アルゴリズム 23－12
（1991．9．20）

動的項書換え計算モデルとその応用

馮速　　坂部俊樹　　稲垣康善

名古屋大学

あらまし　本稿では、書換え規則を動的に変化させながら項を書き換えていく新しい計算モデルとして**動的項書換え計算**（DTRC）を提案する。DTRCは動的に書換え規則を変化させながら項を書き換えていくことができること、階層化されたプログラムが書けて、上位層から下位層への書換え規則の継承が表現できることなどの特徴を持つ。これらの特徴を利用することによってDTRCを定理自動証明に応用できること、ならびに、DTRCの停止性、合流性に関するいくつかの結果を示す。

Dynamic Term Rewriting Calculus and Its Application

Su Feng　　Toshiki Sakabe　　Yasuyoshi Inagaki

Nagoya University

Abstract　Recent researches on term rewriting systems give rise to importance of formally describing and verifying manipulation of term rewriting systems. In this paper, we propose a computation model called Dynamic Term Rewriting Calculus (DTRC for short). Computation of DTRC is basically term rewriting. Moreover DTRC has such features as dynamically changing rules during computation and hierarchically declaring not only function symbols and variables but also rules. These features allow us to describe in DTRC algorithms which manipulate term rewriting systems. Further ,verification procedures of algorithms may be written also in DTRC in a natural way.

We present syntax and operational semantics of DTRC, show an application to automated theorem proving, and give some results on confluency and termination of DTRC.

1 はじめに

定理自動証明、代数的仕様記述に関連して、項書換え系（TRS）に関する研究が盛んに行われている中で、TRS自体の操作を記述し、その記述を検証することの要求が高まりつつある。そこで、本稿では、書換え規則を動的に変化させながら項を書き換えていく新しい計算モデルとして**動的項書換え計算**（Dynamic Term Rewriting Calculus,DTRC）を提案する。

DTRCの計算は基本的には項の書換えによって行なわれる。この意味でDTRCはTRSを包含する計算モデルである。DTRCの特徴は、動的に書換え規則を変化させながら項を書き換えていくことができること、関数記号、変数、書換え規則の階層的な宣言ができることである。この特徴により、階層化されたプログラムにおいて上位層から下位層への書換え規則の継承が表現できる。また、帰納法による証明の形式化の際に必要なメタ変数の概念が自然に表現できる。

本稿では、DTRCの構文および操作的意味論を与えた上で、上述の特徴を説明する例を示すとともに、DTRCの応用としてTRSを用いた帰納的定理の自動証明がDTRCで表現され、実行されることを示す。最後に、計算モデルの基本的性質である合流性と停止性についてのいくつかの結果を示す。

2 DTRC

2.1 DTRCの項

DTRCの典型的項は

$$[F;V{:}R]\{t\}$$

の形をする式であり、プログラムといわれる。$[F;V : R]$の部分はシステムと呼ぶ。FとVはそれぞれ関数記号と変数の宣言部分で、Rは書換え規則の集合である。$[F;V : R]\{t\}$は、直観的には、Rの規則を用いてtを書き換えることを意味する。但し、書換え規則の両辺およびt自体もプログラムを部分項に持つことができる。これにより、階層化された項が書けるようになる。

項をスムーズに定義するために、まず関数記号の集合$\mathcal{F}$と変数の集合$\mathcal{V}$の上の準項の概念を導入する。

【定義 1】関数記号の集合$\mathcal{F}$と変数の集合$\mathcal{V}$に対して、準項の集合$PRT(\mathcal{F},\mathcal{V})$を次の条件を満たす最小集合であるとする。

1. $x \in \mathcal{V}$ならば$x \in PRT(\mathcal{F},\mathcal{V})$
2. $t_1,\ldots,t_n \in PRT(\mathcal{F},\mathcal{V})$, $f \in \mathcal{F}$ならば $f(t_1,\ldots,t_n) \in PRT(\mathcal{F},\mathcal{V})$
3. $f_i \in \mathcal{F}$,$x_i \in \mathcal{V}$, $l_i,r_i,t \in PRT(\mathcal{F},\mathcal{V})$ ならば$[f_1,\ldots,f_m;x_1,\ldots,x_l : l_1 \rightarrow r_1,\ldots,l_n\rightarrow r_n]\{t\} \in PRT(\mathcal{F},\mathcal{V})$
この形の準項をプログラムという。また、$[f_1,\ldots,f_m;x_1,\ldots,x_l{:}l_1\rightarrow r_1,\ldots,l_n\rightarrow r_n]$をシステム、$f_1,\ldots,f_m$を関数宣言、$x_1,\ldots,x_l$を変数宣言という。

【定義 2】文脈は空の場所を表す特別の記号□を含む特別な準項$C[] \in PRT(\mathcal{F},\mathcal{V}\cup\square)$である。但し、□は$C[]$の変数宣言以外の一ヶ所にだけ現れる。□を準項$t$で置き換えて得られる準項は$C[t]$と書く。

