An Axiomatic System of Parametricity

Takeuti, Izumi

Faculty of Science, Tokyo Metropolitan University,
Minami-Osawa, Hatiozi, Tokyo 192-03 Japan

Abstract. Plotkin and Abadi have proposed a syntactic system for parametricity on a second order predicate logic. This paper shows three theorems about that system. The first is consistency of the system, which is proved by the method of relativization. The second is that polyadic parametricities of recursive types are equivalent to each other. The third is that the theory of parametricity for recursive types is self-realizable. As a corollary of the third theorem, the theory of parametricity for recursive types satisfies the term extraction property.

1 Introduction

1.1 Related Works

There are many works on parametric polymorphism. Interest in parametricity arose in two areas, model theory and formal logic. The original meaning of parametricity was as a notion for models of polymorphic calculi, such as System F, the second order lambda calculus studied by Girard [4]. More recently, many researchers have shown interest in parametricity in formal logic. This paper discusses parametricity in formal logic.

In the sense of Reynolds [8], parametricity is the property that if X is of universal type $\Pi\tau.F[\tau]$ then $XA(F[R])XB$ holds for all types A and B and for all relations R between A and B. In the original meaning, parametricity was a notion for universal types. This paper regards the parametricity notion as extended into the notion for all types, just as Plotkin and Abadi [7] do. The parametricity notion for functional types is called naturality by Hasegawa [5, 6].

Abadi, Cardelli and Curien [1] propose System R, a formal system for parametricity. It is a logical system for binary relations between terms of System F. Because it deals only with binary predicates, its expressive power is rather weak.

Hasegawa [5, 6] makes a category-like model for parametricity, which is called a relation-frame. It is a suitable model for System R. The informal logic which is used by Hasegawa [5, 6] is second order predicate logic, and formalizing his informal logic, we can obtain a formal treatment of parametricity in formal second order predicate logic. Hasegawa [5, 6] does not show the existence of the parametric relation-frame for the whole types of System F, although he shows the existence of the parametric relation-frame for some particular types of System F, such as pair types, sum types, natural numbers, and so forth.

Bellucci, Abadi, and Curien [3] construct a model of System R from a partial equivalence relation model of System F. The existence of the model gives a model theoretic proof of the consistency of the theory of parametricity.

Plotkin and Abadi [7] formalize parametricity on a second order predicate logic. In their system, the basic logic is not specialized to the traitment for parametricity, and the axioms implement the parametricity. They also discuss polyadic parametricities. The original notion of parametricity was a notion for binary predicates. Tertiary parametricity for a universal type is the property that if X is of universal type $\Pi\tau.F[\tau]$ then it holds that $F[R](XA)(XB)(XC)$ for all types A, B and C and for all predicates R for terms of type A, B and C. Quaternary parametricity and n-ary parametricity for general n are defined in the similar ways. Plotkin and Abadi propose a problem: Are the theory of binary parametricity and the theory of tertiary parametricity equivalent? How about for quaternary, or n-ary for general n? We solve a part of this problem in this paper, which is an extension of their work.

1.2 Outline

We have three main results in this paper. The first is the consistency of the theory for parametricity. This theory is given by an axiomatic system of parametricity on a basic logic $\mathbf{L}$. The basic logic $\mathbf{L}$ is second order predicate logic over terms of System F. The consistency has been already shown by the model theoretic proof in [3], but there has been no syntactic proof of consistency to the best of the author's knowledge. Our proof is a direct proof, which uses relativization. It is given in Section 3.

The second result is that the theories of polyadic parametricities are equivalent to each others as long as types are restricted to recursive types. Recursive types are defined as

$$\mathbf{T} ::= \mathbf{1} \mid \tau \mid \mathbf{T} \to \mathbf{T} \mid \mathbf{T} \times \mathbf{T} \mid \mathbf{T} + \mathbf{T} \mid \mu\tau.\mathbf{T} \mid \nu\tau.\mathbf{T}.$$

In the definition, the constructors are encoded as the types of the system F. In $\mu\tau.T$ and $\nu\tau.T$, τ occurs only positively. We define an axiom $\mathbf{Par}_n(T)$ which asserts the n-ary parametricity of a type T, and an axiomatic system $\mathbf{Par}_n(\mathbf{T})$ as the set of axioms $\mathbf{Par}_n(T)$ for all the types T in $\mathbf{T}$. We prove that the theories of $\mathbf{Par}_m(\mathbf{T})$ and of $\mathbf{Par}_n(\mathbf{T})$ are equal to each others for arbitrary numbers $m, n \geq 2$. It is a partial answer to the problem of Plotkin and Abadi [7]. This is given in Section 4.

The third result is the realizability interpretation of the theory of the axiomatic system $\mathbf{Par}(\mathbf{T})$, where we write simply $\mathbf{Par}(\mathbf{T})$ for $\mathbf{Par}_2(\mathbf{T})$. We also write $\mathbf{Par}(T)$ for $\mathbf{Par}_2(T)$. We define a realizability interpretation of the basic logic $\mathbf{L}$ into $\mathbf{L}$ itself. and we prove that $\mathbf{Par}(\mathbf{T})$ is self-realisable, that is, for each $\mathbf{Par}(T)$ in $\mathbf{Par}(\mathbf{T})$, there is a term R such that $\mathbf{Par}(\mathbf{T})$ proves that R realizes $\mathbf{Par}(T)$. As the consequence of it, for each theorem P of $\mathbf{Par}(\mathbf{T})$, there is a term R such that $\mathbf{Par}(\mathbf{T})$ proves that R realizes P. This result infers the term extraction property for the axiomatic system $\mathbf{Par}(\mathbf{T})$. It is given in Section 5.

In Section 2 of this paper, we first define a logical system $\mathbf{L}$, and then use it as the basic logic for the axiomatic system of parametricity. $\mathbf{L}$ is a second order predicate logic on terms of the system F. It is slightly modified from the system

of the basic logic of Plotkin and Abadi [7]. Their system and **L** are logically equivalent to each other, that is, there are interpretations of their system into **L**, and of **L** into their system, which preserve provabilities and logical meanings. But there are two main differences between our logic and their logic. One is that **L** has n-ary predicate variables for an arbitrary number n, while their system restricts predicate variables to be binary. This is not an essential difference, because both systems have pairing, so polyadic predicates can be encoded into binary predicates. The other difference is that our logic does not have equality as a primitive symbol; it is defined as Leibniz equality in our logic, while their logic has equality as a primitive symbol. This difference is used essentially in relativizing predicates in the proof of the consistency. From another point of view, **L** is a subsystem of λP2 in the λ-cube which is studied by Barendregt [2], though **L** distinguishes types and formulae but λP2 does not distinguish them. The terms and types of **L** are those of System F. For example, $\forall x.\phi x \to \psi x$ is a formula, but not a type of **L**. But λP2 has $\Pi x.\phi x \to \psi x$ as a type, and $y^{\Pi x.\phi x \to \psi x}$ as a variable.

Continuing Section 2, we next define several notions in order to define the axiomatic system **Par** for parametricity. First, conformity for a type T is defined by induction on the construction of T. We use the notation $\langle T\rangle\{\Phi_1/\tau_1, \ldots, \Phi_i/\tau_i\}$ to denote the notion of conformity for the type T. An outline of the definition of conformity is the following:

– $\langle\tau\rangle\{\ldots, \Phi/\tau, \ldots\} :\equiv \Phi$ for a type variable τ.
– $\langle T \to U\rangle\{\ldots\}xy$ iff for all v and w, if $\langle T\rangle\{\ldots\}vw$ then $\langle U\rangle\{\ldots\}(xv)(yw)$.
– $\langle \Pi\tau.T\rangle\{\ldots\}xy$ iff $\langle T\rangle\{\phi/\tau, \ldots\}(x\alpha)(y\beta)$ for all types α and β, and for all predicates ϕ.

If the assignment $\{\ldots\}$ does not assign a predicate to a type variable τ, then the assignment is regarded as assigning equality to τ. We write $\langle T\rangle^=$ for $\langle T\rangle\{\}$. We say that x^T and y^T conform to each other when $\langle T\rangle^= xy$ holds. x^T is self-conforming when x conforms to x itself. $\phi^{T\to *}$ is extensional when ϕx is equivalent to ϕy if x and y conform to each other. Conformity of a type T is a property that all the terms of the type T are self-conforming. Extensionality of a type T is the property that all the predicates of predicate type $T \to *$ are extensional. Parametricity of T is the conjunction of conformity and extensionality of T. We write $\mathbf{Par}(T)$ for parametricity of T. **Par** is an axiomatic system. Its intuitive meaning is the conjunction of $\mathbf{Par}(T)$ for all types T. The intuitive meaning of parametricity is that x^T and y^T are equal to each other if and only if they conforms to each other. The axiomatic system **Par** is logically equivalent to the axiomatic system in [7]. As compared to [7] however, we divide the notion of parametricity into two notions: conformity and extensionality. The axioms in [7] only say that conformity and equality are equivalent to each other. The two notions, conformity and extensionality, are the essential points in relativizing predicates as well as Leibniz equality.

