決定手続きを用いた項書き換えシステムの帰納的定理自動証明

山口 諒　青戸 等人

等式論理において，自然数やリストなどのデータ構造上で成立する等式を帰納的定理という．帰納的定理の自動証明法として，項書き換えシステムを用いた書き換え帰納法が知られている (Reddy, 1990)．また，自然数の乗加算やリスト構造のいくつかの演算を用いた等式については帰納的定理の決定手続きが知られている (Aoto&Stratulat, 2014)．本報告では，書き換え帰納法と帰納的定理の決定手続きを融合した帰納的定理自動証明法を提案する．提案手法では，必要な等式公理を満たす関数記号を書き換え帰納法を用いて検出することで，予めどのような関数記号が用いられているかといった情報がない場合や，異なる公理が用いられている場合にも，可能な決定手続きを適用する．また，乗加算やリスト演算以外の関数記号や等式公理を持つような項書き換えシステムについても，書き換え帰納法の途中で決定手続きが利用できる補題を検出した場合には決定手続きを適用する．提案手法を実装するとともに，いくつかの具体例に対して実験を行い，提案手法の有効性を確認した．

1　はじめに

プログラムの性質のうちには，自然数やリストといったデータ構造に関する帰納法で証明されるような性質がある．例えば，リストを連結する append 関数は結合性をもつが，この性質はリストの構造に関する帰納法を用いることで証明することが出来る．等式論理の枠組みでは，このような性質は帰納的定理という概念でとらえられており，書き換え帰納法や潜在帰納法といった，帰納的定理の自動証明法が知られている [2] [3].

一方で，Aoto と Stratulat(2014) により，いくつかの特定の項書き換えシステムについて，帰納的定理が決定可能であることが示されている [1]．文献 [1] の手法は書き換え帰納法が失敗する場合にも成功するが，適用範囲は限定的である．一方，書き換え帰納法は失敗することもあるが，適用範囲は広い．

そこで，本報告では，書き換え帰納法と文献 [1] の決定手続きを組み合わせた，強力な帰納的定理の自動証明システムの実現を提案する．

文献 [1] で提案されている帰納的定理の決定手続きは，関数記号が与えられている特定の項書き換えシステムに対して機能する．このため，書き換え帰納法などの一般的な証明法において，この決定手続きを利用するには，その特定の理論に用いる関数記号を前もって知る必要がある．

そこで提案手法では，書き換え帰納法を用いることで，どのような関数記号が必要となる公理を満たすかを検証する．次に，関数記号を推定できたら，決定手続きが適用可能な場合には決定手続きを，そうではない場合には書き換え帰納法を適用することで，書き換え帰納法と文献 [1] の手法の組み合わせを実現する．

2　準備

2.1　項書き換えシステム

本報告で用いる定義及び記法を紹介する．

ソートの集合を $\mathcal{S}$，**多ソート関数記号集合**を $\mathcal{F}$ と表す．関数記号 f のソートを $\alpha_1, \alpha_2, \ldots, \alpha_n \to \beta$ と表す．以下では簡単のため，単一ソートを持つ場合について説明する．**変数集合**を $\mathcal{V}$ と表わし，**項の集**

Proving inductive theorems of term rewriting systems automatically using decision procedures.

Ryo Yamaguchi, Takahito Aoto, 新潟大学大学院自然科学研究科, Graduate School of Science and Technology, Niigata University.

合を $T(\mathcal{F},\mathcal{V})$ と記す．項 t の変数集合を $\mathcal{V}(t)$ と表し，項 t の根記号を $root(t)$ と記す．

関数 $\sigma : \mathcal{V} \to T(\mathcal{F},\mathcal{V})$ で集合 $\{x \mid \sigma(x) \neq x\}$ が有限であるものを**代入**とよぶ．また，集合 $\{x \mid \sigma(x) \neq x\}$ を代入 σ の**定義域**とよび，$dom(\sigma)$ と記す．項 s と t の**最汎単一化子**を $\mathsf{mgu}(s,t)$ と記す．

文脈とは，ホールとよばれる特別の定数 $\Box$ を持つ項のことをいう．また，文脈 $C[\,]$ のホールを項 t に置き換えて得られる項を $C[t]$ と記す．$s = C[t]$ となるとき，t を s の**部分項**とよぶ．

