# PT 関数の逆関数を定義する TRS の生成

Generation of a TRS Implementing the Inverses of Pure Treeless Functions

西田 直樹　酒井 正彦　坂部 俊樹
Naoki NISHIDA　Masahiko SAKAI　Toshiki SAKABE

名古屋大学大学院工学研究科
Graduate School of Engineering, Nagoya University
nishida@sakabe.nuie.nagoya-u.ac.jp　{sakai, sakabe}@nuie.nagoya-u.ac.jp

本稿では，パターン照合の機能を持ち，右辺に入れ子の関数呼び出しを持たない pure treeless 関数定義から指定された関数の逆関数の定義を持つ条件付き項書換え系を生成するアルゴリズムを提案する．さらに，PT 関数定義の右辺が線形のとき，本アルゴリズムにより生成された条件付き項書換え系を項書換え系に変換する方法を示す．

## 1 はじめに

逆向きの計算を行う手法として，関数型言語には，“E-unification”[1, 6] や “Narrowing”[1, 7, 5]，“Inversion Algorithm”[3] などの方法がある．これらの方法は，規則集合と計算結果を与えることによって代入を求めるアルゴリズムであり，解を求める際に，毎回アルゴリズムが実行される．また，計算に対して適用されうるあらゆる規則から代入の可能性のある値を取り出していく．そのため，これらの方法は効率的であるとはいい難いが，2 つ以上の未知変数に対してもあらゆる解を求めることができる．

多くの分野では，逆関数を利用して逆向きの計算を行う方法が一般的である．関数型言語においても逆関数を利用できれば，あらかじめ逆関数を求めておき，それを利用して簡単に何度でも計算を行える．

本稿では，パターン照合の機能を持ち，規則の右辺に入れ子の関数呼び出しがない pure treeless 関数定義 [2] を対象として，関数の効率化を行うフュージョン変換と呼ばれるプログラム変換の考え方を発展させて，プログラム変換により，関数型プログラムから指定された関数の逆関数を持つ条件付き項書換え系を生成するアルゴリズムを提案する．また，このアルゴリズムが必ず停止し，条件付き項書換え系を出力することを示すとともに，提案したアルゴリズムにより得られた関数がその逆関数になっていることを証明する．さらに，PT 関数定義が右線形のとき，本アルゴリズムにより生成された条件付き項書換え系は，項書換え系に容易に変換できることを示す．

## 2 準備

本稿では，項書換え系の一般的な記法に従う [1, 4]．

抽象書換え系は $\mathcal{A} = (\mathcal{S}, \to)$ である．ここで，$\mathcal{S}$ はある集合，$\to$ は $\mathcal{S}$ 上の 2 項関係（書換え関係）である．書換え系列は有限のシーケンス $x_0 \to x_1 \to \cdots \to x_n$ $(n \geq 0)$，あるいは，無限のシーケンス $x_0 \to x_1 \to \cdots$ である．$x, y \in \mathcal{S}$ が同一であるとき，$x \equiv y$ と書く．$x \to y$ となる $y$ が存在しないとき，$x$ は $\mathcal{A}$ の正規形であるという．$\to$ の推移反射閉包を $\xrightarrow{*}$ と書く．$\xrightarrow{k}$，$\xrightarrow{k\geq}$ はそれぞれ $k$ ステップの書換え，$k$ 以下のステップの書換えを表す．任意の $x$ , $y$ , $y' \in \mathcal{S}$ について，$x \xrightarrow{*} y$ かつ $x \xrightarrow{*} y'$ ならば，$z \in \mathcal{S}$ が存在して $y \xrightarrow{*} z$ かつ $y' \xrightarrow{*} z$ が成立するとき，$\mathcal{A}$ は合流性を持つという．

関数記号，変数の集合をそれぞれ $F$，$X$ とする．$F \cup X$ の記号から構成される項は写像 $arity : F \to N$（$N$ は自然数の集合）を用いて，次の (1), (2) を用いて再帰的に定義される．(1) 変数 $x$ $(\in X)$ は項である．(2) $f \in F$, $arity(f) = n$ で $t_1, ..., t_n$ が項のとき，$f(t_1, ..., t_n)$ は項である．ただし，$arity(f) = 0$ のときは $f$ を項とする．$arity(f) = 0$ である関数記号 $f$ を定数と呼ぶ．項の集合を $T(F \cup X)$（あるいは単に $T$）で表す．また，項 $t$ に出現する変数の集合を $\mathcal{V}ar(t)$ で表す．一般に $n$ 個の関数記号 $f$ の入れ子 $\overbrace{f(\cdots f}^{n}(t)\cdots)$ を $f^n(t)$ と書く．項中のどの変数も高だか 1 回しか現れないとき，その項は線形であるという．