In Section 3, we show that the axiomatic system **Par** is consistent. That is the first main result of this paper. We prove consistency by the method of relativization. Because equality is defined as Leibniz equality, an equation works

as a statement for predicates as well as a statement for terms. By using this point, we define the relativization. It relativizes terms by self-conformity, and relativizes predicates by extensionality. The relativization is sound and complete for the theory of **Par**, that is, a formula P is provable in the axiomatic system **Par** if and only if the relativized formula $rel(P)$ is provable in **L**.

In Section 4, we discuss the polyadic parametricity, and give a partial answer to the problem of Plotkin and Abadi [7]. Our answer is that the polyadic parametricities are equivalent to each other as long as the types are restricted to recursive types. That is the second main result of this paper. They [7] define n-ary parametricity: n-ary parametricity of a type T is that $\langle T\rangle^{=}_{n} x_1 \dots x_n$ is equivalent to $x_1 = x_2 = \cdots = x_n$, where $\langle T\rangle^{=}_{n}$ is an n-ary conformity notation defined in a similar way to the binary. We write $\mathbf{Par}_n(T)$ for n-ary parametricity of a type T, and $\mathbf{Par}_n(\mathbf{T})$ for the set of axioms $\mathbf{Par}_n(T)$ for all the types T in **T**. The result is that the theories of $\mathbf{Par}_m(\mathbf{T})$ and of $\mathbf{Par}_n(\mathbf{T})$ are equal to each other for arbitrary numbers $m, n \geq 2$. In other words, for any T in **T**, $\mathbf{Par}_n(T)$ is provable by $\mathbf{Par}_m(\mathbf{T})$.

In order to prove the result, we show that $\langle T\rangle^{=}_{m}$ and $\langle T\rangle^{=}_{n}$ are equivalent under the theory of $\mathbf{Par}_m(\mathbf{T})$. From this fact, we obtain $\langle T\rangle^{=}_{n}$ is equivalent to equality, because $\mathbf{Par}_m(T)$ implies that $\langle T\rangle_m\{\}$ is equivalent to equality. But we cannot compare $\langle T\rangle^{=}_{m}$ and $\langle T\rangle^{=}_{n}$ directly, because $\langle T\rangle^{=}_{m}$ is an m-ary predicate and $\langle T\rangle^{=}_{n}$ is n-ary. Thus we define filters $\mathbf{C}^m_n[\,]$ and $\mathbf{D}^m_n[\,]$ which translate an n-ary predicate into an m-ary predicate. The intuitive meaning of $\mathbf{C}^m_n[\Phi]$ is an internal partial equivalence relation in the domain of Φ, and the intuitive meaning of $\mathbf{D}^m_n[\Phi]$ is the symmetric transitive closure of Φ. The crucial lemma is that $\mathbf{Par}_l(\mathbf{T})$ proves

$$\forall x_1 \dots x_k . \mathbf{D}^k_m[\langle T\rangle_m\{\dots\}]x_1 \dots x_k \supset \mathbf{C}^k_n[\langle T\rangle_n\{\dots\}]x_1 \dots x_k$$

for arbitrary numbers $k, l, m, n \geq 2$. We have also that $\langle T\rangle^{=}_{m}$ implies $\mathbf{D}^m_m[\langle T\rangle^{=}_{m}]$ and $\mathbf{C}^n_n[\langle T\rangle^{=}_{n}]$ implies $\langle T\rangle^{=}_{n}$. Thus from this crucial lemma, we obtain the equivalence of $\langle T\rangle^{=}_{m}$ and $\langle T\rangle^{=}_{n}$.

The proof of the lemma is done by induction on the construction of the type T in **T**. In the step for functional types $\to$, the symmetricity and the transitivity of $\mathbf{C}[\,]$ and $\mathbf{D}[\,]$ do essential work. In the steps for $\mu\tau.T$ and $\nu\tau.T$ of the proof, we use the following fixed point lemma. The fixed point lemma involves fixed point property, whose intuitive meaning is that the conformity of fixed point types is equivalent to the fixed point predicate of conformity. The fixed point lemma says that $\mathbf{Par}_m(\mathbf{T})$ proves the fixed point property for n-ary predicates for any arities m and n independent of each other. This lemma is crucial in the induction steps for $\mu\tau.T$ and $\nu\tau.T$. Only this lemma for $m = n$ appears in [1]. This paper extends their lemma.

In Section 5, we give a realizability interpretation to the theory of $\mathbf{Par}(\mathbf{T})$. First we define a realizability interpretation of the basic logic **L** into **L** itself. This interpretation is sound and complete in the following sense. Soundness means that for each theorem P of **L**, there is a term R such that **L** derives that R realizes P; completeness means that **L** proves that if R realizes P, then P holds.

Next we show that the axiomatic system $\mathbf{Par}(\mathbf{T})$ is self-realizable, that is, there is a term R_T for each axiom $\mathbf{Par}(T) \in \mathbf{Par}(\mathbf{T})$ such that $\mathbf{Par}(\mathbf{T})$ proves that R_T realizes $\mathbf{Par}(T)$. In order to prove the self-realizability of $\mathbf{Par}(\mathbf{T})$, we define a notion $\langle T \rangle^{\models}$. The intuitive meaning of $\langle T \rangle^{\models} rxy$ is that r realizes $\langle T \rangle^{=} xy$. We show the crucial lemma which says that $\mathbf{Par}(\mathbf{T})$ proves that $\langle T \rangle^{=}_3 xyz$ implies $\langle T \rangle^{\models} (K_T x)yz$ for some term K_T, for each type $T \in \mathbf{T}$. The proof of this crucial lemma is done by induction on construction of $T \in \mathbf{T}$.

2 Logic for Parametricity

In this section we define a logical system $\mathbf{L}$, and a set of axioms for parametricity **Par** on $\mathbf{L}$. In order to define **Par**, we introduce the notion of conformity and extensionality in advance.

2.1 The Basic Logic

First we define the basic logic $\mathbf{L}$, on which the theory of parametricity is founded. It is the second order logic on the second order typed lambda calculus, which is studied as System F by Girard [4].

Definition 1. Here we define the second order logic on the second order typed lambda calculus, and we call it $\mathbf{L}$.

The language of $\mathbf{L}$ is defined as follows:

– *Types* and *terms* are those of System F, that is:

$$T ::= \tau \mid T \to T \mid \Pi\tau.T \quad \text{and} \quad X ::= x \mid \lambda x^T.X \mid XX \mid \Lambda\tau.X \mid XT$$

where the application is done only when the types of the two terms involved are suitable. All the variables are explicitly typed, thus all the subterms are typed. But we often omit the superfixes which indicate types.

– *Predicate types*, *predicates*, and *formulae* are constructed as:

$$\begin{array}{l}\text{Predicate types: } A ::= * \mid T \to A \\ \text{Predicates: } \Phi ::= \phi^A \mid (\lambda x^T.\Phi^A)^{T\to A} \mid (\Phi^{T\to A} X^T)^A \mid P^* \\ \text{Formulae: } P ::= \Phi^* \mid P \supset P \mid \forall\tau.P \mid \forall x.P \mid \forall\phi.P\end{array}$$

All the variables and the predicate variables are explicitly typed, thus all the subexpressions are typed. But we often omit the superfixes which indicate types. Formulae are the predicates of predicate type $*$. We use the letters $x, y, \ldots$ for variables, $X, Y, \ldots$ for terms, $\tau, \upsilon, \alpha, \beta, \ldots$ for type variables, and $T, U, A, B, \ldots$ for types. A and B are also used for predicate types. $P, Q, \ldots$ are for formulae, $\phi, \psi, \ldots$ for predicate variables, and Φ, Ψ for predicates.

The inference rules are given below. In the expression "$E, \Gamma \vdash P$", E is an environment which declares types of variables such as $\{x : T, y : U, \ldots\}$, Γ is an assumption, which is a finite set of formulae, and P is a formula. An expression Γ, Δ stands for $\Gamma \cup \Delta$, and Γ, P stands for $\Gamma \cup \{P\}$, and so forth.

When X is a term or a predicate, $Env(X)$ is the least environment which is needed for constructing X. When Γ is a set $\{P_1, P_2, \ldots\}$, $Env(\Gamma) :\equiv \bigcup_i Env(P_i)$.

– Structural Rules:

$$\frac{}{E, \Gamma, P \vdash P}, \qquad \frac{E, \Gamma \vdash P}{E, \Gamma, Q \vdash P}, \qquad \frac{E, \Gamma \vdash P}{E, x : T, \Gamma \vdash P},$$

where $Env(P) \subset E$, $Env(Q) \subset E$ and $Env(\Gamma) \subset E$.