2 つの項 l, r が，$l \in \mathcal{V}$，$\mathcal{V}(r) \subseteq \mathcal{V}(l)$ を満たすとき，$l \to r$ を**書き換え規則**とよぶ．書き換え規則の有限集合を**項書き換えシステム**とよぶ．書き換え規則 $l \to r \in \mathcal{R}$，代入 σ，文脈 $C[\,]$ が存在して，$s = C[l\sigma]$ かつ $t = C[r\sigma]$ となるとき，$s \to_{\mathcal{R}} t$ と記し，**書き換えステップ**とよぶ．$s \to_{\mathcal{R}} t$ となるような t が存在しないとき，項 s を**正規形**とよぶ．$\to_{\mathcal{R}}$ の**反射推移閉包**を $\overset{*}{\to}_{\mathcal{R}}$，$\to_{\mathcal{R}}$ の**等価閉包**を $\overset{*}{\leftrightarrow}_{\mathcal{R}}$ と記す．

項 t が $\mathcal{V}(t) = \emptyset$ を満たすとき，t を**基底項**とよぶ．シグニチャ $\mathcal{F}$ 上の基底項の集合を $T(\mathcal{F})$ と記す．また，任意の $x \in dom(\sigma_g)$ について $\sigma_g(x) \in T(\mathcal{F})$ となっているような代入 σ_g を，項 t の**基底代入**とよぶ．$\mathcal{V}(t) \subseteq dom(\sigma_g)$ となっているとき，項 $t\sigma_g$ を**基底具体化**とよぶ．等式 $s\sigma_g \approx t\sigma_g$ が等式 $s \approx t$ の基底具体化であるとは，$\mathcal{V}(s) \cup \mathcal{V}(t) \subseteq dom(\sigma_g)$ となっているときをいう．

関数記号を**定義記号**と**構成子**に分け，$\mathcal{R}$ の定義記号の集合 $\mathcal{D}$ を，$\mathcal{D} = \{root(l) \mid l \to r \in \mathcal{R}\}$，構成子の集合 $\mathcal{C}$ は，$\mathcal{C} = \mathcal{F} \backslash \mathcal{D}$ と定義する．

ある $f \in \mathcal{D}$ と $t_1, \ldots, t_n \in T(\mathcal{C},\mathcal{V})$ が存在して $t = f(t_1, \ldots, t_n)$ となるとき，項 t が**基本項**であるといい，基本項の集合を $B(\mathcal{D},\mathcal{C},\mathcal{V})$ と記す．また，**基底基本項**の集合を $B(\mathcal{D},\mathcal{C})$ と記し，どの基底基本項も正規形にならないとき，$\mathcal{R}$ を**擬簡約**であるという．項 t の部分項のうち基本項であるものを**基本部分項**とよぶ．また，$\mathcal{B}(s)$ は s の**基本部分項**の集合を表わす．

文脈と代入に関して閉じた関係を**書き換え関係**とよび，整礎な書き換え関係を**簡約関係**とよぶ．簡約関係が半順序 (非反射的かつ推移的) のとき，**簡約順序**とよぶ．

表 1　書き換え帰納法の入出力

入力	擬簡約な項書き換えシステム $\mathcal{R}$, 等式集合 E, $\mathcal{R} \subseteq >$ なる簡約順序 $>$
出力	“証明成功” または “証明失敗”

2.2　帰納的定理

等式 $s \approx t$ が, 項書き換えシステム $\mathcal{R}$ の**帰納的定理**であるとは，$s \approx t$ の任意の基底具体化 $s\theta_g \approx t\theta_g$ について， $s\theta_g \overset{*}{\leftrightarrow}_{\mathcal{R}} t\theta_g$ が成立するときをいう．また，等式集合 E のすべての要素が $\mathcal{R}$ の帰納的定理であるとき，$\mathcal{R} \models_{ind} E$ と表す．

例 2.1 (帰納的定理)

$$\mathcal{R} = \left\{ \begin{array}{ll} +(0,y) & \to y \\ +(s(x),y) & \to s(+(x,y)) \end{array} \right\}$$

