamma = (\mathsf{dom}\ \gamma)\nabla_{XZ}$. Thus $\alpha \odot \beta = \alpha\beta \sqcap \gamma = \alpha\beta \sqcap (\mathsf{dom}\ \gamma)\nabla_{XZ} = (\mathsf{dom}\ \gamma)\alpha\beta$ by Proposition A.1(a).
(d) Assume $u \sqsubseteq \mathrm{id}_X$ and $\nabla_{XZ}\nabla_{ZX} = \nabla_{XX}$, and set $\gamma = \alpha \ominus \beta\nabla_{ZZ}$. Then

$$\begin{array}{lcll} u \sqsubseteq \mathsf{dom}\ \gamma & \Leftrightarrow & \mathsf{dom}\ u \sqsubseteq \mathsf{dom}\ \gamma & \{\ u = \mathsf{dom}\ u\ \} \\ & \Leftrightarrow & u \sqsubseteq \gamma\nabla_{ZX} = \alpha \ominus \beta\nabla_{ZX} & \{\ \text{(b) and Proposition A.3(e):}\nabla_{XZ}\nabla_{ZX} = \nabla_{XX}\ \} \\ & \Leftrightarrow & \alpha^\sharp u \sqsubseteq \beta\nabla_{ZX} & \\ & \Leftrightarrow & u\alpha \sqsubseteq \nabla_{XZ}\beta^\sharp & \{\ \text{conversion}\ \} \\ & \Leftrightarrow & \mathsf{ran}\ (u\alpha) \sqsubseteq \mathsf{ran}\ \beta^\sharp = \mathsf{dom}\ \beta & \{\ \text{(b)}\ \} \end{array}$$

□

Backhouse and van der Woude [1] and Xu et al. [10] also gave the definition of demonic composition. The device used by them to restrict the domain of a relational composition is not intersection, but, instead, composition with a so-called 'monotype', that is, a relation below identity relation. The equivalence of their definition to our definition of demonic composition is clear from (c) and (d) of the last proposition. In [1] there is a proof of associative law for demonic composition using properties of monotype.

Before we see associativity of the demonic compositions we have to show the following lemma.

Lemma 2.4 *Let $\alpha : X \rightharpoonup Y$, $\beta : Y \rightharpoonup Z$ and $\gamma : Z \rightharpoonup W$ be relations in a uniform Dedekind category $\mathcal{D}$. Then the following hold:*

(a) $\alpha \ominus (\beta \odot \gamma)\nabla_{WW} = (\alpha \ominus \beta\gamma\nabla_{WW}) \sqcap (\alpha\beta \ominus \gamma\nabla_{WW})$.

(b) $\alpha(\beta \odot \gamma) \sqcap (\alpha\beta \ominus \gamma\nabla_{WW}) = \alpha\beta\gamma \sqcap (\alpha\beta \ominus \gamma\nabla_{WW})$.

(c) $\alpha \odot (\beta \odot \gamma) = \alpha\beta\gamma \sqcap (\alpha \ominus (\beta\gamma\nabla_{WW} \sqcap (\beta \ominus \gamma\nabla_{WW})))$.

(d) $(\alpha \odot \beta)\gamma = \alpha\beta\gamma \sqcap (\alpha \ominus \beta\nabla_{ZW})$.

(e) $(\alpha \odot \beta) \ominus \gamma\nabla_{WW} = (\alpha \ominus \beta\nabla_{ZW}) \Rightarrow (\alpha\beta \ominus \gamma\nabla_{WW})$.

(f) $(\alpha \odot \beta) \odot \gamma = \alpha\beta\gamma \sqcap (\alpha \ominus (\beta\nabla_{ZW} \sqcap (\beta \ominus \gamma\nabla_{WW})))$.

Proof. (a) It follows from

$$\begin{array}{lll} & \alpha \ominus (\beta \odot \gamma)\nabla_{WW} & \\ = & \alpha \ominus (\beta\gamma \sqcap (\beta \ominus \gamma\nabla_{WW}))\nabla_{WW} & \\ = & \alpha \ominus (\beta\gamma\nabla_{WW} \sqcap (\beta \ominus \gamma\nabla_{WW})) & \{ \text{Propositions A.3(e) and A.1(b)} \} \\ = & (\alpha \ominus \beta\gamma\nabla_{WW}) \sqcap (\alpha \ominus (\beta \ominus \gamma\nabla_{WW})) & \{ \text{Proposition A.3(c)} \} \\ = & (\alpha \ominus \beta\gamma\nabla_{WW}) \sqcap (\alpha\beta \ominus \gamma\nabla_{WW}). & \{ \text{Proposition A.3(d)} \} \end{array}$$

(b) It follows from

$$\begin{array}{lll} & \alpha(\beta \odot \gamma) \sqcap (\alpha\beta \ominus \gamma\nabla_{WW}) & \\ = & \alpha(\beta\gamma \sqcap (\beta \ominus \gamma\nabla_{WW})) \sqcap (\alpha \ominus (\beta \ominus \gamma\nabla_{WW})) & \{ \text{Proposition A.3(d)} \} \\ = & \alpha\beta\gamma \sqcap (\alpha \ominus (\beta \ominus \gamma\nabla_{WW})) & \{ \text{Proposition A.3(f)} \} \\ = & \alpha\beta\gamma \sqcap (\alpha\beta \ominus \gamma\nabla_{WW}). & \{ \text{Proposition A.3(d)} \} \end{array}$$

(c) It is a direct corollary of (a) and (b):

$$\begin{array}{lll} \alpha \odot (\beta \odot \gamma) & = & \alpha(\beta \odot \gamma) \sqcap (\alpha \ominus (\beta \odot \gamma)\nabla_{WW}) \\ & = & \alpha(\beta \odot \gamma) \sqcap (\alpha \ominus \beta\gamma\nabla_{WW}) \sqcap (\alpha\beta \ominus \gamma\nabla_{WW}) \quad \{ \text{(a)} \} \\ & = & \alpha\beta\gamma \sqcap (\alpha \ominus \beta\gamma\nabla_{WW}) \sqcap (\alpha\beta \ominus \gamma\nabla_{WW}) \quad \{ \text{(b)} \} \\ & = & \alpha\beta\gamma \sqcap (\alpha \ominus \beta\gamma\nabla_{WW}) \sqcap (\alpha \ominus (\beta \ominus \gamma\nabla_{WW})) \quad \{ \text{Proposition A.3(d)} \} \\ & = & \alpha\beta\gamma \sqcap (\alpha \ominus (\beta\gamma\nabla_{WW} \sqcap (\beta \ominus \gamma\nabla_{WW}))). \quad \{ \text{Proposition A.3(c)} \} \end{array}$$

(d) It follows from

$$\begin{array}{lll} (\alpha \odot \beta)\gamma & = & (\alpha\beta \sqcap (\alpha \ominus \beta\nabla_{ZZ}))\gamma \\ & = & (\alpha\beta \sqcap (\alpha \ominus \beta\nabla_{ZZ})\nabla_{ZZ})\gamma \quad \{ \text{Proposition A.3(e)} \} \\ & = & \alpha\beta\gamma \sqcap (\alpha \ominus \beta\nabla_{ZZ})\nabla_{ZW} \quad \{ \text{Proposition A.1(b)} \} \\ & = & \alpha\beta\gamma \sqcap (\alpha \ominus \beta\nabla_{ZW}). \quad \{ \text{Proposition A.3(e)} \} \end{array}$$

(e) It is immediate from

$$\begin{array}{lll} & (\alpha \odot \beta) \ominus \gamma\nabla_{WW} & \\ = & (\alpha\beta \sqcap (\alpha \ominus \beta\nabla_{ZZ})) \ominus \gamma\nabla_{WW} & \\ = & (\alpha\beta \sqcap (\alpha \ominus \beta\nabla_{ZZ})\nabla_{ZZ}) \ominus \gamma\nabla_{WW} & \{ \text{Proposition A.3(e)} \} \\ = & (\alpha \ominus \beta\nabla_{ZZ})\nabla_{ZW} \Rightarrow (\alpha\beta \ominus \gamma\nabla_{WW}) & \{ \text{Proposition A.3(h)} \} \\ = & (\alpha \ominus \beta\nabla_{ZW}) \Rightarrow (\alpha\beta \ominus \gamma\nabla_{WW}). & \{ \text{Proposition A.3(e)} \} \end{array}$$

(f) It is a corollary of (d) and (e):

$$\begin{array}{lll} & (\alpha \odot \beta) \odot \gamma & \\ = & (\alpha \odot \beta)\gamma \sqcap ((\alpha \odot \beta) \ominus \gamma\nabla_{WW}) & \\ = & \alpha\beta\gamma \sqcap (\alpha \ominus \beta\nabla_{ZW}) \sqcap ((\alpha \ominus \beta\nabla_{ZW}) \Rightarrow (\alpha\beta \ominus \gamma\nabla_{WW})) & \{ \text{(d), (e)} \} \\ = & \alpha\beta\gamma \sqcap (\alpha \ominus \beta\nabla_{ZW}) \sqcap (\alpha\beta \ominus \gamma\nabla_{WW}) & \{ \text{Proposition A.2(c)} \} \\ = & \alpha\beta\gamma \sqcap (\alpha \ominus (\beta\nabla_{ZW} \sqcap (\beta \ominus \gamma\nabla_{WW}))). & \{ \text{Propositions A.3(d) and A.3(c)} \} \end{array}$$

□

Now we show the associative law of the demonic compositions.

Theorem 2.5 *Let $\alpha : X \rightharpoonup Y$, $\beta : Y \rightharpoonup Z$ and $\gamma : Z \rightharpoonup W$ be relations in a uniform Dedekind category $\mathcal{D}$. Then the associative law $\alpha \odot (\beta \odot \gamma) = (\alpha \odot \beta) \odot \gamma$ of the demonic compositions holds.*

Proof. By Lemmas 2.4(c) and (f) it suffices to see an equality $\beta\nabla_{ZW} \sqcap (\beta \ominus \gamma\nabla_{WW}) = \beta\gamma\nabla_{WW} \sqcap (\beta \ominus \gamma\nabla_{WW})$. Applying Proposition A.3(f) one can see that

$$\begin{array}{ll} & \beta\nabla_{ZW} \sqcap (\beta \ominus \gamma\nabla_{WW}) \\ = & \beta(\nabla_{ZW} \sqcap \gamma\nabla_{WW}) \sqcap (\beta \ominus \gamma\nabla_{WW}) \quad \{ \text{ Proposition A.3(f) } \} \\ = & \beta\gamma\nabla_{WW} \sqcap (\beta \ominus \gamma\nabla_{WW}). \end{array}$$

□

Example 2.6 Take the following homogeneous relations α, α' and β on a set $X = \{1, 2\}$ represented by Boolean matrices:

$$\alpha = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix},\ \alpha' = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \text{ and } \beta = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}.$$

Then $\alpha \sqsubseteq \alpha'$, but $\alpha \odot \beta \not\sqsubseteq \alpha' \odot \beta$ since

$$\alpha \odot \beta = \alpha\beta = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \text{ and}$$

$$\alpha' \odot \beta = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \odot \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

3 Demonic Orderings

As we see in Example 2.6 the demonic composition is not monotonic with respect to the ordering $\sqsubseteq$ on relations. For ensuring the existence of the fixed points of a recursively defined program, we need other orderings among relations on which the demonic composition is monotonic. There are two refinement orderings which are introduced by Xu et al. [10] and Desharnais et al. [5], respectively. In this section we define these refinement orderings in Dedekind categories, and show some of their properties, and finally prove the monotonicity of the demonic composition on these two refinement orderings.

We first recall that each hom-set $\mathcal{D}(X, Y)$ has relative pseudo-complement, that is, for any two relations α and β in $\mathcal{D}$ there is a relation $\alpha \Rightarrow \beta$ such that $\alpha \sqcap \gamma \sqsubseteq \beta$ iff $\gamma \sqsubseteq \alpha \Rightarrow \beta$ for all relations γ.

Define $\alpha^+ = \alpha\nabla_{YY} \Rightarrow \alpha$ for every relation $\alpha : X \rightharpoonup Y$ in a Dedekind category $\mathcal{D}$. A relation α is called *quasi-total* if $\alpha^+ = \alpha$. We can easily see that all total relations are quasi-total as follows: If α is total, then $\nabla_{XY} = \mathrm{id}_X\nabla_{XY} \sqsubseteq \alpha\alpha^\sharp\nabla_{XY} \sqsubseteq \alpha\nabla_{YY}$. Hence $\alpha^+ = \alpha\nabla_{YY} \Rightarrow \alpha = \nabla_{XY} \Rightarrow \alpha = \alpha$. All quasi-total relations are total in uniform Schröder categories. To prove this claim it is enough to show that $\alpha\nabla_{YX} = \nabla_{XX}$ for each quasi-total relation α, because of the fact that $\mathrm{id}_X \sqsubseteq \alpha\alpha^\sharp$ iff $\alpha\nabla_{YX} = \nabla_{XX}$. If α is quasi-total then $\alpha\nabla_{YY} \Rightarrow 0_{XY} \sqsubseteq \alpha\nabla_{YY} \Rightarrow \alpha = \alpha \sqsubseteq \alpha\nabla_{YY}$ and so $\alpha\nabla_{YY} \Rightarrow 0_{XY} = (\alpha\nabla_{YY} \Rightarrow 0_{XY}) \sqcap \alpha\nabla_{YY} = 0_{XY}$. In boolean lattices (or equivalently, in Schröder categories) $\delta \Rightarrow 0_{XY} = \delta^-$ for each relation $\delta : X \rightharpoonup Y$, and so $\alpha\nabla_{YY} = (\alpha\nabla_{YY})^{--} = (\alpha\nabla_{YY} \Rightarrow 0_{XY}) \Rightarrow 0_{XY} = 0_{XY} \Rightarrow 0_{XY} = \nabla_{XY}$. Therefore $\alpha\nabla_{YX} = \alpha\nabla_{YY}\nabla_{YX} = \nabla_{XY}\nabla_{YX} = \nabla_{XX}$ by the uniformity.

Proposition 3.1 *Let $\alpha : X \rightharpoonup Y$ be a relation in a Dedekind category $\mathcal{D}$.*

(a) *$\alpha \sqsubseteq \alpha^+$ and $\alpha^{++} = \alpha^+$. (Every α^+ is quasi-total.)*

(b) *$\alpha\nabla_{YY} = \alpha$ iff $\alpha^+ = \nabla_{XY}$. In particular $0^+_{XY} = \nabla_{XY}$ and $(\alpha\nabla_{YY})^+ = \nabla_{XY}$.*

Proof. (a) It is trivial that $\alpha \sqsubseteq \alpha^+$. Also $\alpha^{++} = \alpha^+\nabla_{YY} \Rightarrow (\alpha\nabla_{YY} \Rightarrow \alpha) = (\alpha^+\nabla_{YY} \sqcap \alpha\nabla_{YY}) \Rightarrow \alpha = \alpha\nabla_{YY} \Rightarrow \alpha = \alpha^+$ by Proposition A.2(d).
(b) Assume $\alpha\nabla_{YY} = \alpha$. Then $\alpha^+ = \alpha \Rightarrow \alpha = \nabla_{XY}$. Conversely assume $\alpha\nabla_{YY} \Rightarrow \alpha = \nabla_{XY}$. Then $\alpha\nabla_{YY} = \alpha\nabla_{YY} \sqcap \nabla_{XY} = \alpha\nabla_{YY} \sqcap \alpha^+ = \alpha$ by Proposition A.2(c). Hence $\alpha = \alpha\nabla_{YY}$. □

In a Dedekind category $\mathcal{D}$ two demonic refinement orderings $\leq$ and $\preceq$ of relations $\alpha, \alpha' : X \rightharpoonup Y$ are respectively defined in [10] and [5] as follows:

$$\begin{array}{lcll} \alpha \leq \alpha' & \overset{def}{\Leftrightarrow} & \alpha \sqsubseteq \alpha' \sqsubseteq \alpha^+ & \{ \text{ Xu et al. [10] } \} \\ & \Leftrightarrow & \alpha' \sqcap \alpha\nabla_{YY} = \alpha & \\ & \Leftrightarrow & \alpha^+ = \alpha\nabla_{YY} \Rightarrow \alpha' & \end{array}$$

$$\alpha \preceq \alpha' \quad \overset{def}{\Leftrightarrow} \quad \alpha\nabla_{YY} \sqsubseteq \alpha'\nabla_{YY} \ \wedge \ \alpha' \sqsubseteq \alpha^+ \quad \{ \text{ Desharnais et al. [5] } \}$$

We can obtain straightforwardly from the above definitions that $\alpha \leq \alpha'$ implies $\alpha \preceq \alpha'$.

Proposition 3.2 *Let $\alpha : X \rightharpoonup Y$ and $\alpha' : X \rightharpoonup Y$ be relations in a Dedekind category $\mathcal{D}$. Then the following hold:*

(a) *If $\alpha\nabla_{YY} = \alpha$, then $\alpha \leq \alpha'$ iff $\alpha \sqsubseteq \alpha'$. In particular $0_{XY} \leq \alpha$ and $0_{XY} \preceq \alpha$.*

(b) *If $\alpha\nabla_{YY} = \alpha$, then $\alpha \preceq \alpha'$ iff $\alpha \sqsubseteq \alpha'\nabla_{YY}$. In particular $\alpha\nabla_{YY} \preceq \alpha$.*

(c) *$\alpha \leq \alpha^+$ and $\alpha \preceq \alpha^+$.*

(d) *$\alpha\alpha^\sharp\alpha \preceq \alpha$.*

(e) *If $\alpha\nabla_{YY} \sqsupseteq \alpha'\nabla_{YY}$ and $\alpha \preceq \alpha'$, then $\alpha' \sqsubseteq \alpha$.*

Proof. (a) Assume $\alpha\nabla_{YY} = \alpha$. Then the assertion is trivial since $\alpha^+ = \nabla_{XY}$ by Proposition 3.1(b).
(b) It is trivial from the definition.
(c) By Proposition 3.1(a) we have $\alpha \sqsubseteq \alpha^+ \sqsubseteq \alpha^+$ which means $\alpha \leq \alpha^+$, and so $\alpha \preceq \alpha^+$.
(d) It follows from $\alpha\alpha^\sharp\alpha\nabla_{YY} \sqsubseteq \alpha\nabla_{YY}$ and $\alpha \sqsubseteq \alpha\alpha^\sharp\alpha \sqsubseteq (\alpha\alpha^\sharp\alpha)^+$ by Proposition 3.1(a).
(e) Assume that $\alpha\nabla_{YY} \sqsupseteq \alpha'\nabla_{YY}$ and $\alpha \preceq \alpha'$. Then we have $\alpha' = \alpha'\nabla_{YY} \sqcap \alpha' \sqsubseteq \alpha\nabla_{YY} \sqcap \alpha^+ = \alpha$ by Proposition A.2(c). □

Next we see the demonic refinement orderings are orderings on the hom-set $\mathcal{D}(X, Y)$.

Proposition 3.3 *Relations $\leq$ and $\preceq$ on the hom-set $\mathcal{D}(X, Y)$ are orderings.*

Proof. (Reflexive law) $\alpha \leq \alpha$ and $\alpha \preceq \alpha$ follows from a fact $\alpha \sqsubseteq \alpha \sqsubseteq \alpha^+$ by Proposition 3.1(a).
(Transitive law) Assume that $\alpha \leq \alpha'$ and $\alpha' \leq \alpha''$, that is, $\alpha \sqsubseteq \alpha' \sqsubseteq \alpha^+$ and $\alpha' \sqsubseteq \alpha'' \sqsubseteq \alpha'^+$. Hence $\alpha \sqsubseteq \alpha' \sqsubseteq \alpha''$ and

$$\begin{array}{lcll} \alpha'' & \sqsubseteq & \alpha'\nabla_{YY} \Rightarrow \alpha' & \{ \ \alpha'' \sqsubseteq \alpha'^+ \ \} \\ & \sqsubseteq & \alpha'\nabla_{YY} \Rightarrow (\alpha\nabla_{YY} \Rightarrow \alpha) & \{ \ \alpha' \sqsubseteq \alpha^+ \ \} \\ & = & (\alpha'\nabla_{YY} \sqcap \alpha\nabla_{YY}) \Rightarrow \alpha & \{ \text{ Proposition A.2(d) } \} \\ & = & \alpha\nabla_{YY} \Rightarrow \alpha. & \{ \ \alpha \sqsubseteq \alpha' \ \} \\ & = & \alpha^+ & \end{array}$$

Similarly $\alpha \preceq \alpha'$ and $\alpha' \preceq \alpha''$ imply $\alpha \preceq \alpha''$.
(Anti-symmetric law) Assume that $\alpha \preceq \alpha'$ and $\alpha' \preceq \alpha$. First note that $\alpha\nabla_{YY} = \alpha'\nabla_{YY}$. Then using Proposition 3.2(e) we have $\alpha \sqsubseteq \alpha'$ and $\alpha' \sqsubseteq \alpha$. Hence $\alpha = \alpha'$. Anti-symmetry of $\leq$ is trivial. □

Example 3.4 Consider the following relations on a set $X = \{1, 2\}$ represented by matrices:

$$\alpha = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} = \alpha\nabla_{XX} \text{ and } \alpha' = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathrm{id}_X.$$

Then $\alpha \preceq \alpha'$ ($\alpha\nabla_{XX} \sqsubseteq \nabla_{XX} = \alpha'\nabla_{XX}$ and $\alpha' \sqsubseteq \nabla_{XX} = \alpha^+$), but $\alpha \not\leq \alpha'$ because $\alpha \not\sqsubseteq \alpha'$.

Lemma 3.5 *Let $\alpha, \alpha' : X \rightharpoonup Y$ be relations in a Dedekind category $\mathcal{D}$. If α' is univalent and $\alpha \sqsubseteq \alpha'$, then $\alpha' \sqsubseteq \alpha^+$ and consequently $\alpha \leq \alpha'$ and $\alpha \preceq \alpha'$.*

Proof. Assume $\alpha'^\sharp \alpha' \sqsubseteq \mathrm{id}_Y$ and $\alpha \sqsubseteq \alpha'$. Then

$$\begin{array}{lcll} \alpha' \sqcap \alpha \nabla_{YY} & \sqsubseteq & \alpha \alpha^\sharp \alpha' & \{ \text{ Dedekind Formula } \} \\ & \sqsubseteq & \alpha \alpha'^\sharp \alpha' & \{ \ \alpha \sqsubseteq \alpha' \ \} \\ & \sqsubseteq & \alpha. & \{ \ \alpha'^\sharp \alpha' \sqsubseteq \mathrm{id}_Y \ \} \end{array}$$

Hence $\alpha' \sqsubseteq \alpha^+$. □

By the above lemma if α is quasi-total and α' is univalent, then $\alpha \sqsubseteq \alpha'$ implies $\alpha = \alpha'$.

The following proposition characterizes maximal elements in the demonic orderings:

Proposition 3.6 (a) *A relation $\alpha : X \rightharpoonup Y$ is maximal in $(\mathcal{D}(X,Y), \leq)$ iff it is quasi-total ($\alpha = \alpha^+$).*

(b) *Suppose a relational axiom[1] of choice. Then a relation $\alpha : X \rightharpoonup Y$ is maximal in $(\mathcal{D}(X,Y), \preceq)$ iff $\alpha = \alpha^+$ and $\alpha^\sharp \alpha \sqsubseteq \mathrm{id}_Y$.*

Proof. (a) Assume that $\alpha = \alpha^+$ and $\alpha \leq \alpha'$. Then $\alpha \sqsubseteq \alpha'$ and $\alpha' \sqsubseteq \alpha^+ = \alpha$. Hence $\alpha = \alpha'$ and so α is maximal. Conversely assume that α is maximal in $(\mathcal{D}(X,Y), \leq)$. Then $\alpha = \alpha^+$ follows from the maximality of α since $\alpha \leq \alpha^+$ by Proposition 3.2(c).
(b) Let $\alpha = \alpha^+$ and $\alpha^\sharp \alpha \sqsubseteq \mathrm{id}_Y$. Assume $\alpha \preceq \alpha'$. Then $\alpha' \sqsubseteq \alpha^+ = \alpha$ and so $\alpha' \preceq \alpha$ by Lemma 3.5. Hence $\alpha = \alpha'$ by the anti-symmetric law of $\preceq$, which proves the maximality of α. Conversely assume that α is maximal in $(\mathcal{D}(X,Y), \preceq)$. Since $\alpha \preceq \alpha^+$ by Proposition 3.2(c) the maximality of α leads $\alpha = \alpha^+$. Now by the relational axiom* of choice there exists a univalent relation $f : X \to Y$ such that $f \sqsubseteq \alpha$ and $f \nabla_{YY} = \alpha \nabla_{YY}$. Then $\alpha \preceq f$ since $\alpha \nabla_{YY} = f \nabla_{YY}$ and $f \sqsubseteq \alpha = \alpha^+$. Again by the maximality of α we have $\alpha = f$, which proves that α is univalent. □

Theorem 3.7 *Let A be a nonempty subset of $\mathcal{D}(X,Y)$.*

(a) *The supremum of the set A in $(\mathcal{D}(X,Y), \leq)$ exists if and inly if*

$$\sqcup_{\alpha \in A} \alpha \sqsubseteq \sqcap_{\alpha \in A} \alpha^+.$$

When this condition is satisfied, the supremum is

$$\sup_{\leq} A = \sqcup_{\alpha \in A} \alpha.$$

(b) *The infimum of A in $(\mathcal{D}(X,Y), \leq)$ always exists, that is,*

$$\inf_{\leq} A = \sqcup \{ \alpha_0 \mid \alpha_0 \sqsubseteq \sqcap_{\alpha \in A} \alpha \text{ and } \sqcup_{\alpha \in A} \alpha \sqsubseteq \alpha_0^+ \}.$$

In particular, $\inf_{\leq} A = \sqcap_{\alpha \in A} \alpha$ *when* $\sqcup_{\alpha \in A} \alpha \sqsubseteq (\sqcap_{\alpha \in A} \alpha)^+$.

Proof. (a) Set $\alpha_0 = \sqcup_{\alpha \in A} \alpha$. We prove the existence condition and the value of the supremum. Let α' be any relation. Then

$$\begin{array}{ll} & \forall \alpha \in A : \alpha \leq \alpha' \\ \Leftrightarrow & \quad \{ \text{ definition } \} \\ & \forall \alpha \in A : \alpha' \sqcap \alpha \nabla_{YY} = \alpha \wedge \alpha_0 \sqsubseteq \sqcap_{\alpha \in A} \alpha^+ \\ \Leftrightarrow & \quad \{ \Rightarrow : \ \alpha' \sqcap \alpha_0 \nabla_{YY} = \sqcup_{\alpha \in A} (\alpha' \sqcap \alpha \nabla_{YY}) = \alpha_0 \\ & \qquad \Leftarrow : \text{ Because } \alpha \sqsubseteq \alpha_0 \sqsubseteq \alpha' \text{ and } \alpha' \sqcap \alpha \nabla_{YY} = \alpha' \sqcap \alpha_0 \nabla_{YY} \sqcap \\ & \qquad \alpha \nabla_{YY} = \alpha_0 \sqcap \alpha \nabla_{YY} \sqsubseteq \alpha^+ \sqcap \alpha \nabla_{YY} = \alpha. \ \} \\ & \alpha' \sqcap \alpha_0 \nabla_{YY} = \alpha_0 \wedge \alpha_0 \sqsubseteq \sqcap_{\alpha \in A} \alpha^+ \\ \Leftrightarrow & \quad \{ \text{ definition } \} \\ & \alpha_0 \leq \alpha' \wedge \alpha_0 \sqsubseteq \sqcap_{\alpha \in A} \alpha^+. \end{array}$$

[1] A relational axiom of choice: for every relation $\alpha : X \rightharpoonup Y$ there exists a univalent relation $f : X \to Y$ such that $f \sqsubseteq \alpha$ and $f \nabla_{YY} = \alpha \nabla_{YY}$.

(b) Denote by A_0 the set of all lower bounds α_0 of A, that is $A_0 = \{\alpha_0 | \alpha_0 \sqsubseteq \sqcap_{\alpha\in A}\alpha \wedge \sqcup_{\alpha\in A}\alpha \sqsubseteq \alpha_0^+\}$, and set $\alpha_* = \sqcup_{\alpha_0\in A_0}\alpha_0$. Obviously A_0 is a nonempty set, since a zero relation 0_{XY} is a lower bound of A. Let α' be any relation, then we obtain

$$
\begin{array}{ll}
 & \forall \alpha \in A : \alpha' \leq \alpha \\
\Leftrightarrow & \quad \{ \text{ definition } \} \\
 & \forall \alpha \in A : \alpha \sqcap \alpha' \nabla_{YY} = \alpha' \\
\Leftrightarrow & \quad \{ \Rightarrow\text{: } \alpha_* \sqcap \alpha' \nabla_{YY} \sqsubseteq \alpha \sqcap \alpha' \nabla_{YY} = \alpha' \sqcap \alpha' \nabla_{YY} \sqsubseteq \alpha_* \sqcap \alpha' \nabla_{YY} \\
 & \quad \text{since } \alpha' \in A_0. \ \} \\
 & \alpha_* \sqcap \alpha' \nabla_{YY} = \alpha' \\
\Leftrightarrow & \quad \{ \text{ definition } \} \\
 & \alpha' \leq \alpha_*,
\end{array}
$$

where the second $\Leftarrow$ follows from $\alpha' \sqsubseteq \alpha_* \sqsubseteq \alpha$ and the next computation

$$
\begin{array}{lll}
\alpha & \sqsubseteq & \sqcap_{\alpha_0\in A_0}\alpha_0^+ \\
 & = & \sqcap_{\alpha_0\in A_0}(\alpha_0 \nabla_{YY} \Rightarrow \alpha_0) \\
 & \sqsubseteq & \sqcap_{\alpha_0\in A_0}(\alpha_0 \nabla_{YY} \Rightarrow \alpha_*) \quad \{ \text{ Proposition A.2(g): } \alpha_0 \sqsubseteq \alpha_* \ \} \\
 & = & (\sqcup_{\alpha_0\in A_0}\alpha_0 \nabla_{YY}) \Rightarrow \alpha_* \quad \{ \text{ Proposition A.2(e) } \} \\
 & = & \alpha_* \nabla_{YY} \Rightarrow \alpha_* \\
 & \sqsubseteq & \alpha' \nabla_{YY} \Rightarrow \alpha'^+ \quad \{ \text{ Proposition A.2(g): } \alpha' \sqsubseteq \alpha_* \sqsubseteq \alpha'^+ \ \} \\
 & = & \alpha'^+. \quad \{ \text{ Proposition A.2(d) } \}
\end{array}
$$

□

We next see the supremum and the infimum of a chain with respect to $\leq$ found in [10].