– Implication Introduction:

$$\frac{E, \Gamma, P \vdash Q}{E, \Gamma \vdash P \supset Q}$$

– Implication Elimination:

$$\frac{E, \Gamma \vdash P \supset Q \quad E, \Gamma \vdash P}{E, \Gamma, \vdash Q}$$

– Term Quantification Introduction:

$$\frac{E, x : T, \Gamma, \vdash P}{\Gamma \vdash \forall x.P}$$

where x does not occur free in Γ.

– Term Quantification Elimination:

$$\frac{E, \Gamma \vdash \forall x^T.P}{E, \Gamma \vdash P[X^T/x]}$$

where $Env(X) \subset E$.

– Predicate Quantification Introduction:

$$\frac{E, \Gamma \vdash P}{E, \Gamma \vdash \forall \phi.P}$$

where ϕ does not occur free in Γ.

– Predicate Quantification Elimination:

$$\frac{E, \Gamma \vdash \forall \phi^A.P}{E, \Gamma \vdash P[\Phi^A/\phi]}$$

where $Env(\Phi) \subset E$.

– Type Quantification Introduction:

$$\frac{E, \Gamma \vdash P}{E, \Gamma \vdash \forall \tau.P}$$

where τ does not occur free in Γ, and moreover, τ does not occur free in the types of free variables, nor the predicate types of free predicate variables in P.

– Type Quantification Elimination:

$$\frac{E, \Gamma \vdash \forall \tau.P}{E, \Gamma \vdash P[T/\tau]}$$

– β-Conversion:

$$\frac{E, \Gamma \vdash P}{E, \Gamma \vdash Q}$$

where there is an one-step β-reduction from P to Q, or from Q to P; and $Env(Q) \subset E$. Both $(\lambda x.X)Y$ and $(\lambda x.\Phi)X$ are regarded to be β-redexes.

If E or Γ is a infinite set, then $E, \Gamma \vdash P$ means that there are finite subsets $E' \subset$ and $\Gamma' \subset \Gamma$ such that $E', \Gamma' \vdash P$. If $\vdash P$ is derivable, then P is called a *theorem* of $\mathbf{L}$, and we write $\mathbf{L} \vdash P$. The set of all theorems of $\mathbf{L}$ is called the *theory* of $\mathbf{L}$. Let $\mathbf{A}$ be a set of formulae regarded as axioms. If P is derivable from $\mathbf{A}$, then P is called a *theorem* of $\mathbf{A}$, and the set of all formulae P such that $\mathbf{A} \vdash P$ is called the *theory* of $\mathbf{A}$.

2.2 Preliminaries

Here are some important propositions and definitions for **L**. The propositions say that **L** is situated between $\lambda 2$ and λP2 in Lambda-Cube, which is studied by Barendregt [2].

Proposition 2. **L** *is consistent.*

Proof. By forgetting all the terms and all the inferences of term quantification, a derivation of **L** is translated into a derivation of second order propositional logic. Hence, the consistency of **L** is reduced to the consistency of second order propositional logic. □

This proof mentions a projection which translates derivations of **L** onto those of second order propositional logic.

Proposition 3. *There is a sound interpretation of* **L** *into* $\lambda P2$.

This proof uses an embedding which translates derivations of **L** into terms of λP2. By regarding a derivation of **L** as a term of λP2, and normalizing the term, we get the disjunction property and the term extracting property. Later we shall prove them again by using the realizability interpretation.

Note that the converse does not hold. **L** does not derive $x : \Pi\tau.\tau \vdash \forall\phi.\phi$, although λP2 derives its interpretation, which is $x : \Pi\tau : *.\tau \vdash x : \Pi\tau : *.\tau$.

Now we define some notations.

Definition 4.

$\bot :\equiv \forall\phi^*.\phi$, $P\wedge Q :\equiv \forall\phi^*.(P\supset Q\supset\phi)\supset\phi$, $P\vee Q :\equiv \forall\phi^*.(P\supset\phi)\supset(Q\supset\phi)\supset\phi$,
$\exists x.P :\equiv \forall\phi^*.(\forall x.P\supset\phi)\supset\phi$, $P\supset\subset Q :\equiv (P\supset Q)\wedge(Q\supset P)$,
$X =_T Y :\equiv \forall\phi^{T\to *}.\phi X\supset\phi Y$, $\mathcal{E}q_T :\equiv \lambda xy.(x =_T y)$,

We often omit the suffix T in $=_T$ and $\mathcal{E}q_T$.

In this paper, $\equiv$ denotes identity of expressions and $=_\beta$ denotes β-equality of expressions. We write $\{\dots\}$ for assignments and $[\dots]$ for substitutions.

2.3 Conformity, Extensionality and Parametricity

Now we define the conformity notation for **L**.

Definition 5. Let T be a type and $\tau_1,\dots,\tau_n$ be type variables. Let $\Phi_1,\dots,\Phi_n$ be binary predicates whose predicate types are $A_1\to B_1\to *,\dots,A_n\to B_n\to *$ respectively. Let T_1, T_2 be types such that $T_1\equiv T[A_1/\tau_1,\dots,A_n/\tau_n]$ and $T_2\equiv T[B_1/\tau_1,\dots,B_n/\tau_n]$.

Then $\langle T\rangle\{\Phi_1/\tau_1,\dots,\Phi_n/\tau_n\}$ is a binary predicate, whose predicate type is $T_1\to T_2\to *$. It is called the *conformity notation*. It is defined by the induction on T as below. In the definition, $\Theta :\equiv \{\Phi_1/\tau_1,\dots,\Phi_n/\tau_n\}$.

– For a type variable τ:
  $\langle\tau\rangle\{\Theta\} :\equiv \Phi$ where Θ assigns Φ to τ.

$\langle\tau\rangle\{\Theta\} :\equiv \mathcal{E}q_\tau$ where Θ does not assign a predicate to τ.
- $\langle T \to U\rangle\{\Theta\} :\equiv \lambda xy.\forall vw.\langle T\rangle\{\Theta\}vw \supset \langle U\rangle\{\Theta\}(xv)(yw)$.
- $\langle \Pi\tau.T\rangle\{\Theta\} :\equiv \lambda xy.\forall\alpha\beta.\forall\phi^{\alpha\to\beta\to *}.\langle T\rangle\{\phi/\tau, \Theta\}(x\alpha)(y\beta)$.

For example:

$$\langle \Pi\tau.\tau\to\tau\rangle\{\}XY =_\beta \forall\alpha\beta.\forall\phi^{\alpha\to\beta\to *}.\forall v^\alpha w^\beta.\phi xy \supset \phi(X\alpha v)(Y\beta w).$$

Definition 6. The notations $\langle\ \rangle^=$, $\mathbf{Conf}()$, $\mathbf{Ext}()$ and $\mathbf{Par}()$ are defined as below:
- $\langle T\rangle^= :\equiv \langle T\rangle\{\}$.
- For a type T and a term X^T: $\mathbf{Conf}_T(X) :\equiv \langle T\rangle^= XX$.

We call this the *self-conformity* of X. When it holds, we say that X is *self-conforming*.
- $\mathbf{Conf}(T) :\equiv \forall x^T.\mathbf{Conf}_T(x)$. We call this the *conformity* of T.
- For predicate type $A \equiv T_1 \to \cdots \to T_i \to *$ and a predicate Φ^A:
  $\mathbf{Ext}_A(\Phi) :\equiv \forall x_1^{T_1}\ldots x_n^{T_n} y_1^{T_1}\ldots y_n^{T_n}.$
  $\langle T_1\rangle^= x_1y_1 \supset \cdots \supset \langle T_n\rangle^= x_ny_n \supset \Phi x_1\ldots x_i \supset \Phi y_1\ldots y_n$.

We call this the *extensionality* of Φ. When it holds, we say that Φ is *extensional*.
- $\mathbf{Ext}(T) :\equiv \forall\phi^{T\to *}.\mathbf{Ext}_{T\to *}(\phi)$. We call this the *extensionality* of T.
- $\mathbf{Par}(T) :\equiv \mathbf{Conf}(T) \wedge \mathbf{Ext}(T)$, which is called the *parametricity* of T.

In **L**, $\mathbf{Conf}(T)$ is equivalent to: $\forall x^T y^T.x = y \supset \langle T\rangle^= xy$.
$\mathbf{Ext}(T)$ is equivalent to: $\forall x^T y^T.\langle T\rangle^= xy \supset x = y$.
And $\mathbf{Par}(T)$ is equivalent to: $\forall x^T y^T.x = y \supset\subset \langle T\rangle^= xy$.

Definition 7. **Par** is the set consisting of the axioms $\forall\tau_1\ldots\tau_n.\mathbf{Par}(T)$ where $\tau_1,\ldots,\tau_n$ are all the free type variables of T, for all types T.