このとき，$+(+(x,y),z) \approx +(x,+(y,z))$ は，$\mathcal{R}$ の帰納的定理となる．よって，$\mathcal{R} \models_{ind} \{+(+(x,y),z) \approx +(x,+(y,z))\}$ が成立する．

2.3　書き換え帰納法

書き換え帰納法は Reddy [2] により提案された帰納的定理の自動証明法である．書き換え帰納法では，等式集合 E と項書き換え規則集合 H の対 $\langle E,H\rangle$ に関する導出を行いながら，証明を行う．表 1 に書き換え帰納法の入出力を，図 1 に書き換え帰納法の導出規則を示す．ここで，Expand 規則に用いられる関数 $Expd$ は以下のように定義される．

$$Expd_u(s,t) = \{C[r]\sigma \approx t\sigma \mid s = C[u], \sigma = \mathsf{mgu}(u,l), l \to r \in \mathcal{R}\}$$

$\uplus$ は直和を表す．等式の向きは区別しない (したがって，Simplify 規則は，等式の左辺も右辺も簡約化する)．$\mathcal{R}$ および $>$ は入力として与えられる項書き換えシステムと簡約順序である．

等式集合と項書き換え規則集合の対について， 図 1 の推論規則を適用して，$\langle E,H\rangle$ から $\langle E',H'\rangle$ が得られることを $\langle E,H\rangle \rightsquigarrow \langle E',H'\rangle$ と記す． $\rightsquigarrow$ の反射推移閉包を $\rightsquigarrow^*$ と記す． 必要に応じて， $\rightsquigarrow$ に添字をつけ Simplify 規則 (Delete 規則， Expand 規則) による導出を $\rightsquigarrow^s$ ($\rightsquigarrow^d$,$\rightsquigarrow^e$) と表す．

Simplify

$$\frac{\langle E \uplus \{s \approx t\}, H\rangle}{\langle E \cup \{s' \approx t\}, H\rangle}\ s \to_{\mathcal{R}\cup H} s'$$

Delete

$$\frac{\langle E \uplus \{s \approx s\}, H\rangle}{\langle E, H\rangle}$$

Expand

$$\frac{\langle E \uplus \{s \approx t\}, H\rangle}{\langle E \cup Expd_u(s,t), H \cup \{s \to t\}\rangle}\ u \in \mathcal{B}(s), s > t$$

図 1　書き換え帰納法の推論規則

直観的には，E はこれから証明しようとする等式を，H は帰納法の仮定および証明済みの定理を表す．等式集合 E が $\mathcal{R}$ の帰納的定理であることを証明するには，$\langle E, \emptyset\rangle$ から導出を始める．最終的に $\langle \emptyset, H\rangle$ が導出されるとき，証明成功となる．

書き換え帰納法は半アルゴリズムである．すなわち，等式集合が空にならないまま無限に導出を繰り返して手続きが停止しないことがあり得る．この場合を手続きが**発散する**という．また，規則を適用する等式の選び方や適用する推論規則に任意性があるので上記の手続きは非決定的である．

例 2.2 (書き換え帰納法の実行例) 項書き換えシステム $\mathcal{R}$ と証明する等式集合 E として，以下が与えられたとする．

$$\mathcal{R} = \left\{ \begin{array}{lcl} +(0,y) & \to & y \\ +(s(x),y) & \to & s(+(x,y)) \end{array} \right\}$$

$$E = \left\{ \begin{array}{lcl} +(+(x,y),z) & \approx & +(x,+(y,z)) \end{array} \right\}$$

$> $ を優位順序 $+ > s > 0$ に基づく辞書式経路順序としたときの書き換え帰納法の実行過程の例を図 2 に示す．

次の定理により，書き換え帰納法の手続きが “証明成功” を返した場合，入力等式集合 E が入力項書き換えシステム $\mathcal{R}$ の帰納的定理であることが保証される．

命題 2.3 (書き換え帰納法の正当性 [2]) $\mathcal{R}$ を擬簡約な項書き換えシステム， E を等式集合，$>$ を $\mathcal{R} \subseteq >$ なる簡約順序とする．ある項書き換え規則集合 H が存在して，$\langle E, \emptyset\rangle \rightsquigarrow^* \langle \emptyset, H\rangle$ となるならば，$\mathcal{R} \models_{ind} E$ である．