Proposition 3.8 *Every chain A in $(\mathcal{D}(X,Y), \leq)$ has the supremum $\sup_{\leq} A = \sqcup_{\alpha\in A}\alpha$ and the infimum $\inf_{\leq} A = \sqcap_{\alpha\in A}\alpha$.*

Proof. (i) By the virtue of the last theorem it suffices to see that every chain A in $(\mathcal{D}(X,Y), \leq)$ satisfies $\sqcup_{\alpha\in A}\alpha \sqsubseteq \sqcap_{\alpha\in A}\alpha^+$. The inequality is equivalent to a fact that $\alpha' \sqsubseteq \alpha^+$ for all $\alpha', \alpha \in A$. But A is a chain, so $\alpha \leq \alpha'$ or $\alpha' \leq \alpha$. In the case of $\alpha \leq \alpha'$ it is trivial that $\alpha' \sqsubseteq \alpha^+$. Also in the case of $\alpha' \leq \alpha$ we have $\alpha' \sqsubseteq \alpha \sqsubseteq \alpha^+$.
(ii) It suffices to show that $\sqcap_{\alpha\in A}\alpha$ is a lower bound of A, that is, $\sqcup_{\alpha\in A}\alpha \sqsubseteq (\sqcap_{\alpha\in A}\alpha)^+$, which is equivalent to $\alpha' \sqcap (\sqcap_{\alpha\in A}\alpha)\nabla_{YY} \sqsubseteq \alpha$ for all $\alpha', \alpha \in A$. But A is a chain in $(\mathcal{D}(X,Y), \leq)$, so $\alpha \leq \alpha'$ or $\alpha' \leq \alpha$. In the case of $\alpha \leq \alpha'$ we have $\alpha' \sqcap (\sqcap_{\alpha\in A}\alpha)\nabla_{YY} \sqsubseteq \alpha' \sqcap \alpha\nabla_{YY} = \alpha$. Also in the case of $\alpha' \leq \alpha$ it is trivial that $\alpha' \sqcap (\sqcap_{\alpha\in A}\alpha)\nabla_{YY} \sqsubseteq \alpha' \sqsubseteq \alpha$. □

We now see the supremum and the infimum with respect to $\preceq$ found in [5].

Proposition 3.9 *Let A be a nonempty subset of $\mathcal{D}(X,Y)$.*

(a) *The supremum of the set A in $(\mathcal{D}(X,Y), \preceq)$ exists if and only if*

$$\sqcup_{\alpha\in A}\alpha\nabla_{YY} \sqsubseteq (\sqcap_{\alpha\in A}\alpha^+)\nabla_{YY}.$$

When this condition is satisfied, the supremum is

$$\sup_{\preceq} A = (\sqcup_{\alpha\in A}\alpha\nabla_{YY}) \sqcap (\sqcap_{\alpha\in A}\alpha^+).$$

(b) *The infimum of A in $(\mathcal{D}(X,Y), \preceq)$ always exists, that is,*

$$\inf_{\preceq} A = (\sqcup_{\alpha\in A}\alpha) \sqcap (\sqcap_{\alpha\in A}\alpha\nabla_{YY}),$$

Proof. (a) Set $\alpha_0 = (\sqcup_{\alpha\in A}\alpha\nabla_{YY}) \sqcap (\sqcap_{\alpha\in A}\alpha^+)$. Noting that when the condition $\sqcup_{\alpha\in A}\alpha\nabla_{YY} \sqsubseteq (\sqcap_{\alpha\in A}\alpha^+)\nabla_{YY}$ holds

$$\alpha_0\nabla_{YY} = (\sqcup_{\alpha\in A}\alpha\nabla_{YY}) \sqcap (\sqcap_{\alpha\in A}\alpha^+)\nabla_{YY} = \sqcup_{\alpha\in A}\alpha\nabla_{YY}$$

and so α_0 can be rewritten to

$$\alpha_0 = \alpha_0\nabla_{YY} \sqcap (\sqcap_{\alpha\in A}\alpha^+),$$

we prove the existence condition and the value of supremum of A. Let α' be any relation. We have

$$\begin{array}{ll}
 & \forall\alpha \in A : \alpha \preceq \alpha' \\
\Leftrightarrow & \quad \{ \text{ definition } \} \\
 & \sqcup_{\alpha\in A}\alpha\nabla_{YY} \sqsubseteq \alpha'\nabla_{YY} \wedge \alpha' \sqsubseteq \sqcap_{\alpha\in A}\alpha^+ \wedge \sqcup_{\alpha\in A}\alpha\nabla_{YY} \sqsubseteq (\sqcap_{\alpha\in A}\alpha^+)\nabla_{YY} \\
\Leftrightarrow & \quad \{ \Rightarrow\text{: } \alpha' \sqcap \alpha_0\nabla_{YY} \sqsubseteq (\sqcap_{\alpha\in A}\alpha^+) \sqcap \alpha_0\nabla_{YY} = \alpha_0 \ \} \\
 & \alpha_0\nabla_{YY} \sqsubseteq \alpha'\nabla_{YY} \wedge \alpha' \sqsubseteq \alpha_0^+ \wedge \sqcup_{\alpha\in A}\alpha\nabla_{YY} \sqsubseteq (\sqcap_{\alpha\in A}\alpha^+)\nabla_{YY} \\
\Leftrightarrow & \quad \{ \text{ definition } \} \\
 & \alpha_0 \preceq \alpha' \wedge \sqcup_{\alpha\in A}\alpha\nabla_{YY} \sqsubseteq (\sqcap_{\alpha\in A}\alpha^+)\nabla_{YY},
\end{array}$$

where the second $\Leftarrow$ follows from

$$\begin{array}{lll}
 & \alpha' \sqcap \alpha\nabla_{YY} & \\
\sqsubseteq & \alpha_0^+ \sqcap \alpha\nabla_{YY} & \{ \ \alpha' \sqsubseteq \alpha_0^+ \ \} \\
= & (\alpha_0\nabla_{YY} \Rightarrow \alpha_0) \sqcap \alpha\nabla_{YY} & \\
= & \alpha_0 \sqcap \alpha\nabla_{YY} & \{ \text{ Proposition A.2(f): } \alpha\nabla_{YY} \sqsubseteq \alpha_0\nabla_{YY} \ \} \\
\sqsubseteq & \alpha_0 & \\
\sqsubseteq & \alpha^+, &
\end{array}$$

which implies $\alpha' \sqsubseteq \alpha\nabla_{YY} \Rightarrow \alpha^+ = \alpha^+$ for each $\alpha \in A$ by Proposition A.2(d).
(b) Set $\alpha_0 = (\sqcup_{\alpha\in A}\alpha) \sqcap (\sqcap_{\alpha\in A}\alpha\nabla_{YY})$. Then $\alpha_0\nabla_{YY} = \sqcap_{\alpha\in A}\alpha\nabla_{YY}$ and so $\alpha_0 = (\sqcup_{\alpha\in A}\alpha) \sqcap \alpha_0\nabla_{YY}$. So we have the following equivalences for any given relation α'

$$\begin{array}{lll}
 & \forall\alpha \in A : \alpha' \preceq \alpha & \\
\Leftrightarrow & \forall\alpha \in A : \alpha'\nabla_{YY} \sqsubseteq \alpha\nabla_{YY} \wedge \alpha \sqsubseteq \alpha'^+ & \{ \text{ definition } \} \\
\Leftrightarrow & \alpha'\nabla_{YY} \sqsubseteq \alpha_0\nabla_{YY} \wedge \sqcup_{\alpha\in A}\alpha \sqsubseteq \alpha'^+ & \{ \text{ definition } \} \\
\Leftrightarrow & \alpha'\nabla_{YY} \sqsubseteq \alpha_0\nabla_{YY} \wedge \alpha_0 \sqsubseteq \alpha'^+ & \{ \Rightarrow\text{: } \alpha_0 \sqsubseteq \sqcup_{\alpha\in A}\alpha \sqsubseteq \alpha'^+ \ \} \\
\Leftrightarrow & \alpha' \preceq \alpha_0, & \{ \text{ definition } \}
\end{array}$$

where the third $\Leftarrow$ is shown as follows. Consider the following computation

$$(\sqcup_{\alpha\in A}\alpha) \sqcap \alpha'\nabla_{YY} \sqsubseteq (\sqcup_{\alpha\in A}\alpha) \sqcap \alpha_0\nabla_{YY} = \alpha_0 \sqsubseteq \alpha'^+,$$

which implies $\sqcup_{\alpha\in A}\alpha \sqsubseteq \alpha'^+$ by Proposition A.2(d). □

Lemma 3.10 *Let $\alpha : X \rightharpoonup Y$ and $\beta : X \rightharpoonup Z$ be relations. Then $\alpha \odot \beta = \alpha\beta \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ})$.*

Proof.

$$\begin{array}{lll}
\alpha \odot \beta & = & \alpha\beta \sqcap (\alpha \ominus \beta\nabla_{ZZ}) \\
 & = & \alpha\beta \sqcap (\alpha\nabla_{YY} \sqcap \alpha^+) \ominus \beta\nabla_{ZZ} \qquad \{ \ \alpha = \alpha\nabla_{YY} \sqcap \alpha^+ \ \} \\
 & = & \alpha\beta \sqcap (\alpha\nabla_{YZ} \Rightarrow (\alpha^+ \ominus \beta\nabla_{ZZ})) \quad \{ \text{ Proposition A.3(h) } \} \\
 & = & \alpha\beta \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ}) \qquad \{ \text{ Proposition A.2(f) } \}
\end{array}$$

□

In the following discussion, a map which is monotonic with respect to $\leq$ or $\preceq$ is called $\leq$-monotonic or $\preceq$-monotonic, respectively. The next proposition shows that the demonic composition $\odot$ is $\leq$-monotonic and $\preceq$-monotonic.

Proposition 3.11 *Let $\alpha, \xi : X \rightharpoonup Y$ and $\beta : Y \rightharpoonup Z$ be relations. Then the following hold:*

(a) *If* $\alpha \leq \alpha'$ *and* $\beta \leq \beta'$, *then* $\alpha \odot \beta \leq \alpha' \odot \beta'$.

(b) *If* $\alpha \preceq \alpha'$ *and* $\beta \leq \beta'$, *then* $\alpha \odot \beta \preceq \alpha' \odot \beta'$.

Proof. (a) Assume that $\alpha \sqsubseteq \alpha' \sqsubseteq \alpha^+$ and $\beta \sqsubseteq \beta' \sqsubseteq \beta^+$. Then

$$\begin{array}{lcll} \alpha \odot \beta & = & \alpha\beta \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ}) & \{ \text{ Lemma 3.10 } \} \\ & \sqsubseteq & \alpha'\beta' \sqcap (\alpha' \ominus \beta'\nabla_{ZZ}) & \{ \alpha \sqsubseteq \alpha' \sqsubseteq \alpha^+,\ \beta \sqsubseteq \beta' \} \\ & = & \alpha' \odot \beta'. & \end{array}$$

and

$$\begin{array}{lll} & (\alpha' \odot \beta') \sqcap (\alpha \odot \beta)\nabla_{ZZ} & \\ \sqsubseteq & \alpha'\beta' \sqcap \alpha\nabla_{YZ} \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ}) & \\ & \{\ (\alpha \odot \beta)\nabla_{ZZ} = \alpha\beta\nabla_{ZZ} \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ}) \sqsubseteq \alpha\nabla_{YZ} \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ})\ \} & \\ \sqsubseteq & \alpha^+\beta' \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ}) \sqcap \alpha\nabla_{YZ} & \{\ \alpha' \sqsubseteq \alpha^+\ \} \\ \sqsubseteq & \alpha^+(\beta' \sqcap \beta\nabla_{ZZ}) \sqcap \alpha\nabla_{YZ} & \{ \text{ Dedekind formula and Proposition A.3(a) } \} \\ \sqsubseteq & \alpha^+\beta \sqcap \alpha\nabla_{YZ} & \{\ \beta' \sqcap \beta\nabla_{ZZ} \sqsubseteq \beta \text{ by } \beta' \sqsubseteq \beta^+\ \} \\ = & (\alpha^+ \sqcap \alpha\nabla_{YY})\beta & \{ \text{ Proposition A.1(b) } \} \\ = & \alpha\beta. & \end{array}$$

Hence $(\alpha' \odot \beta') \sqcap (\alpha \odot \beta)\nabla_{ZZ} \sqsubseteq \alpha \odot \beta$ and so $\alpha' \odot \beta' \sqsubseteq (\alpha \odot \beta)^+$.
(b) Assume that $\alpha\nabla_{YY} \sqsubseteq \alpha'\nabla_{YY}$, $\beta\nabla_{ZZ} \sqsubseteq \beta'\nabla_{ZZ}$, $\alpha' \sqsubseteq \alpha^+$ and $\beta' \sqsubseteq \beta^+$. First note that $\alpha^+ \ominus \beta\nabla_{ZZ} \sqsubseteq \alpha' \ominus \beta'\nabla_{ZZ}$ by the assumptions $\alpha' \sqsubseteq \alpha^+$, $\beta\nabla_{ZZ} \sqsubseteq \beta'\nabla_{ZZ}$ and Proposition A.3(i). Then

$$\begin{array}{lll} & (\alpha \odot \beta)\nabla_{ZZ} & \\ = & \alpha\beta\nabla_{ZZ} \sqcap (\alpha^+ \ominus \beta\nabla_{ZZ}) & \{ \text{ Lemma 3.10, Proposition A.1(b) and A.3(e) } \} \\ \sqsubseteq & \alpha\beta\nabla_{ZZ} \sqcap (\alpha' \ominus \beta'\nabla_{ZZ}) & \\ \sqsubseteq & \alpha'\nabla_{YZ} \sqcap (\alpha' \ominus \beta'\nabla_{ZZ}) & \{ \text{ assumption } \} \\ \sqsubseteq & \alpha'\beta'\nabla_{ZZ} \sqcap (\alpha' \ominus \beta'\nabla_{ZZ}) & \{ \text{ Dedekind formula and Proposition A.3(a) } \} \\ \sqsubseteq & (\alpha' \odot \beta')\nabla_{ZZ}. & \end{array}$$

We have to see $\alpha' \odot \beta' \sqsubseteq (\alpha \odot \beta)^+$, but this claim can be shown by the same argument of the second part in the proof for (a). □

References

[1] R. Backhouse and J. van der Woude, Demonic operators and monotype factors, Math. Struct. in Comp. Science **3**(1993), 417–433.

[2] R. Berghammer, Relational specification of data types and programs, Bericht **9109**, Universität der Bundesweht München, Fakultät für Informatik, 1991.

[3] R. Berghammer and H. Zierer, Relational algebraic semantics of deterministic and nondeterministic programs, *Theoretical Computer Science***43** (1986), 123–147.

[4] R. Bird and O. de Moor, Algebra of programming (Prentice Hall, London, 1997).

[5] J. Desharnais et al., Embedding a demonic semilattice in a relation algebra, *Theoretical Computer Science***149**(1995), 333–360.

[6] Y. Kawahara. Lattices in Dedekind categories. In Ewa Orłowska and Andrzej Szałas, editors. *Relational Methods for Computer Science Applications*, (Springer, Heidelberg, 2001), 247–260.

[7] Y. Kawahara and H. Furusawa. Crispness in Dedekind categories. *Bulletin of Informatics and Cybernetics*, 33:1–18, 2001.

[8] G. Schmidt and T. Ströhlein, *Relations and graphs – Discrete Mathematics for Computer Science* – (Springer-Verlag, Berlin, 1993).

[9] J. van der Woude, Free style specwrestling: Demonic composition and choice. In: *Lambert Meertens, CWI, Liber Amicorum, 1966-1991,* Stichting Mathematisch Centrum, Amsterdam.

[10] L. Xu, M. Takeichi and H. Iwasaki, Relational semantics for locally nondeterministic programs, New Generation Computing **15**(1997), 339–362.

[11] L. Xu, M. Takeichi and H. Iwasaki, Relational definition of UNITY loop, Transactions of Information Processing Society of Japan **39**(1998), 646–655.

A Basic Properties of Relations

In this section we list a few basic properties of relations.

Proposition A.1 *Let $\alpha : X \rightharpoonup Y$, $\beta : Y \rightharpoonup Z$, $\eta : X \rightharpoonup W$ and $u : X \rightharpoonup X$ be relations in a Dedekind category $\mathcal{D}$. Then the following hold:*

(a) *If $u \sqsubseteq \mathrm{id}_X$ then $u\nabla_{XY} \sqcap \alpha = u\alpha$.*

(b) $(\alpha \sqcap \eta\nabla_{WY})\beta = \alpha\beta \sqcap \eta\nabla_{WZ}$.

□

Proposition A.2 *Let $\alpha, \beta, \gamma : X \rightharpoonup Y$ be relations in a Dedekind category $\mathcal{D}$. Then the following hold:*

(a) $\beta \sqsubseteq \alpha \Rightarrow \beta$.

(b) $\alpha \Rightarrow \alpha = \nabla_{XY}$ *and* $\nabla_{XY} \Rightarrow \alpha = \alpha$.

(c) $\alpha \sqcap (\alpha \Rightarrow \beta) = \alpha \sqcap \beta$. *In particular* $\alpha\nabla_{YY} \sqcap \alpha^+ = \alpha$.

(d) $\alpha \Rightarrow (\beta \Rightarrow \gamma) = (\alpha \sqcap \beta) \Rightarrow \gamma$. *In particular* $\alpha\nabla_{YY} \Rightarrow \alpha^+ = \alpha^+$.

(e) $(\alpha \sqcup \alpha') \Rightarrow \beta = (\alpha \Rightarrow \beta) \sqcap (\alpha' \Rightarrow \beta)$.

(f) *If $\alpha \sqsubseteq \beta$, then $\alpha \sqcap (\beta \Rightarrow \gamma) = \alpha \sqcap \gamma$.*

(g) *If $\alpha \sqsupseteq \alpha'$ and $\beta \sqsubseteq \beta'$, then $\alpha \Rightarrow \beta \sqsubseteq \alpha' \Rightarrow \beta'$.*

□

Proposition A.3 *Let $\alpha, \alpha' : X \rightharpoonup Y$, $\beta, \beta' : Y \rightharpoonup Z$, $\delta : Z \rightharpoonup W$ and $\xi : X \rightharpoonup W$ be relations in a Dedekind category $\mathcal{D}$. Then the following hold:*

(a) $\alpha^\sharp(\alpha \ominus \beta) \sqsubseteq \beta$.

(b) $(\alpha \ominus \beta)\delta \sqsubseteq \alpha \ominus (\beta\delta)$.

(c) $\alpha \ominus (\beta \sqcap \beta') = (\alpha \ominus \beta) \sqcap (\alpha \ominus \beta')$.

(d) $(\alpha\beta) \ominus \delta = \alpha \ominus (\beta \ominus \delta)$.

(e) *If $\nabla_{WZ}\nabla_{ZW} = \nabla_{WW}$, then $(\alpha \ominus \beta\nabla_{ZZ})\nabla_{ZW} = \alpha \ominus \beta\nabla_{ZW}$.*

(f) $\alpha(\beta \sqcap \beta') \sqcap \alpha \ominus \beta' = \alpha\beta \sqcap \alpha \ominus \beta'$.

(g) $(\xi\nabla_{WX} \sqcap \mathrm{id}_X) \ominus \alpha = \xi\nabla_{WY} \Rightarrow \alpha$.

(h) $(\xi\nabla_{WY} \sqcap \alpha) \ominus \beta = \xi\nabla_{WZ} \Rightarrow (\alpha \ominus \beta)$.

(i) *If $\alpha \sqsupseteq \alpha'$ and $\beta \sqsubseteq \beta'$, then $\alpha \ominus \beta \sqsubseteq \alpha' \ominus \beta'$.*

□

明示的環境計算体系への部分型の導入

澤田 康秀　(Yasuhide Sawada)
京都大学情報学研究科　(Graduate School of Informatics, Kyoto University)

1　はじめに

型付き λ 計算は、関数型プログラミング言語の基礎理論として位置づけられている体系である。この型付き λ 計算の表現力を拡充していくために様々な研究が行われているが、その中の一つに明示的環境の導入がある。それを実現した体系として挙げられるのが $\lambda\varepsilon$ [1] である。$\lambda\varepsilon$ は、合流性や強正規化性といった良い性質を保持するという点で優れた計算体系である。

環境という概念は、変数の値を保持する箱のようなものを表すが、一般的なプログラミング言語においては暗黙のうちに扱われてしまうことが多い。明示的環境とは、これを syntax の要素として明示的に扱えるようにしたものである。

$\lambda\varepsilon$ における明示的環境は、明示的代入とレコードをさらに一般化した概念でもある。明示的代入とは、単純型付き λ 計算などではメタレベルで行われる代入という操作を、明示的に扱えるようにするものである。一方レコードとは、型の異なる複数の項をまとめて扱うための概念であり、関数の引数や戻り値にもなり得る first class object である。$\lambda\varepsilon$ では、明示的環境を、これらの特徴を全て併せ持つ概念として用いることができる。

レコードの概念を型付き λ 計算に導入した体系はレコード計算と呼ばれており、それはオブジェクト指向言語のメカニズムを λ 計算上で表現することを目指したものでもある。このレコード計算において、いわゆるメソッドの継承に相当するしくみを表現するために部分型という概念が用いられるようになった [2, 3]。部分型を導入することで、より柔軟性の高い型システムを構築することが出来る。

明示的環境がレコードを一般化した概念であるならば、明示的環境計算に部分型を導入することは、レコード計算の場合と同様に有意義であると考えられた。そこで本研究では、$\lambda\varepsilon$ への部分型の導入を行った。

2　明示的環境

計算体系 $\lambda\varepsilon$ における明示的環境の扱いについて述べることにする。明示的環境は、変数の値を保持するためのものであり、例えば

$$\{2/x^{int}, 3/y^{int}\}$$

という形 [1] で記述する。この場合は、x^{int} の値を 2、y^{int} の値を 3 とするような環境を表している。

$\lambda\varepsilon$ は型付きの計算体系であるから、明示的環境にも型が付けられることになる。たとえば、$\{2/x^{int}, 3/y^{int}\}$ の型は $\{x^{int}, y^{int}\}$ である。環境の型は、環境が束縛する変数の集合として定義されることになっている。

また、$e[\![a]\!]$ と書いて、環境 e のもとで項 a を評価することを表す。すなわち、

$$\{2/x^{int}, 3/y^{int}\}[\![x^{int} + y^{int}]\!]$$

という項を書くことができ、これは $\{2/x^{int}, 3/y^{int}\}$ という環境のもとで x と y の値を足すことを表す。この場合は当然、$2 + 3$ という計算が行われることになる。

3　部分型

部分型とは、部分型関係 (型の包含関係にもとづいて定義される関係) を用いて型付けを行う仕組みのことである。$\lambda\varepsilon$ に対してこの部分型を導入することを考える。

3.1　部分型関係

$\lambda\varepsilon$ における型の定義は次のようになっている。

$$\begin{array}{rcll} A, B & ::= & K & \text{(基底型)} \\ & | & A \Rightarrow B & \text{(関数型)} \\ & | & E & \text{(環境型)} \end{array}$$

$$E \quad ::= \quad \{x_1^{A_1}, \ldots, x_n^{A_n}\} \qquad (n \geq 0)$$

この型に対する部分型関係 $<:$ は、以下の規則によって定めることが出来ると考えられる。

[1] $\lambda\varepsilon$ では、全ての変数は型を付けて記述される。

(**S-ref**)
$$\frac{}{A <: A}$$

(**S-trans**)
$$\frac{A <: B \quad B <: C}{A <: C}$$

(**S-arrow**)
$$\frac{B <: A \quad A' <: B'}{A \Rightarrow A' <: B \Rightarrow B'}$$

(**S-env**)
$$\frac{A_i <: B_i \ (i = 1, \ldots, m) \quad n \geq m}{\{x_1^{A_1}, \ldots, x_n^{A_n}\} <: \{x_1^{B_1}, \ldots, x_m^{B_m}\}}$$

規則 (**S-env**) は、環境型の大小関係を規定したものである。これは、レコード計算体系におけるレコード型の大小関係の規定と同様の形式をとっている。

3.2 Coercion

部分型を用いるためには、部分型関係を取り入れた型付け規則が必要になる。レコード計算体系などで最も一般的に用いられるのは、次の規則 (subsumption rule) である。

$$\frac{\Gamma \vdash a : A \qquad A <: B}{\Gamma \vdash a : B}$$

この規則は、文脈 Γ のもとで項 a が型 A を持ち、かつ $A <: B$ という部分型関係が成り立つとき、項 a の型を B に置き換えることを許すというものである。

しかし、この規則を $\lambda\varepsilon$ に導入すると不具合が生じてしまう。subsumption rule は、項を変化させずに型だけを置き換えることを許す規則であるから、項に対する型の唯一性が崩れてしまうという問題がある。そして $\lambda\varepsilon$ では、型をチェックしながら項の計算を行っていくという特徴があり、項に対して型が正しく決められなければ、それが誤った計算につながる恐れも生じてくるのである。

そこで、subsumption rule の導入は行わず、代わりに次の規則を追加することにする。

$$\frac{\Gamma \vdash a : A \qquad A <: B}{\Gamma \vdash a|_B \ : B}$$

つまり、部分型関係を用いて型を置き換える場合、その型の情報を項にも付加しておくことにするということである。$a|_B$ のように、型情報を明記しておいて、計算の際にその型に応じた項の変形を行う仕組みのことを coercion

[4, 5] という。この coercion を導入することによって、項の持つ型の情報を計算時に正しく反映させることが可能となる。

4 体系 $\lambda\varepsilon c$

本研究では、明示的環境計算の体系 $\lambda\varepsilon$ を拡張し、部分型を導入した $\lambda\varepsilon c$ という体系の構築を行った。$\lambda\varepsilon c$ は、coercion をサポートする仕組みを備えた体系であり、部分型を用いる際には coercion の適用を要求するという形をとっている。

4.1 定義

Definition 1 型

$$\begin{array}{rcl} A, B & ::= & K \mid A \Rightarrow B \mid E \\ E & ::= & \{x_1^{A_1}, \ldots, x_n^{A_n}\} \end{array}$$

K は基底型を、E は環境の型を表す。E の定義において、$n \geq 0$ であり、$x_i^{A_i}$ は互いに相異なっていなければならない。

Definition 2 項 [2]

$$\begin{array}{rcl} a, b, e & ::= & x^A \\ & \mid & \lambda x^A.\, b \\ & \mid & ba \\ & \mid & \{a_1/x_1^{A_1}, \ldots, a_n/x_n^{A_n}\} \\ & \mid & e[\![a]\!] \\ & \mid & a|_A \end{array}$$

$a|_A$ は coercion の項であり、a の型が A であることを明示するためのものである。明示的環境 $\{a_1/x_1^{A_1}, \ldots, a_n/x_n^{A_n}\}$ において、$x_1^{A_1}, \ldots, x_n^{A_n}$ $(n \geq 0)$ は互いに異なっていなければならない。

Definition 3 文脈、型判定

文脈は、宣言 (x^A という形の式) の集合である。また、文脈 Γ のもとで項 a が型 A を持つとき、$\Gamma \vdash a : A$ と書き、これを型判定という。

Definition 4 部分型関係 <:

$\lambda\varepsilon c$ における部分型関係 <: は、以下の規則によって定義される、型の二項関係である。

[2] $\lambda\varepsilon c$ の項であることを強調するときには、$\lambda\varepsilon c$-term と呼ぶ。

(**S-ref**)
$$\frac{}{A <: A}$$

(**S-trans**)
$$\frac{A <: B \quad B <: C}{A <: C}$$

(**S-arrow**)
$$\frac{B <: A \quad A' <: B'}{A \Rightarrow A' <: B \Rightarrow B'}$$

(**S-env**)
$$\frac{A_i <: B_i \ (i = 1, \ldots, m) \quad n \geq m}{\{x_1^{A_1}, \ldots, x_n^{A_n}\} <: \{x_1^{B_1}, \ldots, x_m^{B_m}\}}$$

部分型関係 $A <: B$ が成り立っているとき、以下のいずれか 1 つが必ず成り立つことが明らかである。

- A, B はともに基底型で、かつ $A \equiv B$ 。
- A, B はともに関数型。
- A, B はともに環境型。

Definition 5　型付け規則

型判定の導出に用いられる型付け規則は、次のように定義される。

(**assume**)
$$\frac{}{x^A \vdash x^A : A}$$

$(\Rightarrow I)$
$$\frac{\Gamma \vdash b : B}{\Gamma - \{x^A\} \vdash \lambda x^A.\, b : A \Rightarrow B}$$

$(\Rightarrow E)$
$$\frac{\Gamma \vdash b : A \Rightarrow B \quad \Delta \vdash a : A}{\Gamma, \Delta \vdash ba : B}$$

(**env**I)
$$\frac{\Gamma_1 \vdash a_1 : A_1 \quad \cdots \quad \Gamma_n \vdash a_n : A_n}{\Gamma_1, \ldots, \Gamma_n \vdash \{a_1/x_1^{A_1}, \ldots, a_n/x_n^{A_n}\} : \{x_1^{A_1}, \ldots, x_n^{A_n}\}}$$

(**env**E)
$$\frac{\Gamma \vdash e : E \quad \Delta \vdash a : A}{\Gamma, (\Delta - E) \vdash e[\![a]\!] : A}$$

(**coercion**)
$$\frac{\Gamma \vdash a : A \quad A <: B}{\Gamma \vdash a|_B : B}$$