【定義 3】部分準項の概念を次のように帰納的に定義する。

1. tは準項tの部分準項である。
2. 準項$t = f(t_1,\ldots,t_n)$において，t_iの部分準項はtの部分準項である。
3. 準項$t = [F;V{:}\ l_1 \rightarrow r_1,\ldots,l_n \rightarrow r_n]\{t'\}$において、$l_i,r_i(i = 1\ldots n),t'$の部分準項は$t$の部分準項である。

【定義 4】準項$t = [f_1,\ldots,f_m;x_1,\ldots,x_l : R]\{t'\}$において、関数記号$f_i$の宣言の有効範囲は$t$全体である。また、変数宣言$x_i$の有効範囲は$x_1,\ldots,x_l : R$の部分である。

上の有効範囲の定義により，以下の自由と束縛の概念が導入される。

【定義 5】準項tにおける関数記号f（変数x）の出現はその出現を含むfの宣言（xの宣言）がt中に存在するとき、束縛されるといい、そうでないとき自由であるという。tの中に自由な出現のある関数記号および変数はtの自由関数記号、自由変数という。

項は準項の特別な場合として次のように定義される。

【定義 6】準項$t \in PRT(\mathcal{F},\mathcal{V})$が項であるというのは、$t$のすべての部分項$u$に対して、

uの自由な関数記号fの出現がすべて同数の引数を持つとき、かつそのときに限る。

関数記号の集合$\mathcal{F}$と変数の集合$\mathcal{V}$上の項の集合を$TERM(\mathcal{F},\mathcal{V})$と書き、プログラムを部分項に含まない項を単純項という。

2.2 DTRCの操作的意味論

ここでは、DTRCの項の操作的意味論として書換え関係$\rightarrow$を定義する。まず、代入および照合を定義する。

【定義 7】 x_iがすべて異なるような項t_iと変数x_iの対の有限集合$\sigma = \{t_1/x_1, \ldots, t_n/x_n\}$を代入であるという。$\sigma$の定義域$\mathrm{dom}(\sigma)$を$\mathrm{dom}(\sigma)=\{x_1, \ldots, x_n\}$のように定める。項$t$に対して、準項$t\sigma$を次のように定義する。

1. $t \in \mathcal{V}$ならば
$$t\sigma = \begin{cases} t_i & \text{if } t = x_i \\ t & \text{otherwise} \end{cases}$$
2. $f(s_1, \ldots, s_m)\sigma = f(s_1\sigma, \ldots, s_m\sigma)$
3. $[F; V : l_1 \rightarrow r_1, \ldots, l_m \rightarrow r_m]\{t\}\sigma$
$= [F; V : l_1\sigma' \rightarrow r_1\sigma', \ldots, l_m\sigma' \rightarrow r_m\sigma']\{t\sigma\}$
但し、$\sigma' = \{t/x \in \sigma \mid x \notin V\}$

また、t_iにおける変数および関数記号の自由な出現が$t\sigma$においても自由であり、かつ、$t\sigma$が項であるときσはtに関して安全な代入であるという。

【定義 8】 $C[\ ]$を文脈、s,tを項、σをsに関して安全な代入とする。$C[\ ]$の下でsがσによってtに照合するというのは、$s\sigma = t$かつsの自由な関数記号の出現に対応するtでの出現が$C[t]$に於いて自由であるとき、かつそのときに限る。

この代入と照合の概念を用いて書換え関係$\rightarrow$を次のように定義する。

【定義 9】 DTRCのプログラム$[F; V{:}R]\{C[t]\}$において、$l \rightarrow r \in R$及びlに関して安全な代入tの部分項u、そして代入σが存在して、$\mathrm{dom}(\sigma) \subseteq V$かつ$C[\ ]$の下で$l$が$\sigma$によって$t$に照合するとき、
$[F; V{:}R]\{C[t]\} \rightarrow [F; V{:}R]\{C[r\sigma]\}$
であるとする。

また、任意の文脈$C[\,]$及び項t,t'に対して、$t \rightarrow t'$ならば、
$$C[t] \rightarrow C[t']$$
であるとする。

【定義 10】 項tが正規形であるとは$\forall$s $t \not\rightarrow s$のときかつそのときに限る。

【定義 11】 項tが正規形を持つとは、正規形である項sがあって$t \xrightarrow{*} s$のときかつそのときに限る。

また、本質的ではないが、プログラムを短く、また、後の議論を簡単にするために、次のような約束をする。項$t = C[[F; V : R]\{s\}]$が正規形であり、かつ、sにFの関数が現われないならば、tと$C[s]$とを同一視する。

3 DTRCの特徴

この節では、DTRCのいくつかの特徴を述べ、プログラミングの例を示す。

DTRCには以下のような特徴がある。

1. 階層構造： 書き換えられる項がプログラムを部分項に持つことができるので、階層化された項が書ける。階層化された項において、上位層の書換え規則がそのまま下位層の書換え規則として用いられるという意味で、上位層から下位層への書換え規則の継承が表現できる。
2. 動的項書換え： 書換え規則の両辺にプログラムが部分項として現われ得ることから動的に書換え規則を追加、変更させながら項を書き換えていくようなプログラムを書くことができる。
3. メタ変数の表現： 有効範囲および照合の定義により、帰納法の形式化などで必要なメタ変数[4]の概念が簡単に表現できる。