表 2　$DecProc^{N\{\times,+\}}$ の入出力

入力	等式集合 $E \subseteq T(\{f_\times, f_+, f_s, f_0\}, \mathcal{V})^2$
	$f_\times : N, N \to N$
	$f_+ : N, N \to N$
	$f_s : N \to N$
	$f_0 : N$
出力	True　if $\mathcal{R}^N_{(f_\times, f_+, f_s, f_0)} \models_{ind} E$
	False　if $\mathcal{R}^N_{(f_\times, f_+, f_s, f_0)} \not\models_{ind} E$

表 3　$DecProc^{N\{+\}}$ の入出力

入力	等式集合 $E \subseteq T(\{f_+, f_s, f_0\}, \mathcal{V})^2$
	$f_+ : N, N \to N$
	$f_s : N \to N$
	$f_0 : N$
出力	True　if $\mathcal{R}^N_{(f_+, f_s, f_0)} \models_{ind} E$
	False　if $\mathcal{R}^N_{(f_+, f_s, f_0)} \not\models_{ind} E$

3　特定の関数記号に依存しない帰納的定理の決定手続き

Aoto&Stratulat によって，特定の項書き換えシステム $\mathcal{R}$ に対する帰納的定理の決定手続きが与えられている [1]．自然数の乗加算の項書き換えシステムに関する決定手続き $DecProc^{N\{\times,+\}}$ の入出力を表 2 に，自然数の加算の項書き換えシステムに関する決定手続き $DecProc^{N\{+\}}$ の入出力を表 3 に示す．

ここで，$\mathcal{R}^N_{(f_\times, f_+, f_s, f_0)}$ は以下の項書き換えシステムを表す．また，$\mathcal{R}^N_{(f_+, f_s, f_0)}$ は以下の項書き換えシステムの最初の 2 つの書き換え規則から成る．

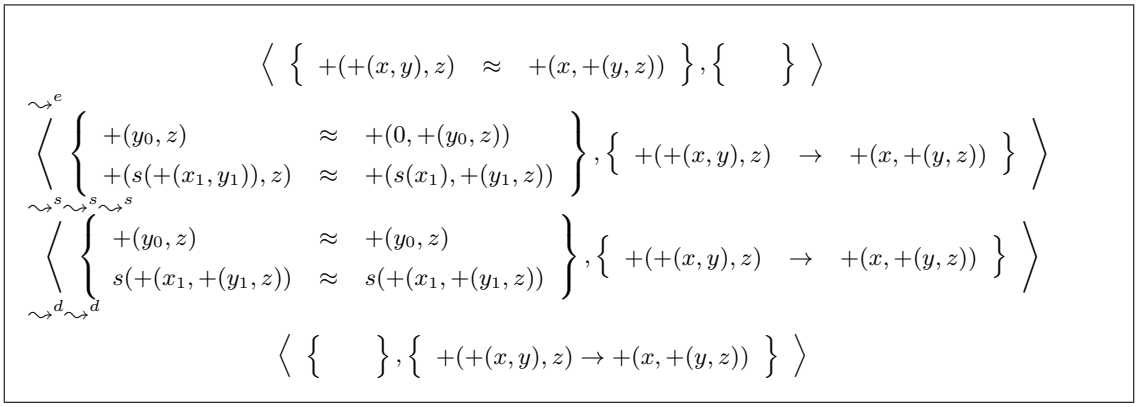

図 2　書き換え帰納法の実行例

$$\left\{ \begin{array}{lcl} f_+(f_0,y) & \to & y \\ f_+(f_s(x),y) & \to & f_s(f_+(x,y)) \\ f_\times(f_0,z) & \to & f_0 \\ f_\times(f_s(x),y) & \to & f_+(y,f_\times(x,y)) \end{array} \right\}$$

ただし，$(f_\times, f_+, f_s, f_0)$ は入力に応じて具体的な関数記号におきかえられる．

しかしこの手続きは，関数記号をパラメータとして必要とする．そのため，書き換え帰納法と同じ枠組みで用いることができない．そこで，この節では与えられた等式が自然数の公理を満たしているかを確かめることで，関数記号を推定して決定手続きを用いる手法 (ProcNatInd) を提案する．

