<table>
<thead>
<tr>
<th>Title</th>
<th>Galois embedding from universal types into existential types: Extended Abstract (Algorithmic problems in algebra, languages and computation systems)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>Fujita, Ken-etsu</td>
</tr>
<tr>
<td>Citation</td>
<td>数理解析研究所講究録 2006(1503): 121-128</td>
</tr>
<tr>
<td>Issue Date</td>
<td>2006-07</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://hdl.handle.net/2433/58467">http://hdl.handle.net/2433/58467</a></td>
</tr>
<tr>
<td>Type</td>
<td>Departmental Bulletin Paper</td>
</tr>
<tr>
<td>Textversion</td>
<td>publisher</td>
</tr>
</tbody>
</table>

Kyoto University
Galois embedding from universal types into existential types
- Extended Abstract -

Ken-etsu Fujita (藤田 憲悦)
Department of Computer Science,
Gunma University (群馬大学)

Abstract

We show that there exist translations between polymorphic \(\lambda\)-calculus and a subsystem of minimal logic with existential types, which form a Galois insertion (embedding). From a programming point of view, this result means that abstract data types can interpret polymorphic functions under CPS-translation.

1 Introduction

We show that polymorphic types can be interpreted by the use of second order existential types. For this, we prove that there exist translations between polymorphic \(\lambda\)-calculus \(\lambda_2\) and subsystem of minimal logic with existential types, which form a Galois connection and moreover a Galois insertion (embedding). From a programming point of view, this result means that abstract data types can interpret polymorphic functions under the so-called modified CPS-translation [Plot75, SF93].

Our main interest is a neat connection and proof duality between polymorphic types (2nd order universally quantified formulae) and existential types (2nd order existentially quantified formulae). It is logically quite natural like de Morgan's duality, and computationally still interesting, since dual of polymorphic functions with universal type can be regarded as abstract data types with existential type [MP85]. Instead of classical systems like [Pari92, Sel01, Wad03], even intuitionistic systems can enjoy that polymorphic types can be interpreted by existential types. That is, computationally polymorphic function with universal type \(\forall X.A\) can be interpreted by abstract data types with existential type, such that the parametric polymorphic function \(\lambda X.M\) for \(X\) can be viewed, under the so-called CPS-translation *, as an abstract data type for \(X\), which is waiting for an implementation with type \(\exists X.A^*\). This interpretation also contains proof duality, such that the universal formulae introduction rule is interpreted by the use of the existential formulae elimination rule, and the universal
elimination by the existential introduction. Moreover, we established not only a Galois connection but also a Galois insertion (embedding) from polymorphic \( \lambda \)-calculus (Girard-Reynolds) into a calculus with existential types. From the neat connection between the calculi, the fundamental properties such as normalization and Church-Rosser are related each other via CPS-translation and left adjoint.

2 Source calculus: \( \lambda 2 \)

We first introduce our source calculus, 2nd order \( \lambda \)-calculus denoted by \( \lambda 2 \). For simplicity, we adopt its domain-free style.

Definition 1 (Types)

\[ A ::= X | A \Rightarrow A | \forall X.A \]

Definition 2 ((Pseudo)\( \lambda 2 \)-terms)

\[ \Lambda 2 \ni M ::= x | \lambda x.M | \lambda x.M | MM | \lambda X.M | MA \]

Definition 3 (Reduction rules)

(\( \beta_1 \)) \( \lambda x.Mx \rightarrow M \), if \( x \notin FV(M) \)

(\( \beta_2 \)) \( \lambda X.MX \rightarrow M[X := A] \)

(\( \eta \)) \( \lambda X.MX \rightarrow M \), if \( X \notin FV(\lambda \sim) \)

\( FV(M) \) denotes a set of free variables in \( M \).

3 Target calculus: \( \lambda ^3 \)

We next define our target calculus denoted by \( \lambda ^3 \), which is logically a subsystem of minimal logic consisting of constant \( \perp \), negation, conjunction and 2nd order existential quantification.