$\Box$ を特別な定数とする．$C[, ..., ]$ と記述される項 $C \in T(F \cup X \cup \{\Box\})$ を文脈と呼ぶ．$n$ 個の $\Box$ を含む

$C[,...,]$ と $t_1,...,t_n \in T(F \cup X)$ に対して，□ を左から順に $t_1,...,t_n$ に置き換えて得られる項を $C[t_1,...,t_n]$ で表す．□ が 1 個の文脈 $C$ を $C[\ ]$ と書く．

代入は変数から項への写像である．変数 $x_1,...,x_n$ をそれぞれ項 $u_1,...,u_n$ に移す代入 $\sigma$ を $\{x_1 \mapsto u_1,...,x_n \mapsto u_n\}$ と表す．また，項 $t$ 中の各変数 $x_i$ を項 $u_i$ に置き換えて得られる項を $t\sigma$ で表す．

$l \to r \Leftarrow \mathcal{C}ond$ は条件付き書換え規則と呼ばれる．ここで，$l \notin X$ と $r$ は項，$\mathcal{C}ond$ は $true$ もしくは $l_1 \to r_1 \wedge \cdots \wedge l_n \to r_n\ (n \geq 1)$ の形式であり，任意の $i$ について，$\mathcal{V}ar(l) \cup (\bigcup_{i=1}^{n} \mathcal{V}ar(l_i) \cup \mathcal{V}ar(r_i)) \supseteq \mathcal{V}ar(r)$ を満たす．$\sigma$ を代入，$\to$ を項上の関係とするとき，すべての $i$ について $l_i\sigma \xrightarrow{*} r_i\sigma$ ならば，$\sigma$ と $\to$ は $\mathcal{C}ond$ を満たすといい，$\mathcal{C}ond(\sigma, \to)$ と書く．条件付き書換え規則の集合を $R$ で表し，$(T(F \cup X), \to_R)$ を**条件付き項書換え系**（CTRS）と呼ぶ．以下では混乱がない限り，CTRS $(T(F \cup X), \to_R)$ を条件付き書換え規則の集合 $R$ のみで表す．ここで，$\to_R$ は次のように定義されるレベル $n$ の書換え $\underset{n}{\to}$ を用いて $\to_R = \bigcup_{n=0}^{\infty} \underset{n}{\to}$ と定義される．

- $\underset{0}{\to} = \emptyset$ .
- $s \underset{n}{\to} t$ ならば，$s \underset{n+1}{\to} t$ .
- $l \to r \Leftarrow \mathcal{C}ond \in R$ かつ $\mathcal{C}ond(\sigma, \underset{n}{\to})$ ならば，$C[l\sigma] \underset{n+1}{\to} C[r\sigma]$ .

特に，条件付き書換え規則 $l \to r \Leftarrow true$ は，単に $l \to r$ と書くことがあり，これを書換え規則と呼ぶ．$R$ が書換え規則のみからなる CTRS を項書換え系（TRS）と呼ぶ．TRS においては，任意の $i \geq 1$ について，$\to_R = \underset{i}{\to}$ が成り立つ．どの条件付き書換え規則の左辺も線形であるとき，CTRS は左線形であるという．2 つの規則 $l \to r \Leftarrow \mathcal{C}ond$ , $l' \to r' \Leftarrow \mathcal{C}ond'$ について，$l\sigma \equiv s\sigma'$ を満たす $l'$ の部分項 $s$ と代入 $\sigma, \sigma'$ が存在するとき，これらの規則は重なるという．左線形かつどの 2 つの規則も重ならない CTRS は直交するという．

## 3 逆関数の生成

### 3.1 Pure Treeless 関数

関数記号の集合 $F$ は，被定義記号の集合 $F_D$ と構成子記号の集合 $F_C$ に分割されているものとする．すなわち，$F = F_D \oplus F_C$ とする．本稿では，被定義記号として $f, g, h$ ，構成子記号として $c, d, e$ ，変数として $x, y, z$ を使用する．