まず，関数記号の情報を用いて関数記号の候補を絞り込む．

定義 3.1 (f_1, f_2, f_3, f_4) が**乗加算記号候補**であるとは，あるソート $\alpha \in \mathcal{S}$ が存在し，$f_1 : \alpha, \alpha \to \alpha$, $f_2 : \alpha, \alpha \to \alpha$, $f_3 : \alpha \to \alpha$, $f_4 : \alpha$, $f_1, f_2 \in \mathcal{D}$, $f_3, f_4 \in \mathcal{C}$ となるときをいう．同様にして，**加算記号候補** (f_2, f_3, f_4) であるとは，乗加算記号候補から f_1 に該当する関数記号を取り除いたものをいう．

適切な問題が発見されたとき，決定手続きを利用した帰納的定理証明手続き (ProcNatInd) を行う．その入出力を表 4 に示す．

定義 3.2 (ProcNatInd)

Step 1. 関数記号を定義記号 $\mathcal{D}$ と構成子 $\mathcal{C}$ に分類.

Step 2. $\mathcal{D}$,$\mathcal{C}$ の情報とソート情報で，$\mathcal{F}(E) \subseteq \{f_1, f_2, f_3, f_4\}$ となるような乗加算記号候補 (f_1, f_2, f_3, f_4) を見つける．候補がなければ Step 5 へ.

Step 3. $\mathcal{R} \models_{\mathsf{ind}} \mathcal{R}^N_{(f_1,f_2,f_3,f_4)}$ を入力された簡約順序 $>$ を用いて書き換え帰納法で証明.
証明成功なら Step 4 へ，失敗なら Step 2 へ戻って異なる候補を選択.

Step 4. $\mathcal{R}^N_{(f_1,f_2,f_3,f_4)} \models_{\mathsf{ind}} E$ を $DecProc^{N\{\times,+\}}$ が True ならば True を，それ以外は証明失敗を返す.

Step 5. $\mathcal{D}$,$\mathcal{C}$ の情報とソート情報で，$\mathcal{F}(E) \subseteq \{f_1, f_2, f_3\}$ となるような加算記号候補 (f_1, f_2, f_3) を見つける．候補がなければ失敗を返す.

Step 6. $\mathcal{R} \models_{\mathsf{ind}} \mathcal{R}^N_{(f_1,f_2,f_3)}$ を入力された簡約順序 $>$ を用いて書き換え帰納法で証明.
証明成功なら Step 7 へ，失敗ならそのまま失敗を返す.

Step 7. $\mathcal{R}^N_{(f_1,f_2,f_3)} \models_{\mathsf{ind}} E$ を $DecProc^{N\{+\}}$ が True ならば True を，それ以外は証明失敗を返す.

定理 3.3 (ProcNatInd の正当性) ProcNatInd を用いて True が返ってきた場合は $\mathcal{R} \models_{ind} E$ である．

表 4　ProcNatInd の入出力

入力	擬簡約な項書き換えシステム $\mathcal{R}$, 等式集合 E, $\mathcal{R} \subseteq >$ なる簡約順序 $>$
出力	True または “証明失敗”

(証明) 以下では $\mathcal{R}^N_{(f_1,f_2,f_3)}$ の場合は同様であるため，$\mathcal{R}^N_{(f_1,f_2,f,f_4)}$ の場合のみ考える．$\mathcal{R}^N_{(f_1,f_2,f,f_4)}$ を $\mathcal{R}^N$ と省略する．

$\mathcal{R} \models_{ind} \mathcal{R}^N$ より，$\forall l \to r \in \mathcal{R}^N.\ l\theta_g \overset{*}{\leftrightarrow}_{\mathcal{R}} r\theta_g$ である．$\mathcal{R}^N \models_{ind} E$ と仮定すると，$\forall u \approx v \in E.\ u\theta_g \overset{*}{\leftrightarrow}_{\mathcal{R}^N} v\theta_g$ であるから，$\forall u \approx v \in E.\ u\theta_g \overset{*}{\leftrightarrow}_{\mathcal{R}} v\theta_g$．よって，$\mathcal{R} \models_{ind} E$ が成立． □