The theory of **Par** is logically equivalent to the system of [7]. In the next subsection we will mention that the parametricities infer the universalities of universal types, where the universalities are in a sense of categorical theory.

2.4 Recursive Types

Here are some data structures which are very often used. The theory of parametricity proves the universalities of these structures.

Definition 8. (Unit Type, Pair Type, Sum Type)
Unit type: $\mathbf{1} :\equiv \Pi\tau.\tau\to\tau$.
Pair type: $T\times U :\equiv \Pi\tau.(T\to U\to\tau)\to\tau$.
Sum type: $T + U :\equiv \Pi\tau.(T\to\tau)\to(U\to\tau)\to\tau$.

We call $T\to U$ a functional type. It is well-known that the theory of parametricity proves the universalities of these structures. For example:

Proposition 9. $\mathbf{Par}(T\times U), \mathbf{Par}(T), \mathbf{Par}(U) \vdash \forall x^{T\times U}.x = \boldsymbol{\pi}(\boldsymbol{pl}x)(\boldsymbol{pr}x)$,
where $\boldsymbol{\pi}$ *is the standard pairing, and* $\boldsymbol{pl}$ *and* $\boldsymbol{pr}$ *are the standard projections.*

Definition 10. (Fixed point type)

$$\mu\tau.T :\equiv \Pi\tau.T \to \tau \to \tau, \quad \nu\tau.T :\equiv \Pi\upsilon.(\Pi\tau.(\tau \to T) \to \tau \to \upsilon) \to \upsilon,$$

where τ occurs only positively in T. $\mu\tau.T$ is called the *least fixed point type*, and $\nu\tau.T$ is called the *greatest fixed point type*.

We discuss the properties of fixed point types in Section 4.3.

Definition 11. $\mathbf{T}$ is the set of all the types which are constructed from $\mathbf{1}$ and type variables by the pair type $\times$, the sum type $+$, the functional type $\to$ and the fixed point types μ and ν. Types in $\mathbf{T}$ are called *recursive types*. $\mathbf{Par}(\mathbf{T})$ is the set consisting of axioms $\mathbf{Par}(T)$ for all types $T \in \mathbf{T}$.

3 Consistency of the Parametricity Axioms

In this section and the next section we prove the consistency of **Par**. It is done by the method of relativization, and is one of main results in this paper.

3.1 Relativized Parametricity

Definition 12. Here we define the notations $\langle T\rangle_{rel}\{\ldots\}$ and $\langle T\rangle^{=}_{rel}$, which are called *relativized conformity notations*. They are defined by mutual induction on T. The definition of $\langle\ \rangle_{rel}\{\ \}$ is as below:

- For a type variable τ:
  $\langle\tau\rangle_{rel}\{\Theta\} :\equiv \Phi$ where Θ assigns Φ to τ.
  $\langle\tau\rangle_{rel}\{\Theta\} :\equiv \mathcal{E}q_\tau$ where Θ does not assign a predicate to τ.
- $\langle T \to U\rangle_{rel}\{\Theta\} :\equiv$
  $\lambda xy.\forall vw.\langle T\rangle^{=}_{rel}vv \supset \langle T\rangle^{=}_{rel}ww \supset \langle T\rangle_{rel}\{\Theta\}vw \supset \langle U\rangle_{rel}\{\Theta\}(xv)(yw)$.
- $\langle\Pi\tau.T\rangle_{rel}\{\Theta\} :\equiv \lambda xy.\forall\alpha\beta\forall\phi.\langle T\rangle_{rel}\{\phi/\tau, \Theta\}(x\alpha)(y\beta)$.

The definition of $\langle T\rangle^{=}_{rel}$ is: $\langle T\rangle^{=}_{rel} :\equiv \langle T\rangle_{rel}\{\}$.

$\langle -\rangle_{rel}\{\ldots\}$ is defined in the similar way to the definition of $\langle -\rangle\{\ldots\}$. The difference between them is that the quantifier is relativized in the definition of $\langle T \to U\rangle_{rel}\{\ldots\}$.

Definition 13. Here are the definitions of relativized self-conformity and relativized extensionality.

$$\mathbf{Conf}^{rel}_T(X) :\equiv \langle T\rangle^{=}_{rel} X\, X,$$

$$\begin{aligned}\mathbf{Ext}^{rel}_{T_1 \to \cdots \to T_n \to *}(\Phi) :\equiv\ & \forall x_1^{T_1} \ldots x_n^{T_n} y_1^{T_1} \ldots y_n^{T_n}.\\ & \mathbf{Conf}^{rel}(x_1) \supset \cdots \supset \mathbf{Conf}^{rel}(x_n) \supset\\ & \mathbf{Conf}^{rel}(y_1) \supset \cdots \supset \mathbf{Conf}^{rel}(y_n) \supset\\ & \langle T_1\rangle^{=}_{rel} x_1 y_1 \supset \cdots \supset \langle T_n\rangle^{=}_{rel} x_n y_n \supset \Phi x_1 \ldots x_n \supset \Phi y_1 .. y_n.\end{aligned}$$

Now we define parametricity relativization. This relativization is a translation from formulae into formulae which rewrites quantifiers. We define the translation for predicates for convenience of definition.

Definition 14. For a predicate P, $rel(P)$ is defined by induction on P as follows:

$rel(\phi) :\equiv \phi$, for a predicate variable ϕ.
$rel(\Phi X) :\equiv rel(\Phi)X$, for a predicate ΦX.
$rel(\lambda x.\Phi x) :\equiv \lambda x.rel(\Phi x)$, for a predicate $\lambda x.\Phi x$.
$rel(P \supset Q) :\equiv rel(P) \supset rel(Q)$.
$rel(\forall x^T.P) :\equiv \forall x^T.\mathbf{Conf}^{rel}_T(x) \supset rel(P)$.
$rel(\forall \phi^A.P) :\equiv \forall \phi^A.\mathbf{Ext}^{rel}_A(\phi) \supset rel(P)$.
$rel(\forall \tau.P) :\equiv \forall \tau.rel(P)$.

The intuitive meaning of this relativization is that terms are relativized by self-conformity, and predicates are relativized by extensionality. Note that Leibniz equality and extensionality play essential roles in relativizing a formula $x = y$.

3.2 Soundness and Completeness

This subsection is aimed to prove the following theorem.

Theorem 15. *The above parametricity relativization is sound and complete for the theory of* **Par**, *that is,* $\mathbf{Par} \vdash P$ *iff* $\mathbf{L} \vdash rel(P)$ *for every closed formula* P.

To prove this, we need to prove some lemmata. Theorem 15 is proved from the soundness and the completeness of the relativization. We shall obtain the consistency of **Par** from the soundness. First we show the soundness.

Lemma 16. *Let* X *be a term of type* T, *and let* $x_1^{T_1} \ldots x_n^{T_n}$ *be all the free variables in* X. *Then* **L** *derives:*

$$Env(X), Env(Y), \langle T_1 \rangle^{=}_{rel} x_1 y_1, \ldots, \langle T_n \rangle^{=}_{rel} x_n y_n \vdash \langle T \rangle^{=}_{rel} XY,$$

where $Y :\equiv X[y_1/x_1, \ldots, y_n/x_n]$.

Proof. By induction on the construction of X. □

Lemma 17. *Let* Φ *be a predicate, and let* $x_1, \ldots, x_m, \phi_1, \ldots, \phi_n$ *be all the variables and the predicate variables in* Φ. *Then* **L** *derives:*

$$Env(\Phi), \mathbf{Conf}^{rel}(x_1), \ldots, \mathbf{Conf}^{rel}(x_m), \mathbf{Ext}^{rel}(\phi_1), \ldots, \mathbf{Ext}^{rel}(\phi_n) \vdash \mathbf{Ext}^{rel}(rel(\Phi)).$$

Proof. By induction on the construction of Φ. □

Lemma 18. (Soundness of Relativization) *Let $P, Q_1, \ldots, Q_n$ be formulae, and let $x_1 \ldots x_l, \phi_1, \ldots, \phi_m$ be all the variables and predicate variables in $P, Q_1, \ldots, Q_n$. If $Env(P, Q_1, \ldots, Q_n), \mathbf{Par}, Q_1, \ldots, Q_n, \vdash P$, then $\mathbf{L}$ derives:*

$$Env(P), \mathbf{Conf}^{rel}(x_1), \ldots, \mathbf{Conf}^{rel}(x_l), \\ \mathbf{Ext}^{rel}(\phi_1), \ldots, \mathbf{Ext}^{rel}(\phi_m), rel(Q_1), \ldots, rel(Q_n) \vdash rel(P).$$

Proof. By induction on the derivation of $Env(P), \mathbf{Par}, Q_1, \ldots, Q_n, \vdash P$. □

Next we show completeness.