次のように再帰的に定義される項を **pure treeless 項**（PT 項）と呼ぶ．

$$PT\text{ 項} ::= \text{変数} \mid c(PT\text{ 項},...,PT\text{ 項}) \mid f(\text{変数},...,\text{変数})$$

PT 項中の被定義記号の引数はすべて変数であるので，PT 項は被定義記号が現れない文脈 $C \in T(F_C \cup X \cup \{\Box\})$ を用いて，$C[f_1(\text{変数},...,\text{変数}),...,f_n(\text{変数},...,\text{変数})]$ と表現できる．

1 つの構成子記号と変数からなる項をパターン項と呼び，次のように定義する．

$$\text{パターン項} ::= c(\text{変数},...,\text{変数})$$

**pure treeless 関数定義**（PT 関数定義）は書換え規則の集合である．それぞれの関数は，次のタイプ 1 もしくはタイプ 2 の形式で定義される[1] .

・タイプ 1　$f(x_1,...,x_n) \to t$

・タイプ 2　$g(p_1,x_1,...,x_n) \to t_1$
$\vdots$
$g(p_m,x_1,...,x_n) \to t_m$

ここで，$t, t_1,...,t_n$ は PT 項，$p_1,...,p_m$ はそれぞれ異なる構成子記号を持つパターン項である．また，左辺項は線形とする．このように，PT 関数定義は制限付きの直交な TRS である．直交な TRS は合流性を持つ [1] ため，PT 関数定義は合流性を持つ．PT 関数定義で定められる被定義記号を PT 関数と呼ぶ．

**例 1** 加算を行う $PT$ 関数 $add$ は次のように定義される．

$$R_1 = \{\ add(0,x_2) \to x_2,\ add(s(x_1),x_2) \to s(add(x_1,x_2))\ \}\ \Box$$

### 3.2 逆関数生成の基本となる考え方

1 引数のタイプ 1 で定義される PT 関数の規則

$$f(x) \to C[f_1(x),...,f_m(x)]$$

について考える．ここで，$C \in T(F_C \cup X \cup \{\Box\})$ である．

[1] 文献 [2] では，タイプ 1 を pure treeless 関数，タイプ 2 を g-type パターン照合関数と呼んでいるが，本稿ではこれらをまとめて pure treeless 関数と呼ぶ．

まず，右辺のすべての $f_i(x)$ をそれぞれ新しい変数 $y_i$ に置き換えて，両辺に $f^{-1}$ を施すと，逆関数の一般的性質 “ $f^{-1}(f(n)) = n$ ” から，左辺は $x$ になる．左右を入れ換えると，

$$f^{-1}(C[y_1, ..., y_m]) \to x$$

が得られる．このとき，$x = f_1^{-1}(y_1)$, ..., $x = f_m^{-1}(y_m)$ であるので，$x$ を $y_1, ..., y_m$ を用いて表せない．そこで，これらを規則の条件とすることにより，逆関数の条件付き書換え規則

$$f^{-1}(C[y_1, ..., y_m]) \to x \;\Leftarrow\; \bigwedge_{i=1}^{m} f_i^{-1}(y_i) \to x$$

が得られる．

### 3.3 アルゴリズム

まず，関数 $f$ の計算に必要となる関数の集合を定義する．

**定義 1** *PT* 関数定義 $(T((F_D \oplus F_C) \cup X), \to_R)$，関数記号の集合 $F_0 \subseteq F_D$ について，$F_0$ 中の関数が依存する関数の集合 $G$ は，次のように再帰的に定義される．

1. $G := F_0$
2. $G := G \cup \{h \in F_D \mid f \in G,$
   $f(...) \to C[..., h(x_1, ..., x_n), ...] \in R\ \}$

このとき $G$ を $\mathcal{D}ep(F_0)$ と書く． □

以下では，3.2 節で考案した手法をアルゴリズム $\mathcal{I}nv$ として形式化する．ここで，多引数関数の逆関数は複数の項を返すのでタプルを導入する．項 $t_1, ..., t_n$ のタプルを，新しい構成子記号 $tp_n$ を用いて $tp_n(t_1, ..., t_n)$ と書くことにする．要素が 1 個のタプル $tp_1(t)$ は単に $t$ とも書く．また，引数がすべて変数のタプル $tp_n(x_1, ..., x_n)$ と第 1 引数がパターン項であるタプル $tp_n(c(x'_1, ..., x'_j), x_2, ..., x_n)$ を $n$-変数タプルと呼ぶ．