例 3.4 入力として次の項書き換えシステム $\mathcal{R}$，等式集合 E が与えられたとする．

$$\mathcal{R} = \left\{ \begin{array}{lcl} plus(zero, x) & \to & x \\ plus(succ(x), z) & \to & succ(plus(x, z)) \\ times(zero, z) & \to & zero \\ times(succ(x), y) & \to & plus(y, times(x, y)) \end{array} \right\}$$

$$E = \left\{ \begin{array}{l} times(x, plus(y, z)) \\ \approx plus(times(x, y), times(x, z)) \end{array} \right\}$$

関数記号を定義記号 $\mathcal{D} = \{times, plus\}$，構成子 $\mathcal{C} = \{succ, zero\}$ に分割する (Step 1)．この情報とソート情報を用いて，乗加算記号候補 $(plus, times, succ, zero)$ が選ばれたとする (Step 2)．$\mathcal{R} \models_{\mathsf{ind}} \mathcal{R}^N_{(plus,times,succ,zero)}$ を書き換え帰納法を用いて検証し，この場合は証明が失敗してしまうため，Step 2 に戻る (Step 3)．前とは別の乗加算記号候補 $(times, plus, succ, zero)$ を選択する (Step 2)．$\mathcal{R} \models_{\mathsf{ind}} \mathcal{R}^N_{(times,plus,succ,zero)}$ を書き換え帰納法を用いて検証し，“証明成功” が返ってきて Step 4 へ移る (Step 3)．$\mathcal{R}^N_{(times,plus,succ,zero)} \models_{\mathsf{ind}} E$ を決定手続きで判定し，帰納的定理であると決定される (Step 4)．

例 3.5 (公理の表現が異なる例) 入力 $\mathcal{R}$，E を以下のとおり与える．

$$\mathcal{R} = \left\{ \begin{array}{lcl} +(y, 0) & \to & y \\ +(y, s(x)) & \to & s(+(x, y)) \end{array} \right\}$$

$$E = \left\{ \begin{array}{lcl} +(+(x, y), z) & \approx & +(y, +(z, x)) \end{array} \right\}$$

ここで ProcNatInd を用いると，関数記号を定義記号 $\mathcal{D} = \{+\}$，構成子 $\mathcal{C} = \{s, 0\}$ に分割する (Step 1)．次に，$\mathcal{R}$ に出現する関数記号が 3 つであることから，乗加算記号候補が見つからず，Step 5 へ移る (Step 2)．$\mathcal{D}, \mathcal{C}$ の情報とソート情報を用いて，加算記号候補 $(+, s, 0)$ が選ばれたとする (Step 5)．$\mathcal{R} \models_{\mathsf{ind}} \mathcal{R}^N_{(+,s,0)}$ を書き換え帰納法を用いて検証し，“証明成功” が返ってきて Step 7 へ移る (Step 6)．$\mathcal{R}^N_{(+,s,0)} \models_{\mathsf{ind}} E$ を決定手続きで判定し，帰納的定理であると決定される (Step 7)．

$\mathcal{R}$ において加算が第 2 引数に関する場合分けで定義されていることに注意する．このように，公理の形が異なる場合にも，手続きをうまく適用することができる．

4　決定手続きを利用した書き換え帰納法

書き換え帰納法を用いて証明をする際，等式をそれ以上 Expand できずに停止してしまう場合 (証明失敗) や，等式集合が空にならないまま無限に導出を繰り返して手続きが停止しないことがあり得る (発散)．しかし，決定手続きは，等式集合が自然数やリストの公理を満たす場合に限定されるものの，発散することなく証明が終了する．そこで，書き換え帰納法と決定手続きの手法を組み合わせることで，より強力な証明システムを実現する．

例 4.1 入力として以下の項書き換えシステム $\mathcal{R}$ と等式集合 E を与える．(pow は自然数を 2 乗する関数である．)

$$E = \left\{ \begin{array}{lcl} pow(x) & \approx & T(x, x) \end{array} \right\}$$

$$\mathcal{R} = \left\{ \begin{array}{lcl} T(Z, y) & \to & Z \\ T(S(x), y) & \to & P(y, T(x, y)) \\ P(Z, y) & \to & y \\ P(S(x), y) & \to & S(P(x, y)) \\ pow(Z) & \to & Z \\ pow(S(x)) & \to & P(pow(x), S(T(x, S(S(Z))))) \end{array} \right\}$$