任意の項 a が型を持つならば、それは 1 つに決まる。その型のことを $\mathrm{TY}(a)$ と書くことにする。

Definition 6　自由変数、束縛変数

項 a 中の自由変数の集合を $\mathrm{FV}(a)$ で表す。$\mathrm{FV}(a)$ の定義は以下の通りである。項 a 中に現れる変数のうち、$\mathrm{FV}(a)$ に含まれないものは束縛変数である。

- $\mathrm{FV}(x^A) \ ::= \ \{x^A\}$
- $\mathrm{FV}(\lambda x^A.\, b) \ ::= \ \mathrm{FV}(b) - \{x^A\}$
- $\mathrm{FV}(ba) \ ::= \ \mathrm{FV}(b) \cup \mathrm{FV}(a)$
- $\mathrm{FV}(\{a_1/x_1^{A_1}, \ldots, a_n/x_n^{A_n}\}) \ ::= \ \mathrm{FV}(a_1) \ \cup \ldots \cup \mathrm{FV}(a_n)$
- $\mathrm{FV}(e[\![a]\!]) \ ::= \ \mathrm{FV}(e) \ \cup \ (\mathrm{FV}(a) - \mathrm{TY}(e))$
- $\mathrm{FV}(a|_A) \ ::= \ \mathrm{FV}(a)$

Definition 7　α 同値

- 項 a に含まれる自由変数 x^A をすべて y^A (a 中に現れない変数) で置き換えたものを b とおくと、$\lambda x^A.\, a$ と $\lambda y^A.\, b$ は α 同値である。
- a の部分項 b を、b と α 同値な b' で置き換えた項を a' とおくと、a と a' は α 同値である。

α 同値な項同士は、同一の項として扱われる。

Definition 8　簡約規則

$\lambda\varepsilon c$ における簡約規則の定義は以下のようになる。

$$(\lambda)\quad (\lambda x^A.b)a \ \mapsto_\lambda \ \{a/x^A\}[\![b]\!]$$

$$\begin{array}{rl}
(\mathbf{gc}) & e[\![a]\!] \ \mapsto_\varepsilon \ a \quad \text{if } \mathrm{TY}(e) \cap \mathrm{FV}(a) = \emptyset \\
(\mathbf{var}) & \{a_1/x_1^{A_1}, \ldots, a_n/x_n^{A_n}\}[\![x_i^{A_i}]\!] \ \mapsto_\varepsilon \ a_i \quad (1 \le i \le n) \\
(\mathbf{abs}) & e[\![\lambda x^A.b]\!] \ \mapsto_\varepsilon \ \lambda x^A.e[\![b]\!] \quad \text{if } x^A \notin \mathrm{TY}(e) \cup \mathrm{FV}(e) \\
(\mathbf{app}) & e[\![ba]\!] \ \mapsto_\varepsilon \ e[\![b]\!]e[\![a]\!] \\
(\mathbf{env}) & e[\![\{a_1/x_1^{A_1}, \ldots, a_n/x_n^{A_n}\}]\!] \ \mapsto_\varepsilon \ \{e[\![a_1]\!]/x_1^{A_1}, \ldots, e[\![a_n]\!]/x_n^{A_n}\} \\
(\mathbf{eval}) & e[\![f[\![x^A]\!]]\!] \ \mapsto_\varepsilon \ e[\![f]\!][\![x^A]\!] \quad \text{if } x^A \in \mathrm{TY}(f) \\
(\mathbf{coe}) & e[\![a|_A]\!] \ \mapsto_\varepsilon \ e[\![a]\!]|_A
\end{array}$$

$$\begin{array}{rl}
(\mathbf{c\text{-}atom}) & a|_K \ \mapsto_c \ a \\
(\mathbf{c\text{-}app}) & (b|_{A \Rightarrow B})\, a \ \mapsto_c \ (b\,(a|_C))|_B \quad (\mathrm{TY}(b) \equiv C \Rightarrow D) \\
(\mathbf{c\text{-}env}) & \{a_1/x_1^{A_1}, \ldots, a_n/x_n^{A_n}\}|_{\{x_1^{B_1}, \ldots, x_m^{B_m}\}} \\
 & \qquad \mapsto_c \ \{a_1|_{B_1}/x^{B_1}, \ldots, a_m|_{B_m}/x^{B_m}\} \qquad (n \ge m)
\end{array}$$

規則 (**abs**) における条件 $x^A \notin \mathrm{TY}(e) \cup \mathrm{FV}(e)$ は、項の α 同値性を利用して変数 x^A を適切に選ぶことで、常に成り立つようにすることが出来る。規則 (**c-atom**) は、基底型による coercion がそのまま外してしまえることを意味する。$a\,|_K$ という項において、a の持つ型は必ず K に一致するため、これは明らかである。

規則 (**c-app**) は関数に対する coercion を移動させるための規則である。たとえば、

$$(\lambda x^A.b)\,|_{C \Rightarrow D}\ c$$

という項があるとき、このままでは (λ) の規則が適用できないため、(**c-app**) のような規則が必要になると考えられる。

規則 (**c-env**) は環境に対する coercion の処理を記述したものである。その最も重要な役割は、不要な変数束縛の除去である。環境 e の中に a/x^A という変数束縛が含まれていても、$x^A \notin \mathrm{TY}(e)$ ならば、e は x^A を束縛しない[3] 。よって e 中の a/x^A は除去してしまうことが出来るのである。

4.2 計算例

従来の $\lambda\varepsilon$ では、たとえば $\lambda z^{\{x^A\}}.z^{\{x^A\}}$ という関数に $\{a/x^A, b/y^B\}$ という引数を与えることは許されなかった。関数の仮引数 z の型が $\{x^A\}$ なのに対し、実引数の型が $\{x^A, y^B\}$ であり、両者が一致しないためである。

しかし $\lambda\varepsilon c$ では、部分型の導入によって環境 $\{a/x^A, b/y^B\}$ の型を $\{x^A, y^B\}$ ではなく $\{x^A\}$ とすることも出来るようになるため、この環境を上の関数に渡すことが可能となる[4] 。

ただし、$\{a/x^A, b/y^B\}$ の型が $\{x^A\}$ になるということは、この環境が x^A しか束縛できなくなることを意味する。そのことが計算時に正しく反映されないと、誤って y^B も束縛してしまうなどの不具合が生じる可能性がある。よって $\{a/x^A, b/y^B\}$ の型を $\{x^A, y^B\}$ から $\{x^A\}$ に置き換える際には、coercion を適用して $\{a/x^A, b/y^B\}\,|_{\{x^A\}}$ と書き、型が $\{x^A\}$ であることを明記しておくことが重要となる。

以下に、

$$((\lambda z^{\{x^{int}\}}.\, z^{\{x^{int}\}})\{1/x^{int}, 10/y^{int}\}\,|_{\{x^{int}\}})\,[\![x^{int} + y^{int}]\!]$$

という項に対する計算の例を挙げる。

---

[3] 自由変数、束縛変数の定義からそれは明らかである。

[4] 勿論、型を置き換えるのは引数ではなく関数の方であってもよい。

$$
\begin{aligned}
&((\lambda z^{\{x^{int}\}}.\, z^{\{x^{int}\}})\{1/x^{int}, 10/y^{int}\}|_{\{x^{int}\}})\,[\![x^{int}+y^{int}]\!] \\
&\to\ \{\{1/x^{int}, 10/y^{int}\}|_{\{x^{int}\}}/z^{\{x^{int}\}}\}[\![z^{\{x^{int}\}}]\!][\![x^{int}+y^{int}]\!] \\
&\to\ \{1/x^{int}, 10/y^{int}\}|_{\{x^{int}\}}[\![x^{int}+y^{int}]\!] \\
&\to\ \{1/x^{int}, 10/y^{int}\}|_{\{x^{int}\}}[\![x^{int}]\!] + \{1/x^{int}, 10/y^{int}\}|_{\{x^{int}\}}[\![y^{int}]\!] \\
&\overset{+}{\to}\ 1 + \{1/x^{int}, 10/y^{int}\}|_{\{x^{int}\}}[\![y^{int}]\!] \\
&\to\ 1 + y^{int}
\end{aligned}
$$

この場合、$\{1/x^{int}, 10/y^{int}\}$ の型は $\{x^{int}\}$ として扱われている。よって $10/y^{int}$ という変数束縛は無効であり、y^{int} に 10 を代入してしまったとしたらそれは誤りである。$\lambda\varepsilon c$ では、coercion の適用によってそのような間違いを防ぐことが出来るようになっている。

4.3 性質

本研究では、$\lambda\varepsilon c$ が以下のような性質を満たすことが証明できた。(証明の詳細は [6] に記載。)

Theorem 9　Subject Reduction
$\Gamma \vdash a : A$ かつ $a \to_{\lambda\varepsilon c} b$ となるとき、$\Delta \subseteq \Gamma$ であるような Δ が存在し、$\Delta \vdash b : A$ が成り立つ。

Theorem 10　$\lambda\varepsilon$ に対する Conservativity
a, b は $\lambda\varepsilon$ の項とする。このとき、$a \overset{*}{\to}_{\lambda\varepsilon} b \ \Leftrightarrow\ a \overset{*}{\to}_{\lambda\varepsilon c} b$ 。

Theorem 11　強正規化性
任意の $\lambda\varepsilon c$-term に対する簡約は必ず有限回で停止する。

Theorem 12　合流性
$a \overset{*}{\to}_{\lambda\varepsilon c} b$ かつ $a \overset{*}{\to}_{\lambda\varepsilon c} c$ ならば、$b \overset{*}{\to}_{\lambda\varepsilon c} d$ かつ $c \overset{*}{\to}_{\lambda\varepsilon c} d$ となる d が必ず存在する。

5　結論

本研究では、明示的環境計算の体系 $\lambda\varepsilon$ を拡張し、部分型を持つ計算体系 $\lambda\varepsilon c$ を構築することが出来た。環境計算において部分型を用いる場合、項の持つ型の情報が計算に正しく反映されなくなる恐れが生じる、という問題点が浮かび上がったが、$\lambda\varepsilon c$ では coercion の導入によってそれを防ぐ仕組みが実現された。そしてこの体系が、合流性や強正規化性などの望ましい性質を満たすことも証明できた。

今後の課題としては、$\lambda\varepsilon c$ をさらに発展させ、coercion の黙示化 (自動補完) を可能にするような計算系を構築することが挙げられる。また、型抽象

などの有力な概念を組み込んでいくことも、興味深いテーマであると考えられる。

謝辞

丁寧な御指導により本研究を支えていただきました、京都大学情報学研究科佐藤雅彦教授、五十嵐淳講師、中澤巧爾助手には心から感謝致します。また、本研究について数々の助言を下さった皆様方にも深く御礼を申し上げます。

参考文献

[1] M. Sato, T. Sakurai, and R. Burstall. Explicit environments. *Fundamenta Informaticae*, Vol. 45, No. 1-2, pp. 79–115, 2001.

[2] L. Cardelli. A semantics of multiple inheritance. *Information and Computation*, Vol. 76, pp. 138–164, 1988.

[3] B. Pierce. *Types and Programming Languages*. The MIT Press, 2002.

[4] V. Breazu-Tannen, T. Coquand, C. Gunter, and A. Scedrov. Inheritance as implicit coercion. *Information and Computation*, Vol. 93, pp. 172–221, 1991.

[5] H Tsuiki. *A Record Calculus with a Merge Operator*. PhD thesis, Keio University, 1992.

[6] 澤田康秀. 部分型を持つ明示的環境計算. 修士論文, 京都大学情報学研究科知能情報学専攻, 2003.

A study on an immune network dynamical system model

Satoko Itaya
ATR Adaptive Communication Research Laboratories Dept. 2

§1. Introduction

Recently, there has been considerable interest in the development of autonomous networks in various fields. Autonomy in information devices in personal and home network and the research related to ad hoc networks are especially active areas. Immune systems seem to be good examples of autonomous network systems that do not have central management. I have been interested in whether dynamical models of immune systems may give us an understanding of this aspect of immune systems. At this research meeting, I report about the characteristics of immunity seen in some dynamical network models.

First, I give an about the outline of immunity in §2, and then explain the dynamical model in §3. In §4, I describe the relation between the formation process of the network and its structure. A summary and guidelines for the future are mentioned at the end.

§2. Immune system

We humans have two immune systems in our body. One is natural immunity, the other is adaptive immunity. All animals have natural immunity. In this system, phagocytes, complements and cytokines work together. Natural immunity has low recognition and fast action (on the order of minutes or hours). But, it does not change with age or experience of transmission, and it has no memory. Only vertebrate animals also have adaptive immunity. In this system, lymphocytes, antibodies, and cytokines work together. It has high recognition and slow action (on the order of days).

Next, I will explain about the solid structure in the antibody which is the leading part of this research. The typical shape of immunoglobulin is like the letter Y (Fig.1).

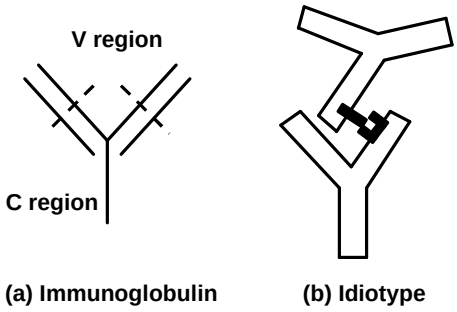


Fig. 1 The basic structure of (a) immunoglobulin, (b) Idiotype

The upper part of the Y is called the “various region” because the arrangement of the gene often causes mutation and rich changes. In contrast, the part at the bottom of the Y is called the “constant region” (Fig. 1 (a)). We call the three-dimensional structure of the V region, which is characteristic of each immunoglobulin, an idiotype (Fig.1 (b)).

There are two kinds of immunoglobulin. One is the membrane-bound immunoglobulin. It uses B-cell antigen receptors. B-cells connect antigens with this receptor. And they have the same structure of antibodies, which they can produce. Generally, B-cells cannot produce only through this action. It is a trigger for the setup of antibody generation in B-cells.

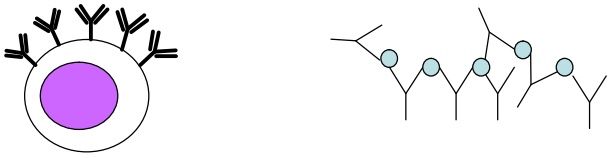


Fig. 2 (a) B-cell and membrane-bound immunoglobulin and (b) anitibodies

The other kind is secreted immunoglobulin. Generally, this is called an antibody. Antibodies have several roles in the immune system. First, antibodies neutralize antigens. And antibodies make it easier to reject invading antigens. For example, the C regions of antibodies excite phagocytes, and the antibodies and antigens make a cross-link structure. They become big lumps, and so they are easily found by other immunocytes.

When B-cells are activated, they produce antibodies. There are some roots that exist for B-cell activation. One is the T-cell-independent response. Two typical materials make this type of reaction.

Material objects, which cause cell division, immediately activate B-cells, which cannot recognize the antigen. This is not remembered.

The antigens that have the structure repeated with the same antigenic determinants can cross-link the receptors of B-cells to recognize the antigen. This reaction is also not remembered.

Another is the T-cell-dependent response.

First, the macrophage preys on an antigen. It expresses the protein of the antigen that resolved it in its cell surface, and gives the antigen presentation. The changed phagocyte is recognized by a helper T-cell, and the helper T-cell influences the maturity of the B-cell by using the chemical substance of interleukin. This interleukin is the cause of fever and inflammation. The activated B-cell matures with splitting. Some parts of the split B-cell maintain memory cells, and the other parts become plasma cells. Plasma cells can produce antibodies, and memory cells can change to plasma cells quickly. As a response proceeds,

its suspension is influenced to the B-cell with the use of interleukin by a suppressor T-cell, and the response is settled.

In 1974, Jerne advocated the theory of the idiotype immune network (ref.1). His theory is as follows.

Generally, an inactive immune system is activated with the invasion of antigens. However, we want to keep minimum number of active immune system members. For this reason, when there is no invasion of antigens, a minimum number of members are activated with the personal protein and the cell. For example, an antibody molecule is formed inside the body due to an unintentional mutation. The new antibody formed at random must be a foreign substance for the antibody that has existed inside the body from the first. Because of that, we can easily imagine that an antibody to react with the antibody is newly formed. In this way, one antibody responds to another antibody as an internal image of external antigens and we keep a minimum number of active immune system members. This is his theory, the network theory. This theory is partly right and it is experimentally well known that the antibodies of the neogenesis mouse have antibody-antibody interaction. But because the experimental study is very difficult to conduct on adult animals, and because other immune cells, T-cells, etc., and some classes of interleukins have been discovered, the network theory has received less attention.

In 1988, Varela et al. proposed a dynamical system model in which both B-cells and antibodies are taken into account (ref. 2). He introduced the effects of T-cells and interleukin as functions of the B-cell. Here, I introduce some models of theoretical immunology and recent fields of application.

In 1989, Bagley et. al. defined the 3-dimensional structure of antigens, and investigated the topology of the network (ref. 3). In 1990, Parisi investigated the capacity of memory with the simple spin-glass idiotype network model (ref. 4). In 1993, De Boer et. al. considered the structure of the cross-link of MIgs, and advocated the B-model (ref. 5). In B-models, B-cells proliferate according to a phenomenological log bell-shaped function. In addition, there is the activated preparation of antigen generation model, the vaccinated model, and so on

§3. Model

In this study, we use Varela's model.

One of the reasons is that there are two equations, for both antibodies and B-cells. And the effects of interleukins and T-cells are described as a function of the maturation and proliferation of B-cells. This model is described by the following two equations (equation 1).

$$\frac{df_i}{dt} = -K_1 s_i f_i - K_2 f_i + K_3 Mat(s_i) b_i$$
$$\frac{db_i}{dt} = -K_4 b_i + K_5 Prol(s_i) b_i + K_6 \quad (1)$$

i=1,…,N、 f_i *;concentration of antibodies,* b_i *;concentration of B-cells* $Mat(s_i)$ *; function of maturation of B-cells,* $Prol(s_i)$ *; function of proliferation of B-cells,* K_1 *; the rate of death by antibody-antibody interaction,* K_2 *; the rate of natural death of antibodies,* K_3 *; the rate of antigens generated by B-cells,* K_4 *; the rate of natural death of B-cells,* K_5 *; the rate of increase of B-cells,* K_6 *; the rate of supply of B-cells from bone marrow*

First, there is the equation of the concentration of antibodies. In this equation, the first term is death by antibody-antibody interaction, the second term is the natural death of antibodies, and the third term is antigen generated by B-cells. Second, there is the equation of the concentration of B-cells. In this equation, the first term is the death of B-cells, the second term is the increase of B-cells division, and the third term is supply of B-cells from bone marrow. This pair of equations describes the behavior of the *i* th clone. If there are two clones, the concentrations of two clones change in the anti-phase. For example, B-cell 1 can generate free antibody-1. There are many B-cell-1s and produced free antibody-1s. We call the whole group “CLONE”.

The constituents of the network, the free antibodies and B-cells, interact with each other through idiotypes. Between two different idiotypes *i* and *j*, there may occur an affinity, which is represented by the connectivity m_{ij}. We set $m_{ij} = 1$ if there is an affinity between *i* and *j*, and $m_{ij} = 0$ if there is none. In some cases, $m_{ij} = 1$ is experimentally measurable. The sensitivity of the network for the *i* th idiotype is defined as

$$s_i = \sum_1^N m_{ij} f_j \quad (2).$$

The probability of the maturation and proliferation of B-cells is assumed to depend on their sensitivity s. An antibody is formed only from a B cell (plasma cell) that has matured. It is well known that both of these functions have dual thresholds depending on affinity.

In order to understand the effect of the maturation and proliferation functions, we change these functions. Even though the function was changed, we found that typical behaviors were maintained (ref. 6). Because of these results, we modify the model by choosing simpler functions (Fig. 3).

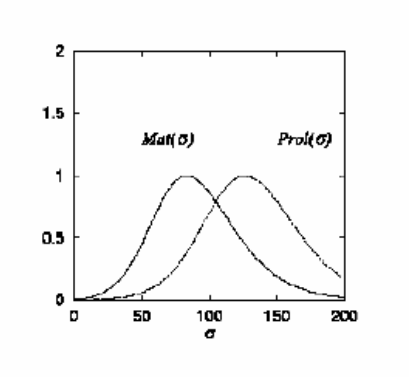


Fig. 3 The mature and proliferation functions used in this study

Here, we introduce a threshold above which antibodies can recognize antibodies and antigens. This is because recognition is not possible if the concentration of antigens inside the body isn't comparatively high. In this case, there are some non-symmetric limit cycles depending on the value of the threshold. We call this condition the differentiating state. Next, we set the elements of a connectivity matrix depending on the concentration of antibodies. If *i* does not equal *j*, when $f_j > f_0$, antibody *j* is recognized by other antibodies, and $m_{ij} = 1$. And when $f_j < f_0$, antibody *j* is not recognized, and $m_{ij} = 0$. Here, because we assume they don't have self connectivity, if *i* equals *j*, $m_{ij} = 0$.

Next, we studied response to the external perturbation in a small network. In a small network of this model, non-symmetric limit cycles exist. In a 3-clone network, each clone has an S-state or L-state condition and there is a differentiating state with two L-state and one S-state clones.

The S-state has a short time over the threshold, and the L-state has a long time over the threshold. Now, clones can respond to an antigen only when their antibody concentration is over the threshold. In view of this, the S-state is unsuitable and the L-state is suitable for reacting with antigens. It can also be said that the short-term memory of the network is suitable when the clone is L-State (ref. 6).

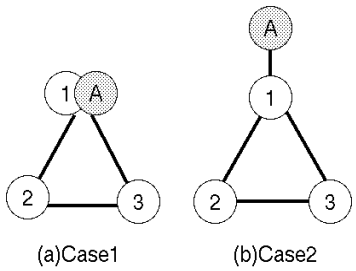


Fig. 4 Antigen invasion

We considered two cases of antigen invasion in a 3-clone system (Fig. 4). In case 1, we consider that external antigens similar to antibodies f_1 invade the 3-clone closed network. In case 2, we consider that external antigens interact only with clone-1. In both cases, we introduce the antigen equation as follows.

$$\frac{dA}{dt} = -K_1 s_A(t) A + K_7 \quad (3)$$

A is the concentration and K_7 is the increase rate of antigens. And in this equation, s_A is defined respectively in case 1 (4) and case 2 (5).

$$s_A(t) = m_{12}(t) f_2(t) + m_{13}(t) f_3(t) \quad (4)$$

$$s_A(t) = \Theta(f_1(t) - g_{1,0}) f_1(t) \quad (5)$$

In both case 1 and case 2, when the reproduction rate of antigens becomes high, it shifts to a more proper attractor arrangement, and the antigens are caught.

We also study antigen invasion in a 4-clone system in 3 cases, (a) the case where a clone that can respond with antigens is L-State, (b) the case where a clone that cannot respond with antigens is L-State, and (c) the case where the system is chaotic. In this figure, we compared the average time of antigen moderation.

The relaxation time Ta for case (a) is the shortest, and Tb for case (b) is the longest. Relaxation time Tc for case (c) is between Ta and Tb. These results suggest that a chaotic state is more effective than the differentiating states for preparing for various types of antigens.

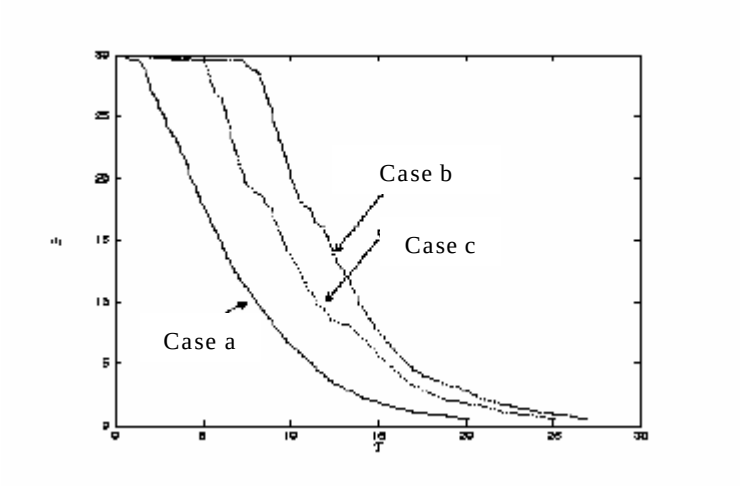


Fig. 5 (a) The clone that can respond with antigen A is L-state, (b) the clone that can NOT respond with antigen A is L-state, (c) the system is chaotic

We have written in detail about these results in ref. 6.

§4. Formation process and network structure

Before considering a large network, it is an important problem to consider whether a network is truly one network or whether it can be divided into sub-networks. If the network can be divided into sub-networks, the study of the smaller networks is very important. As for this model, the condition that the concentration of each clone's antibody and the B cell oscillates is called "activation".

We analyze the activated conditions of a large network for the case in which each clone interacts with all other clones or only a few other clones in the network (ref. 7). When each clone interacted with all clones in the network, we did not find any activated clone in the network. Furthermore, when each clone interacted with only a few other clones, we found that many clones are activated in the network. Therefore, in this model, each clone must have only a few other connections to activate the network.

We also studied the effect of threshold as a localized mechanism of immune response (ref. 8). The roles of the threshold in units are as follows. When it is attacked by antibodies, a network with a threshold doesn't break easily from a network without a threshold. By introducing a threshold, the independence of each clone is increased, the collapse of the network is made more difficult, and the size of the parameter areas where the system functions as a network increases.

To check the role of the threshold in the network, we connect three basic units loosely (Fig. 3).

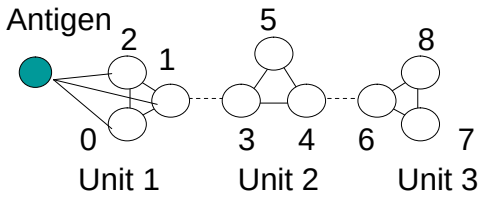


Fig. 6　Connection of 3-clone units

There is no threshold in each basic unit. We set the threshold between each unit, to ku=0 or ku=50. We examined the way that of fluctuation spreads when unit 1 was disturbed. In both cases, the disturbance does not spread significantly to unit 3. Further, we have found that the disturbance in unit 2 is reduced considerably for ku=50, while it is still large for ku=0. As a result, the threshold prevents the spread of local fluctuation through the whole network (Fig. 4).

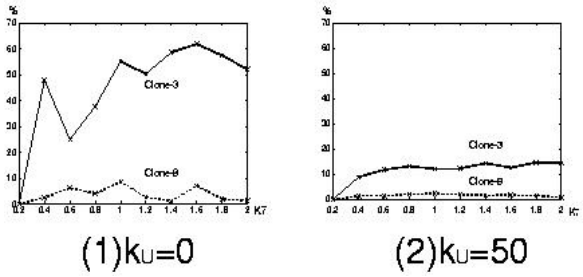


Fig. 7 K7 dependence of the magnitude of fluctuation s in clones 2 and 8 relative to that in clone 0

As for the actual reaction of immunity, it is very important that antibody molecules cross-link for phagocyte prey antigens. In real immune systems as well, it is feasible that a concentration threshold exists in some way.

Next, I will explain the relation between the generation process and the structure of the network.

What kind of difference is there in the structure of the network through the different formation processes? How does the network divide into sub-networks? I want to know what kind of difference occurs in the structure of the network through the different formation processes.

In the network in which we introduced a threshold, we can measure the time over the threshold. These times are characteristic depending on the condition of each clone. For example, when we investigate the running average of this time, if the clone is L-state, the time T_L is about 45, if the clone is S-state, the time T_S is about 22. If the clone has a limit

cycle, this time is about constant. So, I investigate the running average of the time over the threshold.

First, I define the values of connectivity keeping the characters of a small network. When I connect ten 3-clone systems loosely, each clone wanders about around the S-state and the L-state. At that time, the group that has many connections is the motive power in the network. I also found that responses to the invasion of antigens to each unit have characteristics in common with the small network case. The sub-network structure and the threshold are also important for preventing the spread of a local fluctuation throughout the whole network. I think that if a large network can be divided into small sub-networks, the study of the small networks is important. For this reason, I study what type of mechanism I can introduce into this model to make a sub-network structure.

First, I define the values of connection at random. If the number of nodes in the network is small enough, there are some cases where networks have sub-networks. However, when the network size becomes large enough, there are no sub-networks.

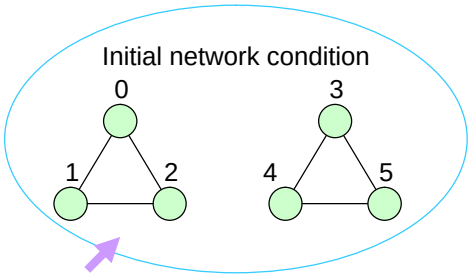
So, I introduce a newer phenomenon. This is the phenomenon of affinity maturation (ref. 9). Because of the substitution of amino-acid due to mutation, the affinity of the antibody changes. B cells cause mutation frequently in the process of the immune response. There is a possibility that the new antibody has more or less sensitivity than the parent antibody. The state of activation of these B-cells changes corresponding to the concentration of the antigens. When the network has a high concentration of antigens, both sensitive and insensitive antigens are activated. Also, when the network has a low concentration of antigens, only sensitive antibodies are activated. Therefore, as the immunity reaction proceeds, the affinity of the antigens is sensitive. I imitate this phenomenon, and change the value of the connectivity corresponding to the amount of time it is over the threshold.

I define values of connectivity constantly at random in a 30-clone system, and change them in accordance with a condition. Here, I change the values of connectivity depending on the number of times over the threshold. Then, some limit cycles appear, but there are no sub-networks.

Next, I will introduce the mutation mechanism in the model.

B-cells cause mutation frequently in the process of the immune response. So we introduce meta-dynamics - not the whole network generated at once, but the growth of the network by mutation (Fig. 5). A clone exceeding a threshold affects the mutation of other B-cells. However, the new kind of clone generated due to the mutation does not always have a good response to the antibodies that led to that mutation. The initial condition of the network is two different 3-clone systems. Here, 1, 2, 3, 4 or 5 clones become new members of the network in every unit time. The occurrence probability of how many clones come to the

network is 20% each. Various network structures are seen depending on the generation rules.