アルゴリズム $\mathcal{I}nv$ は，PT 関数定義 $R$，逆関数を求めたい関数記号の集合 $F_0 \subseteq F_D$ から，CTRS を生成する．

$$\begin{aligned}&\mathcal{I}nv(R, F_0) \;=\\ &\quad \{\ h^{\#}(s) \to tp_n(t_1, x_2, ..., x_n) \;\Leftarrow\; Cd\\ &\qquad\qquad \mid\ h \in \mathcal{D}ep(F_0),\\ &\qquad\qquad\ \ h(t_1, x_2, ..., x_n) \to t \in R,\\ &\qquad\qquad\ \ \mathcal{I}nv\mathcal{R}ule(t) = \langle\ s\ ;\ Cd\ \rangle\ \}\end{aligned}$$

ここで，$\mathcal{I}nv\mathcal{R}ule$ は，PT 項 $t$ を入力とし，項 $s$ と書換え条件 $Cd$ の並び，$\langle\ s\ ;\ Cd\ \rangle$ を出力する手続きである．$s$ は，$\mathcal{C}nd := true$ を初期値として，$\mathcal{T}[\![t]\!]$ に $\mathcal{T}[\![...]\!]$ が消えるまで次の変換を適用して得られる項である．また，このときの $\mathcal{C}nd$ の式が $Cd$ である．

- $t \equiv x$ のとき，

$$\mathcal{T}[\![x]\!] \;\Rightarrow\; x$$

- $t \equiv c(t_1, ..., t_n)$ のとき，

$$\mathcal{T}[\![c(t_1, ..., t_n)]\!] \;\Rightarrow\; c(\mathcal{T}[\![t_1]\!], ..., \mathcal{T}[\![t_n]\!])$$

- $t \equiv f(x_1, ..., x_n)$ のとき，

$$\begin{cases} \mathcal{T}[\![f(x_1, ..., x_n)]\!] \;\Rightarrow\; y\\ \mathcal{C}nd := \mathcal{C}nd \wedge f^{\#}(y) \to tp_n(x_1, ..., x_n)\end{cases}$$

ここで，$y$ は新しい変数とする．

条件付き書換え規則の集合 $\mathcal{I}nv(R, F_0)$ が有限であること，手続き $\mathcal{I}nv\mathcal{R}ule$ における $\mathcal{T}$ の計算が必ず停止することは明らかでる．

**例 2** 例 *1* の *PT* 関数定義 $R_1$ に対して加算 $add$ の逆関数を求める．$R_2 = \mathcal{I}nv(R_1, \{add\})$ とすると，$R_2$ は次のようになる．

$$\begin{aligned}R_2 \;=\; \{\;& add^{\#}(x_2) \to tp_2(0, x_2),\\ & add^{\#}(s(y)) \to tp_2(s(x_1), x_2)\\ & \qquad \Leftarrow\; add^{\#}(y) \to tp_2(x_1, x_2)\ \}\end{aligned}$$ □

### 3.4 アルゴリズムの正当性

本節では，アルゴリズムにより得られた CTRS が逆関数になっていることを証明する．

**命題 1** $R$ を *PT* 関数定義とする．このとき，$\mathcal{D}ep(F_0)$ に含まれるすべての $f$ に対して，$f(...) \to r \in R$ ならば，$r$ 中に出現する被定義記号は $\mathcal{D}ep(F_0)$ に属する．

**証明** $\mathcal{D}ep(F_0)$ の作り方より明らか． □

**定理 1** $R$ を $PT$ 関数定義，$R' = \mathcal{I}nv(R, F_0)$ とする．このとき，任意の $f \in \mathcal{D}ep(F_0)$ と $t_1, ..., t_n$, $s \in T(F_C)$ について，$f(t_1, ..., t_n) \xrightarrow{*}_R s$ ならば，$f^{\#}(s) \to_{R'} tp_n(t_1, ..., t_n)$ である．