Definition 4 (Types)

\[ A ::= \perp | X | \neg A | A \wedge A | \exists X.A \]

Definition 5 ((Pseudo)\( \lambda ^3 \)-terms)

\[ \Lambda ^3 \ni M ::= x | \lambda x.M | MM | \langle M, M \rangle | \text{let } (x, x) = M \text{ in } M \]

\[ | \langle A, M \rangle | \text{let } (X, x) = M \text{ in } M \]

Definition 6 (Reduction rules)

(\( \beta \)) \( \lambda x.M_1.M_2 \rightarrow M_1[x := M_2] \)

(\( \eta \)) \( \lambda x.Mx \rightarrow M \), if \( x \notin FV(M) \)
(let) let \( (x_1, x_2) = (M_1, M_2) \) in \( M \rightarrow M[x_1 := M_1, x_2 := M_2] \)

(let_\lambda) let \( (x_1, x_2) = M_1 \) in \( M[z := (x_1, x_2)] \rightarrow M[z := M_1] \),
\[ \text{if } x_1, x_2 \notin FV(M) \]

(let_\exists) let \( (X, x) = (A, M_1) \) in \( M \rightarrow M[X := A, x := M_1] \)

(let_\lambda) let \( (X, x) = M_1 \) in \( M[z := (X, x)] \rightarrow M_2[z := M_1] \),
\[ \text{if } X, x \notin FV(M_2) \]

We write simply (let) for either (let_\lambda) or (let_\exists), and (let_\eta) for (let_\lambda) or (let_\exists).

4 CPS-translation * from \( \Lambda 2 \) into \( \Lambda^3 \)

We define a translation so-called modified CPS-translation * from pseudo \( \lambda 2 \)-terms into pseudo \( \lambda^3 \)-terms, which preserves not only reduction relation but also typing relation introduced later. In each case, a fresh and free variable \( a \) is introduced, which is called a continuation variable.

**Definition 7**

1. \( x^* = xa \)
2. \( (\lambda x.M)^* = \text{let } (x, a) = a \text{ in } M^* \)
3. \( (M_1, M_2)^* = \left\{ \begin{array}{ll}
M_1^*[a := (x, a)] & \text{for } M_2 \equiv x \\
M_1^*[a := (\lambda a.M_2^*, a)] & \text{otherwise}
\end{array} \right. \)
4. \( (\lambda X.M)^* = \text{let } (X, a) = a \text{ in } M^* \)
5. \( (MA)^* = M^*[a := (A^*, a)] \)
6. \( X^* = X; \ (A_1 \Rightarrow A_2)^* = \neg A_1^* \land A_2^*; \ (\forall X.A)^* = \exists X.A^* \)

Remarked that \( M^* \) contains exactly one free occurrence of a continuation variable \( a \), and \( M^* \) has neither \( \beta \)-redex nor \( \eta \)-redex. Let \( \lambda X.M \) have type \( \forall X.A \). Then, under the translation, parametric polymorphic function \( (\lambda X.M)^* \) with respect to \( X \) becomes an abstract data type for \( X \), which is waiting for an implementation with type \( \exists X.A^* \) together with an interface (a signature) with type \( A^* \), i.e.,

\[ \text{abtype } X \text{ with } a : A^* \text{ is } a \text{ in } M^* \]

in a familiar notation.

**Lemma 1**

1. We have \( M_1^*[x := \lambda a.M_2^*] \rightarrow^\beta_\eta (M_1[x := M_2])^* \).
   In particular, \( M_1^*[x := \lambda a.M_2^*] \rightarrow^\beta_\lambda (M_1[x := M_2])^* \) provided that \( M_2 \) is not a variable.
2. If \( M_1 \rightarrow^\beta M_2 \), then \( M_1^* \rightarrow^\beta_{\eta \text{let}} M_2^*. \)
3. If \( M_1 \rightarrow^\eta M_2 \), then \( M_1^* \rightarrow^+_{\eta \text{let}_a} M_2^*. \)
For inverse translation, the following mutual induction defines $Univ$ and $C$, respectively for denotations and for continuations.

\[
\begin{align*}
C_a \in C & \quad \implies \quad xC_a \in Univ \\
C_a \in C & \quad \land \quad P \in Univ & \quad \implies \quad (\lambda a.P)C_a \in Univ
\end{align*}
\]

\[
\begin{align*}
C_a \in C & \quad \land \quad P \in Univ \land \quad \text{let } \langle x, a \rangle = C_a \text{ in } P \in Univ \quad \implies \quad \langle x, C_a \rangle \in C \\
C_a \in C & \quad \land \quad P \in Univ \land \quad \text{let } \langle X, a \rangle = C_a \text{ in } P \in Univ \quad \implies \quad \langle A^*, C_a \rangle \in C
\end{align*}
\]

We write $\langle R_1, R_2, \ldots, R_n \rangle$ for $\langle (R_1, (R_2, \ldots, R_n)) \rangle$ with $n > 1$, and $\langle R_1 \rangle$ for $R_1$ with $n = 1$. $C_a \in C$ is in the form of $\langle R_1, \ldots, R_n, a \rangle$ where $R_i$ $(1 \leq i \leq n)$ is $x$, $\lambda a.P$, or $A^*$ with $n \geq 0$.