この場合，導出過程は図 4 のとおりになり，証明に失敗する．しかしながら，最後の等式集合に着目してみると，$\mathcal{R}^N_{(T,P,S,Z)}$ の帰納的定理である．よって，前節の特定の関数記号に依存しない決定手続きを用いることで，証明に成功する．

そこで，次のように書き換え帰納法を拡張する．

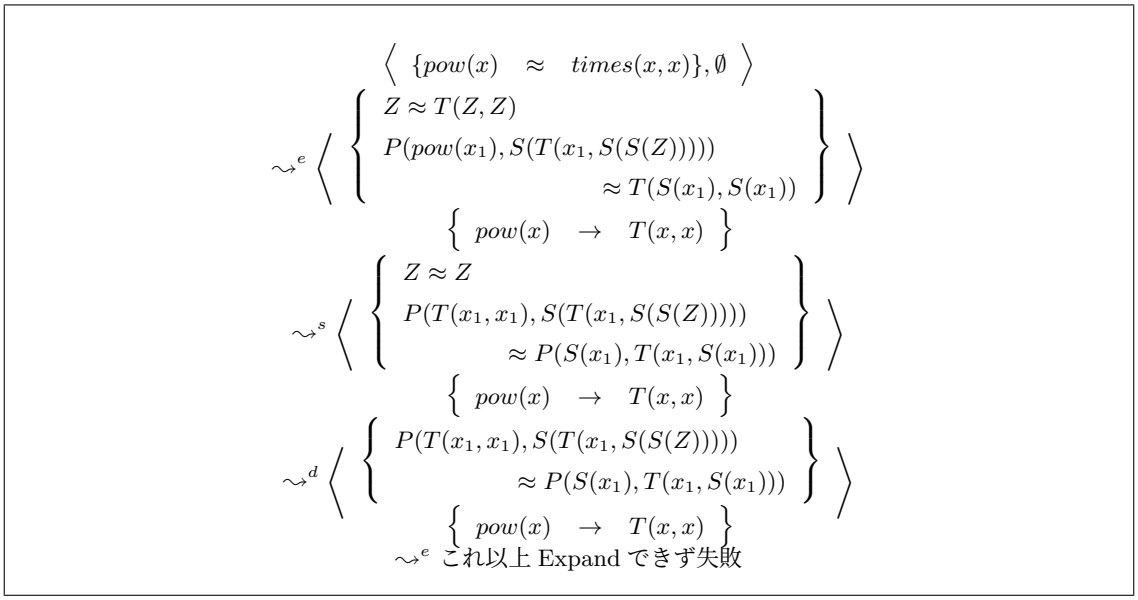

図 3　pow 関数を用いた等式の書き換え帰納法での導出例

表 5　融合証明法の入出力

入力	擬簡約な項書き換えシステム $\mathcal{R}$, 等式集合 E, $\mathcal{R} \subseteq >$ なる簡約順序 $>$
出力	“証明成功” または “証明失敗”

定義 4.2 (書き換え帰納法の拡張)　書き換え帰納法に以下の規則を追加する．

E-Delete

$$\frac{\langle E \uplus \{s \approx s''\}, H\rangle}{\langle E, H\rangle} \ \mathrm{ProcNatInd}(\mathcal{R}, \{s \approx s''\}, >) = \mathrm{true}$$

この拡張した書き換え帰納法を融合証明法と呼び，その入出力を表 4 に示す．

E-Delete による導出を $\rightsquigarrow^{Ed}$ と表す．この規則は，導出過程の等式集合 E に含まれる等式 $s \approx s''$ について，決定手続き (ProcNatInd) を用いて帰納的定理が証明成功する場合，該当する等式 $s \approx s''$ を等式集合 E から取り除く．