**証明** 書換え系列 $f(t_1, ..., t_n) \xrightarrow{k}_R s$ のステップ数 $k$ に関する帰納法により証明できる [8]. □

**例 3** 例 *2* で $R_1$ から生成した $R_2$ について考える．$add(s(0), s^2(0)) \xrightarrow{*}_{R_1} s^3(0)$ を考えたとき，定理 *1* より，

$$add^{\#}(s^3(0)) \to_{R_2} tp_2(s(0), s^2(0)) \qquad (1)$$

である．実際，$add^{\#}(x_2) \to_{R_2} tp_2(0, x_2)$ より，$add^{\#}(s^2(0)) \to_{R_2} tp_2(0, s^2(0))$ が示せる．これより，規則 $add^{\#}(s(y)) \to tp_2(s(x_1), x_2) \Leftarrow add^{\#}(y) \to tp_2(x_1, x_2)$ が適用できるので，式 *(1)* がいえる． □

PT 関数は合流性を持つが，アルゴリズムによって得られた CTRS は合流性を持つとは限らない．実際，多対 1 関数では，$s \neq s'$ かつ $s, s' \in T(F_C)$ について，$f(s) \equiv f(s')\ (\equiv t)$ であるが，定理 1 より，$f^{\#}(t) \to s$ かつ $f^{\#}(t) \to s'$ となり，合流性を持たない．

### 3.5 TRS への変換

すべての規則の右辺がそれぞれ線形である TRS を右線形であるという．本節では，右線形である PT 関数定義から 3.3 節のアルゴリズムにより生成された CTRS を TRS に変換する方法について述べる．

**命題 2** $R$ を $PT$ 関数定義とする．このとき，$R' = \mathcal{I}nv(R, F_0)$ 中の条件付き書換え規則 $f^{\#}(t) \to r \Leftarrow \bigwedge_{i=1}^{n} f_i^{\#}(y_i) \to r_i$ 中の変数は次の性質を満たす．ここで，$f, f_i \in \mathcal{D}ep(F_0)$, $t \in T(F_C \cup X)$ ，$r$, $r_i$ はそれぞれ $m$-変数タプル（$m = arity(f)$），$m_i$-変数タプル（$m_i = arity(f_i)$）である．

- $\mathcal{V}ar(t) \cap \bigcup_{i=1}^{n} \mathcal{V}ar(r_i) = \emptyset$
- $\forall i,\ y_i \in \mathcal{V}ar(t),\ y_i \notin \mathcal{V}ar(r)$
- $(\mathcal{V}ar(t) - \{y_1, ..., y_n\}) \cup \bigcup_{i=1}^{n} \mathcal{V}ar(r_i) = \mathcal{V}ar(r)$

また，$R$ が右線形であるとき，次の性質も満たす．

$$\forall i, j,\ i \neq j \Rightarrow \mathcal{V}ar(r_i) \cap \mathcal{V}ar(r_j) = \emptyset$$

**証明** $R'$ の作り方と $R$ が右線形であることより明らか． □

PT 関数定義 $R$ が右線形であるとする．このときアルゴリズムによって生成される CTRS $R'$ $(= \mathcal{I}nv(R, F_0))$ を次の手順で TRS $R''$ に変換できる．

1. $R'' := \{\ l \to r \Leftarrow true \mid l \to r \Leftarrow true \in R'\ \}$
2. $R'$ 中のすべての条件付き書換え規則 $f^{\#}(t) \to r \Leftarrow \bigwedge_{i=1}^{n} f_i^{\#}(y_i) \to r_i$ （$n \geq 1$）に対して，規則ごとに新しい関数記号 $f_{new}$ を用意し，次のように $R''$ に書換え規則を追加する．

$$\begin{aligned} R'' := R'' \cup \{ & \\ & f^{\#}(t) \to f_{new}(z_1, ..., z_j, f_1^{\#}(y_1), ..., f_n^{\#}(y_n)), \\ & f_{new}(z_1, ..., z_j, r_1, ..., r_n) \to r\ \ \} \end{aligned}$$

ここで，$t$ 中の変数を $\mathcal{V}ar(t) - \{y_1, ..., y_n\} = \{z_1, ..., z_j\}$ とする．

この変換により，CTRS $R'(= \mathcal{I}nv(R, F_0))$ から TRS $R''$ が得られたとき，$R'' = \mathcal{D}el\mathcal{C}ond(R')$ と書く．