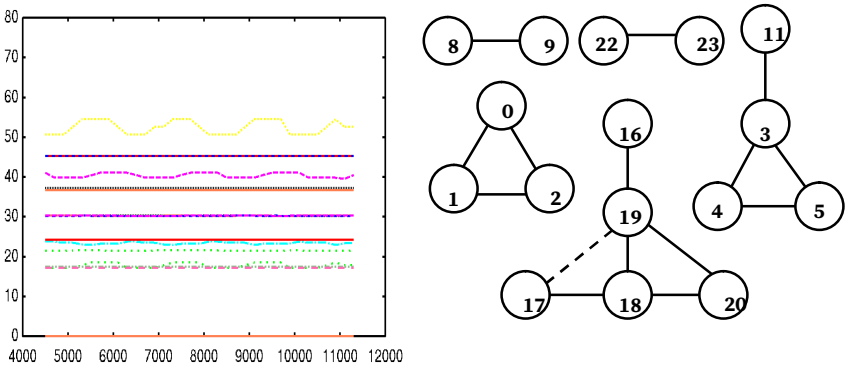


1, 2, 3 ,4 or 5 clones become new members of the network in every unit time. (Occurrence probability is 20% each.)

Fig. 8　Initial network condition and mechanism of mutation model

As one case, I define the value of connectivity between the new members generated due to mutation and the existing member, which caused the mutation. These values are decided at random in accordance with the following ratios: the proportion of 0.0 is 20%, 0.02 is 20% and 1.0 is 50%. The values of connectivity between new members are decided at random in accordance with the following ratios: the proportion of 0.0 is 30%, 0.02 is 30% and 1.0 is 40%

.



(a)　　(b)

Fig. 9　One case of a network configuration of a mutation model

When a network grows gradually by a mechanism that imitates mutation, we can show that some sub-networks are generated. Possibly, the mutation mechanism may be the cause of the sub-network structure being made.

§5. Summary

First, at least in a small network, it is found that the idiotype network model proposed by F. J. Varela can appropriately work for external perturbation.

Second, in the case of some clones connected at random, if the number of clones in the network is large, we cannot see any sub-networks, because the system behavior is chaotic. In the case of some clones connected at random, if the values of connectivity are changed in accordance with some assumptions, some limit cycle states may appear.

Fourth, a network has various structures which are dependent on the formation process.

Also possibly, a mutation-like generation mechanism creates the sub-network structure.

In this study, some characteristics of the immune network model have been revealed. In the future I want to examine applications to other fields.

References

1) N. K. Jerne, Ann. Immunol. (Inst. Pasteur) 125C (1974), 373.
2) F. J. Varela, A. Countinho, B. Dupire and N. M. Vaz, Theor. Immunol. Vol. II ed. A. S. Perleson, P.359 (Addison-Wesley, New York, 1988).
3) R. J. Bagley, J.D. Farmer, S. A. Kauffman, N. H. Packard, A. S. Perleson and I. M. Stadnyk, BioSystems 23 (1989), 113.
4) G. Parisi, Proc. Natl. Acad. Sci. USA 87 (1990) 429.
5) R. J. De Boer, A. S. Perelson and I.G. Kevrekidis, Bull. Math. Biol. 55 (1993), 745.
6) S. Itaya and T. Uezu, Prog. Theor. Phys. Vol. 104, No. 5 (2000), 903.
7) T. Uezu, S. Itaya and A. C. C. Coolen, Proceeding of RIMS (2001).
8) S. Itaya and T. Uezu, Prog. Theor. Phys. Vol. 107, No. 4 (2002), 827.
9) D. M. Weir and J. Stewart, Immunology 8/e (1999).

Coherence of the Double Negation in Linear Logic *

Masahito Hasegawa （長谷川 真人）
RIMS, Kyoto University （京都大学数理解析研究所）
PRESTO, JST （科学技術振興事業団 さきがけ研究２１）

Many formulations of proof nets and sequent calculi for *Classical Linear Logic (CLL)* [7, 8] take it for granted that a type A is *identical* to its double negation $A^{\perp\perp}$. On the other hand, since Seely [13], it has been assumed that $*$*-autonomous categories* [1, 2] are the appropriate semantic models of (the multiplicative fragment of) CLL. However, in general, in a $*$-autonomous category an object A is only *canonically isomorphic* to its double involution A^{**}. For instance, in the category of finite dimensional vector spaces and linear maps, a vector space V is only isomorphic to its double dual V^{**}. This raises the questions whether $*$-autonomous categories do not, after all, provide an accurate semantic model for these proof nets and whether there could be semantically non-identical proofs (or morphisms), which must be identified in any system which assumes a type is identical to its double negation. Whether this can happen is not completely obvious even when one examines purely syntactic descriptions of proofs with the isomorphism between A and $A^{\perp\perp}$ present such as [11, 9] or the alternative proof net systems of [4] which are faithful to the categorical semantics.

Fortunately, there is no such semantic gap: in this talk we provide a *coherence theorem* on the double involution on $*$-autonomous categories, which tells us that there is no difference between the up-to-identity approach and the up-to-isomorphism approach, as far as this double-negation problem is concerned.

> **Theorem.** *Any free $*$-autonomous category is strictly equivalent to a free $*$-autonomous category in which the double-involution $(-)^{**}$ is the identity functor and the canonical isomorphism $A \simeq A^{**}$ is an identity arrow for all A.*

This remains true under the presence of linear exponential comonads and finite products (the semantic counterpart of exponentials and additives respectively). Our proof is fairly short and simple, and we suspect that this

*Joint work with J.R.B. Cockett and R.A.G. Seely [5]

is folklore among specialists (at least everyone would expect such a result), though we are not aware of an explicit treatment of this issue in the literature.

This result should be compared with the classical coherence theorem for monoidal categories, as found e.g. in [12, 10]. In fact, we follow the proof strategy by Joyal and Street in [10]. We first show a weaker form of coherence theorem which turns a $*$-autonomous category into an equivalent one with "strict involution" (where A^{**} is identical to A), for which we make use of (a simplified version of) a construction of Cockett and Seely [6]. We then strengthen it to a form of "all diagrams commute" result by some additional arguments on the structure-preserving functors. In this way, this work also demonstrates the applicability of the Joyal-Street argument (which actually can be seen an instance of a general flexibility result on free algebras of 2-monads developped by Blackwell, Kelly and Power [3]) to other sorts of coherence problems.

References

[1] Barr, M. (1979) $*$-*Autonomous Categories*. Springer Lecture Notes in Math. **752**.

[2] Barr, M. (1991) $*$-autonomous categories and linear logic. *Math. Struct. Comp. Sci.* **1**, 159–178.

[3] Blackwell, R., Kelly, G.M. and Power, A.J. (1989) Two-dimensional monad theory. *J. Pure All. Algebra* **59**, 1–41.

[4] Blute, R.F., Cockett, J.R.B., Seely, R.A.G. and Trimble, T.H. (1996) Natural deduction and coherence for weakly distributive categories. *J. Pure Appl. Algebra* **113**(3), 229–296.

[5] Cockett, J.R.B., Hasegawa, M. and Seely, R.A.G. (2003) Coherence of the double involution on $*$-autonomous categories. Manuscript.

[6] Cockett, J.R.B. and Seely, R.A.G. (1999) Linearly distributive functors. *J. Pure Appl. Algebra* **143**, 155–203.

[7] Girard, J.-Y. (1987) Linear logic. *Theoret. Comp. Sci.* **50**, 1–102.

[8] Girard, J.-Y. (1995) Linear logic: its syntax and semantics. In *Advances in Linear Logic*, London Mathematical Society Lecture Note Series **222**, pp. 1–42.

[9] Hasegawa, M. (2002) Classical linear logic of implications. In *Proc. Computer Science Logic (CSL'02)*, Springer Lecture Notes in Comp. Sci. **2471**, pp. 458–472.

[10] Joyal, A. and Street, R. (1993) Braided tensor categories. *Adv. Math.* **102**, 20–78.

[11] Koh, T.W. and Ong, C.-H.L. (1999) Explicit substitution internal languages for autonomous and $*$-autonomous categories. In *Proc. Category Theory and Computer Science (CTCS'99)*, Electron. Notes Theor. Comput. Sci. **29**.

[12] Mac Lane, S. (1971) *Categories for the Working Mathematician.* Graduate Texts in Mathematics **5**, Springer-Verlag.

[13] Seely, R.A.G. (1989) Linear logic, $*$-autonomous categories and cofree coalgebras. In *Categories in Computer Science*, AMS Contemporary Mathematics **92**, pp. 371–389.

一般設計学と抽象設計論に関する考察

(A Note on General Design Theory and Abstract Design Theory)

菊池 誠 (Makoto Kikuchi)

神戸大学工学部情報知能工学科

(Department of Computer and Systems Engineering, Kobe University)

mkikuchi@kobe-u.ac.jp

1 はじめに

論理学とは推論の科学である．数学とは最も典型的な，ある種の形式を持つ推論の体系であり，19 世紀末から 20 世紀初頭にかけての数学の基礎についての論争を通して，真理，定理，証明といった推論と関わる概念が数学的ないし哲学的な観点から考察された．述語論理はその結果得られた形式的な推論の体系であり，そこでは知識や情報は論理式という記号の有限列によって，推論は論理式の有限列によって表現される．ここで推論とは演繹的推論，すなわち正しい命題から新たな正しい命題を導く関数とされ，推論は推論規則と呼ばれる基本的な推論の組み合わせによって得られるとされた．このとき証明とは公理もしくは仮定から始まる推論規則の有限列のことに他ならず，定理が証明の最後に現れる命題として形式化されることで，真な命題と定理の区別が与えられた．

設計とは推論が重要な役割を果たす人間の典型的な活動の一つである．もしも道具を作るときには設計があるとするならば，設計は人間の存在と同等の長さの歴史を持つであろうが，設計それ自身が科学的ないし哲学的な考察の対象として見られるようになったのは極めて最近のことである．1969 年に Simon [18] が人工物に関する科学を提唱してから，現在までに幾つもの設計や人工物に対する理論が相次いで提案されてきた．1977 年には設計の方法論への体系的な取り組みとして Paul and Beitz [16] が現れ，1979 年に吉川弘之 [31, 34] は設計についての公理的理論として一般設計学を提案した．より最近では，1990 年の Suh [21] による設計の原理原則についての議論や，1991 年の柳生孝昭 [29] による論理学ないし論理プログラミングを背景に持つ設計論，Braha and Maimon [1] の集合論に基づく設

計論がある．こうした流れの背景には 20 世紀の科学および工学の成熟と，その結果としての無視できぬ困難さの出現と共に，特に Simon の議論に顕著に見られるように，論理学に基礎を持つ計算機科学の発展に基づく人工知能への試みがあることは明らかであろう．

柳生 [29] は設計論が設計対象のモデル化と設計過程のモデル化から成ることを示唆する．この二種類のモデル化の区別は，述語論理における論理式による知識や情報の表現と推論の形式化の区別と対応し，種々の設計論の位置付けの理解に役に立つ．設計対象のモデル化は，広範な領域に対象を持つ工学設計においていかに統一的な枠組みを設けるかという問題であり，容易な作業ではない．そうした広範な領域に対処するために設計対象のモデル化は抽象的ないし数学的方法に拠ることになり，半ば必然的に設計論は数学的な装いを持つ．例えば，柳生 [29] は論理学を設計対象のモデル化の道具に用い，Braha and Maimon [1] は集合論を用いた．一方，設計過程とは設計における推論そのもののことであろうが，そのモデル化は容易ではない．たとえ設計対象の数学的な記述が得られても，それが自身は真偽のような概念との対応を持たない以上，述語論理における推論の形式化と同じように設計対象のモデルを並べるだけでは設計過程がモデル化されたとは考えにくく，述語論理の考え方をそのまま設計論に応用することはできない．そもそも発見的推論が重要な役割を持つ設計と，演繹的推論に特化した述語論理は折り合いが悪く，たとえ論理学が推論の科学であるとしても，未だ設計は論理学の対象とは成り得ていない．

吉川弘之の一般設計学は人工物のモデル化それ自身には全く興味を示していないという点で，他の設計論とは大きく性格が異なる．一般設計学では設計に現れる概念を実体概念と抽象概念に分類し，その概念自身の構造を議論するのではなく，そうした概念間の関係を公理として記述することで設計を論じる．この意味で一般設計学は，その原典 [31] が副題として「一般設計学のための公理的方法」と掲げるように，設計についての公理的な理論を目指すものであり，その公理の記述には位相空間論が用いられた．この一般設計学が目指したことは工学の領域に依存しない一般的な設計学の構築にあると言えようが，これは具体的な言語に依存しない，一般的な言語の性質を明らかにしようとしたソシュールの一般言語学の影響があろう．こうした一般設計学の考え方は工学の外からも関心を持たれることが少なくない．

さて，一般設計学は位相空間論を用いるなど，他の設計論と比較しても数学的な色合いが強いが，一般設計学それ自身が数学として展開されている訳ではなく，[28, 11] での分析にも見られるように，数学的には曖昧な主張も含まれる．角田譲 [4, 5, 8, 9] によって提唱された抽象設計論は，情報の流れについての数学的理論で

あるチャンネル理論 [2] に基づき，まず，一般設計学の数学的な基礎付けを目指すものであった．この抽象設計論は，数学的な設計論なのではなく，数学としての設計論であることに大きな特徴を持つが，設計を意識と実世界の間の情報の流れと見る新しい設計論の提示でもあり，特に角田譲による最近の展開はその方向を明確に打ち出している [6, 7]．また，三木大史 [14, 15] による抽象設計論に基づく規範的教育論の形式化など，抽象設計論の応用の試みも始まっている．この小論では一般設計学と初期の段階の抽象設計論を比較しながら，こうした設計論が何を，どのように論じようとしているのか考察したい．

2　一般設計学の公理

一般設計学は設計についての公理的理論である．一般に公理的理論は，理論が対象とする基本的な概念の決定と，その語彙に関する基本的な命題を公理として公理の選定から始まり，その理論の基本的な考え方は定義と公理の選び方に反映される．こうした形の理論が成り立つとすることは，概念と命題に関する還元主義的な立場に立つことでもある．

一般設計学では基本的な概念として「実体」,「実体概念」,「属性概念」が選ばれており，それが何であるかは以下の定義によって与えられる．

> 定義 1．実体集合とはすべての実体を元とするような集合である．すべての実体とは，存在するもの，存在したもの，存在するであろうものを含む．これを集合 S' とする．
>
> 定義 4．実体概念とは，人間が実体を体験することによって成立させた概念である．これはその実体や属性や機能などのように，抽象化の結果得られる抽象概念とはまったく独立である．しかし，抽象概念はこの実体概念から発生する．
>
> 定義 5．抽象概念とは，人間が意味ないし価値に導かれて実体概念を分類したときに，その各類に関する概念を言う．

哲学的な立場からも，工学的な立場からもこの定義には議論が多い．最も議論が多いのが定義 1 であろう．果たしてそのような集合 S' が存在するのか，そのような集合の存在を認めることは設計を論じる上で余りにも強い仮定ではないか．数学的な立場からは，こうした語彙に対応する集合が与えられれば，問題とすべき事柄はその集合にどのような性質を仮定するのかであって，とりあえず今の段階でこうした問題は保留しても構わない．

定理 2，定理 3 では属性と機能という語彙が導入され，それに応じて抽象概念はさらに属性概念，形態概念，機能概念に分類される．

> 定義 6．属性概念とは抽象概念の一つであるが，実在するものの属性は，人間が認識可能であり，従って属性を知ることによって実体を想定することができる．これは特に価値という側面は陽に意識はされない．むしろ意図としては没価値的にあるいはできるだけ客観的に実体を分類しようとするときに採用される類で，自然科学の態度に近いものである．

> 定義 7．形態概念とは，属性概念の一つであるが，属性のうち特に形態に注目すると形態概念が成立する．

> 定義 8．機能概念とは，抽象概念の一つであるが，特に実体の持つ機能的価値に注目するときに成立するものである．

機能や属性，形態がいったい何なのかは哲学的な観点からはやはり議論が尽きないが，ここでは三種類の概念の区別が与えられれば十分である．

一般設計学では知識を理想的知識と現実的知識とに分類し，理想的知識のもとでの実体，実体概念，抽象概念に関する公理として以下の三つの命題を導入する．これが一般設計学の公理であり，設計はこの公理のもとで議論される．

> 公理 1（認識公理）．実体は属性（あるいは機能，形態などの抽象概念）によって認識あるいは記述することが可能である．

> 公理 2（存在物と概念との対応関係）．実体集合と（理想的な）実体概念集合とは１対１に対応する．

> 公理 3（概念に関する位相公理，または概念の操作公理）．抽象概念集合は実体概念集合の位相である．

公理 1 は一般設計学の立場表明のようなものであって，実際には後の議論では公理として用いられることはない．哲学的にはこれもまた議論が多い．公理 2 によって定義 1 で導入された実体集合 S' は早々と役割を終える．この公理の後に話題になるのは実体概念であって，実体ではない．

数学的な意味で公理として実際に機能するのは公理 3 である．この公理には明記されていないが，この公理の意味することの一つは，一つの抽象概念は実体概念集合上の開集合となることである．数学的な観点からはここで多くの疑問が生じる．開集合の特徴は抽象概念のどのような性質に対応するのか．開集合における和集合と積集合に関する有限性の条件の違いは抽象概念のどのような性質に根

拠を持つのか．抽象概念の補集合は必ずしも抽象概念とはならないのか．おそらく，公理 3 が意図したことは（直接的にであれ，間接的にであれ）抽象概念が実体概念集合上に位相を与えるということ，すなわち，実体概念間の近さないし類似性を与えるという見方であり，その文面そのものが数学的に意味することではなかろう．しかし数学的には，この曖昧さが後の議論の曖昧さに繋がる．

設計論の多くでは，設計とは与えられた設計仕様に対して設計解を与えることとされている．一般設計学では設計仕様は機能概念の集合によって，設計解は属性概念の集合によって表現されると仮定され，設計は次のように定義される．

> 定義 12．設計とは，抽象概念空間上に示された領域に対応する属性（概念）空間上の領域を指定することである．

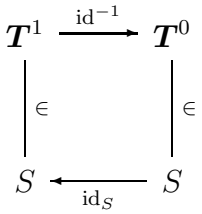
この設計の定式化には機械設計を考えることの影響が見られ，例えばソフトウェアを考える際には，この定式化によって設計を考えることには色々な疑問がありえる．また，設計対象は仕様と実現の組みによって表現されるとする柳生孝昭の提唱にも見られるように，そもそも機能や属性と実体の存在を独立に仮定し，その関係によって設計を考えることへの問題提起もある．

3　一般設計学の逆理

一般設計学の定義と公理の選び方の背景の一つには，明示的にであれ，非明示的にであれ，いかにして『ならば』を形式化するかという問題がある．言うまでもなく『ならば』の理解は哲学においても論理学においても古典的で重要な，難しい問題である．設計においても，例えば機械の設計であれば設計仕様と設計解は物理的な因果に基づく『ならば』によって結び付けられ,『ならば』の解釈の問題は避けて通ることができない．工学においては物理的な因果であったり，計算の過程であったりと分野ごとに『ならば』の背景は大きく異なり，このことが設計の一般論の構築を難しくする．一般設計学の定義や公理は，この『ならば』を形式的に，集合論の部分集合によって解釈するために選ばれていると考えられ，この『ならば』に対する態度が，場合によっては位相空間論を用いていること以上に，一般設計学を強く特徴付けることになる．

一般設計学の枠組みは，形式的には次のように要約できる．まず，１対１対応のつく二つの集合として実体集合 S'，実体概念集合 S が用意され，次に，抽象概念集合 $\boldsymbol{T}$，属性概念集合 $\boldsymbol{T}^0$，機能概念集合 $\boldsymbol{T}^1$ が，いずれも S の部分集合の族（すなわち S の冪集合 $\mathcal{P}(S)$ の部分集合）として導入される．$\langle S, \boldsymbol{T}\rangle, \langle S, \boldsymbol{T}^0\rangle, \langle S, \boldsymbol{T}^1\rangle$ は位相空間であると仮定され，それぞれ抽象概念空間，属性概念空間，機能概念

空間と呼ばれる．以上の定義のもと，一般設計学では設計仕様は機能概念の集合 $\boldsymbol{F} \subseteq \boldsymbol{T}$ によって，設計解は属性概念の集合 $\boldsymbol{A} \subseteq \boldsymbol{T}^0$ によって与えられるとされ，設計とは与えられた $\boldsymbol{F} \subseteq \boldsymbol{T}$ に対して $\cap\boldsymbol{A} \subseteq \cap\boldsymbol{F}$ を満たす $\boldsymbol{A} \subseteq \boldsymbol{T}^0$ を求めることとして定式化される．



この一般設計学の枠組みは，設計を上のような四角形の図式で議論することと，$\boldsymbol{T}^0, \boldsymbol{T}^1$ を与えられた集合 S の部分集合の族として定式化するという二つの特徴を持つ．後者の特徴を冨山と吉川は [24, 25] で抽象概念の「外延的」表現と呼ぶが，この外延的表現が選ばれたことの理由は，[24, 25] での議論からは，本来「ある実体が $\boldsymbol{A}$ に属する属性を全て持つ『ならば』，その実体は $\boldsymbol{F}$ に属する機能を全て持つ」と書かれるべき $\boldsymbol{A}$ と $\boldsymbol{F}$ の関係において，『ならば』という概念の分析を回避し，$\subseteq$ という集合間の二項関係で置き換えようとしたことにあると考えられる．S が「過去に存在したもの，現在存在するもの，未来に存在しうるもの全て」を含む実体集合と 1 対 1 対応を持つという，一般設計学の定義と対応公理から導かれる強い条件は，この『ならば』と $\subseteq$ の同値性の保証のために必要な仮定とみると分かりやすい．

こうした選択の結果として，一般設計学では定理として「理想的知識での設計は仕様を記述したときに完了する」，すなわち「理想的知識では設計は不要である」と語られることになる．同時に別の定理によって「現実的知識では，機能概念集合系の元と属性概念集合系の元との間に，実体概念を媒介としない何らかの直接的な対応が見い出されるとき，またそのときに限り設計が可能である」と主張されるが，この主張は「現実的知識では一般設計学の公理は放棄せざるを得ない」と読める．理想的知識のもとでは設計は不要であり，現実的知識のものでは公理を放棄しなければならないのであれば，これは設計論としては逆理的な状況であり，この状況を [12] では「一般設計学の逆理」と呼んだ．

現実的知識のもとでの公理を回復するために，[23, 24, 25] では現実的な知識として S の部分集合 $\tilde{S}$ を導入することで一般設計学の拡張が試みられている．すなわち，一般設計学の公理は S の上では成立しないが，$\tilde{S}$ の上では成立するという．しかし，この拡張では一般設計学の逆理は完全には解消できない．設計解が $\tilde{S}$ の中に存在する場合には S の場合と同様，設計は仕様を記述したときに完了し，設計は不要となる．一方，設計解が $\tilde{S}$ の中には存在しない場合には，たとえ $\tilde{S}$ の上で一般設計学の公理が成立したとしても，そのことと設計解の間には直接の関係

はなく，一般設計学の公理が本来の役割を果たすことは難しい．

4　チャンネル理論と抽象設計論

数学的な観点からは，こうした逆理的な現象に一般設計学の問題がある訳ではない．いくつかの主張や証明が数学的に曖昧なのが問題なのであり，これは一般設計学の中での位相空間論の役割の曖昧さにも繋がる．角田譲によって提唱された抽象設計論は，まず，位相空間論よりも一般的な数学的な枠組みであるチャンネル理論を用いて一般設計学の基本的な考え方を数学的に厳密に記述し，その後に，その枠組みと位相空間論が関係するかを論じることで，一般設計学の数学的基礎付けを試みるものであった．現在までに，この当初の目的は大方達成されていると考えられるが，後に見るように，抽象設計論によって結果として一般設計学の逆理的状況も解消されることとなった．

チャンネル理論は Dretske の情報の流れについての哲学的考察 [3] に基づいて Barwise と Seligman によって提唱された，情報の流れについての数学的理論である [2]．チャンネル理論の最も基本的な構造として，情報を表現するための数学的な構造である分類と，二つの分類を射である情報射が導入される．分類 $\boldsymbol{A}$ とはトークンと呼ばれるものの集合 $\mathrm{tok}(\boldsymbol{A})$，タイプと呼ばれるものの集合 $\mathrm{typ}(\boldsymbol{A})$，その間の二項関係 $\models_{\boldsymbol{A}}$ の三つ組みから成る．トークンは情報を担うものを，タイプとは担われる情報を意味する．$\boldsymbol{A}$ から $\boldsymbol{B}$ への情報射 $f : \boldsymbol{A} \rightleftarrows \boldsymbol{B}$ とは二つの関数 $f^{\wedge} : \mathrm{typ}(\boldsymbol{A}) \to \mathrm{typ}(\boldsymbol{B})$ と $f^{\vee} : \mathrm{tok}(\boldsymbol{B}) \to \mathrm{tok}(\boldsymbol{A})$ の組みであり，全ての $\alpha \in \mathrm{typ}(\boldsymbol{A})$ と $b \in \mathrm{tok}(\boldsymbol{B})$ に対して $f^{\vee}(b) \models_{\boldsymbol{A}} \alpha$ と $b \models_{\boldsymbol{B}} f^{\wedge}(\alpha)$ が同値となるものでる．

$$\begin{array}{ccc} \mathrm{typ}(\boldsymbol{A}) & \xrightarrow{f^{\wedge}} & \mathrm{typ}(\boldsymbol{B}) \\ \Big| \models_{\boldsymbol{A}} & & \Big| \models_{\boldsymbol{B}} \\ \mathrm{tok}(\boldsymbol{A}) & \xleftarrow[f^{\vee}]{} & \mathrm{tok}(\boldsymbol{B}) \end{array} \tag{1}$$

二つの情報射 $f : \boldsymbol{A} \rightleftarrows \boldsymbol{C}$，$g : \boldsymbol{B} \rightleftarrows \boldsymbol{C}$ の集合は（二項）チャンネルと呼ばれ，チャンネルを介して $\boldsymbol{A}$ から $\boldsymbol{B}$ へと情報は流れる．

位相空間はその台集合の元をトークン，開集合をタイプ，$\in$ をその間の二項関係とみることで分類の一種と考えられる．このとき，位相空間の間の連続写像 f は，f と f^{-1} の組みによって，この二つの位相空間の間の情報射を与える．この意味で分類と情報射は位相空間と連続写像の一般化である．抽象設計論とは機能概念空間と属性概念空間を分類として，その間の関係を情報射として定式化するものである．情報射によって結ばれるこの二つの分類という図式を角田譲は機能

スキームと呼んだが，この図式は一般設計学において恒等写像によって結ばれる二つの位相空間の一般化と言える．ここで，機能スキームにおける属性概念空間は一般設計学と同様のものであり，そのトークンは（実体と同一視された）実体概念と考えてよいが，機能概念空間においては，タイプ集合上にタイプ間の導出関係を意味する「理論」が与えられていることが仮定され，機能概念空間のトークンは，この理論に対して無矛盾で完全なタイプの集合のこととされた．この条件は一般設計学にはない抽象設計論独自のものであり，一般設計学と抽象設計論の違いを論じるときに重要である．

機能スキームによって一般設計学の基本的枠組みはチャンネル理論の上に再構築された．角田譲は分類から生成される位相空間の定義を与えているが [4, 5]，それによって一般設計学上の位相空間に関係する議論，例えば「設計仕様とは機能概念空間上のフィルターである」という主張などの機能スキームの上での解釈が可能になっている [9]．

角田譲はチャンネルの考え方をもとに，媒体空間という機能概念空間と属性概念空間を結ぶ空間の存在を仮定したが，これは一般設計学におけるメタモデルの考え方と対応を持つ．メタモデルとは，一般設計学の初期においては，機能概念空間と属性概念空間を結ぶための中間的な構造とされていたが，後に，機能概念空間と属性概念空間を人工物に関する異なるモデル化の枠組みと考え，そうした異種のモデルを統括するための，より抽象度の高い人工物のモデル化のための枠組みと考えられるようになった [22, 13]．このメタモデルの考え方はチャンネルの考え方と極めて近い．メタモデルという概念の獲得は一般設計学の最大の成果の一つとされたが，抽象設計論においても，一般設計学の数学的基礎付けという問題から離れ，設計を情報の流れとみる数学的な設計論であろうとしたときに，チャンネルという概念はより重要になっている．

なお，現段階の抽象設計論 [6, 7] では，設計は機能と属性の対立という構図によってではなく，設計者の欲求と人工物の挙動の関係によって議論されている．そこでは，機能スキームにおける機能概念空間のタイプは機能から欲求に，属性概念空間のタイプを属性から挙動に置き換えられた．この見方は一般設計学の考え方よりも，設計対象と設計仕様と実現の組みで考える柳生孝昭の立場に近い．

5　実体概念の形式化について

そもそも抽象設計論は，一般設計学で選ばれた実体，機能，属性という基本的な用語とそれらが置かれる図式を保ちながら，一般設計学を数学的に基礎付ける試みから始まっている．従って，抽象設計論と一般設計学は基本的な枠組みを共有

しており，二つの理論の違いは数学的な表現方法にしかなく，思想的には抽象設計論は一般設計学に対して新しさは無いとの見方も可能であろう．特に初期の抽象設計論では，設計を機能概念と属性概念を結ぶ行為とするという一般設計学の基本的な考え方は，そのまま踏襲されている．しかし一般に，表現の違いは考え方の違いを反映し，必ずしも一般設計学の思想の全てが抽象設計論に引き継がれている訳ではない．[12] で論じたように，抽象設計論と一般設計学の最も大きな違いは，実体概念の捉え方にあると言えよう．