これらの特徴により、項書換え系の階層化、等価変換の記述ができるばかりでなく、項書換え系を用いた定理証明手続きの記述への応用も可能である。

ここでは、特徴1.と2.を利用したプログラミング例を挙げ、DTRCにおける構造化プログラミング手法を示唆すると共に、書換えの行ない方についても説明する。なお、特徴3.については次節で述べる。

【例 1】

```
[ recursive;x,y:
  recursive(0,y) → base(y),
  recursive(s(x),y)
→ rec(y,recursive(x,y))]
```

これは2引数帰納的関数を抽象化し、その構造を表すシステムである。このシステムを Sys1 とする。base と rec は自由な関数記号で、Sys1 をとりまく上位のシステムによって任意に定義が与えられる。このことを次の例で示す。

【例 2】

```
 [recur,add,mult,0,s;u,v:
    recur(u,v) → Sys1{recursive(u,v)},
    add(u,v) → [base,rec;x,y:
                  base(x) → x,
                  rec(x,y) → s(y)]
                {recur(u,v)},
    mult(u,v) → [base,rec;x,y:
                   base(x) → 0,
                   rec(x,y) → add(x,y)]
                 {recur(u,v)}
 ]
```

このシステム（Sys とする）では、加算 add と乗算 mult を定義する add(u,v) → [...]{...} と mult(u,v) → [...]{...} の構造がほとんど同じであって、base(x) と rec(x,y) の定義のみが異なっている。これは、加算と乗算が共通の再帰構造を持つことに注目した結果のプログラムである。Sys{mult(s(s(0)),s(s(0)))} は s(s(0)) と s(s(0)) の乗算（2×2）をするプログラムで、次のように評価されていく。

```
  Sys{mult(s(s(0)),s(s(0)))}
→ [recur,add,mult,0,s;u,v:
     recur(u,v) → Sys1{recursive(u,v)},
     add(u,v) → [base,rec;x,y:
                   base(x) → x,
                   rec(x,y) → s(x)]
                 {recur(u,v)},
     mult(u,v) → [base,rec;x,y:
                    base(x) → 0,
                    rec(x,y) → add(x,y)]
                  {recur(u,v)}
   ]
   {[base,rec;x,y:
       base(x) → 0,
       rec(x,y) → add(x,y)]
     {recur(s(s(0)),s(s(0)))}
→ [...]
  {[base,rec;x,y:
      base(x) → 0,
      rec(x,y) → [base,rec;x,y:
                    base(x) → x,
                    rec(x,y) → s(y)]
                  {recur}
   ]
   {Sys1{recursive(s(s(0)),s(s(0)))}}}
→ [...]
  {[basee,rec;x,y:
      base(x) → 0,
      rec(x,y) → [base,rec;x,y:
                    basse(x) → x,
                    rec(x,y) → s(y)]
                  {Sys1{recursive(x,y)}}
   ]{Sys1{recursive(s(s(0)),s(s(0)))}}}
→ [...]{s(s(s(s(0))))}
```

4 定理自動証明への DTRC の応用

酒井ら [4] はメタ変数を用いて構造帰納法による帰納的定理の証明を形式化した。さらに、それを拡張した被覆集合帰納法による帰納的定理の証明方法を提案し、それらを項書換え系で自動実行する方法を示した。本節では、そのような帰納的定理の自動証明への DTRC の応用について述べる。なお、構造帰納法、被覆集合帰納法およびメタ変数などの詳細については酒井ら [4] を参照されたい。

メタ変数は、直感的には、任意の基礎項を表現し、かつ、定数と同等に扱われる変数である。DTRC では、システム

$$[F; V: R]$$

において、変数 p は V で宣言されていなければ、このシステムの中で定数と同等に扱われ、メタ変数の働きをする。このことを用いれば、構造帰納法や被覆集合帰納法を自動実行する DTRC のプログラムが自然に記述できることを示す。

等式 $e(x)$ が等式集合 E の帰納的定理であることの構造帰納法による証明は、簡単にいえば、p をメタ変数として、次の2つを示すこ

とで行なわれる。

1. 定数記号 c に対して $e[c/p]$ がEの下で成り立つこと。
2. $e(p)$ がEの下で成り立つなら、すべての構成子 f に対して、$e[f(p)/p]$ がEの下で成り立つこと。

【例 3】等式の集合

$$\{0+y=y, s(x)+y=s(x+y)\}$$

の下で

$$(x+y)+z=x+(y+z)$$

が帰納的定理であることの構造帰納法による証明の手続きはDTRCの項で次のように記述できる。

```
[true,eq;u,v:
   eq(u,u,v,v) → true]
{[0,s,+;x,y:
    0+y → y,
    s(x)+y → s(x+y)]
 {[;:
     (p+q)+r → p+(q+r)]
  {eq((0+q)+r,0+(q+r),
      (s(p)+q)+r,s(p)+(q+r))}}}
```