According to the general property of Galois connection, for $P \in Univ$, an upper adjoint (left adjoint) $\sharp$ can be defined as follows, where a preorder $\subseteq$ is defined by $\supseteq^*$, the reflexive and transitive closure of one step reduction $\rightarrow$.

\[
P^\sharp \overset{\text{def}}{=} \sup \{ M \in \Lambda 2 \mid M^* \subseteq P \} \quad \text{where } P \subseteq Q \overset{\text{if }}{\rightarrow} Q \rightarrow^* P.
\]

In fact, this definition works well, which can be verified by case analysis on $P \in Univ$, in the following recursive way:

- **Case $P \equiv xC = x(R_1, \ldots, R_n, a)$ with $n \geq 0$**
  
  From the definition of $^*$, $P^\sharp$ is in the form of $xM_1 \ldots M_n$ for some $M_i$, where
  
  - If $R_i \equiv x_i$, then $M_i \equiv x_i$.
  - If $R_i \equiv \lambda a.P_i$, then find similarly $M_i$ such that $M_i^* \subseteq P_i$.
  - If $R_i \equiv A_i^*$, then $M_i \equiv A_i$.

- **Case $P \equiv (\lambda a.P')C'$**
  
  We have no $M$ such that $M^* \equiv (\lambda a.P')C$. Then we should find $M$ such that $M^* \subseteq P'[a := C] \subseteq (\lambda a.P')C$, where $a$ is a linear variable.

- **Case $P \equiv \text{let } \langle x, a \rangle = C \text{ in } P'$ with $C = \langle R_1, \ldots, R_n, a \rangle$ $(n \geq 0)$**
  
  $P^\sharp$ is in the form of $(\lambda x.M)N_1 \ldots N_n$ for some $M$ and $N_i$, where we should find $M$ such that $M^* \subseteq P'$, and:
  
  - If $R_i \equiv x_i$ then $N_i \equiv x_i$.
  - If $R_i \equiv \lambda a.P_i$ then find $N_i$ such that $N_i^* \subseteq P_i$.
  - If $R_i \equiv A_i^*$ then $N_i \equiv A_i$.
We have a valid induction measure, since continuation variable is linear and we always choose strictly smaller subterms to find an upper adjoint. This definition is summarized inductively as follows, where we write $C[ ]$ for $C \in C$ with a hole $[ ]$.

**Definition 8**

1. $(xC)^{\dagger} = C^{\dagger}[x^{\dagger}]$
2. $((\lambda a.P)C)^{\dagger} = C^{\dagger}[(\lambda a.P)^{\dagger}]$
3. $(\text{let } \langle x,a \rangle = C \text{ in } P)^{\dagger} = C^{\dagger}[\lambda x.P^{\dagger}]$
4. $(\text{let } \langle X,a \rangle = C \text{ in } P)^{\dagger} = C^{\dagger}[\lambda X.P^{\dagger}]$
5. $a^{\dagger} = [ ]$
6. $\langle x,C \rangle^{\dagger} = C^{\dagger}[[x^{\dagger}]$
7. $(\lambda a.P, C)^{\dagger} = C^{\dagger}[[\lambda a.P)^{\dagger}]$
8. $(A^{\dagger}, C)^{\dagger} = C^{\dagger}[[A^{\dagger})^{\dagger}]$

Note that $C_{a}^{\dagger} = [ ]R_{1}^{\dagger} \ldots R_{n}^{\dagger}$ with left associativity, if $C_{a} \in C$ is in the form of $\langle R_{1}, \ldots, R_{n}, a \rangle.$ We also remark that if we have $P[a := C]$, then there uniquely exists a free occurrence of $a$ in $P \in \text{Univ}.$