Proposition 19. $\mathbf{Par}, Env(P) \vdash P \supset\subset rel(P)$ *for any formula P.*

Lemma 20. (Completeness of Relativization) *Let P be a formula, and let $x_1, \ldots, x_m, \phi_1 \ldots \phi_n$ be all the variables and the predicate variables in P. If $\mathbf{L}$ derives:*

$$Env(P) \vdash \mathbf{Conf}^{rel}(x_1) \supset \cdots \supset \mathbf{Conf}^{rel}(x_m) \supset \\ \mathbf{Ext}^{rel}(\phi_1) \supset \cdots \supset \mathbf{Ext}^{rel}(\phi_n) \supset rel(P),$$

then $\mathbf{Par} \vdash P$.

Proof. By induction on P. □

Corollary 21. (Consistency of Par) **Par** *is consistent.*

Proof. From soundness, if $\mathbf{Par} \vdash \bot$, then $\mathbf{L} \vdash rel(\bot)$. But $rel(\bot) \equiv \bot$. Thus, if **Par** is inconsistent, then $\mathbf{L}$ is inconsistent, so second order propositional logic is inconsistent. □

4 Polyadic Parametricity

We discussed parametricity only for binary relations until the section before. Plotkin and Abadi [7] give the definition of polyadic parametricity. In this section we discuss the equivalence among the theories of polyadic parametricities.

4.1 Definitions, Conjecture and Theorem

The definition of polyadic parametricity is quite similar to the binary one. It is essentially equivalent to that in [7] just the same as for binary relations.

Definition 22. $\mathcal{E}q^n_T :\equiv \lambda x_1 \ldots x_n.(x_1 =_T x_2 \wedge x_1 =_T x_3 \wedge \cdots \wedge x_1 =_T x_n)$.

Definition 23. Let T be a type, n be a number ≥ 2 and $\Phi_1, \ldots, \Phi_m$ be n-ary predicate variables, where Φ_i is of type $U_{i,1} \to \cdots \to U_{i,n} \to *$.

Then $\langle T \rangle_n \{\Phi_1/\tau_1, \ldots, \Phi_m/\tau_m\}$ is a predicate of type $T_1 \to \cdots \to T_n \to *$, where $T_i :\equiv T[U_{1,i}/\tau_1, \ldots, U_{m,i}/\tau_m]$. It is defined by induction on T as below.

– For a type variable: $\langle \tau \rangle_n \{\Theta\} :\equiv \Phi$ where Θ assigns Φ to τ.

$\langle \tau \rangle_n \{\Theta\} :\equiv \mathcal{E}q^n_\tau$ where Θ does not assign a predicate to τ.

– $\langle T \to U \rangle_n \{\Theta\} :\equiv$
$\lambda x_1 \ldots x_n . \forall y_1 \ldots y_n . \langle T \rangle_n \{\Theta\} y_1 \ldots y_n \supset \langle U \rangle_n \{\Theta\} (x_1 y_1) \ldots (x_n y_n)$.

– $\langle \Pi \tau . T \rangle_n \{\Theta\} :\equiv$
$\lambda x_1 \ldots x_n . \forall \alpha_1 \ldots \alpha_n . \forall \phi^{\alpha_1 \to \cdots \to \alpha_n \to *} . \langle T \rangle_n \{\phi/\tau, \Theta\} (x_1 \alpha_1) \ldots (x_n \alpha_n)$.

Definition 24. The definitions of $\langle T \rangle^=_n$, $\mathbf{Par}_n(T)$, $\mathbf{Par}_n$ and $\mathbf{Par}_n(\mathbf{T})$ are similar to those for the binary case. $\langle T \rangle^=_n$ and $\mathbf{Par}_n(T)$ are defined as:

$$\langle T \rangle^=_n :\equiv \langle T \rangle_n \{\}, \quad \mathbf{Par}_n(T) :\equiv \forall x^T_1 \ldots x^T_n . \mathcal{E}q x_1 \ldots x_n \supset\subset \langle T \rangle^=_n x_1 \ldots x_n .$$

$\mathbf{Par}_n$ is the set consisting of the axioms $\forall \tau_1 \ldots \tau_n . \mathbf{Par}_n(T)$ where $\tau_1, \ldots, \tau_n$ are all the free type variables of T, for all types T. $\mathbf{Par}_n(\mathbf{T})$ is the set consisting of axioms $\mathbf{Par}_n(T)$ for all types $T \in \mathbf{T}$.

Lemma 25. *$\mathbf{Par}_n$ is consistent. Moreover, $\bigcup_{m \le n} \mathbf{Par}_m$ is consistent for any n. Therefore, $\bigcup_n \mathbf{Par}_n$ is consistent.*

Proof. Similarly to the case for binary parametricity, there is a suitable relativization which is sound and complete for the theory of $\bigcup_{m \le n} \mathbf{Par}_m$. □

We make this conjecture for the polyadic parametricity: $\mathbf{Par}_m \vdash \mathbf{Par}_n(T)$ for all type T and $m, n \ge 2$. This conjecture says that all of $\mathbf{Par}_n$'s are equivalent to each other for arbitrary numbers $n \ge 2$. However, we have only the following theorem yet.

Theorem 26. *$\mathbf{Par}_m(\mathbf{T}) \vdash \mathbf{Par}_n(T)$ for all types T in $\mathbf{T}$ and all $m, n \ge 2$.*

This theorem says that all of $\mathbf{Par}_n(\mathbf{T})$'s are equivalent to each other for arbitrary numbers $n \ge 2$. We will spend the remaining part of this section in showing the proof of it.

4.2 Functor

We have to define the notion of functor and some terms in order to prove Theorem 26.

Definition 27. When a type variable τ occurs only positively in a type T, then the operation which makes $T[U/\tau]$ from U is called a *functor*. When a type variable τ occurs only negatively in a type T, then the operation which makes $T[U/\tau]$ from U is called a *contravariant functor*.

Suppose that $C[\,]$ is a context which makes a type $C[T]$ from a type T, and that a functor or contravariant functor F makes $C[T]$ from T. Then $C[-]$ denotes the functor or contravariant functor F.

Definition 28. A functor or a contravariant functor F is said to be in $\mathbf{T}$ if $F\tau \in \mathbf{T}$.

Definition 29. $I_T :\equiv \lambda x^T.x, \quad X \circ Y :\equiv \lambda x.X(Yx)$.

Definition 30. We define the operation of functors and contravariant functors for some terms. Let X be a term of type $T \to U$, F be a functor, or a contravariant functor, then $\langle X \rangle_F$ is a term of type $FT \to FU$ if F is a functor, or of type $FU \to FT$ if F is a contravariant functor, and is defined as follows.

If τ does not occur in $F\tau$, then $\langle X \rangle_F :\equiv I_{F\tau}$.

Otherwise:

- When $F\tau \equiv \tau$, then $\langle X \rangle_F :\equiv X$.
- When $F\tau \equiv G\tau \to H\tau$, then $\langle X \rangle_F :\equiv \lambda x.\langle X \rangle_H \circ x \circ \langle X \rangle_G$.
- When $F\tau \equiv \Pi \upsilon.C[\tau, \upsilon]$, then $\langle X \rangle_F :\equiv \Lambda \upsilon.\langle X \rangle_{C[-,\upsilon]}$.

Definition 31. We define substitution of types which occur in functors. When F is a functor and θ is a substitution of types, then $F\theta$ is a functor which makes $F\theta\tau \equiv (F\tau)\theta$ from a fresh variable τ. Substitution of types for contravariant functors is defined similarly.

These identities hold: $FT\theta \equiv F\theta(T\theta)$, $(\langle X \rangle_F)\theta \equiv \langle X\theta \rangle_{F\theta}$, and $(\mu\tau.F\tau)\theta \equiv \mu\tau.F\theta\tau$.

Definition 32. Let F be a functor or a contravariant functor, Θ be an assignment of n-ary predicates for type variables, and Φ be an n-ary predicate. Then: $F^{\Theta}\Phi :\equiv \langle F\tau \rangle_n \{\Phi/\tau, \Theta\}$.

4.3 Fixed Point Type and Fixed Point Predicate

In this subsection, we define fixed point predicates, and show some properties of fixed point types and fixed point predicates.

Definition 33. Let Φ, Ψ be predicates with the same predicate type. Then:

$$\Phi \sqsubseteq \Psi :\equiv \forall x_1 \ldots x_n.\Phi x_1 \ldots x_n \supset \Psi x_1 \ldots x_n, \quad \text{and} \quad \Phi \cong \Psi :\equiv \Phi \sqsubseteq \Psi \wedge \Psi \sqsubseteq \Phi.$$

Definition 34. Let $\Phi[\phi^A]$ be a predicate of predicate type $A \equiv T_1 \to \cdots T_n \to *$, where ϕ occurs only positively. Then:

$$\mu\phi.\Phi[\phi] :\equiv \lambda x_1 \ldots x_n.\forall\phi.(\Phi[\phi] \sqsubseteq \phi) \supset \phi x_1 \ldots x_n,$$
$$\nu\phi.\Phi[\phi] :\equiv \lambda x_1 \ldots x_n.\exists\phi.(\phi \sqsubseteq \Phi[\phi]) \wedge \phi x_1 \ldots x_n.$$

$\mu\phi.\Phi[\phi]$ is called the *least fixed point predicate*, and $\nu\phi.\Phi[\phi]$ is called the *greatest fixed point predicate*.