この項が [...]{true} の形の項に書き換えられれば証明成功であり、実際そうなる。

上の項の一番内側の書換え規則 (p+q)+r → p + (q + r) は帰納法の仮定に当たり、p,q,r はそこで宣言されていないので、メタ変数の働きをし、任意の基礎項を表す。

一般に次のような命題が成立する。

【命題 1 】等式集合E[1]の下で、等式 $\alpha(\mathbf{x}) = \beta(\mathbf{x})$ が x に対して帰納的定理であることの証明の手続きはDTRCの項で次のように記述できる。

```
[ true,eq;u,v:
   eq(u,u,v,v) → true ]
{ [ F;V:E ]
  { [;: α (p) → β (p) ]
    { eq(α (a), β (a),
         α (s(p)), β (s(p))) }}}
```

但し、F,VはEに現れる関数記号と変数の集合である。また、すべての基礎項は定数記号aと関数記号sとからなる基礎項にE等価になるものとする。上の項を書き換えることによって、[...]{true} の形の項に書き換えられるなら、$\alpha(\mathbf{x}) = \beta(\mathbf{x})$ が帰納的定理である。

次に被覆集合帰納法へのDTRCの応用を示す。被覆集合帰納法は、被覆集合と項の上の整礎順序に基づいて行なわれる帰納法である。被覆集合は、変数を基礎項上で走らせることによって、すべての基礎項の組の組合せを被うことができるような項の組の集合である。被覆集合M、等式集合E、等式 e および整礎順序 $<$ が与えられたとき、次のことを証明すれば、e がEの帰納的定理であることが示せる。すべての $\xi \in M$ に対して

$$E \cup \{e[\xi'/p] \mid \xi' < \xi\} \vdash e[\xi/p]$$

このような被覆集合帰納法に対しても命題1と同様の命題が成り立つ。

【命題 2 】等式集合Eの下で、等式 $\alpha(\vec{x}) = \beta(\vec{x})$ とそれに関する被覆集合Mが与えられたとき、$\alpha(\vec{x}) = \beta(\vec{x})$ が変数の組 $\vec{x}$ に対して帰納的定理であることの証明の手続きはDTRCの項で次のように記述できる。

```
[ true,eq,and;t:
   eq(t,t) → true,
   and(true,...) → true ]
{ [ F;V:E ]
  { and(...) }}
```

但し、FとVはEの関数記号と変数の集合であり、andの引数の数はMの要素の数と同じで、各引数は $\xi \in$M に対して

$$[;: \alpha(\xi'_1) \to \beta(\xi'_1), \ldots, \alpha(\xi'_n) \to \beta(\xi'_n)]$$
$$\{\mathrm{eq}(\alpha(\xi), \beta(\xi))\}$$

の形をしている。ここに、ξ'_i は ξ より小さい項（の組）である。

上の項を書き換えることによって、[...]{true} の形の項に書き換えられるなら、$\alpha(\vec{x}) = \beta(\vec{x})$ が帰納的定理である。

【例 4】前例と同じ仮定の下で

$$x+y=y+x$$

が与えられたとする。このとき、p,q がメタ変数とすると、

$$\{<0,0>, <s(p),0>, <0,s(q)>, <s(p),s(q)>\}$$

が被覆集合となる。従って

$$x+y=y+x$$

[1] Eの要素が書換え規則の形で表せると仮定する。

が帰納的定理であることの被覆集合帰納法による証明の手続きはDTRCの項で次のように表現できる。

```
[true,eq,and;t:
   eq(t,t) → true,
   and(true,true,true,true) → true ]
{ [0,s,+;x,y:
     0+y → y,
     s(x)+y → s(x+y)]
   {and( [;:]{eq(0+0,0+0)},
         [;;p+0 → 0+p ]
         {eq(s(p)+0,0+s(p))},
         [;:0+q → q+0 ]
         {eq(0+s(q),s(q)+0)},
         [;:p+q → q+p,
            p+s(q) → s(q)+p,
            s(p)+q → q+s(p) ]
         {eq(s(p)+s(q),s(q)+s(p))})}}
```

この項の正規形は[...]{true}となるので証明が成功となる。

ここで用いた項の順序については[4]を参照されたい。

5 合流性

この節では、同じ枠組みを持つプログラムのクラスにおける合流性について議論する。

5.1 準備

まず、準備としてフレームの概念を導入する。

【定義 12】

1. 項$[F;V{:}R]\{t\}$に対して、$[F;V{:}R]\{\triangle\}$はフレームである。
2. 項$[F;V:R]\{t\}$とフレームfrに対して、$[F;V{:}R]\{fr\}$はフレームである。