**定理 2** $R$ を右線形な $PT$ 関数定義，$R' = \mathcal{I}nv(R, F_0)$，$R'' = \mathcal{D}el\mathcal{C}ond(R')$ とする．このとき，任意の $f \in \mathcal{D}ep(F_0)$ と $t$ , $t_1, ..., t_m \in T(F_C)$ について，次の関係が成り立つ．ここで，$m = arity(f)$ とする．

$$\begin{aligned} & f^{\#}(t) \to_{R'} tp_m(t_1, ..., t_m) \\ & \quad \overset{\text{iff}}{\Longleftrightarrow}\ f^{\#}(t) \xrightarrow{*}_{R''} tp_m(t_1, ..., t_m) \end{aligned}$$

**証明** $\Rightarrow$). 書換え系列 $f^{\#}(t) \underset{k}{\to}_{R'} tp_m(t_1, ..., t_m)$ のレベル $k$ に関する帰納法により証明する．

- $f^{\#}(u) \to r \in R'$ で書換えられたとき，$f^{\#}(u) \to r \in R''$ より，$f^{\#}(t) \to_{R''} tp_m(t_1, ..., t_m)$.
- $f^{\#}(u) \to r \Leftarrow \bigwedge_{i=1}^{n} f_i^{\#}(y_i) \to r_i \in R'$ で書換えられたとき，ある代入 $\sigma$ が存在して，$t \equiv u\sigma$, $tp_m(t_1, ..., t_m) \equiv r\sigma$, $f_i^{\#}(y_i)\sigma \underset{k-1}{\to}_{R'} r_i\sigma$ である．帰納法の仮定より，

$$f_i^{\#}(y_i)\sigma \xrightarrow{*}_{R''} r_i\sigma\ \ (1 \leq i \leq n) \qquad (2)$$

が成り立つ．

また，$\mathcal{V}ar(u) - \{y_1, ..., y_n\} = \{z_1, ..., z_j\}$ とすると，$f^{\#}(u) \to f_{new}(z_1, ...z_j, f_1^{\#}(y_1), ..., f_n^{\#}(y_n)) \in R''$, $f_{new}(z_1, ..., z_j, r_1, ..., r_n) \to r \in R''$ である．よって，これらの規則と式 (2) より，

$$\begin{aligned} f^{\#}(t) \quad & \equiv f^{\#}(u)\sigma \\ & \to_{R''} f_{new}(z_1, ..., z_j, \\ & \qquad f_1^{\#}(y_1), ..., f_n^{\#}(y_n))\sigma \\ & \equiv f_{new}(z_1\sigma, ..., z_j\sigma, \\ & \qquad f_1^{\#}(y_1)\sigma, ..., f_n^{\#}(y_n)\sigma) \\ & \xrightarrow{*}_{R''} f_{new}(z_1\sigma, ..., z_j\sigma, r_1\sigma, ..., r_n\sigma) \\ & \equiv f_{new}(z_1, ..., z_j, r_1, ..., r_n)\sigma \\ & \to_{R''} r\sigma \equiv tp_m(t_1, ..., t_m) \end{aligned}$$

⇐). 書換え系列 $f^{\#}(t) \xrightarrow{k}_{R''} tp_m(t_1,...,t_m)$ のステップ数 $k$ に関する帰納法により証明する．最初に規則 $f^{\#}(u) \to r$ で書換えられたとする．このとき，$f^{\#}(t) \to_{R''} t' \xrightarrow{k-1}_{R''} tp_m(t_1,...,t_m)$ である．

- $r$ が $m$-変数タプルのとき，$t'$ は $R''$ の正規形であるから，$t' \equiv tp_m(t_1,...,t_m)$ である．このとき，$f^{\#}(u) \to r \in R'$ であるから，$f^{\#}(t) \to_{R'} t' \equiv tp_m(t_1,...,t_m)$ である．
- $r \equiv f_{new}(z_1,...,z_j,f_1^{\#}(y_1),...,f_n^{\#}(y_n))$ のとき，$f^{\#}(u) \to r_0 \Leftarrow \bigwedge_{i=1}^{n} f_i^{\#}(y_i) \to r_i \in R'$，$f_{new}(z_1,...,z_j,r_1,...,r_n) \to r_0 \in R''$ である．よって，ある代入 $\sigma$ が存在して，

$$\begin{aligned} f^{\#}(t) &\equiv f^{\#}(u)\sigma \\ &\to_{R''} f_{new}(z_1,...,z_j, \\ &\qquad f_1^{\#}(y_1),...,f_n^{\#}(y_n))\sigma \\ &\equiv f_{new}(z_1\sigma,...,z_j\sigma, \\ &\qquad f_1^{\#}(y_1)\sigma,...,f_n^{\#}(y_n)\sigma) \\ &\xrightarrow{*}_{R''} f_{new}(z_1\sigma,...,z_j\sigma,r_1\sigma,...,r_n\sigma) \\ &\equiv f_{new}(z_1,...,z_j,r_1,...,r_n)\sigma \\ &\to_{R''} r_0\sigma \equiv tp_m(t_1,...,t_m) \end{aligned}$$