Proposition 35. (Inferences of Fixed Point Predicate)

$$Env(\Phi), Env(\Psi), \Phi[\Psi] \sqsubseteq \Psi \vdash \mu\phi.\Phi[\phi] \sqsubseteq \Psi, \quad Env(\Phi) \vdash \Phi \cong \mu\phi.\Phi[\phi],$$
$$Env(\Phi), Env(\Psi), \Psi \sqsubseteq \Phi[\Psi] \vdash \Psi \sqsubseteq \nu\phi.\Phi[\phi], \quad Env(\Phi) \vdash \Phi \cong \nu\phi.\Phi[\phi].$$

For a functor F, we write μF for $\mu\tau.F\tau$, and νF for $\nu\tau.F\tau$. It is well-known that μF can encode recursive structures, and νF can encode lazy structures.

Definition 36. For a functor F and a type T, The terms $\boldsymbol{c}_{\mu F,T}$ and $\boldsymbol{c}_{\nu F,T}$ are the standard encodings of the constructors of type $(FT \to T) \to \mu F \to T$ and of $(T \to FT) \to T \to \nu F$. $\boldsymbol{f}_{\mu F}$ and $\boldsymbol{f}_{\nu}$ are the standard encodings of the unfold operators of type $\mu F \to F\mu F$ and of type $\nu F \to F\nu F$. $\boldsymbol{g}_{\mu F}$ and $\boldsymbol{g}_{\nu}$ are the standard encodings of the unfold operators of type $F\mu F \to \mu F$ and of type $F\nu F \to \nu F$.

We have by an easy calculation that $\boldsymbol{c}_{\mu F,T}X \circ \boldsymbol{f}_{\mu F} =_\beta X \circ \langle \boldsymbol{c}_{\mu F}X\rangle_F$ for each term $X^{FT\to T}$, and also $\boldsymbol{g}_{\nu F} \circ \boldsymbol{c}_{\mu F,T}X =_\beta \langle \boldsymbol{c}_{\mu F}X\rangle_F \circ X$ for each term $X^{T\to FT}$. These are a standard encoding of recursive structure and lazy structure. It is well-known that the axioms of parametricities prove that μF is a initial algebra and νF is a final coalgebra.

We will prove the fixed point lemma 40. We define some notations in advance.

Definition 37. $T^0 \to * :\equiv *$, $T^{n+1} \to * :\equiv T \to T^n \to *$.

Definition 38. Let Φ be a predicate of predicate type $T^n \to *$, and X be a term of type $U \to T$. Then, $\Phi \circ X :\equiv \lambda x_1^U \ldots x_n^U.\Phi(Xx_1)\ldots(Xx_n)$.

Definition 39. Let F be a functor, $\phi_1, \ldots, \phi_m$ be predicate variables where ϕ_i is of predicate type $U_m^n \to *$, θ be a substitution of types $[U_1/\tau_1, \ldots, U_m/\tau_m]$, and Θ be a assignment $\{\phi/\tau_1, \ldots, \phi/\tau_m\}$. Then

$$\mathbf{Fix}_n(\mu F, \theta) :\equiv \forall\phi_1 \ldots \phi_m.(\langle \mu F\rangle_n\{\Theta\} \cong \mu\psi^{\mu F\theta^n \to *}.(F^\Theta\psi) \circ \boldsymbol{g}_{\mu F\theta})$$

$$\mathbf{Fix}_n(\nu F, \theta) :\equiv \forall\phi_1 \ldots \phi_m.(\langle \nu F\rangle_n\{\Theta\} \cong \mu\psi^{\nu F\theta^n \to *}.(F^\Theta\psi) \circ \boldsymbol{g}_{\nu F\theta})$$

We call $\mathbf{Fix}_n(T, \theta)$ *fixed point property* of T for n-ary predicates.

The intuitive meaning of fixed point property is that the conformity of fixed point type is equivalent to the fixed point of conformity.

Lemma 40. (Fixed Point Lemma) *Let F be a functor in $\mathbf{T}$, and $F\theta$ also be a functor in $\mathbf{T}$. Then $\mathbf{Par}_m(\mathbf{T}) \vdash \mathbf{Fix}_n(\mu F, \theta)$ and $\mathbf{Par}_m(\mathbf{T}) \vdash \mathbf{Fix}_n(\nu F, \theta)$ for any $m, n \geq 2$.*

Note that we can choose the suffix numbers m and n independently. In [7], this lemma is mentioned, but it is restricted for the same suffix n.

4.4 Crucial Lemma

This subsection is spent on proving Lemma 45. It is crucial for the proof of our main theorem 26.

Definition 41. Let Φ be a predicate of predicate type $T^n \to *$. Then:

$$\hat{\mathbf{b}}[\Phi] \equiv \lambda x^T y^T.(\Phi x \ldots x \wedge \bigwedge_{i=1,\ldots,n} \forall z_1 \ldots z_{i-1} z_{i+1} \ldots z_n. \Phi z_1 \ldots z_{i-1} x z_{i+1} \ldots z_n \supset\subset \Phi z_1 \ldots z_{i-1} y z_{i+1} \ldots z_n),$$

$$\bar{\mathbf{b}}[\Phi] :\equiv \lambda x^T y^T.\exists z_1 \ldots z_n.(\Phi z_1 \ldots z_n \wedge (\bigvee_{i=1,\ldots,n} x = z_i) \wedge (\bigvee_{i=1,\ldots,n} y = z_i)),$$

$$\check{\mathbf{b}}[\Phi] :\equiv \mu\phi^{T\to T\to *}.\lambda xy.\exists z.\bar{\mathbf{b}}[\Phi] xz \wedge (z = y \vee \phi z y).$$

The binary relation $\hat{\mathbf{b}}[\Phi]$ denotes a partial equivalence relation in a domain in which Φ holds. If $\hat{\mathbf{b}}[\Phi]xy$, then x and y behave in the same way for Φ. The binary relation $\bar{\mathbf{b}}[\Phi]$ is the symmetric closure of Φ, and $\check{\mathbf{b}}[\Phi]$ is the transitive closure of $\bar{\mathbf{b}}[\Phi]$.

Definition 42. Let Φ be a predicate of predicate type $T^n \to *$. Then:

$$\mathbf{C}_n^m[\Phi] :\equiv \lambda x_1 \ldots x_m.\hat{\mathbf{b}}[\Phi] x_1 x_2 \wedge \hat{\mathbf{b}}[\Phi] x_1 x_3 \wedge \ldots \wedge \hat{\mathbf{b}}[\Phi] x_1 x_m,$$

$$\mathbf{D}_n^m[\Phi] :\equiv \lambda x_1 \ldots x_m.\check{\mathbf{b}}[\Phi] x_1 x_2 \wedge \check{\mathbf{b}}[\Phi] x_1 x_3 \wedge \ldots \wedge \check{\mathbf{b}}[\Phi] x_1 x_m.$$

$\mathbf{C}_n^m[\,]$ and $\mathbf{D}_n^m[\,]$ are partial equivalence relations of arity m. $\mathbf{C}_n^m[\Phi]$ is an internal relation over the domain of Φ, and $\mathbf{D}_n^m[\Phi]$ is the closure of Φ.