**Lemma 2**

1. $(P[a := C])^{\dagger} \equiv C^{\dagger}[P^{\dagger}]$

2. Let $P, P_{0} \in \text{Univ}$ and $C \in C$.
   
   $(P[x := \lambda a.P_{0}])^{\dagger} = P^{\dagger}[x := P_{0}^{\dagger}]$
   
   $(C[x := \lambda a.P_{0}])^{\dagger} = C^{\dagger}[x := P_{0}^{\dagger}]$

**Proposition 1** Let $P_{1}, P_{2} \in \text{Univ}$.

1. If $P_{1} \rightarrow_{\beta} P_{2}$, then $P_{1}^{\dagger} \equiv P_{2}^{\dagger}$.
2. If $P_{1} \rightarrow_{\eta} P_{2}$, then $P_{1}^{\dagger} \equiv P_{2}^{\dagger}$.
3. If $P_{1} \rightarrow_{\text{let}} P_{2}$, then $P_{1}^{\dagger} \rightarrow_{\beta} P_{2}^{\dagger}$.
4. If $P_{1} \rightarrow_{\text{let}}, P_{2}$, then $P_{1}^{\dagger} \rightarrow_{\eta} P_{2}^{\dagger}$.

**Proposition 2** Let $M \in \Lambda 2$ and $P \in \text{Univ}$.

1. $M^{*} \equiv M$ and $P \rightarrow^{*}_{\beta, \eta} P^{*}$

2. If $M$ is in $\lambda 2$-normal, then $M^{*}$ is in $\lambda^{3}$-normal.
   
   If $P$ is in $\lambda^{3}$-normal, then $P^{*}$ is in $\lambda 2$-normal.
Theorem 1 (Galois insertion) \( \langle \Lambda 2, \text{Univ}, \ast, \# \rangle \) forms a Galois connection, in particular Galois insertion such that \( M^{\ast} \equiv M \). That is, let \( M, M_1, M_2 \in \Lambda 2 \) and \( P, P_1, P_2 \in \text{Univ} \). Then we have the following properties:

1. If \( M_1 \rightarrow^{*}_{\lambda 2} M_2 \) then \( M_1^{*} \rightarrow^{*}_{\text{Univ}} M_2^{*} \).
2. If \( P_1 \rightarrow^{*}_{\text{Univ}} P_2 \) then \( P_1^{*} \rightarrow^{*}_{\lambda 2} P_2^{*} \).
3. \( M^{\ast} \rightarrow^{*}_{\lambda 2} M \) and \( P \rightarrow^{*}_{\text{Univ}} P^{*} \).

In other words:

\[ P \rightarrow^{*}_{\text{Univ}} M^{*} \text{ if and only if } P^{*} \rightarrow^{*}_{\lambda 2} M \]

Corollary 1

1. Strong normalization of \( \text{Univ} \) implies that of \( \lambda 2 \).
2. \( \lambda 2 \) is weakly normalizing iff \( \text{Univ} \) is weakly normalizing.
3. There exists a one-to-one correspondence between \( \lambda 2 \)-normal forms and \( \text{Univ} \)-normal forms.
4. \( \lambda 2 \) is Church-Rosser iff \( \text{Univ} \) is Church-Rosser.

We remark that \( \lambda^{3} \) itself is NOT Church-Rosser.

5. Let \( \downarrow P \overset{\text{def}}{=} \{ Q \in \text{Univ} \mid P \rightarrow^{*}_{\lambda 3} Q \} \) for \( P \in \text{Univ} \). Then an inverse image of \( \downarrow P \) is principal, in the sense that the inverse image of \( \downarrow P \) is equal to \( \downarrow (P^{4}) \), that is, generated by \( P^{4} \).

6. Let \( \downarrow_{\lambda 3}[\Lambda 2] \overset{\text{def}}{=} \{ P \mid M^{*} \rightarrow^{*}_{\lambda 3} P \text{ for some } M \in \Lambda 2 \} \).

Let \( \uparrow_{\beta \eta}[\Lambda 2] \overset{\text{def}}{=} \{ P \in \text{Univ} \mid P \rightarrow^{*}_{\beta \eta} M^{*} \text{ for some } M \in \Lambda 2 \} \).

Then we have \( \downarrow_{\lambda 3}[\Lambda 2] \subseteq \uparrow_{\beta \eta}[\Lambda 2] = \text{Univ} \).