先の pow 関数を用いた例 4.1 が E-Delete 規則を適用した拡張書き換え帰納法を用いることで証明に成功する．

5　実装と実験

文献 [1] の決定手続き (DP)，書き換え帰納法 (RI)，本報告で提案した融合証明法 (FP) の 3 つの証明法を実装した．実装には関数型言語である SML/NJ を用いた．プログラムコードは約 2,000 行である．

20 個の例で実験を行った結果を表 5 に示す．この例では，基本的に奇数番目が特定の関数記号 $(\times, +, s, 0)$ に依存した等式集合，偶数番目がそれらに依存しない等式集合である．決定手続きや書き換え帰納法を用いた場合よりも，強力な証明を実現できた．

6　おわりに

文献 [1] の自然数の乗加算の記号を用いた決定手続き ($DecProc^N$) の手法と書き換え帰納法を融合することで，帰納的定理証明の能力を高める手法を提案した．今後の課題としては，書き換え帰納法と文献 [1] のリスト構造の演算を用いた等式の決定手続きを融合することで，より強力な証明システムを構築することが挙げられる．リストの関数記号においても，多ソート情報や，定義記号と構成子の分類を有効に用

表 6　提案手法を用いた帰納的定理証明の実験結果 (○:定理，×:失敗，∞:発散，−:適用不可)

	等式集合 E	DP	RI	FP
1	$s(+(x,+(y,0))) \approx +(y,+(s(0),x))$	○	×	○
2	$succ(plus(x,plus(y,z))) \approx plus(y,plus(succ(z),x))$	–	×	○
3	$s(+(+(x,x),+(y,x))) \approx +(+(x,x),s(+(y,x)))$	○	×	○
4	$succ(plus(plus(x,x),plus(y,x))) \approx plus(plus(x,x),succ(plus(y,x)))$	–	×	○
5	$+(+(y,+(0,x)),0) \approx +(y,x)$	○	○	○
6	$P(P(y,P(Z,x)),Z) \approx P(y,x)$	–	○	○
7	$+(+(s(0),x),0) \approx s(+(0,+(x,0)))$	○	○	○
8	$p(p(succ(zero),x),z) \approx succ(p(z,p(x,zero)))$	–	×	○
9	$*(x,+(0,0)) \approx *(+(x,x),0)$	○	×	○
10	$T(x,P(z,z)) \approx T(P(x,x),z)$	–	×	○
11	$s(s(+(+(s(x),*(x,x)),x))) \approx +(s(x),s(s(+(x,*(x,x)))))$	○	∞	○
12	$succ(succ(plus(plus(succ(x),times(x,x)),x)))$ $\approx plus(succ(x),succ(succ(plus(x,times(x,x)))))$	–	∞	○
13	$s(0) \approx +(s(0),*(*(x,0),s(x)))$	○	×	○
14	$s(z) \approx plus(s(z),times(times(x,z),s(x)))$	–	×	○
15	$*(*(*(*(s(x),0),x),0),x)$ $\approx *(*(0,+(s(*(x,y)),x)),*(0,*(s(*(y,y)),*(*(y,x),x))))$	○	∞	○
16	$T(T(T(T(S(x),Z),x),Z),x)$ $\approx T(T(Z,P(S(T(x,y)),x)),T(Z,T(S(T(y,y)),T(T(y,x),x))))$	–	∞	○
17	$pow(x) \approx *(x,x)$	–	×	○
18	$pow(x) \approx T(x,x)$	–	×	○
19	$rev(rev(@(xs,ys))) \approx @(xs,ys)$	–	∞	∞
20	$R(R(app(xs,ys))) \approx app(xs,ys)$	–	∞	∞
成功数		9/20	3/20	18/20

いることで，関数記号を推定することが可能であると考える．また，決定手続きは反証も可能であるため，融合証明法を反証に応用することも課題である．

参 考 文 献

[1] Aoto, T. and Stratulat, S.: Decision procedures for proving inductive theorems without induction, *Proc. of the 16th PPDP*, ACM Press, 2014, pp. 237–248.

[2] Reddy, U. S.: Term rewriting induction, *Proc. of the 10th CADE*, Vol. 449 of LNAI, Springer-Verlag, 1990, pp. 162–177.

[3] 小池広高, 外山芳人: 潜在帰納法と書換え帰納法の比較, Vol. 17(2000), pp. 1–12.