Proposition 43. $\mathbf{D}[\,]$ *is a monotone context, that is,* ϕ *occurs only positively in* $\mathbf{D}_n^m[\phi]$. *Thus* $\mathbf{L}$ *derives:*

$$Env(\Phi), Env(\Psi), \Phi \sqsubseteq \Psi \vdash \mathbf{D}[\Phi] \sqsubseteq \mathbf{D}[\Psi].$$

Definition 44. Let Θ be an assignment $\{\Phi_1^{T_1^n \to *}/\tau_1, \ldots, \Phi_l^{T_l^n \to *}/\tau_l\}$. Then:

$$\mathbf{C}_n^m[\Theta] :\equiv \{\mathbf{C}_n^m[\Phi_1]/\tau_1, \ldots, \mathbf{C}_n^m[\Phi_l]/\tau_l\}, \mathbf{D}_n^m[\Theta] :\equiv \{\mathbf{D}_n^m[\Phi_1]/\tau_1, \ldots, \mathbf{D}_n^m[\Phi_l]/\tau_l\}.$$

Lemma 45. *Let* $T, T_1, \ldots, T_i, T'_1, \ldots, T'_{i'}, U_1, \ldots, U_j, U'_1, \ldots, U'_{j'}$ *be types in* $\mathbf{T}$. *Suppose that type variables* $\tau_1, \ldots, \tau_i, \tau'_1, \ldots, \tau'_{i'}$ *occur only positively in* T, *and* $\upsilon_1, \ldots, \upsilon_j, \upsilon'_1, \ldots, \upsilon'_{j'}$ *occur only negatively in* T. *Let*

$$\Phi_1{}^{T_1^m \to *}, \ldots, \Phi_i{}^{T_i^m \to *}, \Phi'_1{}^{T'^n_1 \to *}, \ldots, \Phi'_{i'}{}^{T'^n_{i'} \to *},$$

$$\Psi_1{}^{U_1^n \to *}, \ldots, \Psi_j{}^{U_j^n \to *}, \Psi'_1{}^{U'^m_1 \to *}, \ldots, \Psi'_{j'}{}^{U'^m_{j'} \to *}$$

be predicates. Let assignments Θ, Θ', Ξ *and* Ξ' *be defined as:*

$$\Theta :\equiv \{\Phi_1/\tau_1, \ldots, \Phi_i/\tau_i\}, \quad \Theta' :\equiv \{\Phi'_1/\tau'_1, \ldots, \Phi'_{i'}/\tau'_{i'}\},$$

$$\Xi :\equiv \{\Psi_1/\upsilon_1, \ldots, \Psi_j/\upsilon_j\}, \quad \Xi' :\equiv \{\Psi'_1/\upsilon'_1, \ldots, \Psi'_{j'}/\upsilon'_{j'}\}.$$

Then for arbitrary numbers $k, l, m, n \geq 2$, *it holds that:*

$$\mathbf{Par}_l(\mathbf{T}) \vdash \mathbf{D}_m^k[\langle T\rangle_m\{\Theta, \mathbf{C}_n^m[\Theta'], \mathbf{D}_n^m[\Xi], \Xi'\}] \sqsubseteq \mathbf{C}_n^k[\langle T\rangle_n\{\mathbf{D}_m^n[\Theta], \Theta', \Xi, \mathbf{C}_m^n[\Xi']\}].$$

Proof. The proof is by induction on the construction of T. We use the following notations:

$$M :\equiv \{\Theta, \mathbf{C}_n^m[\Theta'], \mathbf{D}_n^m[\Xi], \Xi'\}, \quad \text{and}, \quad N :\equiv \{\mathbf{D}_m^n[\Theta], \Theta', \Xi, \mathbf{C}_m^n[\Xi']\}.$$

M is an assignment of predicates with m arguments, and N is an assignment of predicates with n arguments. The substitution θ is defined as:

$$\theta :\equiv [T_1/\tau_1, \ldots, T_i/\tau_i, T_1'/\tau_1', \ldots, T_{i'}'/\tau_{i'}, U_1/\upsilon_1, \ldots, U_j/\upsilon_j, U_1'/\upsilon_1', \ldots, U_{j'}'/\upsilon_{j'}]$$

For an arbitrary type T, $\langle T\rangle_m\{M\}$ is a predicate of predicate type $T\theta^m \to *$, and $\langle T\rangle_n\{N\}$ is a predicate of predicate type $T\theta^n \to *$.

We show the proof step only for functional types $\to$ and for least fixed types μ. The steps for unit type $\mathbf{1}$, pair type $\times$, and sum type $+$ are easy, and the step for greatest fixed point types ν are similar to that for μ.

The step for least fixed point types μ is shown by using the fixed point lemma 40. The outline of the step for the least fixed point types is as below:

By the fixed point lemma, we have

$$\langle \mu F\rangle_m\{M\} \cong \mu\phi.F^M\phi \circ \boldsymbol{g}_{\mu F\theta}, \quad \text{and} \quad \langle \mu F\rangle_n\{N\} \cong \mu\phi.F^N\phi \circ \boldsymbol{g}_{\mu F\theta}.$$

The induction hypothesis is $F^M(\mathbf{C}_n^m[\langle \mu F\rangle_n\{N\}]) \sqsubseteq \mathbf{C}_n^m[F^N(\langle \mu F\rangle_n\{N\})]$, thus

$$\mu\phi.F^M\phi \sqsubseteq \mathbf{C}_n^m[F^N(\langle \mu F\rangle_n\{N\})] \circ \boldsymbol{g}_{\mu F\theta}.$$

Therefore we obtain $\langle \mu F\rangle_m\{M\} \sqsubseteq \mathbf{C}_n^n[\langle \mu F\rangle_n\{N\}]$.

The proof of the step for a functional type $T \to U$ is as below:

Our goal in this step is to derive

$$\mathbf{D}_m^k[\langle T \to U\rangle_m\{M\}] \sqsubseteq \mathbf{C}_n^k[\langle T \to U\rangle_n\{N\}].$$

It is enough to show

$$\bar{\mathbf{b}}[\langle T \to U\rangle_m\{M\}] \sqsubseteq \hat{\mathbf{b}}[\langle T \to U\rangle_n\{N\}],$$

because then we get the target formula by operating with $\mathbf{D}_2^k[\,]$ on both sides. The induction hypotheses of the case $k = 2$ are:

$$\check{\mathbf{b}}[\langle T\rangle_m\{N\}] \sqsubseteq \hat{\mathbf{b}}[\langle T\rangle_m\{M\}], \quad \check{\mathbf{b}}[\langle U\rangle_m\{M\}] \sqsubseteq \hat{\mathbf{b}}[\langle U\rangle_n\{N\}].$$

We can derive $\hat{\mathbf{b}}[\langle T \to U\rangle_m\{N\}]xy$ from $\bar{\mathbf{b}}[\langle T \to U\rangle_m\{M\}]xy$. In the derivation, the symmetricity of $\bar{\mathbf{b}}[\,]$, and the transitivity of $\hat{\mathbf{b}}[\,]$ and $\mathbf{b}[\,]$ do essential work. □

4.5 Proof of the Theorem

Proof of Theorem 26. For each type $T \in \mathbf{T}$, we have

$$\mathbf{Par}_m(\mathbf{T}) \vdash \mathbf{D}_m^n[\langle T\rangle_m^=] \sqsubseteq \mathbf{C}_n^n[\langle T\rangle_n^=], \quad \mathbf{Par}_m(\mathbf{T}) \vdash \mathbf{D}_n^n[\langle T\rangle_n^=] \sqsubseteq \mathbf{C}_m^n[\langle T\rangle_m^=]$$

by Lemma 45. $\mathbf{Par}_m(\mathbf{T})$ also proves $\mathcal{E}q_T^m \cong \langle T\rangle_m^=$. $\mathbf{L}$ derives $\mathcal{E}q_T^n \cong \mathbf{D}_m^n[\mathcal{E}q_T^m]$, and $\mathcal{E}q_T^n \cong \mathbf{C}_m^n[\mathcal{E}q_T^m]$. Thus we obtain the theorem.

5 Realizability Interpretation

In this section we discuss the realizability interpretation for **L**, and we show the self-realizability of **Par**(**T**). By using this self-realizability, we obtain the term extraction property of **Par**(**T**).

5.1 Realizability of the Basic Logic

First we define the realizability interpretation of **L** into **L** itself.

Definition 46. A figure $\{(\Phi_1, \Psi_1)/\phi_1, \ldots, (\Phi_n, \Psi_n)/\phi_n\}$ denotes the assignment which assigns predicates (Φ_i, Ψ_i) to ϕ_i, where the predicate type of Φ_i is A, which is the same as the predicate type of ϕ_i, and the predicate type of Ψ_i is $T \to A$ for some type T. It is called a *double assignment*.

Definition 47. For a double assignment $\Xi \equiv \{(\Phi_1, \Psi_1)/\phi_1, \ldots, (\Phi_n, \Psi_n)/\phi_n\}$, $[\Xi]$ stands for the substitution such that $[\Xi] :\equiv [\Phi_1/\phi_1 \ldots \Phi_n/\phi_n]$.

Definition 48. We define a type $\mathbf{r}^{\Xi}(\Phi)$ for a predicate Φ and a double assignment Ξ. It will be the type of a realizer of Φ with the double assignment Ξ.

$\mathbf{r}^{\Xi}(\phi) :\equiv T$, where Ξ assigns $(\Phi^A, \Psi^{T\to A})$ to ϕ,
$\mathbf{r}^{\Xi}(\phi) :\equiv \mathbf{1}$, where Ξ does not assign predicates to ϕ,
$\mathbf{r}^{\Xi}(\lambda x.\Phi) :\equiv \mathbf{r}^{\Xi}(\Phi)$,
$\mathbf{r}^{\Xi}(\Phi X) :\equiv \mathbf{r}^{\Xi}(\Phi)$,
$\mathbf{r}^{\Xi}(P \supset Q) :\equiv \mathbf{r}^{\Xi}(P) \to \mathbf{r}^{\Xi}(Q)$,
$\mathbf{r}^{\Xi}(\forall x^T.P) :\equiv T \to \mathbf{r}^{\Xi}(P)$,
$\mathbf{r}^{\Xi}(\forall \tau.P) :\equiv \Pi\tau.\mathbf{r}^{\Xi}(P)$,
$\mathbf{r}^{\Xi}(\forall \phi^A.P) :\equiv \Pi\tau.\mathbf{r}^{\Xi'}(P)$, where $\Xi' \equiv \{(\phi, \psi)/\phi, \Xi\}$ for a fresh $\psi^{\tau\to A}$.