フレームfrの$\triangle$に単純項tを代入することによって得られるプログラムを$fr\{t\}$と書く。

次に、プログラムおよびフレームについての合流性、弱合流性を定義する。

【定義 13】

1. **プログラムの合流性**
   プログラムPにおいて、任意の項TとT'に対して、$P \xrightarrow{*} T$, $P \xrightarrow{*} T'$[2]ならば、項T''があって、$T \xrightarrow{*} T''$, $T' \xrightarrow{*} T''$が成り立つとき、Pは合流性を持つという。
2. **プログラムの弱合流性**
   プログラムPにおいて、任意の項TとT'に対して、$P \to T$, $P \to T'$ならば、項T''があって、$T \xrightarrow{*} T''$, $T' \xrightarrow{*} T''$が成り立つとき、Pは弱合流性を持つという。
3. **フレームの合流性**
   すべての単純項tに対して$fr\{t\}$が合流性を持つなら、フレームfrは合流性を持つという。
4. **フレームの弱合流性**
   すべての単純項tに対して$fr\{t\}$が弱合流性を持つなら、フレームfrは弱合流性を持つという。

さらに、規則の重なり、システムの整合性、無矛盾性、プログラムの展開を定義する。

【定義 14】

1. $[F;V:R]$において$l_1 \to r_1 \in R$が文脈$C[\ ]$と代入$\sigma_1,\sigma_2(\mathrm{dom}(\sigma_1)\subseteq V)$の下で$l \to r \in R$に重なるというのは、$l$の変数でない部分項$l_2$が存在して、
   $$C[l_2] = l,\quad l_1\sigma_1 = l_2\sigma_2$$
   が成り立つときである。
2. すべての重なる$l_1 \to r_1, l \to r \in R$ ($C[\ ],\sigma_1,\sigma_2$の下で)に対して、
   $$C[r_1]\sigma_1 = r\sigma_2$$
   が成り立つとき、$[F;V:R]$が整合性を持つという。
3. すべての重なる$l_1 \to r_1, l \to r \in R$ ($C[\ ],\sigma_1,\sigma_2$の下で)に対して、項Tがあって、
   $$[F;V{:}R]\{C[r_1]\sigma_1\} \xrightarrow{*} T$$
   $$[F;V{:}R]\{r\sigma_2\} \xrightarrow{*} T$$
   が成り立つとき、$[F;V:R]$が無矛盾であるという。

【定義 15】プログラムPの展開$\overline{P}$を次のように定める。

1. $t \in \mathcal{V}$のとき、$\overline{t} = t$

[2] $\xrightarrow{*}$は$\to$の反射、推移的閉包である。

2. $t=f(t_1,\ldots,t_n)$, $\overline{t_i}=[F_i;V_i:R_i]\{u_i\}$ のとき、$\overline{t}=[\bigcup F_i;\bigcup V_i:\bigcup R_i]\{f(u_1,\ldots,u_n)\}$
3. $t=[F;V:l_1\to r_1,\ldots,l_n\to r_n]\{t'\}$, $\overline{t'}=[F';V':R']\{u'\}$, $\overline{r_i}=[H_i;W_i:P_i]\{r_i'\}$ のとき、$\overline{t}=[F\cup F'\cup\bigcup H_i;V\cup V'\cup\bigcup W_i:\{l_i\to r_i\}\cup R'\cup\bigcup P_i]\{u'\}$

フレーム fr を $\triangle$ を定数記号として展開したものを fr の展開とし、$\overline{fr}$ と書く。

5.2 合流性

次に、三つの場合に分けて、フレームの合流性について考察する。

CASE I: フレーム fr は

$$fr=[F;V:R]\{\triangle\}$$

という形をいていて、すべての $l\to r\in R$ に対して、r は単純項である。

【定理 1 】

1. fr が整合性を持つならば、合流性を持つ。
2. fr が無矛盾であることと弱合流性を持つことは等価である。

【証明】任意の固定した単純項 t に対して、$P=fr\{t\}$ とする。

まず、2. を示す。P が無矛盾と仮定し、$P\to M,P\to N$ に対して、$M\xrightarrow{*}T$ かつ $N\xrightarrow{*}T$ となる T を見つければ良い。$P\to M,P\to N$ にそれぞれ $l\to r$ と $l_1\to r_1$ が適用されたとする。また、$P\to N,P\to M$ でそれぞれ t の部分項 s と s_1 を書き換えたとする。このとき、s と s_1 が重なりがなければ、t の出現 s に対応する N の部分項に $l\to r$ を適用させて、T が得られるとすると、明らかに $M\to T,N\to T$ が成り立つ。s と s_1 が重なりがあるとする。このとき、文脈 $C[\ ]$ と代入 σ_1,σ_2 が存在して、$l_1\to r_1$ が $(C[\ ],\sigma_1,\sigma_2$ の下で) $l\to r$ に重なるか、または、$l\to r$ が $((C[\ ],\sigma_1,\sigma_2$ の下で) $l_1\to r_1$ に重なる。前者の場合、文脈 $C'[\ ]$ が存在して、$t=C'[s]$, $N=[F;V:R]\{C'[C[r_1]\sigma_1]\}$, $M=[F;V:R]\{C'[r\sigma_2]\}$ が成り立つ。無矛盾性の定義により、$C[r_1\sigma_1]$ または $r\sigma_2$ の部分だけを書き換えていくと、項 T があって、$N\xrightarrow{*}T,M\xrightarrow{*}T$ が成り立つ。後者についても同様に証明できる。