抽象設計論では，設計を機能概念空間と属性概念空間という二つの空間の関係という図式で議論することと，属性概念空間の定式化において S の存在を仮定することに関しては一般設計学の枠組みを踏襲されている．しかし，機能概念空間においては，$\boldsymbol{T}^1$ を与えられた集合とし，対応する実体概念集合を S ではなく,(単純化してしまえば）$\boldsymbol{T}^1$ の冪集合 $\mathcal{P}(\boldsymbol{T}^1)$ とされ，つまり冨山と吉川が「内包的」と呼ぶ表現が選ばれている．その結果，一般設計学で機能空間と属性空間を結んでいた id_S でこの二つの空間を繋ぐことはできず，二つの写像の組み f, g を考える必要が生じる．その結果として，抽象設計論では一般設計学の逆理は回避される．

$$\begin{array}{ccc} \boldsymbol{T}^1 & \xrightarrow{f} & \boldsymbol{T}^0 \\ \Big| \in & & \Big| \in \\ \mathcal{P}(\boldsymbol{T}^1) & \xleftarrow[g]{} & S \end{array}$$

一般設計学では一種類の実体概念集合しか用いられなかったが，抽象設計論では二種類の実体概念集合が用いられている．この点に二つの理論の最も大きな違いがあると言えよう．一般設計学では常に実体集合と実体概念集合の同一視が可能であるが，抽象設計論では属性概念空間においてのみその同一視は可能である．機能概念空間の実体概念は実体ではなく，あくまで実体概念であり，写像 g を実体から実体概念への抽象化の写像と考えることもできよう．

実体のタイプを e，真理値のタイプを t とする．実体の集合とは特性関数を考えることで $\langle e,t\rangle$ のタイプを持ち，実体の集合の集合は $\langle\langle e,t\rangle,t\rangle$ のタイプを持つ．抽象設計論では機能概念空間の実体概念は機能概念の集合として形式化されるが，このとき機能概念はタイプ $\langle e,t\rangle$ を持ち，従って機能概念空間の実体概念のタイプは $\langle\langle e,t\rangle,t\rangle$ となる．さて，モンタギュー文法では名詞句とは $\langle\langle e,t\rangle,t\rangle$ のタイプを持つものとされる．これは任意 $\forall$，存在 $\exists$ という量化子を一般化した概念である一般量化子と呼ばれるものでもある（[17, 10, 26, 19, 20]）．つまり，通常の述語がタイプ $\langle e,t\rangle$ を持つ関数（タイプ e をもつ対象を変数の値にとって真理値を返す関数）であるのに対し，量化子とは述語を変数の値にとって真理値を返すタイプ

$\langle\langle e,t\rangle,t\rangle$ の関数である．この意味において，抽象設計論の機能概念空間における実体概念は，言語学における名詞句（もしくは一般量化子）と対応すると考えられる．抽象設計論における属性概念空間の実体概念が実体のタイプ e を持つとするならば，実体から実体概念への抽象化の写像 g とは，e のタイプを持つ対象から $\langle\langle e,t\rangle,t\rangle$ のタイプを持つ対象への写像であり，このとき設計は言語的対象と実体とを結び付ける行為と考えられる．

一般設計学では設計仕様，設計解はいずれも抽象概念の集合として，すなわち実体概念（＝実体）の集合の集合として形式化された．このとき，一般設計学における設計仕様や設計解はいずれも名詞句であると考えられ，従って，一般設計学における設計とは，機能概念のみを用いて書かれた名詞句を，属性概念のみを用いて書かれた名詞句に変換する過程であると言うこともできよう．このとき設計とは実体に即した活動ではなく，より言語的な活動であると考えることができ，設計仕様も設計解も適当なタイプを持つ記号列であると考えることで，実体概念を参照することなく，直接，設計仕様と設計解の関係を論じることも可能になる．これは吉川弘之自身によって初期の段階でその可能性と必要性が既に指摘されていたことであり，実際，一般設計学はそうした方向で発展したとも考えられようが，そもそもの一般設計学では，属性概念空間と機能概念空間の実体概念集合にタイプ e を持つ元から成る共通の集合を割り当て，その二つの集合の間を恒等写像で結び付けており，その結果，この二つの空間の関係の多くは自明なものに陥ってしまった．その結果が一般設計学の逆理であったと言えよう．

実体概念とは実体，すなわち人工物のモデルである．名詞句とは実体を指し示す表現であり，これもまた実体のモデルである．設計仕様や設計解もまた，実体のモデルであると考えられ，結局，一般設計学でも初期の段階の抽象設計論でも，そうした人工物のモデル化のための枠組みを提示することはないが，多様な人工物のモデルの間の関係を論じることが主題であったと言えよう．タイプ e を持つと考えられる実体概念は実体に関する固有名詞であり，それに対して設計仕様や設計解は普通名詞と対応する．属性概念空間の実体概念に固有名詞を，機能概念空間の実体概念に普通名詞を割り当てたことに，一般設計学とは異なる抽象設計論の特徴がある．

6　おわりに

ここで論じたことは一般設計学と抽象設計論を対象とした分析であり，その意味で，理論に関する理論として，メタ設計論と呼ぶことができよう．この小論では最初に，発見的推論が設計において本質的であるとに触れたが，結局は最後までそ

のことについては何も論じることはなかった．もしかしたら設計に関する数学は，このまま発見の周囲を回りながら最後まで発見的推論そのものについては何も語られることはないのかも知れないが，もしも発見的推論について何らかの見識が得られれば，それは論理学にとっても新たな視点と成り得るものであろう．そのためには工学的応用や論理学との関係という視点に留まらず，幅広い問題意識と共に設計が論じられることが必要であろう．

なお，ここで論じた事柄は神戸大学の数理設計論グループや，2000 年より半年ごとに神戸にて開催してきた設計論と機能論に関する研究集会での議論，特に角田譲，柳生孝昭との議論に負うことが多い．また，この小論は基本的に長坂一郎との共同研究に基づき，一般設計学の公理の分析と実体概念と名詞句の関係についての議論は [12] において，柳生（[27, 29]）によってアリストテレス的世界観対仏教的世界観と名付けられた対比の構図の中で論じられている．柳生孝昭は [27, 29] や [30] で，この小論や [12] で論じた話題について大変に興味深い議論を展開しており，それは数学と設計論の関係そのものについても示唆に富む．

References

[1] Braha, D. and Maimon, O., A Mathematical Theory of Design: Foundations, Algorithms and Applications, Kluwer, 1998.

[2] Barwise, J. and Seligman, J., Information Flow: The Logic of Distributed Systems, Cambridge University Press, Cambridge, 1997.

[3] Dretske, F. Knowledge and the Flow of Information. *MIT Press*, 1981, reprinted by *CSLI Publications*, 1999.

[4] Kakuda, Y. A mathematical description of GDT: A marriage of Yoshikawa's GDT and Barwise-Seligman's theory of information flow, handout in Workshop on General Design Theory, Cambridge, UK, 1999.

[5] Kakuda, Y., A mathematical definition of synthetic emergence, In: *Proceedings of IWES'99*, 13–20, 1999.

[6] 角田譲「設計と情報の流れ」, 科学基礎論研究, Vol. 29, No. 1, 1–5, 2001.

[7] 角田譲「抽象設計論 –情報の流れの理論と設計–」, 数学基礎論シンポジウム 2002 講義録, 2002.

[8] Kakuda, Y. and Kikuchi, M., Abstract Design Theory, *Annals of the Japan Association for Philosophy of Science* 10 (3), 19–35, 2001.

[9] Kakuda, Y. and Kikuchi, M. Topology on Classifications in Abstract Design, In: *Proceedings of IWES'01*, 123–130, 2001.

[10] Keenan, E.L. and Westerstahl, D., Generalized Quantifiers in Linguistics and Logic In: *Handbook of Logic and Language*, van Benthem, J. and ter Mulen, A. (eds.), Elseviver, Amsterdam, 837–893, 1997.

[11] 菊池誠「吉川の一般設計学序説を読む」, 一般設計学勉強会資料, 神戸, 1999.

[12] Kikuchi, M. and Nagasaka, I., On the Three Axioms of General Design Theory, In: *Proceedings of IWES'02*, 69–76, 2002.

[13] 桐山孝司, 冨山哲男, 吉川弘之「設計対象モデル統合化のためのメタモデルの研究」, 人工知能学会誌, Vol. 6, No. 3, 426–434, 1991.

[14] 三木大史「抽象設計論による教育の数理的形式化」, 学位論文, 神戸大学, 2003.

[15] 三木大史「教育における抽象設計モデル」, 科学基礎論研究, 掲載予定.

[16] Paul, G. and Beitz, W., Engineering Design: A Systematic Approach (2nd ed.), Springer, 1996. （邦訳：ポール, バイツ (翻訳：設計工学研究グループ)「工学設計 – 体系的アプローチ」, 培風館, 1995.）

[17] Partee, B.H. and Hendriks, H.L.W. Montague Grammar, In: *Handbook of Logic and Language*, van Benthem, J. and ter Mulen, A. (eds.), Elseviver, Amsterdam, 5–91, 1997.

[18] Simon, H.A., The Sience of Artificial (3rd ed.), MIT Press, 1996. （邦訳：サイモン (翻訳：稲葉元吉, 吉原 英樹)「システムの科学（第 3 版)」, パーソナルメディア, 1999.）

[19] 白井賢一郎「形式意味論入門—言語・論理・認知の世界」, 産業図書, 1985.

[20] 白井賢一郎「 自然言語の意味論—モンタギューから「状況」への展開」, 産業図書, 1991.

[21] Suh, N.P., The Principles of Design, Oxford University Press, 1990. （邦訳：スー (翻訳：畑村 洋太郎)「設計の原理 – 創造的機械設計論」, 朝倉書店, 1992.）

[22] Tomiyama, T., Kiriyama, T. Takeda, H., Xue, D. and Yoshikawa, H., Meta-model: A Key to Intelligent CAD Systems, Re- search in Engineering Design, Vol. 1, No. 1, 19–34, 1990.

[23] 冨山哲男, 吉川弘之「一般設計学の展開（第１報）－概念空間のコンパクト化－」, 精密機械 51 (4), 809–815, 1985.

[24] 冨山哲男, 吉川弘之「一般設計学の展開（第２報）－概念空間の距離付けとＣＡＤへの応用－」精密工学会誌 52 (8), 1406–1411, 1986.

[25] Tomiyama, T. and Yoshikawa, H. Extended General Design Theory, In: *Design Theory for CAD*, Yoshikawa, H. and Warman, E.A. (eds.), 95–130, North-Holland, Amsterdam, 1987.

[26] van Benthem, J., Language in Action: Categories, Lambdas and Dynamic Logic, North-Holland, Amsterdam, 1991.

[27] Yagiu, T., Modeling Design Objects and Processes, Springer-Verlag, 1991.

[28] Yagiu, T., Problems with the General Design Theory and the Proposal for an Alternate Theory, handout in Workshop on General Design Theory, Cambridge, UK, 1997.

[29] Yagiu, T., Modeling Design Objects and Processes, handout in 3rd. Workshop of Design and Function, Kobe, Japan, 2001.

[30] 柳生孝昭「近代科学・技術における抽象化の原理と功罪」, 総合知学会誌, 知的文明研究会報告, 2002.

[31] 吉川弘之「一般設計学序説」, 精密機械 45 (8) 20–26, 1979.

[32] 吉川弘之「一般設計過程」, 精密機械 47 (4) 19–24, 1981.

[33] Yoshikawa, H. General Design Theory and a CAD system, In: *Man-Machine Communication in CAD/CAM*, Sata, T. and Warman, E. (eds.), 35–58, North-Holland, Amsterdam, 1981.

[34] 吉川弘之, 冨山哲男「設計学 –ものづくりの理論–」, 放送大学教材, 2000.

Exponential Free Typed Böhm Theorem

Satoshi Matsuoka （松岡聡）
National Institute of Advanced Industrial Science and Technology
（産業技術総合研究所）

In [DP01], Dosen and Petric called Statman's "Typical Ambiguity theorem" [Sta83] *typed Böhm theorem*. Moreover, they gave a new proof of the theorem based on set-theoretical models of typed lambda calculus. In this paper, we give the linear version of the typed Böhm theorem: given two closed intuitionistic implicational proof nets that have different cut-free η-long normal forms, we can have a context that separates the two proof nets in a clear way.
Such a context is the composition of the following contexts and type instantiations:

1. contexts that decrease the orders of proof nets
2. type instantiation operators
   In these operators additive connectives may occur.
3. choice contexts
   These contexts can pick up obviously different proof nets by exploiting the information of given different two proof nets with an order less than fourth order.
4. The final type instantiation and context
   You can choose any type and any two proof nets with the type.

References

[DP01] Kosta Dosen and Zoran Petric.
The Typed Bohm Theorem, *Electronic Notes in Theoretical Computer Science*, vol. 50, no. 2, Elsevier Science Publishers, 2001.

[Mat03] Satoshi Matsuoka. Exponential-Free Typed Böhm Theorem. Full version available from `http://staff.aist.go.jp/s-matsuoka/`, 2003.

[Sta83] R. Statman. λ-definable functionals and βη-conversion. *Arch. math. Logik*, 23:21-26. 1983.

Embedding into Wreath Product and the Yoneda Lemma

Ryu Hasegawa (長谷川立)

Graduate School of Mathematical Sciences, The University of Tokyo, Komaba 3-8-1, Meguro-ku, Tokyo 153-8914, Japan
(東京大学大学院数理科学研究科, 〒 153-8914 東京都目黒区駒場 3-8-1)

Abstract
The Kaloujnine-Krasner theorem is an immediate consequence of an easy theorem for the 2-category of groupoids and a 2-categorical variation of the Yoneda lemma.

1 Introduction

We give a new proof of the Kaloujnine-Krasner theorem in group theory [6]. This theorem asserts that every group extension embeds into a wreath product. Indeed we give a proof of the generalized theorem for twisted wreath product by Neumann [4].

The purpose of this note is to show that the pure group-theoretic Kaloujnine-Krasner theorem naturally arises as a consequence of a general fact for groupoids and a fundamental tool of general category theory, the Yoneda lemma. In particular, the use of the Yoneda lemma elucidates the reason wreath product naturally arises.

Given two groups N and G, an extension of N by G is a group F satisfying the short exact sequence

$$1 \longrightarrow N \longrightarrow F \longrightarrow G \longrightarrow 1.$$

In other words, an extension F is a group having N as a normal subgroup and satisfying $F/N \cong G$. The Kaloujnine-Krasner theorem asserts that every extension of N by G embeds into the standard wreath product $N \operatorname{Wr} G$.

The standard wreath product is defined as semidirect product $N^G \rtimes G$ where N^G is simply cartesian product of m copies of N where m is the order of G. An element of $N \operatorname{Wr} G$ is a pair of an element g and an m-tuple $\langle n_h \rangle_{h \in G}$ of elements of N. This is best illustrated in the form of matrices. Regarding G acting on the set G of m elements with regular action, each element $g \in G$ yields an $m \times m$ permutation matrix. Then the pair $(g, \langle n_h \rangle)$ may be regarded as a matrix where the non-zero entry in each column of the permutation matrix is replaced with the corresponding element n_h.

Example: (i) Let S_2 be the symmetric group of all permutations over two letters, and let C_n be the cyclic group of order n. Direct product $C_n \times S_2$ is an extension of C_n by S_2, generated by two element x and y satisfying three relations $x^2 = 1$, $y^n = 1$ and $xyx = y$. The group $C_n \times S_2$ embeds into $C_n \operatorname{Wr} S_2$ as

$$x \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad y \mapsto \begin{pmatrix} y & 0 \\ 0 & y \end{pmatrix}$$

where, in the last matrix, we abuse y also as the generator of C_n. It is easy to see that these matrices fulfill the three relations above.

(ii) The dihedral group D_{2n} is defined as the semidirect product $C_n \rtimes S_2$. Namely it is generated by two elements x and y subject to relations $x^2 = 1$, $y^n = 1$, and $xyx = y^{-1}$. The group D_{2n} embeds into C_n Wr S_2 as

$$x \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad y \mapsto \begin{pmatrix} y & 0 \\ 0 & y^{-1} \end{pmatrix}$$

These matrices fulfill the defining three relations of the dihedral group.

We assume knowledge of category theory and bicategory theory. The standard literature is [3, 1]. With respect to the terminology for bicategories, we use pseudo-functors, quasi-natural transformation, and modifications. These are weak ones. For example, pseudo-functor preserves composition up to invertible 2-cells satisfying due coherence conditions.

2 Groupoids and the Yoneda lemma

A *groupoid* is a small category where all morphisms are invertible. Every group is regarded as a groupoid such that there is only one object: the elements of the group become the morphisms on the single object. Given a groupoid G, we write $G(x, y)$ the homset of all morphisms $x \xrightarrow{g} y$ in G.

A *groupoid homomorphism* is simply a functor between groupoids. A *natural isomorphism* (also called homotopy) between homomorphisms is defined as the usual natural transformation in category theory. Since all morphisms in groupoids are invertible, a natural transformation automatically turns out to be an isomorphism.

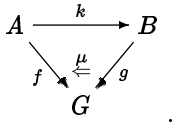
We let **Gpoid** denote the 2-category of groupoids, groupoid homomorphisms, and natural isomorphisms. Every 2-cell is invertible. Namely $\mathrm{Hom}(A, B)$ is a groupoid. In general, we call a 2-category where all 2-cells are invertible a *groupoid-enriched* category.

We will encounter several other groupoid-enriched categories in this paper:

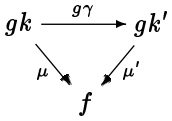
Example: Let G be a groupoid.

(i) The groupoid enriched-category $\mathbf{Gpoid}^G$ is defined. Its objects are all pseudo-functors from G into **Gpoid**. Its 1-cells are quasi-natural transformations, and its 2-cells are modifications.

(ii) The slice groupoid-enriched category $\mathbf{Gpoid}/G$ is defined. The objects are all pairs (A, f) of a groupoid A and a groupoid homomorphism $A \xrightarrow{f} G$. The 1-cells $(A, f) \xrightarrow{(k,\mu)} (B, g)$ are such that $A \xrightarrow{k} B$ is a groupoid homomorphism and $gk \overset{\mu}{\Rightarrow} f$ is a natural isomorphism:



The 2-cells $(k, \mu) \overset{\gamma}{\Rightarrow} (k', \mu')$ are such that γ is a natural isomorphism between k and k' rendering the diagram



of natural isomorphisms commutative.

The following definition plays an important role throughout the rest of the paper.

2.1 Definition
Let t be an object of $\mathbf{Gpoid}^G$ where G is a groupoid.

The *Grothendieck construction* $\int_{x \in G} t(x)$ is the groupoid defined as follows: Its objects are the pairs (x, a) of all objects $x \in G$ and all objects $a \in t(x)$; Its morphisms $(x, a) \xrightarrow{(g,\gamma)} (x', a')$ are such that $x \xrightarrow{g} x'$ is a morphism in G and $g \cdot a \xrightarrow{\gamma} a'$ a morphism in $t(x')$, where $g \cdot a$ denotes $t(g)(a)$.

Remark: The Grothendieck construction $\int_{x \in G} t(x)$ turns out to be a bicolimit of the pseudo-functor $G \xrightarrow{t}$ **Gpoid** for groupoid G.

The following lemma gives a simple computational rule for the Grothendieck construction. The first one is an analogy of representing double integral by iteration of ordinary integral. The second is of Fubini's theorem.

2.2 Lemma
(i) *Let A be a groupoid. Moreover, let $A \xrightarrow{t}$* **Gpoid** *and $\int_{x \in A} t(x) \xrightarrow{u}$* **Gpoid** *be pseudo-functors. Then equivalence of groupoids*

$$\int_{(x,a)\in\int_{x\in A} t(x)} u(x,a) \;\cong\; \int_{x\in A}\int_{a\in t(x)} u(x,a)$$

holds.

(ii) *Let A and B be groupoids, and let $A\times B \xrightarrow{u}$ **Gpoid** be a pseudo-functor. Then equivalence of groupoids*

$$\int_{x\in A}\int_{y\in B} u(x,y) \;\cong\; \int_{x\in B}\int_{y\in A} u(x,y)$$

holds.

(*Proof*) (i) is straightforward (or follows from a general fact for bicolimits). (ii) is derived from (i) by taking as t the constant functor to B and remarking $A\times B \cong \int_{x\in A} B \cong B\times A$. $\square$

The Yoneda lemma is one of the most fundamental machinery in ordinary category theory [3]. The following lemma is an extension of the lemma to groupoid-enriched categories (the lemma for general bicategories is found in [5]). In the statement, x/G denotes the slice category (indeed groupoid) where the objects are all pairs (z,f) of an object z and a morphism $x \xrightarrow{f} z$ in G, and the morphisms $(z,f)\xrightarrow{g}(z',f')$ is a morphism $z\xrightarrow{g} z'$ in G satisfying $gf=f'$. In a dual way, G/x is defined.

2.3 Lemma
*Let G be a groupoid and let $G \xrightarrow{t}$ **Gpoid** be a pseudo-functor.*

The following equivalences of groupoids hold:

$$\varprojlim_{(z,f)\in x/G} t(z) \;\cong\; t(x) \;\cong\; \varinjlim_{(z,f)\in G/x} t(z).$$

Moreover these equivalences are quasi-natural in x.

The leftmost groupoid in this lemma denotes a bilimit of the pseudo-functor t preceded by obvious projection $x/G \longrightarrow G$. Likewise the rightmost is a bicolimit. The proof of the lemma is parallel to the one for ordinary categories.

Remark: The lemma remains to hold for general groupoid-enriched category $\mathbf{C}$ and a pseudo-functor $\mathbf{C}\xrightarrow{t}$ **Gpoid**. In this paper, we use only the special form above.

We can write the equivalences of the lemma in terms of hom-groupoid and the Grothendieck construction:

$$\mathrm{Hom}_{\mathbf{Gpoid}^G}(G(x,\text{-}),\, t) \;\cong\; t(x) \;\cong\; \int_{z\in G} t(z)G(z,x).$$

In the rightmost, concatenation is cartesian product of groupoid $t(z)$ with set (i.e., discrete groupoid) $G(z,x)$.

The following simple fact is the first vehicle for verification of the Kaloujnine-Krasner theorem.

2.4 Theorem
Let G be a groupoid.

Then biequivalence $\mathbf{Gpoid}^G \cong \mathbf{Gpoid}/G$ between groupoid-enriched categories holds.

(*Proof*) A 2-functor $\mathbf{Gpoid}^G \xrightarrow{F} \mathbf{Gpoid}/G$ is simply the Grothendieck construction. Namely F carries each object $t \in \mathbf{Gpoid}^G$ to $\int_{z\in G} t(z)$ with an obvious projection to G. Also a 2-functor in the reverse direction $\mathbf{Gpoid}/G \xrightarrow{U} \mathbf{Gpoid}^G$ involves the construction. Given an object (A,p) of $\mathbf{Gpoid}/G$, the object $U(A,p)$ is defined as the strict (pseudo-)functor $\int_{x\in A} G(p(x),\text{-})$.

Let us prove $UF \cong id$ (an identity functor). We have

$$UF(t) \;\cong\; \int_{(z,a)\int_{z\in G} t(z)} G(z,\text{-}).$$

By Lem. 2.2, it is equivalent to $\int_{z\in G}\int_{a\in t(z)} G(z,\text{-})$, that is, $\int_{z\in G} t(z)G(z,\text{-})$. By the Yoneda lemma, the last is equivalent to t. Next we prove $FU \cong id$. We have

$$FU(A,p) = \int_{z\in G}\int_{x\in A} G(p(x),z),$$

which is equivalent to $\int_{x\in A}\int_{z\in G} G(p(x),z)$ by Fubini. The internal integral is equivalent to a trivial groupoid 1 by the Yoneda lemma, noticing $G(p(x),z) \cong G(z,p(x))$ (or by a simple direct argument). Hence $FU(A,p) \cong A$. $\square$

Remark: The construction $FU(A,p)$ in the proof above gives an fibration [2] (also called opfibration in the literature) equivalent to the original $A \xrightarrow{p} G$.

This theorem looks straightforward, but it conceals a deep consequence. Let us consider the special case where G is a group. Given another group N, we want to characterize all groups F satisfying the short exact sequence

$$1 \longrightarrow N \longrightarrow F \longrightarrow G \longrightarrow 1.$$

In other words, F is a group having (a copy of) N as a normal subgroup and satisfying $F/N \cong G$. By the theorem, we can associate an object of $\mathbf{Gpoid}^G$ given as $t = \int_{x \in F} G(px, \text{-})$ where $F \xrightarrow{p} G$ is the canonical surjection. For a unique object $z \in G$, the groupoid $t(z)$ is equivalent to the group N. Conversely, if $t(z)$ is a group, the associated Grothendieck construction $\int_{z \in G} t(z)$ is obviously a group endowed with a surjection onto G. Therefore the problem to characterize all group extensions F amounts to the problem to give all pseudo-functors $t \in \mathbf{Gpoid}^G$ such that $t(z) = N$.

This observation yields the traditional theory of Schreier's factor sets in group theory [6]. This theory tells us that all group extensions are obtained from the following data: a family of automorphisms $g \cdot (\text{-})$ on N for $g \in G$, and a family of elements $\tilde{\varphi}_{g,h}$ of N for $g, h \in G$, all these satisfying the equalities:

$$\begin{aligned} gh \cdot x &= \tilde{\varphi}_{g,h}(g \cdot h \cdot x)\tilde{\varphi}_{g,h}^{-1} \\ \tilde{\varphi}_{gh,k}\tilde{\varphi}_{g,h} &= \tilde{\varphi}_{g,hk}(g \cdot \tilde{\varphi}_{h,k}) \end{aligned}$$

where $g, h, k \in G$ and $x \in N$. The second equality is exactly the coherence condition for pseudo-functors. The first simply says that $\tilde{\varphi}_{g,h}$ is a natural isomorphism. (For a general pseudo-functor, we have also structural 2-cells involving identities. In this case, however, we do not need them, for they are determined from the condition of $\tilde{\varphi}$.) For the reader's convenience, we record the construction of group F from a factor set. The underlying set is $N \times G$. Multiplication is given as $(x, g) \cdot (y, h) = (x(g \cdot y)\tilde{\varphi}_{g,h}^{-1}, gh)$. The unit is $(\tilde{\varphi}_{e,e}, e)$ where e is the unit of G.

Two different factor sets may yield isomorphic groups. Quasi-natural equivalence between pseudo-functors yields the condition for that. Two factors sets $(\cdot, \tilde{\varphi})$ and $(\cdot', \tilde{\psi})$ give isomorphic groups if there is a family of elements $z_g \in N$ for $g \in G$, subject to the equalities:

$$\begin{aligned} g \cdot' x &= z_g(g \cdot x)z_g^{-1} \\ \tilde{\psi}_{g,h} &= z_{gh}\tilde{\varphi}_{g,h}z_g^{-1}(g \cdot' z_h)^{-1} \end{aligned}$$

where $x \in N$ and $g, h \in G$. Again the second equality is exactly the coherence condition for quasi-natural transformations.

3 Twisted wreath product and the Kaloujnine-Krasner theorem

Let G be a group acting on a set Y from right. For a group N, the *general wreath product* is defined as semidirect product $N^Y \rtimes G$ [6]. Here G acts on cartesian product N^Y from left as permutation of components. Namely, for a function $Y \xrightarrow{\theta} N$ and $g \in G$, we define the function $g \cdot \theta$ by equality $(g \cdot \theta)(y) = \theta(y \cdot g)$ for $y \in Y$.

The group N^Y is equivalent to the hom-groupoid $\mathrm{Hom}_{\mathbf{Gpoid}}(Y, N)$, where Y is regarded as a discrete groupoid and N as a groupoid with a single object. Therefore the general wreath product can be written $\int_{z \in G} \mathrm{Hom}_{\mathbf{Gpoid}}(\varphi(z), N)$ where $G^{\mathrm{op}} \xrightarrow{\varphi} \mathbf{Gpoid}$ is the functor carrying the unique object $z \in G$ to the discrete groupoid Y and morphisms of G to the corresponding right actions on Y.

Our proof of the Kaloujnine-Krasner theorem involves standard wreath product as well as twisted wreath product [6].

3.1 Definition
Let N and G be a group. Let us assume for (ii), in addition, that H is a subgroup of G endowed with a left action on N (note that H acts also on G by left multiplication).

(i) The *standard wreath product* $N \mathrm{Wr} G$ is defined as the semidirect product $N^G \rtimes G$ where N^G is the group of all functions on G into N with pointwise multiplication and the left action of G on N^G is induced by the regular right action of G on G itself.

(ii) The *twisted wreath product* $N \mathrm{Wr}_H G$ is defined similarly to the standard wreath product, except that the first component is restricted to those functions $G \to N$ commuting with left actions of H.

The standard wreath product $N \mathrm{Wr} G$ is a special case of general wreath product $N^Y \rtimes G$ where Y is equal to G. Furthermore, the standard wreath product is a special case of the twisted wreath product $N \mathrm{Wr}_H G$ where H is the unit group.

From the observation above for general wreath product, the standard wreath product $N \mathrm{Wr} G$ can be written $\int_{z \in G} \mathrm{Hom}_{\mathbf{Gpoid}}(\varphi(z), N)$ where the functor $G^{\mathrm{op}} \xrightarrow{\varphi} \mathbf{Gpoid}$ carries the unique object $z \in G$ to the set (i.e., discrete groupoid) of elements of G, and the morphisms of G to the right multiplication by the corresponding elements.

We want to give a similar characterization for twisted wreath product. Let us consider the groupoid-enriched category $\mathbf{Gpoid}^H$. Then φ in the previous paragraph, in fact, yields a functor into $\mathbf{Gpoid}^H$, where the left action of H on $\varphi(z) = G$ is given by left multiplication. Moreover, we can consider the given action $H \to \mathrm{Aut}(N)$ as the (strict) functor $t \in \mathbf{Gpoid}^H$ carrying the unique object of H to N. We show that $\int_{z \in G} \mathrm{Hom}_{\mathbf{Gpoid}^H}(\varphi(z), t)$ is equivalent to the twisted wreath product $G \mathrm{Wr}_H N$.