Definition 49. For a predicate Φ, a double assignment Ξ and a term R of type $\mathbf{r}^{\Xi}(\Phi)$, we define the realizability interpretation $R\{\Xi\} \models \Phi$. It is defined by induction on Φ.

$R\{\Xi\} \models \phi :\equiv \lambda x_1 \ldots x_n.(\Phi x_1 \ldots x_n \wedge \Psi R x_1 \ldots x_n,)$
where Ξ assigns (Φ, Ψ) to ϕ,
$R\{\Xi\} \models \phi :\equiv \phi$, where Ξ does not assign predicates to ϕ,
$R\{\Xi\} \models \lambda x.\Phi :\equiv \lambda x.(R\{\Xi\} \models \Phi)$,
$R\{\Xi\} \models \Phi X :\equiv (R\{\Xi\} \models \Phi)X$,
$R\{\Xi\} \models P \supset Q :\equiv (\forall r.(r\{\Xi\} \models P) \supset (Rr\{\Xi\} \models Q)) \wedge (P \supset Q)[\Xi]$,
$R\{\Xi\} \models \forall x.P :\equiv \forall x.Rx\{\Xi\} \models P$,
$R\{\Xi\} \models \forall \tau.P :\equiv \forall \tau.R\tau\{\Xi\} \models P$,
$R\{\Xi\} \models \forall \phi^A.P :\equiv \forall \tau.\forall \phi^A \psi^{\tau\to A}.R\tau\{(\phi, \psi)/\phi, \Xi\} \models P$.

$\mathbf{r}(\Phi)$ stands for $\mathbf{r}^{\{\ \}}(\Phi)$ and $R \models P$ stands for $R\{\} \models P$.

Proposition 50. (Completeness) *For each formula P and each assignment Ξ, **L** derives $E, (R\{\Xi\} \models P) \vdash P[\Xi]$ for a suitable environment E.*

Proof. By induction on P. □

Theorem 51. (Soundness) *If* $\vdash P$ *then there is a closed term* R *such that* **L** *derives* $\vdash (R \models P)$.

Proof. By induction on the proof of $\vdash P$. □

5.2 Self-Realizability of Parametricity

Lemma 52. *Let* **A** *be a set of axioms. If* $\mathbf{A} \vdash (R \models \forall x.\exists y.P)$, *then there is a term* X *such that* $\mathbf{A} \vdash \forall x.P[Xx/y]$.

Lemma 53. *Let* **A** *be a set of axioms. Suppose that for each* $A \in \mathbf{A}$ *there is a term* R *such that* $\mathbf{A} \vdash (R \models A)$. *If* $\mathbf{A} \vdash P$, *then there is a term* R *such that* $\mathbf{A} \vdash (R \models P)$.

Corollary 54. *Let* **A** *be a set of axioms. Suppose that for each* $A \in \mathbf{A}$ *there is a term* R *such that* $\mathbf{A} \vdash (R \models A)$. *If* $\mathbf{A} \vdash \forall x.\exists y.P$, *then there is a term* X *such that* $\mathbf{A} \vdash \forall x.P[Xx/y]$.

We call **A** *self-realizable* when there is a term R_A such that $\mathbf{A} \vdash (R_A \models A)$ for each $A \in \mathbf{A}$. If **A** is self-realizable, the theory of **A** satisfies term extraction.

We shall show that $\mathbf{Par}(\mathbf{T})$ is self-realizable. We want to show that for each T in **T** there is a term R_T such that $\mathbf{Par}(\mathbf{T}) \vdash R_T \models \mathbf{Conf}(T)$, which is equivalent to $\mathbf{Par}(\mathbf{T}), x^T \vdash R_T x \models \langle T \rangle^{=} xx$. For the purpose of showing that, we first will define a predicate $\langle T \rangle^{\models}$ for each T, and show that $\mathbf{Par}(\mathbf{T}) \vdash \forall x^T.\langle T \rangle^{\models}(K_T x)xx$ for some term K_T.

Definition 55. Let $\tau_1, \ldots, \tau_n$ be all the free variables in T. Then define a predicate

$$\langle T \rangle^{\models} :\equiv \lambda r.(r\{\Xi\} \models \langle T \rangle\{\phi_1/\tau_1, \ldots, \phi_n/\tau_n\})$$

where $\phi_1^{\tau_1 \to \tau_1 \to *}, \ldots, \phi_n^{\tau_n \to \tau_n \to *}$ are fresh predicate variables, and Ξ is a double assignment such that $\Xi \equiv \{(\mathcal{E}q^2_{\tau_1}, \mathcal{E}q^3_{\tau_1})/\phi_1, \ldots, (\mathcal{E}q^2_{\tau_n}, \mathcal{E}q^3_{\tau_n})/\phi_n\}$.

Lemma 56. *Let* T *be a type in* **T**. *Then there are terms* J_T *and* K_T *such that*

$$\mathbf{Par}(\mathbf{T}) \vdash \forall xyz.\langle T \rangle^{=}_3 xyz \supset \langle T \rangle^{\models}(K_T x)yz,$$

$$\mathbf{Par}(\mathbf{T}) \vdash \forall rxy.\langle T \rangle^{\models} rxy \supset \langle T \rangle^{=}_3 (J_T r)xy$$

Proof. By induction of construction on T. □

Lemma 57. (Realizability of Conformity) *For a type* $T \in \mathbf{T}$, *there is a term* R *such that* $\mathbf{Par}(\mathbf{T}) \vdash R \models \mathbf{Conf}(T)$.

Proof. By Lemma 56. □

Lemma 58. (Self-Realizability of Extensionality) *For each type* T, *there is a term* R *such that* $\mathbf{Ext}(T) \vdash (R \models \mathbf{Ext}(T))$.

By using these lemmata, we obtain the next theorem.

Theorem 59. *For each type $T \in \mathbf{T}$, there is a term R such that* $\mathbf{Par}(\mathbf{T}) \vdash R \models \mathbf{Par}(T)$.

Corollary 60. $\mathbf{Par}(\mathbf{T})$ *satisfies the term extraction property.*

6 Conclusion

This paper has discussed the system of parametricity of Plotkin and Abadi [7]. We have pointed out three new results.

The first is consistency of the parametricity. The second is that the polyadic parametricities for recursive types are equivalent to each other. The third is a sound realizability interpretation of the theory of parametricity for recursive types.

Acknowledgement
I am greatly indebted to M. Sato for his total support for this work. I am deeply indebted to J. R. Hindley for carefully reading the draft of this paper and checking my English expression. I am very indebted to G. D. Plotkin, and P. Dybjer for many previous discussions and comments. I would like to thank very much T. Ito, and M. H. Takahashi for their encouragement and comments. I would like to thank R. Hasegawa, P. O'Hearn, Y. Akama, D. Suzuki, M. Hasegawa, and the referees for discussions and comments.

References

1. M. Abadi, L. Cardelli and P.-L. Curien. Formal parametric polymorphism. *Theoretical Computer Science,* 121:9–58, 1993.
2. H. Barendregt. Lambda Calculi with Types. In *Handbook of Logic in Computer Science,* Vol. 2, 117–309, Oxford Univ. Press, 1992.
3. R. Bellucci, Martin Abadi, P.-L. Curien. A Model for Formal Parametric Polymorphism: A PER Interpretation for system R. In *Typed Lambda Calculi and Applications,* Lecture Notes in Computer Science, 902, pages 32–46, 1995.
4. J.-Y. Girard, et al., *Proofs and Types* Cambridge Univ. Press, 1989.
5. R. Hasegawa. Parametricity of extensionally collapsed term models of polymorphism and their categorical properties. In *Proceeding of Theoretical Aspects of Computer Science.* Lecture Notes In Computer Science 526, pages 495–512 Springer-Verlag, 1991.
6. R. Hasegawa. Categorical data types in parametric polymorphism. In *Mathematical Structures in Computer Science,* 4:71–109, 1994.
7. G. Plotkin and M. Abadi. A Logic for parametric polymorphism. In *Typed Lambda Calculi and Applications,* Lecture Notes in Computer Science, 664, pages 361–375, 1993.
8. J. C. Reynolds. Types, abstraction and parametric polymorphism. In R. E. A. Mason, editor, *Information Processing 83,* pages 513–523, Elsevier Science Publishers B. V., 1983.

This article was processed using the LaTeX macro package with LLNCS style