5 Proof duality

Finally, we give sets of type assignment rules for \( \lambda 2 \) and \( \lambda^{3} \), respectively, as follows.

\[ \lambda 2: \]

\[ x : A \in \Gamma \]

\[ \begin{array}{c}
\Gamma \vdash x : A \\
\end{array} \]

\[ \begin{array}{c}
\Gamma, x : A_1 \vdash M : A_2 \\
\end{array} \]

\[ \Gamma \vdash \lambda x : A_1. M : A_1 \Rightarrow A_2 \quad (\Rightarrow I) \]

\[ \begin{array}{c}
\Gamma \vdash M_1 : A_1 \Rightarrow A_2 \\
\Gamma \vdash M_2 : A_2 \\
\end{array} \]

\[ \Gamma \vdash \lambda x^* \rightarrow^{*}_{\text{Univ}} M_2^{*} \Rightarrow A_1 \quad (\Rightarrow E) \]

\[ \begin{array}{c}
\Gamma \vdash \lambda x. M : \forall x. A \\
\end{array} \]

\[ \begin{array}{c}
\Gamma \vdash \lambda x. M : A[X := A_1] \\
\end{array} \]

\[ (\forall I)^{\ast} \]

\[ (\forall E) \]
where $(\forall I)^*$ denotes the eigenvariable condition $X \notin FV(\Gamma)$.

\[ \lambda^3 : \]

\[ \frac{x : A \in \Gamma}{\Gamma \vdash x : A} \]

\[ \frac{\Gamma, x : A \vdash M : \bot}{\Gamma \vdash \lambda x : A. M : \neg A} (\neg I) \]

\[ \frac{\Gamma \vdash M_1 : \neg A \quad \Gamma \vdash M_2 : A}{\Gamma \vdash M_1 M_2 : \bot} (\neg E) \]

\[ \frac{\Gamma \vdash M_1 : A_1 \quad \Gamma \vdash M_2 : A_2}{\Gamma \vdash \langle M_1, M_2 \rangle : A_1 \wedge A_2} (\wedge I) \]

\[ \frac{\Gamma \vdash \Lambda I_1 : A_1 \quad \Gamma, x_1 : A_1, x_2 : A_2 \vdash M : A}{\Gamma \vdash \langle x_1, r_2, \rangle = \Lambda f_1 \in A f : A} (\wedge E) \]

\[ \frac{\Gamma \vdash \Lambda J_1 : A_1 \wedge A_2}{\Gamma \vdash 1 \mathrm{e} \mathrm{t} \langle x_1, r_2, \rangle = \Lambda f_1 \in A f : A} (\exists I) \]

\[ \frac{\Gamma \vdash \Lambda J_1 : A_1 \wedge A_2 \quad \Gamma, x_1 : A_1, x_2 : A_2 \vdash M : A}{\Gamma \vdash \langle x_1, r_2, \rangle = \Lambda f_1 \in A f : A} (\exists F_{d})^* \]

where $(\exists E)^*$ denotes the eigenvariable condition $X \notin FV(\Gamma, A_1)$.

**Proposition 3** $\Gamma \vdash_{\chi^2} M : A$ if and only if $\neg \Gamma^*, a : A^* \vdash_{\lambda^3} M^* : \bot$

**Theorem 2 (Proof duality)** Let $\Pi$ be a normal deduction of $\Gamma \vdash_{\chi^2} M : A$, and in $\Pi$, let $\pi$ be a path:

\[ A_1(E_1)A_2(E_2)\ldots A_i(E_i)A_{i+1}(I_{i+1})\ldots A_{n-1}(I_{n-1})A_n. \]

Then, in the deduction of $\neg \Gamma^*, a : A^* \vdash_{\lambda^3} M^* : \bot$, there exists a path $\pi^*$ as follows:

\[ A_1^*(E_1)^*A_2^*(E_2)^*\ldots A_{i+1}^*(E_{i+1})^*A_i^*\ldots A_{n-1}^*(E_{n-1})^*A_n^* \]

under the following correspondence:

\[ (\Rightarrow I)^* = (\wedge E); (\Rightarrow E)^* = (\forall I); (\forall I)^* = (\exists I); (\forall E)^* = (\exists E)^*. \]

**References**