以上により、fr が無矛盾であれば、弱合流性を持つことが証明できた。

fr が無矛盾でなければ、書換え規則 $l\to r,l_1\to r_1$ が存在して、$l_1\to r_1$ が $(C[\ ],\sigma_1,\sigma_2$ の下で) $l\to r,l$ に重なり、

$$fr\{l_1\}\to fr\{C[r_1\sigma_1]\}$$
$$fr\{l_1\}\to fr\{r\sigma_2\}$$

が成り立つが、任意の T に対して、

$$fr\{C[r_1]\sigma_1\}\xrightarrow{*}T$$
$$fr\{r\sigma_2\}\xrightarrow{*}T$$

が成り立たないので、fr は弱合流性を持たない。

次に、1. を示す。$P\xrightarrow{i}M,P\xrightarrow{*}N$ に対して、$M\xrightarrow{*}T$ かつ $N\xrightarrow{*}T$ となる T を見つける。i に関する帰納法により、次を証明すれば良い。

(1) $P\to M,P\xrightarrow{j}N\Rightarrow(\exists T)M\xrightarrow{*}T,N\xrightarrow{*}T$

(1) を証明するために、j に関する帰納法を用いて、次を証明すれば良い。

(2) $P\to M,P\to N\Rightarrow(\exists T)M\xrightarrow{*}T,N\to T$

(2) は、P が整合性を持つことを利用すれば、弱合流性の証明と同じように証明できる。

また、下の例で示すように、無矛盾であるが、合流性を持たないフレームが存在する。

【例 5】

```
fr=[ ; :b→a,b→c,c→b,c→d ]
   {△}
```

とすると、fr が無矛盾で、弱合流性を持つが、fr の下で、$b\xrightarrow{*}d$, $b\to a$ が成り立つにもかかわらず、任意の t に対して、$d\xrightarrow{*}t,a\xrightarrow{*}t$ が成り立たない。

CASE II: フレーム fr は

$fr=[F_1;V_1:R_1]\{\ldots\{[F_n;V_n:R_n]\{\triangle\}\}\ldots\}$

の形をしていて、すべての $l\to r\in\bigcup R_i$ に対して、l と r が単純項である。[3]

【定理 2 】

1. 次の条件を満たすならば、fr は合流性を持つ。

   (a) fr の展開 $\overline{fr}$ は合流性を持つ。

[3]束縛されている記号はすべて異なるように名前替されていると仮定する。

(b) すべての $1 \leq j < i \leq n$ に対して、$l \to r \in R_i, l_1 \to r_1 \in R_j$ なら、l が l_1 と r_1 に照合しない。

2. fr が弱合流性を持つことと $\overline{fr}$ が弱合流性を持つことは等価である。

【証明】書換えの過程で、書換え規則を変化させることがないということに着目すれば、CASE I と同様に証明できる。

【例 6】

```
P=[;:c → b,c → d]
    {[;:a → d,a → b,b → c]{a}}
```

とすると、P は弱合流性を持つが、

```
P → [;:c → b,c → d]
    {[;:a → d,a → b,b → c]{d}}
  → d

P → [;:c → b,c → d]
    {[;:a → d,a → b,b → b]{a}}
  → [;;c → b,c → d]
    {;:a → d,a → b,b → b]{b}}
```

が成り立ち、明らかに合流性を持たない。

CASE III: フレーム fr は

$$fr = [F; V{:}R]\{\triangle\}$$

の形をしていて r が単純項でない $l \to r \in R$ が存在する。

この場合、fr が弱合流性を持つ条件は CASE II と同じである。

【定理 3 】次の条件を満たすとき、f が合流性を持つ：

1. fr の展開 $\overline{fr}$ は合流性を持つ。
2. r が単純項でないようなすべての $l \to r \in R$ に対して、任意の $l_1 \to r_1 \in R$ の l_1 が r の中のあらゆる書換え規則の右辺と左辺に照合しない。

【証明】CASE I とほぼ同様にできる。

【例 7】

```
P= [;:e → [;:a → d,a → b,b → c]{a},
       c → b,
       c → d]{e}
  → [;:e → [...]{a},
       c → b,
       c → d:]
    {[;:a → d,a → b,b → c]{a}}
```

とすると、P は弱合流性を持つが、

```
P → [;:e → [;:a → d,a → b,b → c]{a},
       c → b,c → d]
    {[;:a → d,a → b,b → c]{d}}
  → d

P → [;:e → [;:a → d,a → b,b → c]{a},
       c → b,c → d]
    {[;:a → d,a → b,b → b]{a}}
  → [;:e → [;:a → d,a → b,b → c]{a},
       c → b,c → d]
    {;:a → d,a → b,b → b]{b}}
```

となり、合流性を持たない。

以上より、すべての場合、フレーム fr の展開 $\overline{fr}$ が弱合流性を持つならば、fr も弱合流性を持つ。また、弱合流性の条件は必要十分条件である。しかし、合流性について、同じことがいえない。原因は書換え規則を変化させることにあって、これこそが DTRC の重要な特徴の一つでもある。このような場合、合流性を期待すべきもない。どうしても合流性を持つ項にしたいとき、各階層ごとに相異なる記号を使えば良い。