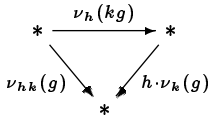
3.2 Lemma
Let H be a subgroup of a group G, and let N be a group endowed with a left action $H \to \mathrm{Aut}\, N$, regarded as a functor $H \xrightarrow{t} \mathbf{Gpoid}$. Moreover, let $G^{\mathrm{op}} \xrightarrow{\varphi}$

$\mathbf{Gpoid}^H$ *be the functor carrying the unique object* $z \in G$ *to the set* G *with left action of* H *given by multiplication.*

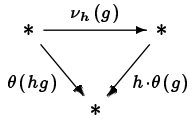
Then the hom-groupoid $\mathrm{Hom}_{\mathbf{Gpoid}^H}(\varphi(z), t)$ *is equivalent to a group. The elements of the group are the functions* θ *on* G *into* N *commuting with the left action of* H*, that is, satisfying* $\theta(hg) = h \cdot \theta(g)$*. Multiplication is pointwise.*

(*Proof*) First we verify that the hom-groupoid is equivalent to a group. It suffices to show every quasi-natural transformation $\varphi(z) \xrightarrow{\nu} t$ is isomorphic to a strict (quasi-)natural transformation $\varphi(z) \xrightarrow{\tilde{\nu}} t$ (strictness means all structural 2-cells are identities), since such a strict one is unique for N has only one object.

Quasi-naturality of ν amounts to giving a family of elements $\nu_h(g)$ in group $t(z)$ for $h \in H$ and $g \in G$, rendering the diagram



commutative for $h, k \in H$ and $g \in G$, where $*$ denotes the unique object of group $t(z)$, and $h \cdot (\text{-})$ denotes $t(h)(\text{-})$. Modification $\nu \xrightarrow{\theta} \tilde{\nu}$ amounts to giving a family of $\theta(g) \in t(z)$ for $g \in G$, rendering



commutative for $h \in H$ and $g \in G$, noticing that $\tilde{\nu}_h(g)$ is identity by assumption of strictness of $\tilde{\nu}$.

Let X be a right transversal of H in G, that is, the set of chosen representatives from the right cosets of H in G. We fix X once and for all. For each $g \in G$, there is a unique $g^\tau \in X$ such that $Hg = Hg^\tau$. We note also $(hg)^\tau = g^\tau$ for $h \in H$. Now let us define $\theta(g)$ by $\nu_{h_0}(g^\tau)$ where $h_0 = g(g^\tau)^{-1}$ is an element of H. Then it is easy to see that the triangle diagram of quasi-naturality of ν implies that of modification. This ends the proof that every quasi-natural transformation is isomorphic to the strict $\tilde{\nu}$.

Furthermore, if $\nu = \tilde{\nu}$, each modification $\tilde{\nu} \xrightarrow{\theta} \tilde{\nu}$ should satisfy $\theta(hg) = h \cdot \theta(g)$ as immediately seen from the diagram above for modification. □

Remark: Let us suppose that the functor t is defined on G. Then a consequence of this lemma (indeed equivalent to it) is that every quasi-natural transformation $\varphi(z) \xrightarrow{\nu} t$ in $\mathbf{Gpoid}^H$ extends to $\mathbf{Gpoid}^G$. Namely every ν defined on subgroup H extends to full domain G. This is proved by defining $\varphi(z) \xrightarrow{\nu^\dagger} t$ in $\mathbf{Gpoid}^G$ with equality $\nu^\dagger_g(g') = (g \cdot \theta(g'))^{-1}\theta(gg')$ for all $g, g' \in G$.

From this lemma, we conclude that the twisted wreath product $N \operatorname{Wr}_H G$ is equivalent to $\int_{z \in G} \operatorname{Hom}_{\mathbf{Gpoid}^H}(\varphi(z), t)$.

We verify an extension of the Kaloujnine-Krasner theorem to twisted wreath product. We start with the following general lemma without a proof.

3.3 Lemma
Let $\mathbf{C} \xrightarrow{F} \mathbf{D}$ *be a pseudo-functor between bicategories.*

If we are given a family of objects $G(X) \in \mathbf{D}$ *satisfying* $F(X) \cong G(X)$ *for all objects* $X \in \mathbf{C}$*, then* G *gives rise to a pseudo-functor, and* F *and* G *turn out to be quasi-naturally equivalent.*

3.4 Theorem
Let F *be a group extension of* N *by* G*. Let us assume that* F *splits on a subgroup* $H \leq G$*, that is, there is an injective group homomorphism* $H \xrightarrow{\sigma} F$ *yielding an identity on* H *by composing the canonical projection* $F \to G$*.*

Under these conditions, F *embeds into the twisted wreath product* $N \operatorname{Wr}_H G$ *where the left action of* H *on* N *is given by* $h \cdot n = \sigma(h) n \sigma(h)^{-1}$*.*

(*Proof*) By Thm. 2.4, the canonical projection $F \xrightarrow{p} G$ corresponds to the functor $t = \int_{x \in F} G(p(x), \text{-})$ in $\mathbf{Gpoid}^G$. We can construct a quasi-natural transformation ν in $\mathbf{Gpoid}^G$ such that

$$\nu_z \ : \ t(z) \ \longrightarrow \ \operatorname{Hom}_{\mathbf{Gpoid}^H}(\varphi(z), t)$$

where $G^{\mathrm{op}} \xrightarrow{\varphi} \mathbf{Gpoid}^H$ is defined as in Lem. 3.2. This construction is given as a slight modification of the Yoneda lemma. Indeed, if $H = G$ then ν is nothing but the equivalence appearing in the Yoneda lemma, observing that φ is the Yoneda embedding: $\varphi(z) = G(z, \text{-})$. For general subgroup $H \leq G$, there is still a quasi-natural transformation ν. Each ν_z is faithful as a groupoid homomorphism.

We note that $t(z) = \int_{x \in F} G(p(x), z)$ is equivalent to the group N. So, from the preceding lemma, t is quasi-naturally equivalent to a pseudo-functor $t' \in \mathbf{Gpoid}^H$ satisfying that $t'(z) = N$. Hence, without loss of generality, we can replace t in the codomain of ν_z by t' satisfying $t'(z) = N$. Now, returning to the world of $\mathbf{Gpoid}/G$ by the Grothendieck construction using Thm. 2.4, we have a groupoid homomorphism $F \xrightarrow{e} \int_{z \in G} \operatorname{Hom}_{\mathbf{Gpoid}^H}(\varphi(z), t')$. It is easy to prove that e is faithful.

This appears the end of the proof: $\int_{z \in G} \operatorname{Hom}_{\mathbf{Gpoid}^H}(\varphi(z), t')$ looks equivalent to the twisted wreath product $N \operatorname{Wr}_H G$. This is not true, however, since t' has no assurance to be a strict functor. The definition of twisted wreath product requires that H act on N in an ordinary sense, that is, the corresponding pseudo-functor $t' \in \mathbf{Gpoid}^H$ to be strict. By analysis of $t \cong t'$, the automorphism

$t'(h) : N \to N$ carries n to $h^\tau n(h^\tau)^{-1}$ where h^τ is a chosen element in $p^{-1}(h)$. Hence, if there is a splitting on H, we can choose h^τ in a functorial way (that is $(hk)^\tau = h^\tau k^\tau$), yielding the strictness of t'. □

3.5 Corollary
Let F be a group extension of N by G.

Then F embeds into the standard wreath product N Wr G.

(*Proof*) Set $H = \{1\}$. We note that the last paragraph in the proof of the theorem turns out to be unnecessary, since t' becomes a strict functor trivially. □

The direct proof of the theorem is found in [6]. See also [7]. We emphasize that the proof above uses only simple general facts in the theory of groupoids and bicategories. Moreover the use of (a slight modification of) the Yoneda lemma in the proof suggests, someways, why the wreath product arises as the target of the embedding.

References

[1] J. Bénabou, Introduction to bicategories, in: *Reports of the Midwest Category Seminar*, (Springer, 1967) pp. 1–77.

[2] R. Brown, Fibrations of groupoids, *J. Algebra* **15** (1970) 103–132.

[3] S. Mac Lane, *Categories for the Working Mathematician*, (Springer, 1971).

[4] B. H. Neumann, Twisted wreath product of groups, *Arch. Math.* **14** (1963) 1–6.

[5] R. Street, Fibrations in bicategories, *Cahiers Topologie Géometrie Différentielle* **21** (1980) 111–160; Corrections, **28** (1987) 53–56.

[6] M. Suzuki, *Group Theory, I*, Grundlagen der mathematischen Wissenshaften 247, (Springer, 1982).

[7] C. Wells, Some applications of the wreath product construction, *American Math. Monthly* **83** (1976) 317–338.

Geometry of Interaction explained

Masaru Shirahata
白旗　優
Division of Mathematics, Keio University, Hiyoshi Campus
慶應義塾大学日吉数学研究室

sirahata@math.hc.keio.ac.jp

1 Introduction

The purpose of this paper is mostly expository. We first review the axiomatic framework recently proposed by Abramsky, Haghverdi and Scott [1] for Girard's Geometry of Interaction [3] in terms of traced symmetric monoidal categories. We then work out in some detail how the new proposal captures Girard's original formulation.

The Geometry of Interaction is introduced by Girard as the mathematical model of the dynamics of cut-elimination. It is formulated in terms of operator algebra, and the cut-elimination is represented by a single *execution formula*. This is very much interesting, but the intuitive meaning of this mathematical model does not seem to be perfectly clear.

Abramsky and Jagadeesan [2] proposed their own formulation of Geometry of Interaction, which is very much similar to their game semantics of linear logic. The machinery is fairly simple and clear, but the precise relationship to the original formulation is not fully explicated.

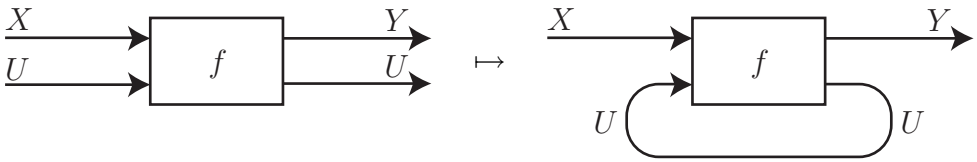
The axiomatic framework of Geometry of Interaction proposed by Abramsky, Haghverdi and Scott is supposed to fill the gap between the two formulations. In any case it gives us a very clear and intuitive picture. The framework is based on a traced symmetric monoidal category, and it yields a certain compact closed category as a model of linear combinatory algebra, covering as much as Girard's original formulation works.

The precise relationship of this framework to the original Geometry of Interaction is, however, only claimed in Abramsky, Haghverdi and Scott [1] and sketched in Haghverdi [4]. It may be obvious to them, but we find it helpful to work it out in some detail. This is what we intend to do in the present paper.

2 The axiomatic framework

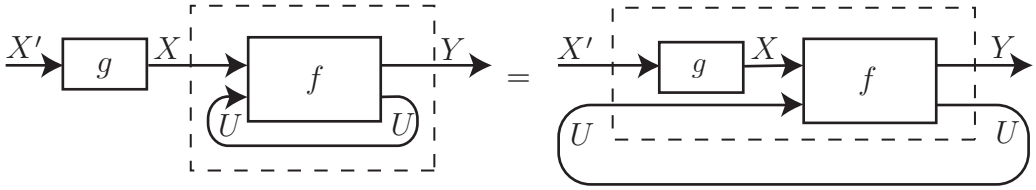
2.1 Traced symmetric monoidal categories

A traced symmetric monoidal category $\mathbb{C}$ is a symmetric monoidal category enhanced with the trace operations $\mathtt{Tr}^U_{X,Y}(f)$ from $\mathbb{C}(X \otimes U, Y \otimes U)$ to $\mathbb{C}(X, Y)$, represented by the diagrams:

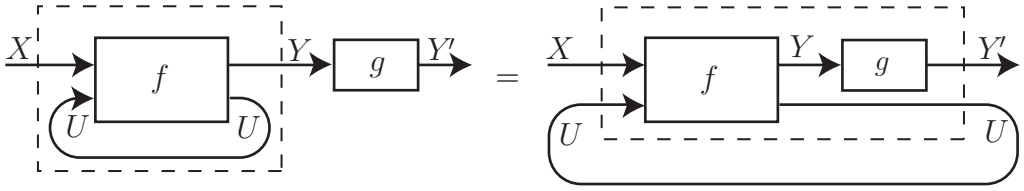


$\mathtt{Tr}^U_{X,Y}(f)$ must satisfy the following conditions. To simplify the presentation we assume that $\mathbb{C}$ is a strict monoidal category.

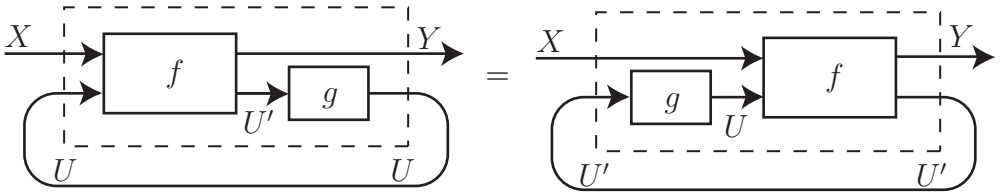
1. $\mathtt{Tr}^U_{X,Y}(f)g = \mathtt{Tr}^U_{X',Y}(f(g \otimes 1_U))$ for $f : X \otimes U \to Y \otimes U$ and $g : X' \to X$:



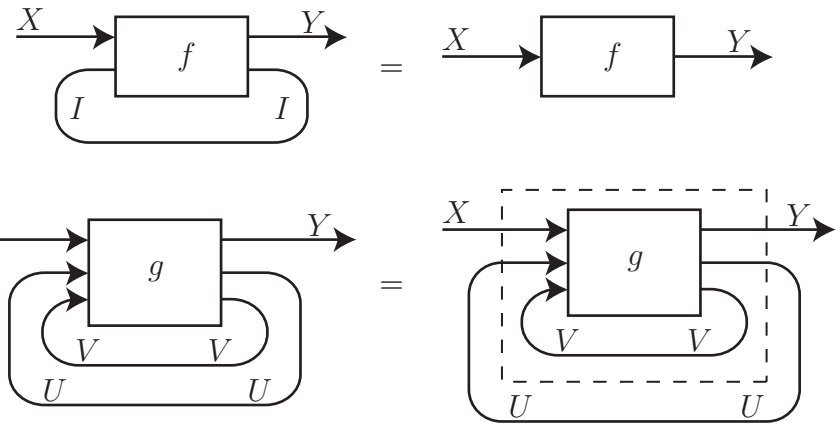
2. $g\mathtt{Tr}^U_{X,Y}(f) = \mathtt{Tr}^U_{X,Y'}((g \otimes 1_U)f)$ for $f : X \otimes U \to Y \otimes U$ and $g : Y \to Y'$:



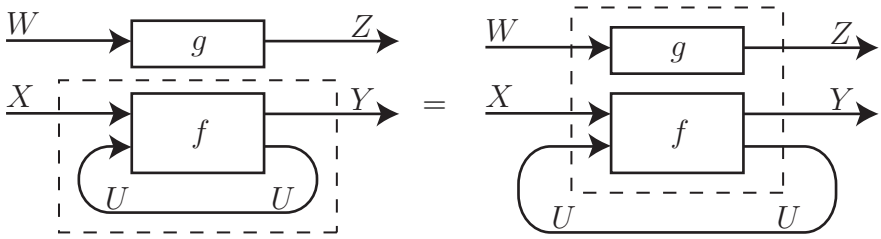
3. $\mathtt{Tr}^U_{X,Y}((1_Y \otimes g)f) = \mathtt{Tr}^{U'}_{X,Y}(f(1_X \otimes g))$ for $f : X \otimes U \to Y \otimes U'$ and $g : U' \to U$:



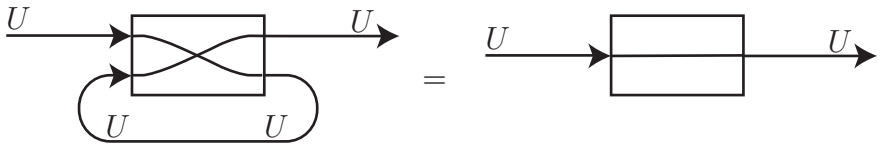
4. $\mathtt{Tr}^I_{X,Y}(f) = f$ and $\mathtt{Tr}^{U\otimes V}_{X,Y}(g) = \mathtt{Tr}^U_{X,Y}(\mathtt{Tr}^V_{X\otimes U,Y\otimes U}(g))$ for $f : X \to Y$, where $X \otimes I = X$ and $Y \otimes I = Y$, and $g : X \otimes U \otimes V \to Y \otimes U \otimes V$:



5. $g \otimes \mathtt{Tr}^U_{X,Y}(f) = \mathtt{Tr}^U_{W\otimes X,Z\otimes Y}(g \otimes f)$ for $f : X \otimes U \to Y \otimes U$ and $g : W \to Z$:



6. $\mathtt{Tr}^U_{U,U}(\sigma_{U,U}) = 1_U$, where $\sigma_{U,U}$ is the canonical morphism for symmetry;



For traced symmetric monoidal categories $\mathbb{C}$ and $\mathbb{D}$, a monoidal functor (F, ϕ, ϕ_I) from $\mathbb{C}$ to $\mathbb{D}$ is called *traced* if it is symmetric and it satisfies

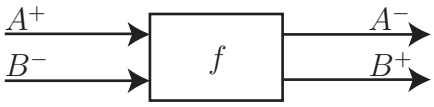
$$\mathtt{Tr}^{FU}_{FX,FY}(\phi^{-1}_{Y,U}(Ff)\phi_{X,U}) = F(\mathtt{Tr}^U_{X,Y}(f))$$

for $f : X \otimes U \to Y \otimes U$.

2.2 The Geometry of Interaction construction

Given a traced symmetric monoidal category $\mathbb{C}$, we construct a compact closed category $\mathcal{G}(\mathbb{C})$, which gives a basic framework for the Geometry of Interaction.

The objects of $\mathcal{G}(\mathbb{C})$ are the pairs (A^+, A^-) of objects of $\mathbb{C}$. Morphisms f from (A^+, A^-) to (B^+, B^-) are the morphisms $f : A^+ \otimes B^- \to A^- \otimes B^+$ of $\mathbb{C}$:

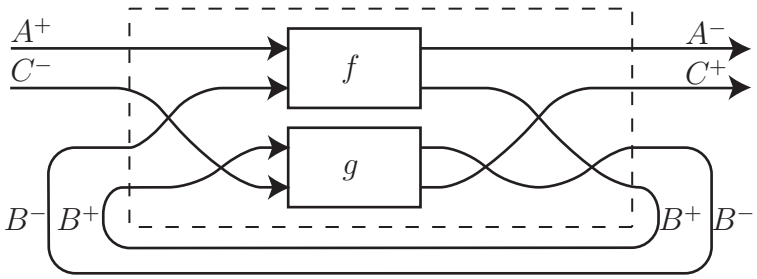


The identity for an object (A^+, A^-) is given as the canonical morphism $\sigma_{A^+,A^-} : A^+ \otimes A^- \to A^- \otimes A^+$ for symmetry in $\mathbb{C}$. Sometimes it is helpful to add extra subscripts to distinguish *occurrences* of objects. We then write $\sigma_{A^+,A^-} : A_1^+ \otimes A_2^- \to A_1^- \otimes A_2^+$ to indicate that it is a morphism from (A_1^+, A_1^-) to (A_2^+, A_2^-).

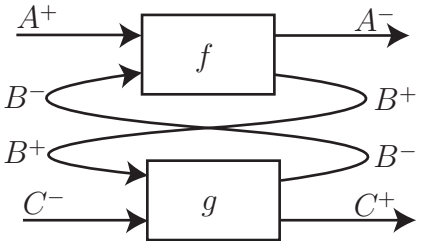
The composition $gf : (A^+, A^-) \to (C^+, C^-)$ of morphisms $f : (A^+, A^-) \to (B^+, B^-)$ and $g : (B^+, B^-) \to (C^+, C^-)$ in $\mathcal{G}(\mathbb{C})$ is defined as

$$\mathtt{Tr}^{B^- \otimes B^+}_{A^+ \otimes C^-, A^- \otimes C^+}(\beta(f \otimes g)\alpha)$$

in $\mathbb{C}$, where $\alpha = (1_{A^+} \otimes 1_{B^-} \otimes \sigma_{C^-,B^+})(1_{A^+} \otimes \sigma_{C^-,B^-} \otimes 1_{B^+})$ and $\beta = (1_{A^-} \otimes 1_{C^+} \otimes \sigma_{B^+,B^-})(1_{A^-} \otimes \sigma_{B^+,C^+} \otimes 1_{B^-})(1_{A^-} \otimes 1_{B^+} \otimes \sigma_{B^-,C^+})$, represented by the diagram:



Since the coherence of the symmetric monoidal category allows us to permute the tensor products in $\mathbb{C}$ through the canonical morphisms in any order, we can make the use of permutations implicit and depict the above diagram more intuitively:

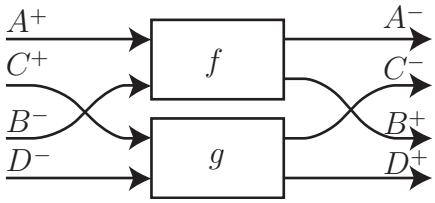


$\mathcal{G}(\mathbb{C})$ is equipped with the tensorial structure. The tensor product of (A^+, A^-) and (B^+, B^-) is given by $(A^+ \otimes B^+, A^- \otimes B^-)$, *i.e.* by taking the tensor products in $\mathbb{C}$ pointwise. The unit is the pair (I, I) of the unit I in $\mathbb{C}$.

The tensor product of $f \otimes g : (A^+ \otimes C^+, B^- \otimes D^-) \to (A^- \otimes C^-, B^+ \otimes D^+)$ of $f : (A^+, A^-) \to (B^+, B^-)$ and $g : (C^+, C^-) \to (D^+, D^-)$ is given by

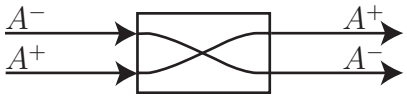
$$f \otimes g = (1_{A^-} \otimes \sigma_{B^+,C^-} \otimes 1_{D^+})(f \otimes g)(1_{A^+} \otimes \sigma_{C^+,B^-} \otimes 1_{D^-})$$

in $\mathbb{C}$, *i.e.* by taking the tensor product of f and g in $\mathbb{C}$ and composing it with the appropriate permutations, represented by the diagram:



.

$\mathcal{G}(\mathbb{C})$ has the structure of a compact closed category as well. The left adjoint $(A^+, A^-)^*$ of (A^+, A^-) is given by (A^-, A^+), *i.e.* by exchanging the two components. Then the unit $\eta : (I, I) \to (A^+, A^-) \otimes (A^+, A^-)^*$ should be a morphism from the unit object (I, I) to $(A^+ \otimes A^-, A^- \otimes A^+)$, which is in turn a morphism from $A^- \otimes A^+$ to $A^+ \otimes A^-$ in $\mathbb{C}$. In fact we can simply take σ_{A^-,A^+} in $\mathbb{C}$ as the unit η:



.

The counit $\delta : (A^+, A^-)^* \otimes (A^+, A^-) \to (I, I)$ can be similarly given by $\sigma_{A^-,A^+} : A^- \otimes A^+ \to A^+ \otimes A^-$ in $\mathbb{C}$.

2.3 The GoI Situation

To yield a model of intuitionistic linear logic, the traced symmetric monoidal category $\mathbb{C}$ needs to have an extra structure, which is summarized as a GoI Situation.

Let us recall that A is a *retract* of B when there exists morphisms $f : A \to B$ and $g : B \to A$ such that $gf = 1_A$. In such a case we call (f, g) a *retraction* and write $f : A \triangleleft B : g$. The *GoI Situation* is a triple $(\mathbb{C}, T, U)$, where $\mathbb{C}$ is a traced symmetric monoidal category, T is a traced symmetric monoidal functor on $\mathbb{C}$ with the following retractions as monoidal natural transformations:

1. $e : TT \triangleleft T : e'$ (Comultiplication),
2. $d : \mathtt{Id} \triangleleft T : d'$ (Dereliction),

3. $c : T \otimes T \triangleleft T : c'$ (Contraction),

4. $w : \mathcal{K}_I \triangleleft T : w'$ (Weakening), where $\mathcal{K}_I$ is the constant I functor;

and U is a *reflexive object* in $\mathbb{C}$ with the retractions:

1. $j : U \otimes U \triangleleft U : k$,

2. $l : I \triangleleft U : m$,

3. $u : TU \triangleleft U : v$.

The functor T is intended to induce the exponential operator ! of linear logic in $\mathcal{G}(\mathbb{C})$, as suggested by the names of the retractions.

For any categories $\mathbb{C}, \mathbb{D}$ and functors $F, G : \mathbb{C} \to \mathbb{D}$, we say that a family of morphisms $m_A : FA \to GA$ is a *pointwise* natural transformation from F to G if the naturality condition holds only for morphisms $f : I \to A$, *i.e.* the diagram

$$\begin{array}{ccc} FA & \xrightarrow{m_A} & GA \\ {\scriptstyle Ff}\uparrow & & \uparrow{\scriptstyle Gf} \\ FI & \xrightarrow{m_I} & GI \end{array}$$

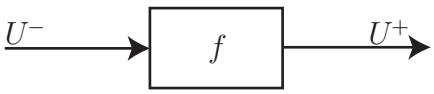
commutes for any such f.

Given a GoI Situation $(\mathbb{C}, T, U)$, the compact closed category $\mathcal{G}(\mathbb{C})$ becomes a *weakly* linear category, in the sense that the standard maps for the exponential are only pointwise natural.

This is, however, sufficient to obtain a model of intuitionistic linear logic, since we only consider the morphisms from (I, I) to (U, U). In fact $\mathcal{G}(\mathbb{C})((I, I), (U, U))$ is a linear combinatory algebra, *i.e.*, the algebraic model of intuitionistic linear logic.

The construction of linear combinatory algebra from the GoI Situation is fully worked out in Abramsky, Haghverdi and Scot [1], and we do not give its detail here. In the present paper we are more interested in how this setting fits Girard's original formulation of Geometry of Interaction.

At this moment we only note that a morphism $f : (I, I) \to (U, U)$ in $\mathcal{G}(\mathbb{C})$ is nothing but the morphism $f : U \to U$ in $\mathbb{C}$, assuming that $\mathbb{C}$ is a strict monoidal category. In this case it is more perspicuous to distinguish the two occurrences of U in (U, U) as (U^+, U^-), and write $f : U^- \to U^+$ for f in $\mathbb{C}$:



2.4 The category PInj

A typical example of a traced symmetric monoidal category with a GoI Situation is the category of sets and partial injective functions. This category is equipped with the tensorial structure defined by the disjoint unions of sets and functions.

Given the disjoint union $A \uplus B = \{(0,x) \mid x \in A\} \cup \{(1,y) \mid y \in B\}$ of sets A and B, we have the injections $\iota_1 : A \to A \uplus B$ and $\iota_2 : B \to A \uplus B$ defined by

$$\iota_1 : x \mapsto (0,x), \qquad \iota_2 : y \mapsto (1,y)$$

and the quasi (partial) projections $\pi_1 : A \uplus B \to A$ and $\pi_2 : A \uplus B \to B$ defined by

$$\pi_1 : (0,x) \mapsto x, \qquad \pi_2 : (1,y) \mapsto y.$$

They can be naturally extended to the n-ary injections $\iota_i^n : A_1 \to A_1 \uplus \cdots \uplus A_n$ and quasi projections $\pi_i^n : A_1 \uplus \cdots \uplus A_n \to A_i$. Note that they are all partial injective functions and hence morphisms of **PInj**.

If partial injective functions $f_i : A \to B$ $(i \in I)$ have mutually disjoint domains $\{x \mid \exists y\ f_i(x) = y\}$ and mutually disjoint codomains $\{y \mid \exists x\ f_i(x) = y\}$, they can be summed up simply by taking the union $\bigcup_{i\in I} f_i$. We write $\sum_{i\in I} f_i$ for $\bigcup_{i\in I} f_i$.

By means of ι_i^n and π_i^n any partial function $f : A_1 \uplus \cdots \uplus A_n \to A_1 \uplus \cdots \uplus A_m$ can be decomposed as

$$f = \sum_{i\in\{1,\ldots,m\}} \sum_{j\in\{1,\ldots,n\}} f_{ij}$$

where $f_{ij} = \pi_i^m f \iota_j^n$. Furthermore the trace of $f : A \oplus U \to B \oplus U$ is given by

$$\mathtt{Tr}_{A,B}^U(f) = f_{AA} + \sum_{n\in\omega} f_{UB} f_{UU}^n f_{AU}$$

where $f_{AA} = f_{11}, f_{AU} = f_{12}, f_{UB} = f_{21}, f_{UU} = f_{22}$.

3 Girard's formulation

3.1 The preliminary setting

Girard's original Geometry of Interaction is formulated in terms of operator algebra. The canonical example is the Banach space $\mathcal{B}(\mathbb{H})$ of bounded operators on $\mathbb{H}$, where $\mathbb{H}$ is the Hilbert space ℓ^2 of square summable infinite sequences of complex numbers.

It turns out that the full internal structure of $\mathcal{B}(\mathbb{H})$ is not really necessary. For this reason we only state some of the basic definitions. The infinite sequence $\boldsymbol{z} = (z_i)_{i\in\omega}$ of complex numbers is *square summable* if $\sum_{i=0}^{\infty} z_i \bar{z}_i$ converges. In that

case the square root of this value is denoted $\|\boldsymbol{z}\|$. The *bounded operator* u on $\mathbb{H}$ is a linear transformation on $\mathbb{H}$ such that $\sup\{\|u(\boldsymbol{z})\| \mid \|\boldsymbol{z}\| = 1\}$ is finite.

For $\boldsymbol{x} = (x_i)$ and $\boldsymbol{y} = (y_i)$, the scalar product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is defined as $\sum x_i \bar{y}_i$, and we have the adjoint operation $u \mapsto u^*$ on $\mathcal{B}(\mathbb{H})$ such that $\langle u\boldsymbol{x}, \boldsymbol{y} \rangle = \langle \boldsymbol{x}, u^*\boldsymbol{y} \rangle$. A bounded operator u is

- *unitary* if $uu^* = u^*u = 1$, where 1 is the identity operator,
- *hermitian* if $u = u^*$,
- a *projector* if u is hermitian and $u^2 = u$,
- a *symmetry* if u is hermitian and unitary,
- a *partial isometry* if uu^* and u^*u are projectors.