である．よって，

$$f_i^{\#}(y_i)\sigma \xrightarrow{k-2\geq}_{R''} r_i\sigma \quad (1 \leq i \leq n)$$

が成り立つ．これより，帰納法の仮定から，

$$f_i^{\#}(y_i)\sigma \to_{R'} r_i\sigma \quad (1 \leq i \leq n) \qquad (3)$$

が成り立つ．式 (3) より，規則 $f^{\#}(u) \to r_0 \Leftarrow \bigwedge_{i=1}^{n} f_i^{\#}(y_i) \to r_i \in R'$ が適用できるので，

$$f^{\#}(t) \equiv f^{\#}(u)\sigma \to_{R'} r_0\sigma \equiv tp_m(t_1,...,t_m) \quad \square$$

**例 4** 例 *1* の *PT* 関数定義 $R_1$ は右線形であるので，$R_1$ から生成した *CTRS* $R_2$ *(* $= \mathcal{I}nv(R_1,\{add\})$ *)* より，次の $R_3 = \mathcal{D}el\mathcal{C}ond(R_2)$ が得られる．

$$\begin{aligned} R_3 = \{ \quad & add^{\#}(x_2) \to tp_2(0,x_2), \\ & add^{\#}(s(y)) \to add'(add^{\#}(y)), \\ & add'(tp_2(x_1,x_2)) \to tp_2(s(x_1),x_2)) \ \} \end{aligned}$$

$add^{\#}(s^3(0)) \xrightarrow{*}_{R_2} tp_2(s(0),s^2(0))$ を考えたとき，

$$\begin{aligned} add^{\#}(s^3(0)) &\to_{R_3} add'(add^{\#}(s^2(0)) \\ &\to_{R_3} add'(tp_2(0,s^2(0))) \\ &\to_{R_3} tp_2(s(0),s^2(0)) \end{aligned}$$

となり，定理 *2* が成り立つことが確認できる． $\square$

## 4 まとめ

本稿では，PT 関数定義から指定した関数記号の逆関数を持つ条件付き項書換え系を生成するアルゴリズムを提案し，アルゴリズムにより得られた CTRS が，逆関数になっていることを証明した．さらに，PT 関数定義の右辺が線形のとき，本アルゴリズムにより生成された条件付き項書換え系を項書換え系に変換する方法を示した．

今後の課題として，入力の条件を緩めた場合，つまり PT 項において関数記号の内側に構成子記号が出現する場合についてアルゴリズムを拡張する．

また，生成された逆関数の CTRS が合流性を持つクラスについて調べることも今後の課題である．

### 参考文献


[1] F. Baader, T. Nipkow: Term Rewriting and All That. CAMBRIDGE Univ. Press, 1998.

[2] W. CHIN: Safe Fusion of Functional Expressions. In ACM Conference on Lisp and Functional Programming, San Francisco, Ca., pp.11-20, ACM, June 1992.

[3] N. Dershowitz, Subrata Mitra: Jeopardy. LNCS 1631 Rewriting Techniques and Applications , pp.16-29, 1999.

[4] J.W.Klop: Term Rewriting Systems. In S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors, Hnadbook of Logic in Computer Science, volume 2, pp.2-116. Oxford University Press, 1992.

[5] D.Lankford: Canonical Algebraic Simplification in Computational Logic. Technical Report ATP-25, Department of Mathematics, University of Texas, Austin, 1975.

[6] G.Plotkin: Building-in Equational Theories. Machine Intelligence, 7: pp.73-90, 1972.

[7] J.R.Slagle: Automated Theorem Proving for Theories with Simplifiers, Commutativity and Associativity. J.ACM, 21: pp.622-642. 1974.

[8] 西田, 酒井, 坂部: PT 関数の逆関数を定義する条件付き TRS の生成. 電子情報通信学会技術研究報告, COMP 2001-14, pp.9-16, 2001.