6 DTRC の停止性

TRS と同様 DTRC に於いては項の停止性が重要な意味を持つ。ここでは、TRS で停止性の研究でよく用いられた整礎擬順序を DTRC の項の集合に拡張し、更に新たな補助関係を付けることによって DTRC の停止性について考察する。

6.1 準備

【定義 16】次の条件を満たす関数

$$\mathcal{S}: TERM(\mathcal{F}, \mathcal{V}) \to TERM(\mathcal{F}, \mathcal{V})$$

を戦略という。

1. $t \equiv \mathcal{S}(t)$ t が正規形のとき
2. $t \to \mathcal{S}(t)$ そうでないとき

【定義 17 】戦略 $\mathcal{S}$ が正規化戦略であるとは、任意の項 t に対して、t が正規形を持つなら、$\{\mathcal{S}^n(t) | n \geq 0\}$ は t の正規形を含む。

【定義 18】正規形を持つ項 t が戦略 $\mathcal{S}$ に対して停止するとは $\{\mathcal{S}^n(t) | n \geq 0\}$ が t の正規形を含むことをいう。

6.2 順序と停止性

TRSの停止性は整礎擬順序を用いて議論される[2]。ここではDTRCの単純項に擬順序$\geq$が定義されたとき、$\geq$に対応するDTRCの擬順序$\succeq$を定義し、更に、書き換え規則の間の擬順序$\gg$を付ける。

【定義 19】 $TERM(\mathcal{F},\mathcal{V})$上のシステム$Sys=[F;V:R], Sys'=[F';V':R']$に対して、次の条件を満たす写像$\alpha:\mathcal{F}\bigcup\mathcal{V}\rightarrow\mathcal{F}\bigcup\mathcal{V}$を$Sys$から$Sys'$への埋込写像という。

1. $f\in\mathcal{F}\Rightarrow\alpha(f)\in\mathcal{F}$
2. $v\in\mathcal{V}\Rightarrow\alpha(v)\in\mathcal{V}$
3. $\alpha|_{F\bigcup V}$は単射かつ range$(\alpha|_{F\bigcup V})=F'\bigcup V'$
4. $l\rightarrow r\in R$ならば$\alpha(l)\rightarrow\alpha(r)\in R'$

【定義 20 】単純項の間の擬順序$\geq$に対して、DTRCの項、書き換え規則及びシステムのそれぞれの間の関係$\succeq$は次の条件を満たす最小の擬順序であるとする。

1. s,tが単純項なら、$s\geq t\Rightarrow s\succeq t$
2. 項l,l',r,r'に対して、$l\succeq l',\ r\succeq r'\Rightarrow l\rightarrow r\succeq l'\rightarrow r'$
3. システム$Sys=[F;V:R],\ Sys'=[F';V':R']$に対して、$Sys$から$Sys'$への埋込写像$\alpha$が存在して、すべての$l\rightarrow r\in R$に対して、$\alpha(l\rightarrow r)\succeq l\rightarrow r$であるとき、Sys$\succeq$Sys'
4. 項s,t, システム$Sys=[\ldots],\ Sys'=[\ldots]$に対して、$s\succeq t, Sys\succeq Sys'\Rightarrow Sys\{s\}\succeq Sys'\{t\}, Sys\{s\}\succeq t$

以下では$s\geq t$なら$\forall\sigma\ s\sigma\geq t\sigma$とする。また、$s\succeq$かつ$t\not\geq s$のとき、$s\succ t$と書く。

【定理 4 】 $\geq$が整礎であるとき、$\succeq$は整礎である。

各プログラム$P=[F;V:l_1\rightarrow r_1,\ldots,l_n\rightarrow r_n]\{t\}$ に$P'(F'(F),V'(V),l_1,r_1,\ldots,l_n,r_n,t)$の形の項を対応させてTRSの場合とほぼ同様に証明できる。但し、P',F',V'は$\mathcal{F}$に現れない特別な関数で、かつ、各Pに対して、これらの関数記号がそれぞれ異なるとする。

【定義 21】 $\succ$から、次のようにして$\gg$を定める。

項l,l',r,r'に対して、すべての文脈$C[\ ]$と代入σに対して、$C[l]\sigma=l'\ \Rightarrow\ C[r]\sigma\succ r'$が成り立つなら、$l\rightarrow r\gg l'\rightarrow r'$とする。

【命題 3 】 $\geq$が擬順序であるなら、$\gg$は推移律を満たす。

【定義 22】書換え規則r,r'が比較不能であるのは、r,r'の左辺がお互いに他方に照合しないときである。

【定義 23 】 DTRCの項の集合Cは次の条件を満たすとき、$\succ$減少閉クラスであるという。

1. tCに出現する任意の書き換え規則l→rに対してl$\succ$r ($\succ$減少)。
2. Cにあるすべての項sに対して、sがtに書換えられるなら、tもCの要素である(閉クラス)。