Any projector defines a closed subspace $\mathbb{H}' = \{u\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{H}\}$ of $\mathbb{H}$. Conversely given any closed subspace $\mathbb{H}'$ of $\mathbb{H}$, the unique decomposition $\boldsymbol{x} = \boldsymbol{x}' + \boldsymbol{x}''$ of $\boldsymbol{x} \in \mathbb{H}$ into $\boldsymbol{x}'$ in $\mathbb{H}'$ and $\boldsymbol{x}''$ in its orthogonal complement $\mathbb{H}''$ gives a projector $\boldsymbol{x} \mapsto \boldsymbol{x}'$.

A partial isometry u can be regarded then as a scalar product preserving map (isometry) from the subspace $\{u^*u\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{H}\}$ onto the subspace $\{uu^*\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{H}\}$. Clearly $uu^*u\boldsymbol{x}$ belongs to the latter, and it is onto since $uu^*\boldsymbol{x} = u((u^*u)(u^*\boldsymbol{x}))$. The scalar product is preserved since

$$\langle uu^*u\boldsymbol{x}, uu^*u\boldsymbol{y} \rangle = \langle u^*u\boldsymbol{x}, u^*uu^*u\boldsymbol{y} \rangle = \langle u^*u\boldsymbol{x}, u^*u\boldsymbol{y} \rangle$$

holds.

3.2 The partial isometries p and q

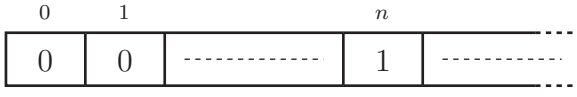
What is really necessary from $\mathcal{B}(\mathbb{H})$ is the existence of partial isometries p and q, which are used to internalize the direct sum $\mathbb{H} \oplus \mathbb{H}$ within $\mathbb{H}$. In fact it suffices to have any p and q such that

(1) $p^*q = q^*p = 0$,

(2) $p^*p = q^*q = 1$.

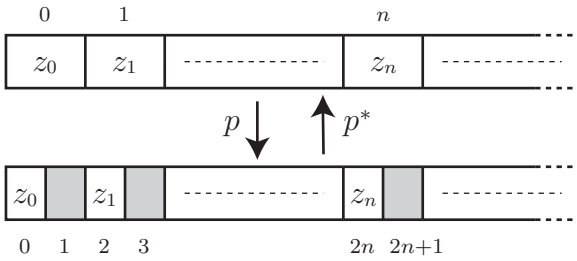
As a matter of fact (2) implies that p and q are partial isometries.

The concrete examples of p and q can be given by introducing the canonical base $(\boldsymbol{b}^n)$ of ℓ^2. Each $\boldsymbol{b}^n = (b^n_m)$ is an infinite sequence of 0 and 1 such that $b^n_m = 1$ iff $n = m$:

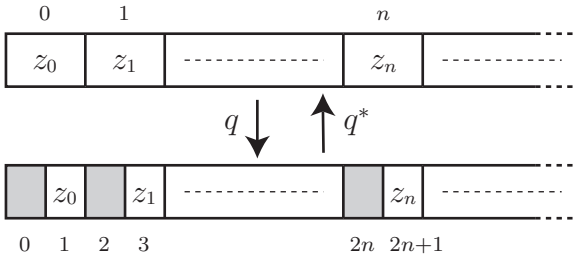


Clearly any $\boldsymbol{z} = (z_n)$ is expressed as the infinitary linear combination $\boldsymbol{z} = \sum z_n \boldsymbol{b}^n$.

Then p is given by $p\boldsymbol{z} = \sum z_n \boldsymbol{b}^{2n}$ and its adjoint p^* by $p^*\boldsymbol{z} = \sum z_{2n} \boldsymbol{b}^n$:



Similarly $q\boldsymbol{z} = \sum z_n \boldsymbol{b}^{2n+1}$ and $q^*\boldsymbol{z} = \sum z_{2n+1} \boldsymbol{b}^n$:



Note that p may be regarded as an isometry from $\mathbb{H} = \{p^*p\boldsymbol{z} \mid \boldsymbol{z} \in \mathbb{H}\}$ onto $\{\sum z_n \boldsymbol{b}_{2n} \mid z_n \in \mathbb{C}\} = \{pp^*\boldsymbol{z} \mid \boldsymbol{z} \in \mathbb{H}\}$, hence a bijection between them. Similarly q may be regarded as a bijection between $\mathbb{H}$ and $\{\sum z_n \boldsymbol{b}_{2n+1} \mid z_n \in \mathbb{C}\}$.

In those examples of p and q the equation

(1′) $pp^* + qq^* = 1$

holds, which is stronger than (1). From (1′) we have

$$p^*q = p^*(pp^* + qq^*)q = p^*pp^*q + p^*qq^*q = p^*q + p^*q$$

and $p^*q = 0$ holds. $q^*p = 0$ similarly follows from (1′).

3.3 Internalizing the direct sum

The direct sum $\mathbb{H} \oplus \mathbb{H}'$ of the Hilbert spaces $\mathbb{H}$ and $\mathbb{H}'$ can simply given as the vector space of formal expressions $\boldsymbol{x} \oplus \boldsymbol{x}'$ for $\boldsymbol{x} \in \mathbb{H}$ and $\boldsymbol{x}' \in \mathbb{H}'$, where the vector addition and the scalar multiplication are defined pointwise, and

$$\langle \boldsymbol{x} \oplus \boldsymbol{x}', \boldsymbol{y} \oplus \boldsymbol{y}' \rangle = \langle \boldsymbol{x}, \boldsymbol{y} \rangle + \langle \boldsymbol{x}', \boldsymbol{y}' \rangle.$$

The direct sum $f \oplus g$ of morphisms f and g is defined similarly as

$$(f \oplus g)(\boldsymbol{x} \oplus \boldsymbol{y}) = f\boldsymbol{x} \oplus g\boldsymbol{y}.$$

We take $\boldsymbol{x} \oplus (\boldsymbol{y} \oplus \boldsymbol{z})$ to be identical to $(\boldsymbol{x} \oplus \boldsymbol{y}) \oplus \boldsymbol{z}$, simply denoted $\boldsymbol{x} \oplus \boldsymbol{y} \oplus \boldsymbol{z}$, and make the canonical isomorphisms for associativity identity maps. Recall that direct sum is just another name of biproduct in the category of vector spaces.

For $\mathbb{H} = \ell^2$, the direct sum $\mathbb{H} \oplus \mathbb{H}$ has the base consisting of $\boldsymbol{b}^n \oplus \boldsymbol{0}$ and $\boldsymbol{0} \oplus \boldsymbol{b}^n$. Then the mapping

$$\begin{cases} \boldsymbol{b}^n \oplus \boldsymbol{0} & \mapsto & \boldsymbol{b}^{2n}, \\ \boldsymbol{0} \oplus \boldsymbol{b}^n & \mapsto & \boldsymbol{b}^{2n+1} \end{cases}$$

induces the isomorphism $j : \mathbb{H} \oplus \mathbb{H} \to \mathbb{H}$. For $\boldsymbol{x} = (x_n)$ and $\boldsymbol{y} = (y_n)$

$$j(\boldsymbol{x} \oplus \boldsymbol{y}) = j(\boldsymbol{x} \oplus \boldsymbol{0} + \boldsymbol{0} \oplus \boldsymbol{y}) = \sum x_n \boldsymbol{b}^{2n} + \sum y_n \boldsymbol{b}^{2n+1} = p\boldsymbol{x} + q\boldsymbol{y},$$

and for $\boldsymbol{z} = (z_n)$

$$\begin{aligned} j^{-1}\boldsymbol{z} &= j^{-1}\left(\sum z_{2n}\boldsymbol{b}^{2n} + \sum z_{2n+1}\boldsymbol{b}^{2n+1}\right) \\ &= \left(\sum z_{2n}\boldsymbol{b}^n\right) \oplus \boldsymbol{0} + \boldsymbol{0} \oplus \left(\sum z_{2n+1}\boldsymbol{b}^n\right) \\ &= p^*\boldsymbol{z} \oplus q^*\boldsymbol{z}. \end{aligned}$$

Hence we can regard $p\boldsymbol{x} + q\boldsymbol{y} \in \mathbb{H}$ as the internal representation of $\boldsymbol{x} \oplus \boldsymbol{y} \in \mathbb{H} \oplus \mathbb{H}$, and any $\boldsymbol{z} \in \mathbb{H}$ can be regarded as such.

Given j we have the isomorphisms $1_{\mathbb{H}'} \oplus j : \mathbb{H}' \oplus \mathbb{H} \oplus \mathbb{H} \to \mathbb{H}' \oplus \mathbb{H}$ and this is enough to establish the existence of isomorphism $j^n : \mathbb{H}^n \to \mathbb{H}$ for $n \geq 3$.

Under the general setting $j : \boldsymbol{x} \oplus \boldsymbol{y} \mapsto p\boldsymbol{x} + q\boldsymbol{y}$ does not necessarily give an isomorphism but constitutes a retraction with $k : \boldsymbol{z} \mapsto p^*\boldsymbol{z} \oplus q^*\boldsymbol{z}$. This follows immediately from the conditions (1) and (2) for p and q. It can be generalized to the retraction $j^n : \mathbb{H}^n \triangleleft \mathbb{H} : k^n$ as well.

3.4 Matrix representation of operators

$\mathbb{H}^n$ is a biproduct, and we have the projections $\pi_i : \mathbb{H}^n \to \mathbb{H}$ $(1 \leq i \leq n)$ given by

$$\boldsymbol{x}_1 \oplus \cdots \oplus \boldsymbol{x}_n \mapsto \boldsymbol{x}_i$$

and the injections $\iota_i : \mathbb{H} \to \mathbb{H}^n$ $(1 \leq i \leq n)$ given by

$$\boldsymbol{x} \mapsto \boldsymbol{0} \oplus \cdots \oplus \underset{\substack{\uparrow \\ i\text{th}}}{\boldsymbol{x}} \oplus \cdots \oplus \boldsymbol{0}.$$

This *additive* structure allows the decomposition of a map $f : \mathbb{H}^n \to \mathbb{H}^m$ into the maps (f_{ij}) $(1 \leq i \leq m$ and $1 \leq j \leq n)$ by

$$f_{ij} = \pi_i f \iota_j : \mathbb{H} \to \mathbb{H}$$

in such a way that

$$f(\boldsymbol{x}_1 \oplus \cdots \oplus \boldsymbol{x}_n) = \sum_{i=1}^{n} f_{1i}\boldsymbol{x}_i \oplus \cdots \oplus \sum_{i=1}^{n} f_{mi}\boldsymbol{x}_i.$$

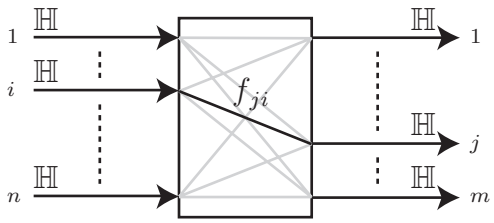
Writing the direct sum $(\boldsymbol{x}_1 \oplus \cdots \oplus \boldsymbol{x}_n)$ as a column vector, we can rewrite the above formula as the familiar equation

$$\begin{pmatrix} \sum_{i=1}^{n} f_{1i}\boldsymbol{x}_i \\ \vdots \\ \sum_{i=1}^{n} f_{mi}\boldsymbol{x}_i \end{pmatrix} = \begin{pmatrix} f_{11} & \cdots & f_{1n} \\ \vdots & \ddots & \vdots \\ f_{m1} & \cdots & f_{mn} \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \end{pmatrix}$$

of matrix computation, *i.e.* the map $f : \mathbb{H}^n \to \mathbb{H}^m$ can be expressed as the matrix

$$\begin{pmatrix} f_{11} & \cdots & f_{1n} \\ \vdots & \ddots & \vdots \\ f_{m1} & \cdots & f_{mn} \end{pmatrix}$$

and this is represented graphically as:



For $f : \mathbb{H}^n \to \mathbb{H}^m$ and $g : \mathbb{H}^{n'} \to \mathbb{H}^{m'}$, the direct sum $f \oplus g$ is then represented by the matrix

$$\begin{pmatrix} f_{11} & \cdots & f_{1n} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ f_{m1} & \cdots & f_{mn} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & g_{11} & \cdots & g_{1n'} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & g_{m'1} & \cdots & g_{m'n'} \end{pmatrix}$$

and the diagram for $f \oplus g$ is obtained by stacking the diagrams for f and g.

Since we have the retraction (possibly isomorphism) $j^n : \mathbb{H}^n \triangleleft \mathbb{H} : k^n$, any map $f : \mathbb{H}^n \to \mathbb{H}^m$ can be regarded as the map $\hat{f} = j^m f k^n : \mathbb{H} \to \mathbb{H}$ as depicted below.

$$\begin{array}{ccc} \mathbb{H}^n & \xrightarrow{f} & \mathbb{H}^m \\ {\scriptstyle k^n}\uparrow & & \downarrow{\scriptstyle j^m} \\ \mathbb{H} & \xrightarrow{\hat{f}} & \mathbb{H} \end{array}$$

We call $\hat{f}$ *the internalized version* of f. Note that f can be recovered from $\hat{f}$ by $\hat{f} \mapsto k^m \hat{f} j^n$. Hence we can officially stay inside the endomorphisms on $\mathbb{H}$, while working informally on maps from $\mathbb{H}^n$ to $\mathbb{H}^m$.

Similarly any map $f : \mathbb{H}^{n+2} \to \mathbb{H}^{n+2}$ $(n \geq 0)$ can be regarded as the map

$$\begin{cases} (1_{\mathbb{H}^n} \oplus j) f (1_{\mathbb{H}^n} \oplus k) & \text{if } n \geq 1, \\ jfk & \text{if } n = 0, \end{cases}$$

from $\mathbb{H}^{n+1}$ to $\mathbb{H}^{n+1}$. Note that

$$j = \begin{pmatrix} p & q \end{pmatrix}, \qquad k = \begin{pmatrix} p^* \\ q^* \end{pmatrix}$$

and $(1_{\mathbb{H}^n} \oplus j) f (1_{\mathbb{H}^n} \oplus k)$ can be written as

$$\begin{pmatrix} 1 & \cdots & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 & 0 \\ 0 & \cdots & 0 & p & q \end{pmatrix} \begin{pmatrix} f_{11} & \cdots & f_{1n} & \alpha_1 & \alpha_2 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ f_{m1} & \cdots & f_{mn} & \beta_1 & \beta_2 \\ \alpha_1' & \cdots & \alpha_2' & \gamma_1 & \gamma_2 \\ \beta_1' & \cdots & \beta_2' & \delta_1 & \delta_2 \end{pmatrix} \begin{pmatrix} 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \\ 0 & \cdots & 0 & p^* \\ 0 & \cdots & 0 & q^* \end{pmatrix}$$

which is equal to the matrix:

$$\begin{pmatrix} f_{11} & \cdots & f_{1n} & \alpha_1 p^* + \alpha_2 q^* \\ \vdots & \ddots & \vdots & \vdots \\ f_{m1} & \cdots & f_{mn} & \beta_1 p^* + \beta_2 q^* \\ p\alpha_1' + q\beta_1' & \cdots & p\alpha_2' + q\beta_2' & p\gamma_1 p^* + p\gamma_2 q^* + q\delta_1 p^* + q\delta_2 q^* \end{pmatrix}$$

We write Φ for the operation $f \mapsto (1_{\mathbb{H}^n} \oplus j) f (1_{\mathbb{H}^n} \oplus k)$ or $f \mapsto jfk$, and Φ will be called *contraction* of matrices (f_{ij}). Note that any two rows (columns) of a matrix can be exchanged by the left (right) action of the isomorphism:

$$\begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 0 & \cdots & 1 & & \\ & & \vdots & \ddots & \vdots & & \\ & & 1 & \cdots & 0 & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{pmatrix}$$

Hence we can contract any two rows and columns of a matrix by moving them last, contracting them and moving them back.

3.5 The interpretation of proofs

For now we concentrate on the multiplicative fragment of classical linear logic without exponentials.

We consider a proof together with all the cut formulas within it. A proof of a sequent $\vdash A_1, \ldots, A_n$ with cut formulas $B_1, \ldots B_m$ is said to be of type $\vdash [B_1, \ldots, B_m]A_1, \ldots, A_n$. It is interpreted by an $(2m+n, 2m+n)$ matrix of the elements of $\mathcal{B}(\mathbb{H})$, which is officially transposed to an element of $\mathcal{B}(\mathbb{H})$ through the retraction.

The interpretation of an axiom $\vdash A, A^\perp$ is the permutation σ:

$$\sigma = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Given a proof of type $\vdash [\Delta]\ \Gamma, A, B$ with the interpretation Π, a proof of type $\vdash [\Delta]\ \Gamma, A \parr B$ obtained from it by the $\parr$ rule is interpreted just by $\Phi\Pi$, where Φ is the contraction of the last two rows and columns of a matrix.

Given proofs of type $\vdash [\Delta]\ \Gamma, A$ and of type $\vdash [\Delta']\ \Gamma', A'$ with interpretations Π and Π'

$$\Pi = \begin{pmatrix} \Sigma & & \alpha \\ & & \vdots \\ \beta & \cdots & \gamma \end{pmatrix}, \qquad \Pi' = \begin{pmatrix} \Sigma' & & \alpha' \\ & & \vdots \\ \beta' & \cdots & \gamma' \end{pmatrix}$$

respectively, a proof of type $\vdash [\Delta, \Delta']\ \Gamma, \Gamma', A \otimes A'$ obtained from them by the $\otimes$ rule is interpreted by

$$\Phi \begin{pmatrix} \Sigma & 0 & \boxed{\begin{matrix}\alpha \\ \vdots\end{matrix}} & 0 \\ & & & \vdots \\ 0 & \Sigma' & 0 & \boxed{\begin{matrix}\alpha' \\ \vdots\end{matrix}} \\ & & \vdots & \\ \boxed{\beta \ \cdots} & 0 \ \cdots & \boxed{\gamma} & 0 \\ 0 \ \cdots & \boxed{\beta' \ \cdots} & 0 & \boxed{\gamma'} \end{pmatrix}$$

where the matrix to be contracted is obtained by moving the last row and column of Π right before the last row and column of Π' in $\Pi \oplus \Pi'$.

Similarly given proofs of type $\vdash [\Theta]\ A, \Gamma$ and of type $\vdash [\Theta']\ A^\perp, \Delta$ with interpretations Π and Π' as below

$$\Pi = \begin{pmatrix} \alpha & \cdots & \beta \\ \vdots & \Sigma & \\ \gamma & & \end{pmatrix}, \qquad \Pi' = \begin{pmatrix} \alpha' & \cdots & \beta' \\ \vdots & \Sigma' & \\ \gamma' & & \end{pmatrix}$$

a proof of type $\vdash [A, \Theta, \Theta'] \; \Gamma, \Delta$ obtained from them by the cut rule is interpreted by the matrix:

$$\begin{pmatrix} \boxed{\alpha} & 0 & \boxed{\cdots \quad \beta} & 0 \quad \cdots \\ 0 & \boxed{\alpha'} & 0 \quad \cdots & \boxed{\cdots \quad \beta'} \\ \boxed{\begin{matrix}\vdots \\ \gamma\end{matrix}} & \begin{matrix}0 \\ \vdots\end{matrix} & \Sigma & 0 \\ 0 & \boxed{\begin{matrix}\vdots \\ \gamma'\end{matrix}} & 0 & \Sigma' \\ \vdots & & & \end{pmatrix}$$

Note that we move the last rows and columns of Π and Π' to the first two rows and columns in $\Pi \oplus \Pi'$ and we do not apply the contraction Φ here.

3.6 The execution formula

The interpretation Π of a proof of type $\vdash [B_1, \ldots, B_m] \; A_1, \ldots A_n$ is an $(2m+n, 2m+n)$ matrix. From this we can obtain a proof of type $\vdash A_1, \ldots, A_n$ by cut elimination. This process is expressed by the *execution formula*:

$$\mathtt{Ex}(\Pi, \sigma_{m,n}) = (I_{2m+n} - \sigma_{m,n}^2)\Pi(I_{2m+n} - \sigma_{m,n}\Pi)^{-1}(I_{2m+n} - \sigma_{m,n}^2)$$

where I_{2m+n} is the unit matrix and $\sigma_{m,n}$ is given by

$$\sigma_{m,n} = \underbrace{\sigma \oplus \cdots \oplus \sigma}_{m \text{ times}} \oplus 0_n$$

or

$$\sigma_{m,n} = \begin{pmatrix} 0 & 1 & \cdots & 0 & 0 & & \\ 1 & 0 & \cdots & 0 & 0 & & \\ \vdots & \vdots & \ddots & \vdots & \vdots & & 0 \\ 0 & 0 & \cdots & 0 & 1 & & \\ 0 & 0 & \cdots & 1 & 0 & & \\ & & & & & & \\ & & 0 & & & & 0 \end{pmatrix}.$$

Acting from the left $\sigma^2_{m,n}$ is the map

$$\begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{2m-1} \\ \boldsymbol{x}_{2m} \\ \boldsymbol{x}_{2m+1} \\ \vdots \\ \boldsymbol{x}_{2m+n} \end{pmatrix} \quad \mapsto \quad \begin{pmatrix} \boldsymbol{x}_2 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_{2m} \\ \boldsymbol{x}_{2m-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

and $I_{2m+n} - \sigma^2_{m,n}$ is nothing but

$$\begin{pmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_{2m} \\ \boldsymbol{x}_{2m+1} \\ \vdots \\ \boldsymbol{x}_{2m+n} \end{pmatrix} \quad \mapsto \quad \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \boldsymbol{x}_{2m+1} \\ \vdots \\ \boldsymbol{x}_{2m+n} \end{pmatrix}.$$

Recall that if the infinite series $I + X + X^2 + \cdots$ converges for a matrix X, it is equal to the matrix $(I-X)^{-1}$. In our case the matrix $\sigma_{m,n}\Pi$ is shown to be *nilpotent*, *i.e.* $(\sigma_{m,n}\Pi)^i = 0_{2m+n}$ for some i. This in fact corresponds to the normalization of a proof. Hence the infinite series $I + \sigma_{m,n}\Pi + (\sigma_{m,n}\Pi)^2 + \cdots$ converges and

$$\Pi(I_{2m+n} - \sigma_{m,n}\Pi)^{-1} = \Pi + \Pi\sigma_{m,n}\Pi + \Pi\sigma_{m,n}\Pi\sigma_{m,n}\Pi + \cdots$$

holds.

3.7 Exponentials

The exponentials ! and ? are handled by internalizing the tensor product $\mathbb{H} \otimes \mathbb{H}'$, which is defined as the space of all linear combinations of $\boldsymbol{x} \otimes \boldsymbol{y}$ ($\boldsymbol{x} \in \mathbb{H}$ and $\boldsymbol{y} \in \mathbb{H}'$) with complex coefficients, quotiented by the equivalence relations:

$$\boldsymbol{x} \otimes (\boldsymbol{x}' + \boldsymbol{y}') = \boldsymbol{x} \otimes \boldsymbol{x}' + \boldsymbol{x} \otimes \boldsymbol{y}', \qquad (\boldsymbol{x} + \boldsymbol{y}) \otimes \boldsymbol{x}' = \boldsymbol{x} \otimes \boldsymbol{x}' + \boldsymbol{y} \otimes \boldsymbol{x}'$$
$$(\lambda \boldsymbol{x}) \otimes \boldsymbol{x}' = \boldsymbol{x} \otimes (\lambda \boldsymbol{x}') = \lambda(\boldsymbol{x} \otimes \boldsymbol{x}').$$

The tensor product $u \otimes v$ of bounded operators $u : \mathbb{H} \to \mathbb{H}$ and $v : \mathbb{H}' \to \mathbb{H}'$ is defined as the completion of

$$(u \otimes v)(\boldsymbol{x} \otimes \boldsymbol{y}) = u\boldsymbol{x} \otimes v\boldsymbol{y}.$$

In particular the tensor product $\mathbb{H} \otimes \mathbb{H}$, where $\mathbb{H} = \ell^2$, has the canonical base $(\boldsymbol{c}^{mn})$. Each $\boldsymbol{c}^{mn}$ is an infinite double sequence of 0 and 1 such that $\boldsymbol{c}^{mn}(m', n') = 1$ iff $m = m'$ and $n = n'$. We then have the isomorphism $\beta : \mathbb{H} \to \mathbb{H} \otimes \mathbb{H}$ induced from the bijection between $\mathbb{N}$ and $\mathbb{N} \times \mathbb{N}$.

We write $\langle m, n \rangle$ for the number corresponding to an ordered pair (m, n) by the bijection between $\mathbb{N}$ and $\mathbb{N} \times \mathbb{N}$. The internalized version of the associativity map between $\mathbb{H} \otimes (\mathbb{H} \otimes \mathbb{H})$ and $(\mathbb{H} \otimes \mathbb{H}) \otimes \mathbb{H}$ is then obtained as the map $t : \mathbb{H} \to \mathbb{H}$ induced by the bijection

$$\langle m, \langle n, p \rangle \rangle \mapsto \langle \langle m, n \rangle, p \rangle.$$

t^* is the inverse t^{-1} of t.

We also consider the bounded operators p and q on $\mathbb{H}$ which are induced from the maps

$$n \mapsto \langle 0, n \rangle, \qquad n \mapsto \langle 1, n \rangle$$

respectively. They are different from p and q previously defined, but they satisfy the conditions

1. $p^*q = q^*p = 0$,
2. $p^*p = q^*q = 1$.

Hence they can be used to obtain the retraction $j : \mathbb{H} \oplus \mathbb{H} \triangleleft \mathbb{H} : k$ by

$$j : \boldsymbol{x} \oplus \boldsymbol{y} \mapsto p\boldsymbol{x} + q\boldsymbol{y}, \qquad k : \boldsymbol{z} \mapsto p^*\boldsymbol{z} \oplus q^*\boldsymbol{z}.$$

Note however that j and k are not isomorphisms anymore.

When a proof of the type $\vdash [\Delta]\ ?\,\Gamma, !\,A$ is obtained from a proof of the type $\vdash [\Delta]\ ?\,\Gamma, A$ by an application of the promotion rule, the matrix changes in the following way.

$$\begin{pmatrix} \alpha & \cdots & \beta \\ \vdots & \ddots & \vdots \\ \gamma & \cdots & \delta \end{pmatrix} \quad \mapsto \quad \begin{pmatrix} t(1 \otimes \alpha)t^* & \cdots & t(1 \otimes \beta) \\ \vdots & \ddots & \vdots \\ (1 \otimes \gamma)t^* & \cdots & 1 \otimes \delta \end{pmatrix}$$

For the dereliction rule from $\vdash [\Delta]\ \Gamma, A$ to $\vdash [\Delta]\ \Gamma, ?\,A$, we use:

$$\begin{pmatrix} \alpha & \cdots & \beta \\ \vdots & \ddots & \vdots \\ \gamma & \cdots & \delta \end{pmatrix} \quad \mapsto \quad \begin{pmatrix} \alpha & \cdots & \beta p^* \\ \vdots & \ddots & \vdots \\ p\gamma & \cdots & p\delta p^* \end{pmatrix}$$

where p and q are the new p and q we just defined. For the weakening from $\vdash [\Delta]\ \Gamma$ to $\vdash [\Delta]\ \Gamma, ?\,A$, we use:

$$\begin{pmatrix} \alpha & \cdots \\ \vdots & \ddots \end{pmatrix} \quad \mapsto \quad \begin{pmatrix} \alpha & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{pmatrix}$$

For the contraction rule from $\vdash [\Delta]\ \Gamma,\ ?\,A,\ ?\,A$ to $\vdash [\Delta]\ \Gamma,\ ?\,A$, we change the matrix

$$\begin{pmatrix} \cdots\cdots & \alpha_1 & \alpha_2 \\ \vdots & \ddots & \vdots & \vdots \\ \alpha'_1 & \cdots & \gamma_1 & \gamma_2 \\ \beta'_1 & \cdots & \delta_1 & \delta_2 \end{pmatrix}$$

to the matrix:

$$\begin{pmatrix} \cdots\cdots\cdots\cdots\cdots & \alpha_1(p^* \otimes 1) + \alpha_2(q^* \otimes 1) \\ \vdots \qquad\qquad \ddots & \vdots \\ (p \otimes 1)\alpha'_1 + (q \otimes 1)\beta'_1 \quad \cdots & \begin{array}{c}(p \otimes 1)\gamma_1(p^* \otimes 1) + (p \otimes 1)\gamma_2(q^* \otimes 1) \\ +(q \otimes 1)\delta_1(p^* \otimes 1) + (q \otimes 1)\delta_2(q^* \otimes 1)\end{array} \end{pmatrix}$$

4 Working out the relationship

4.1 The category $\mathbf{Hilb}_2$

In this section we work out how the axiomatic framework captures Girard's original formulation, following Haghverdi's sketch in [4]. The category we are working with is not the category of Hilbert spaces but its subcategory $\mathbf{Hilb}_2$ defined by M. Barr.

The key observation is that there exists a monoidal contravariant functor, called ℓ^2, from the category **PInj** to the category of Hilbert spaces. A set X is mapped to the space of those complex valued functions a on X which are *square summable* in the sense that $\sum_{x \in X} |a(x)|^2$ is finite. A quasi injective function $f : X \to Y$ is mapped to the function which sends $a \in \ell^2(Y)$ to

$$\ell^2(f)(x) = \begin{cases} af(x) & \text{if } f(x) \text{ is defined} \\ 0 & \text{otherwise.} \end{cases}$$

The category $\mathbf{Hilb}_2$ is defined as the image of ℓ^2.

It is known that $\ell^2(X \times Y) \cong \ell^2 X \otimes \ell^2 Y$ and $\ell^2(X \uplus Y) \cong \ell^2 X \oplus \ell^2 Y$ where $\ell^2 X \otimes \ell^2 Y$ and $\ell^2 X \oplus \ell^2 Y$ are a tensor product and a direct sum, respectively, in the category of Hilbert spaces. In $\mathbf{Hilb}_2$ they are both tensor products, but $\ell^2 X \oplus \ell^2 Y$ is no longer a direct sum.