【定理 5 】 DTRCの単純項上の整礎な順序に対して、$\succ$減少閉クラス中のすべての項は任意の戦略に対して停止する。

【定理 6 】 各$C_i(i=1\ldots n)$が整礎な擬順序$\geq$について$\succ$減少閉クラスであるとき、

$\bigcup_{i=1}^{n} C_i$

は$\succ$減少閉クラスである。

上の2つの定理は$\succ$減少閉クラスの定義と定理1によって明らかである。

以下では、DTRCの単純項上の整礎な擬順序$\geq$を固定して話を進める。また、項tの部分項$[F;V:R]\{t'\}$のRに含まれる書換え規則$l\rightarrow r$もtの書換え規則と呼ぶ。

【定理 7 】以下の条件を満たす項tの集合C_1は$\succ$減少閉クラスである。

1. tが定義23の1を満たす。
2. 同階層にないtの書き換え規則r,r'は
   (a) tの出現$[\ldots;\ldots:R]\{t'\}$が存在して、rがRに含まれて、r'がt'に含まれるなら、rの左辺はr'の左辺の部分項に照合しない。そうでなければ、
   (b) r,r'は比較不能である。

【系 1】

- C_1中のすべての項は停止する。
- すべての戦略はC_1中の項に対して正規化戦略である。

【例 8】項 t を

```
[ true,eq;u,v:
    eq(u,u,v,v) → true ]
{ [0,s,+;x,y:
      0+y → y,
      s(x)+y → s(x+y) ]
  { [;:(p+q)+r → p+(q+r)]
    {eq((0+q)+r,0+(q+r),
        (s(p)+q)+r,s(p)+(q+r)) }}}
```

とする。t にある記号上の単純項 w に対して、$|w|$ を w の長さ（要素の数）を表すものとして、次のように $>$ を定義する：

$|w| > |w'| \Rightarrow w > w'$

$|w| = |w'|, w = x + y, w' = s(z) \Rightarrow w > w'$

$|w| = |w'|, w = x + x', w' = y + y', x > y$ or $x = y, x' > y' \Rightarrow w > w'$

$\geq$ から定義される順序 $\succeq$ は整礎で、この順序に関して t$\in C_1$ なので t は停止する。

【例 9】

```
[ ; :f(a) → g(a), f(a) → b, g(a) → a ]
{ [ ; :f(a) → a ]
  {a}}
```

整礎な順序 $\geq$ が存在して、上の項 t は定義 23 の 1 の条件を満たし、書換え規則 r によって書換え規則 r' が書き換えられるとき、$r \gg r'$ が成り立つが、t が正規形を持つにも かかわらず、下のように正規形に到達できない書換えが存在する。

```
t →[ ; :f(a) → g(a), f(a) → b, g(a) → a ]
    { [ ; :g(a) → a ]
      {a}}
  →[ ; :f(a) → g(a), f(a) → b, g(a) → a ]
    { [ ; :a → a ]
      {a}}
```

C_1 は書き換え規則が他のすべての書き換え規則の左辺を書き換えることのできない項の集合である。これに対して、書き換え規則が他のすべての書き換え規則の右辺を書き換えることのできない項の集合 C_2 も $\succ$ 減少閉クラスである。

【定理 8 】以下の条件を満たす項 t の集合 C_2 は $\succ$ 減少閉クラスである。

1. t が定義 23 の 1 を満たす。
2. t の書き換え規則 $l \to r, l' \to r'$ に対して、
   (a) l が l' の部分項に照合すれば $l \to r \gg l' \to r'$
   (b) r は l' の変数でない部分項と単一化できない。

【系 2】

- C_2 中のすべての項は停止する。
- すべての戦略は C_2 中の項に対して正規化戦略である。

【例 10】整礎な順序 $\geq$ が存在して、次の項は $\geq$ に対して、C_2 に入っているので、停止する。

```
[ ;x,y:f(x,y) → b ]
{ [ ;x:f(x,x) → a ]
  {f(0,0)}}
```

7 まとめ

本稿では、TRS の操作の記述、検証などへの応用を目的として、書換え規則を動的に変化させながら項を書き換えていく新しい計算モデル — 動的項書換え計算（DTRC）を提案し、DTRC の特徴、性質および応用について考察した。DTRC に表示的意味を与えることなどは今後の課題である。

謝辞

日頃御指導、御討論賜る平田富夫助教授、直井徹岐阜大学助教授、外山勝彦中京大学講師、杉野花津江助手、酒井正彦助手、結縁祥治助手並びに研究室の皆様に感謝致します。

参考文献

[1] Baker F. T.,*Organizing for Structured Programming*,Lecture Notes in Computer Science 23,pp38-49,1975

[2] Dershowitz.N., *Orderings for term-rewriting systems*, Theoretical Computer Science,17(3),1982,

[3] Klop J. W.,*Term Rewriting Systems:a tutorial*, Centre for Mathematics and Computer Science, Kruislaan 413 (1980)

[4] 酒井，坂部，稲垣，代数的仕様の検証のための被覆集合帰納法，電子情報通信学会，COMP90-5, 1990, pp.37-46