The trace in $\mathbf{Hilb}_2$ can simply defined from the trace in **PInj** as below.

$$\mathtt{Tr}^{\ell^2(U)}_{\ell^2(X),\ell^2(Y)}(\ell^2(f)) = \ell^2(\mathtt{Tr}^U_{X,Y}(f)).$$

4.2 The basic structure

A proof of $\vdash [C_1, \ldots, C_m]\ A_1, \ldots, A_n$ is interpreted by a $(2m+n, 2m+n)$ matrix, understood as an operator from $\mathbb{H}^{2m+n}$ to $\mathbb{H}^{2m+n}$, which can be further internalized as an operator on $\mathbb{H}$.

In particular the interpretation of an axiom $\vdash A, A^\perp$, which is σ, is nothing but the canonical morphism for symmetry in $\mathbf{Hilb}_2$ as we expected. The linear logic tensor and par are both interpreted as the direct sum in the category of Hilbert spaces.

4.3 Cut as composition in $\mathcal{G}(\mathbf{Hilb}_2)$

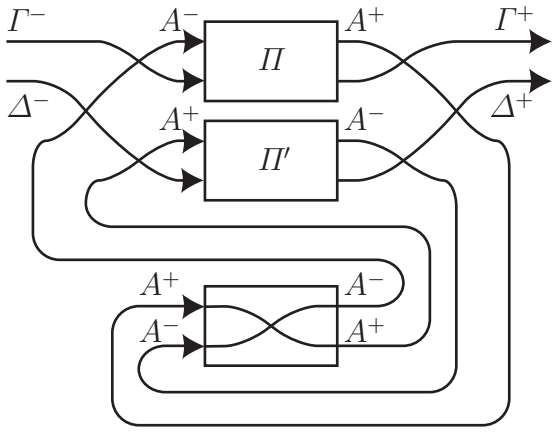
Cut in a sequent calculus corresponds to composition in a category. Consider proofs Π and Π' of sequents $\vdash A, \Gamma$ and $\vdash A^\perp, \Delta$, respectively. In our setting they are interpreted as the morphisms $\Pi : (I, I) \to (A^+, A^-) \oplus (\Gamma^+, \Gamma^-)$ and $\Pi' : (I, I) \to (A^-, A^+) \oplus (\Delta^+, \Delta^-)$ in $\mathcal{G}(\mathbf{Hilb}_2)$. Since we are in a compact closed category, we can obtain the desired morphism by the composition with the counit

$$\delta : (A^+, A^-) \otimes (A^-, A^+) \to (I, I)$$

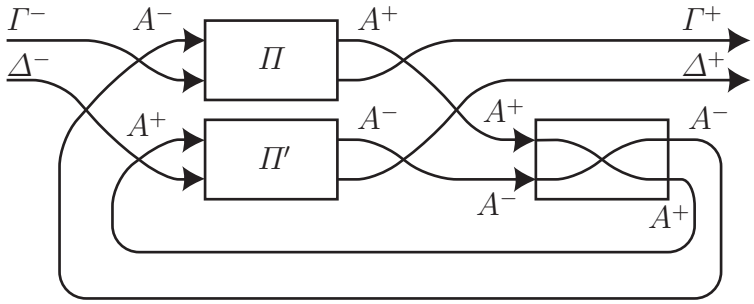
in the following way:

$$(I, I) \xrightarrow{\Pi \otimes \Pi'} (A^+, A^-) \oplus (\Gamma^+, \Gamma^-) \oplus (A^-, A^+) \oplus (\Delta^+, \Delta^-) \to$$
$$(\Gamma^+, \Gamma^-) \oplus (\Delta^+, \Delta^-) \oplus (A^+, A^-) \oplus (A^-, A^+) \xrightarrow{1 \oplus \delta} (\Gamma^+, \Gamma^-) \oplus (\Delta^+, \Delta^-).$$

This morphism is depicted by the diagram:



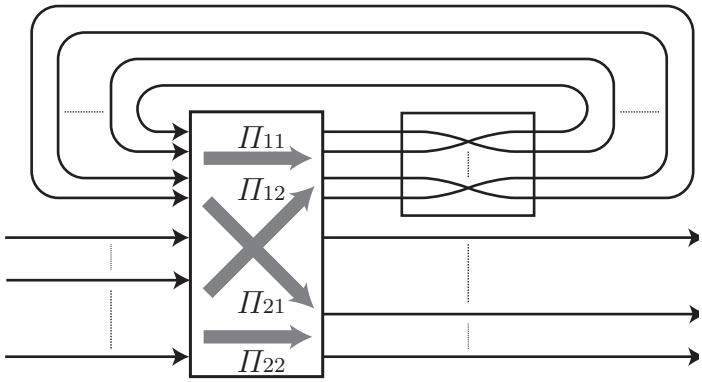
which can be simplified to the following.



Although we adopt the convention to take the trace at the right component U of the products $X \oplus U$ and $Y \oplus U$, the permutation allows us to formulate the trace at the left component U of $U \oplus X$ and $U \oplus Y$ as well. Using the latter convention we can represent the morphism $\widehat{\Pi}$

$$(I, I) \stackrel{\Pi}{\to} (A_1^+, A_1^-) \oplus (A_1^-, A_1^+) \oplus \cdots \oplus (A_m^+, A_m^-) \oplus (A_m^-, A_m^+) \oplus (\Gamma^+, \Gamma^-) \stackrel{\delta \oplus \cdots \delta \oplus 1}{\longrightarrow} (\Gamma^+, \Gamma^-)$$

by the following diagram:



where $\Pi_{11}, \Pi_{12}, \Pi_{21}$ and Π_{22} are obtained as the submatrices of the matrix Π as below:

$$\Pi = \left(\begin{array}{ccc|ccc} f_{11} & \cdots & f_{1\,2m} & f_{1\,2m+1} & \cdots & f_{1\,2m+n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ f_{2m\,1} & \cdots & f_{2m\,2m} & f_{2m\,2m+1} & \cdots & f_{2m\,2m+n} \\ \hline f_{2m+1\,1} & \cdots & f_{2m+1\,2m} & f_{2m+1\,2m+1} & \cdots & f_{2m+1\,2m+n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ f_{2m+n\,1} & \cdots & f_{2m+n\,2m} & f_{2m+n\,2m+1} & \cdots & f_{2m+n\,2m+n} \end{array}\right) = \begin{pmatrix} \Pi_{11} & \Pi_{12} \\ \Pi_{21} & \Pi_{22} \end{pmatrix}$$

$\widehat{\Pi}$ is the morphism which corresponds to the proof Π of the type $\vdash [A_1, \ldots, A_m]\ \Gamma$. Writing

$$\begin{cases} \widehat{\sigma} = \underbrace{\sigma \oplus \cdots \oplus \sigma}_{m \text{ times}} \\ \widehat{\sigma}_{m,n} = \widehat{\sigma} \oplus 1_n, \end{cases}$$

we can express $\widehat{\Pi}$ by the formula

$$\begin{aligned} \widehat{\Pi} &= \Pi_{22} + \sum_{n=0}^{\infty} \Pi_{21}(\widehat{\sigma}\Pi_{11})^n \widehat{\sigma}\Pi_{12} \\ &= \Pi_{22} + \Pi_{21}\widehat{\sigma}\Pi_{12} + \Pi_{21}(\widehat{\sigma}\Pi_{11})\widehat{\sigma}\Pi_{12} + \Pi_{21}(\widehat{\sigma}\Pi_{11})(\widehat{\sigma}\Pi_{11})\widehat{\sigma}\Pi_{12} + \cdots \\ &= \mathtt{Tr}_{\Gamma^-,\Gamma^+}^{A_1 \oplus \cdots \oplus A_m}(\widehat{\sigma}_{m,n}\Pi), \end{aligned}$$

where $\widehat{\sigma}_{m,n}\Pi$ is the matrix:

$$\widehat{\sigma}_{m,n}\Pi = \begin{pmatrix} \widehat{\sigma}\Pi_{11} & \widehat{\sigma}\Pi_{12} \\ \Pi_{21} & \Pi_{22} \end{pmatrix}$$

Furthermore by way of the projection

$$\alpha : \begin{pmatrix} x_1 \\ \vdots \\ x_{2m} \\ x_{2m+1} \\ \vdots \\ x_{2m+n} \end{pmatrix} \mapsto \begin{pmatrix} x_{2m+1} \\ \vdots \\ x_{2m+n} \end{pmatrix}$$

and the injection

$$\alpha' : \begin{pmatrix} x_{2m+1} \\ \vdots \\ x_{2m+n} \end{pmatrix} \mapsto \begin{pmatrix} 0 \\ \vdots \\ 0 \\ x_{2m+1} \\ \vdots \\ x_{2m+n} \end{pmatrix}$$

we have

$$\begin{aligned} \alpha\mathtt{Ex}(\pi, \sigma_{m,n})\alpha' &= \alpha\Pi\alpha' + \alpha\Pi\sigma_{m,n}\Pi\alpha' + \alpha\Pi\sigma_{m,n}\Pi\sigma_{m,n}\Pi\alpha' + \cdots \\ &= \Pi_{22} + \Pi_{21}\widehat{\sigma}\Pi_{12} + \Pi_{21}(\widehat{\sigma}\Pi_{11})\widehat{\sigma}\Pi_{12} + \ldots \\ &= \widehat{\Pi}. \end{aligned}$$

4.4 Exponentials from a GoI Situation

The exponential operator ! is modelled by the functor

$$\mathbb{X} \mapsto \mathbb{H} \otimes \mathbb{X}, \qquad f \mapsto 1_{\mathbb{H}} \otimes f$$

where $\mathbb{H} \otimes \mathbb{X}$ is the tensor product in Hilbert spaces.

We then need to check that the GoI Situation holds with with $T = \mathbb{H} \otimes \mathtt{Id}$ and $U = \mathbb{H}$. The retractions for a reflexive object U become

1. $j : \mathbb{H} \oplus \mathbb{H} \triangleleft \mathbb{H} : k$,
2. $l : I \triangleleft \mathbb{H} : m$,
3. $u : \mathbb{H} \otimes \mathbb{H} \triangleleft \mathbb{H} : v$.

in the present situation.

We have already seen that p and q give us the retraction $j : \mathbb{H} \oplus \mathbb{H} \triangleleft \mathbb{H} : k$ by

$$j : \boldsymbol{x} \oplus \boldsymbol{y} \mapsto p\boldsymbol{x} + q\boldsymbol{y}, \qquad k : \boldsymbol{z} \mapsto p^*\boldsymbol{z} \oplus q^*\boldsymbol{z}.$$

Recall however that there are many possibilities to choose specific p and q, and j and k may or may not become isomorphisms depending on the choice.

The additive unit object I is obtained as $\ell^2(\emptyset)$, which is indeed the singleton $\{\emptyset\}$. Clearly

$$l : \mathbf{0} \mapsto \mathbf{0}, \qquad m : \boldsymbol{x} \mapsto \mathbf{0}$$

give us the required retraction $l : I \triangleleft \mathbb{H} : m$.

For $u : \mathbb{H} \otimes \mathbb{H} \triangleleft \mathbb{H} : v$ we have already seen the existence of an isomorphism $\beta : \mathbb{H} \to \mathbb{H} \otimes \mathbb{H}$. Hence $v = \beta$ and $u = \beta^{-1}$ suffice.

The retractions for the functor T are

1. $e : TT \triangleleft T : e'$ (Comultiplication),
2. $d : \mathtt{Id} \triangleleft T : d'$ (Dereliction),
3. $c : T \oplus T \triangleleft T : c'$ (Contraction),
4. $w : \mathcal{K}_I \triangleleft T : w'$ (Weakening)

where $T : X \mapsto \mathbb{H} \otimes X,\ f \mapsto 1 \otimes f$.

The retraction $e : TT \triangleleft T : e'$ is obtained as follows.

$$e : \mathbb{H} \otimes (\mathbb{H} \otimes \mathbb{X}) \xrightarrow{a} (\mathbb{H} \otimes \mathbb{H}) \otimes \mathbb{X} \xrightarrow{\beta^{-1} \otimes 1} \mathbb{H} \otimes \mathbb{X}, \qquad e' = e^{-1}.$$

where a is a canonical associativity map.

When $\mathbb{X} = \mathbb{H}$ we the following diagram commutes:

$$\begin{array}{ccccc}
\mathbb{H} \otimes (\mathbb{H} \otimes \mathbb{H}) & \xrightarrow{a} & (\mathbb{H} \otimes \mathbb{H}) \otimes \mathbb{H} & \xrightarrow{\beta^{-1} \otimes 1} & \mathbb{H} \otimes \mathbb{H} \\
{\scriptstyle 1 \otimes \beta}\uparrow & & \downarrow{\scriptstyle \beta^{-1} \otimes 1} & & \| \\
\mathbb{H} \otimes \mathbb{H} & & \mathbb{H} \otimes \mathbb{H} & & \mathbb{H} \otimes \mathbb{H} \\
{\scriptstyle \beta}\uparrow & & \downarrow{\scriptstyle \beta^{-1}} & & \downarrow{\scriptstyle \beta^{-1}} \\
\mathbb{H} & \xrightarrow{t} & \mathbb{H} & = & \mathbb{H}
\end{array}$$

Hence t is in fact the internal version of e. Similarly t^* is the internal version of e'.

For the retraction $d : \mathtt{Id} \triangleleft T : d'$ consider the Hilbert space $\mathbb{I} = \{a \mid a : 1 \to \mathbb{C}\}$. Clearly $\mathbb{I} = \ell^2(1)$ and the isomorphism $X \times 1 \cong 1 \times X \cong X$ in **PInj** induces the isomorphisms $\ell^2(X) \otimes \mathbb{I} \cong \mathbb{I} \otimes \ell^2(X) \cong \ell^2(X)$ in $\mathbf{Hilb}_2$. We have the partial injection

$$X \longrightarrow 1 \times X \xrightarrow{(0 \mapsto 0) \times 1} \mathbb{N} \times X$$

and this induces our d'. Similarly

$$\mathbb{N} \times X \xrightarrow{(0 \mapsto 0) \times 1} 1 \times X \longrightarrow X$$

induces d. For $X = \mathbb{N}$ the internal versions of d and d' coincide with our new p and p^* respectively, since the following diagrams commute:

$$\begin{array}{ccc}
\mathbb{H} & \xrightarrow{d} & \mathbb{H} \otimes \mathbb{H} \\
\| & & \downarrow{\scriptstyle \beta^{-1}} \\
\mathbb{H} & \xrightarrow{p} & \mathbb{H}
\end{array}
\qquad
\begin{array}{ccc}
\mathbb{H} \otimes \mathbb{H} & \xrightarrow{d'} & \mathbb{H} \\
{\scriptstyle \beta}\uparrow & & \| \\
\mathbb{H} & \xrightarrow{p^*} & \mathbb{H}
\end{array}$$

The retraction $c : T \oplus T \triangleleft T : c'$ is obtained through the isomorphism

$$(\ell^2(X) \oplus \ell^2(Y)) \otimes \ell^2(Z) \cong (\ell^2(X) \otimes \ell^2(Z)) \oplus (\ell^2(Y) \otimes \ell^2(Z))$$

in $\mathbf{Hilb}_2$ induced from the isomorphism $(X \uplus Y) \times Z \cong (X \times Z) \uplus (Y \times Z)$ in **PInj**. The map c is

$$(\mathbb{H} \otimes \mathbb{X}) \oplus (\mathbb{H} \otimes \mathbb{X}) \longrightarrow (\mathbb{H} \oplus \mathbb{H}) \otimes \mathbb{X} \xrightarrow{j \otimes 1} \mathbb{H} \otimes \mathbb{X},$$

and c' is

$$\mathbb{H} \otimes \mathbb{X} \xrightarrow{k \otimes 1} (\mathbb{H} \oplus \mathbb{H}) \otimes \mathbb{X} \longrightarrow (\mathbb{H} \otimes \mathbb{X}) \oplus (\mathbb{H} \otimes \mathbb{X}).$$

We then have

$$\begin{aligned}
c((\boldsymbol{x} \otimes \boldsymbol{z}) \oplus (\boldsymbol{y} \otimes \boldsymbol{w})) &= (j \otimes 1)((\boldsymbol{x} \oplus \boldsymbol{0}) \otimes \boldsymbol{z} + (\boldsymbol{0} \oplus \boldsymbol{y}) \otimes \boldsymbol{w}) \\
&= p\boldsymbol{x} \otimes \boldsymbol{z} + q\boldsymbol{y} \otimes \boldsymbol{w} \\
&= (p \otimes 1)(\boldsymbol{x} \otimes \boldsymbol{z}) + (q \otimes 1)(\boldsymbol{y} \otimes \boldsymbol{w})
\end{aligned}$$

and

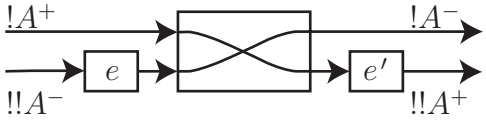
$$\begin{aligned} c'(\boldsymbol{x} \otimes \boldsymbol{y}) &= (p^* \boldsymbol{x} \otimes \boldsymbol{y}) \oplus (q^* \boldsymbol{x} \otimes \boldsymbol{y}) \\ &= (p^* \otimes 1)(\boldsymbol{x} \otimes \boldsymbol{y}) \oplus (q^* \otimes 1)(\boldsymbol{x} \otimes \boldsymbol{y}). \end{aligned}$$

The retraction $w : \mathcal{K}_I \triangleleft T : w'$ is obtained by

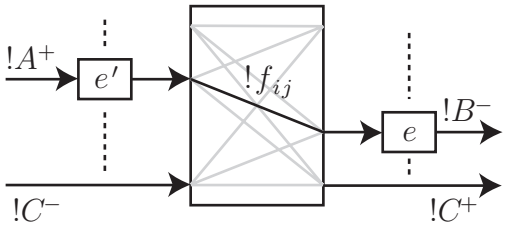
$$w : \mathbf{0} \mapsto \mathbf{0}, \qquad w' : \boldsymbol{x} \otimes \boldsymbol{y} \mapsto \mathbf{0}.$$

Those retraction maps give the promotion, dereliction, contraction and weakening maps in $\mathcal{G}(\mathbf{Hilb}_2)$.

The promotion map $!\,(A^+, A^-) \to !\,!\,(A^+, A^-)$ is the one depicted by the diagram:



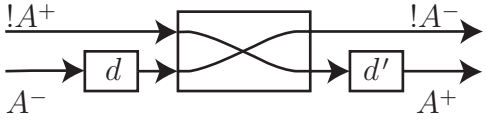
The interpretation of a proof obtained by an application of the promotion rule is given by the composition with this morphism, and the result can be depicted as follows:



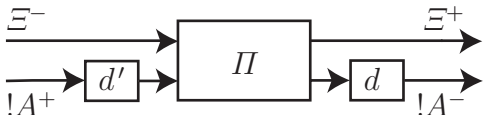
Since the internalized versions of e and e' are t and t^*, respectively, this in fact gives the matrix:

$$\begin{pmatrix} t(1 \otimes \alpha)t^* & \cdots & t(1 \otimes \beta) \\ \vdots & \ddots & \vdots \\ (1 \otimes \gamma)t^* & \cdots & 1 \otimes \delta \end{pmatrix}$$

The dereliction map $!\,(A^+, A^-) \to (A^+, A^-)$ is:



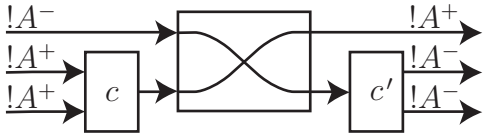
and the composition with this map yields:



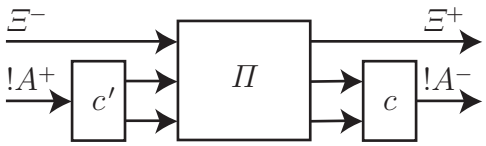
The internalized versions of d and d' are p and p^*, respectively. Hence we have the matrix:

$$\begin{pmatrix} \alpha & \cdots & \beta p^* \\ \vdots & \ddots & \vdots \\ p\gamma & \cdots & p\delta p^* \end{pmatrix}$$

The contraction map $!(A^+, A^-) \to !(A^+, A^-) \oplus !(A^+, A^-)$ is:



and the composition gives:



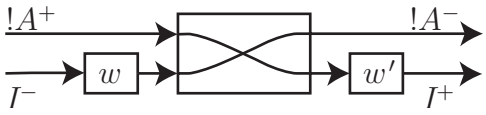
Since we are writing the direct sum $\boldsymbol{x} \oplus \boldsymbol{y}$ as a column vector, $c : \boldsymbol{x} \oplus \boldsymbol{y} \mapsto (p \otimes 1)\boldsymbol{x} + (q \otimes 1)\boldsymbol{y}$ and $c' : \boldsymbol{x} \mapsto (p^* \otimes 1)\boldsymbol{x} \oplus (q^* \otimes 1)\boldsymbol{x}$ are represented by the matrices:

$$c = \begin{pmatrix} p \otimes 1 & q \otimes 1 \end{pmatrix}, \qquad c' = \begin{pmatrix} p^* \otimes 1 \\ q^* \otimes 1 \end{pmatrix}$$

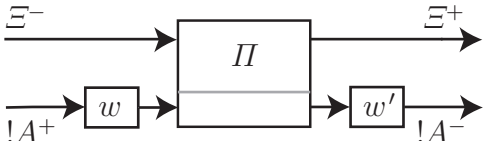
Hence the proof obtained by an application of the contraction rule is represented by the following matrix as we expected:

$$\begin{pmatrix} \cdots\cdots\cdots\cdots\cdots\cdots & \alpha_1(p^* \otimes 1) + \alpha_2(q^* \otimes 1) \\ \vdots \qquad\qquad \ddots & \vdots \\ (p \otimes 1)\alpha_1' + (q \otimes 1)\beta_1' \quad \cdots & \begin{array}{c}(p \otimes 1)\gamma_1(p^* \otimes 1) + (p \otimes 1)\gamma_2(q^* \otimes 1) \\ +(q \otimes 1)\delta_1(p^* \otimes 1) + (q \otimes 1)\delta_2(q^* \otimes 1)\end{array} \end{pmatrix}$$

The weakening map $!(A^+, A^-) \to (I, I)$ is:



and the interpretation of a proof is:



whose matrix is

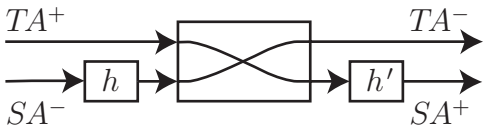
$$\begin{pmatrix} \alpha & \cdots \\ \vdots & \ddots \end{pmatrix} \quad \mapsto \quad \begin{pmatrix} \alpha & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{pmatrix}$$

since w and w' are the constant zero operators.

5 Discussion on the naturality

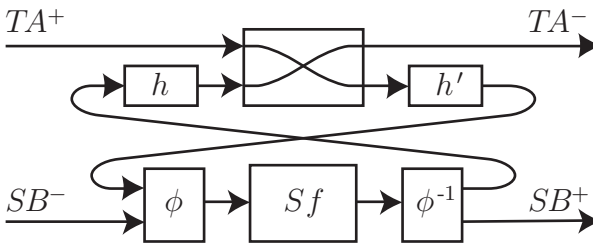
It has been shown in [1] that the promotion, dereliction, contraction and weakening maps in $\mathcal{G}(\mathbb{C})$ become natural transformations iff the corresponding retraction maps are isomorphisms. The argument can be easily generalized and we now state and prove its generalized version.

Let (S, ϕ, ϕ_I) and (T, ψ, ψ_I) be monoidal functors on $\mathbb{C}$. Suppose that we have a family of retractions $h : SA \triangleleft TA : h'$ which is a monoidal natural transformation from S to T. Consider a family of morphisms in $\mathcal{G}(\mathbb{C})$ which have the form:

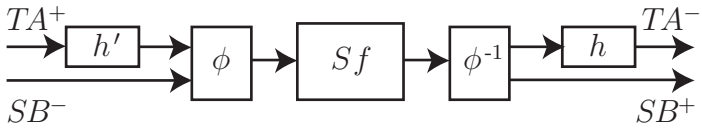


Such a family of morphisms becomes a natural transformation in $\mathcal{G}(\mathbb{C})$ iff $hh' = 1_{TA}$ for all objects A in $\mathbb{C}$.

We give a proof as a sequence of diagrams. When we precompose such a morphism to another morphism Sf we obtain the morphism represented by the diagram

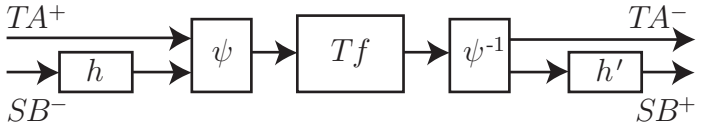


where $\phi : SA^+ \otimes SB^- \to S(A^+ \otimes B^-)$ is the isomorphism provided by the monoidal functor S. This diagram can be simplified to:



(1)

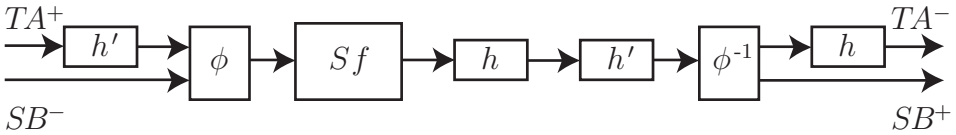
Similarly when we postcompose the morphism to Tf we obtain:



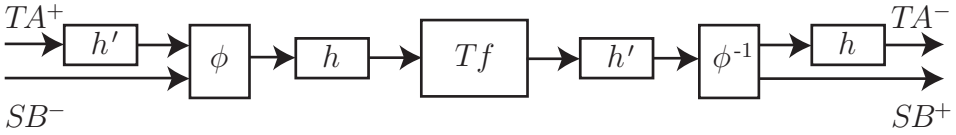
(2)

where $\psi : TA^+ \otimes TB^- \to T(A^+ \otimes B^-)$ is the isomorphism provided by T.

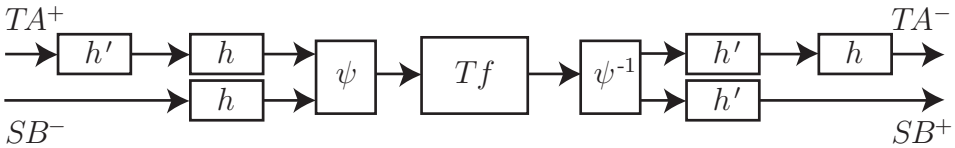
The naturality is the claim that the diagrams (1) and (2) always represent the same morphism. To see when it holds, we first insert $h'h$, which is an identity since (h, h') is a retraction, in the diagram (1) as follows:



The naturality of h then allows us to transform it to the below:



Since h and h' are monoidal natural transformations, we can then make the diagram (1) in the following form:



(3)

If $hh' = 1_{TA}$- the diagram (3) immediately becomes the same as the diagram (2) and the naturality holds. For the other direction let $f = 1_{A\otimes B}$. Then (2) becomes the map $1_{TA} \otimes 1_{SB}$ and (3) becomes $hh' \otimes 1_{SB}$. If the naturality holds we have $1_{TA} \otimes 1_{SB} = hh' \otimes 1_{SB}$ for any objects A and B in $\mathbb{C}$. In particular we can choose $I = B$. Then $SB = I$ and the naturality of the isomorphisms $\lambda_A : A \otimes I \to A$ makes the following diagrams commute.

$$\begin{array}{ccc} TA\otimes I & \xleftarrow{\lambda^{-1}_{TA}} & TA \\ {\scriptstyle 1_{TA}\otimes 1_I}\downarrow & & \downarrow{\scriptstyle 1_{TA}} \\ TA\otimes I & \xrightarrow[\lambda_{TA}]{} & TA \end{array} \qquad \begin{array}{ccc} TA\otimes I & \xleftarrow{\lambda^{-1}_{TA}} & TA \\ {\scriptstyle hh'\otimes 1_I}\downarrow & & \downarrow{\scriptstyle hh'} \\ TA\otimes I & \xrightarrow[\lambda_{TA}]{} & TA \end{array}$$

Hence $1_{TA} \otimes 1_I = hh' \otimes 1_I$ implies $1_{TA} = hh'$.

The naturality of the promotion, dereliction, contraction and weakening maps is necessary to make the Geometry of Interaction interpretation sound for the full cut-elimination. In Girard's original formulation, the soundness for the cases involving exponentials is obtained only when the context formulas are empty. This is due to the fact that the maps for exponentials are only pointwise natural in $\mathcal{G}(\mathbf{Hilb}_2)$.

The result stated in this section, however, tells us that we should not expect more than the pointwise naturality in this setting. We can make the retractions for contraction and promotion isomorphic, but the retractions for dereliction and weakening should not be isomorphic. As shown in [4] and [1], the pointwise naturality suffices to construct a linear combinatory algebra, which is good for the analysis of computation. If the purpose of the Geometry of Interaction is the analysis of the cut-elimination or the analysis of classical logic, however, the situation is not quite satisfactory.

The machinery of the Geometry of Interaction, either in its original formulation or the axiomatic framework, is very much symmetric. It seems however that the exponential rules, in particular dereliction and weakening, require us to re-introduce asymmetry in one way or another.

References

[1] S. Abramsky, E. Haghverdi and P. Scott, "Geometry of Interaction and linear combinatory algebra," *Mathematical Structures in Computer Science* (2002), *vol.* 12, *pp.* 625-665.

[2] S. Abramsky and R. Jagadeesan, "New Foundations for the Geometry of Interaction," *Proceedings 7th Annual IEEE Symp. on Logic in Computer Science, LICS'92*, Santa Cruz, CA, USA, 22–25 June 1992.

[3] J. Y. Girard, Geometry of Interaction I: Interpretation of system F, *Logic Colloquium '88*, Ferro, Bonotto, Valentini and Zarnado (Editors), Elsevier (North-Holland), 1989, pp. 221-260.

[4] E. Haghverdi, "Unique decomposition categories, Geometry of Interaction and combinatory logic," *Mathematical Structures in Computer Science* (2000), *vol.* 10, *pp.* 205-231.