LA-009

項書き換えシステムの合流性自動判定

Automating Confluence Check of Term Rewriting Systems

吉田 順一 † 青戸 等人 † 外山 芳人 †
Junichi Yoshida Takahito Aoto Yoshihito Toyama

1. はじめに

停止性と合流性は項書き換えシステムの重要な性質であり，さまざまな判定法が提案されている [1, 6]．停止性については，依存対解析に基づいた強力な停止性自動判定システムがすでに開発されている [2, 3]．一方，合流性については，定理自動証明などで広く利用されているにもかかわらず，依存対解析のような強力な判定方法は知られていない．また，現在提案されているさまざまな判定方法 [6] も適用範囲が限られているため，強力な合流性自動判定システムの開発は困難であった．

本研究では，複数の判定方法を組み合わせた合流性自動判定システムを提案する．我々の提案するシステムでは，判定条件が直接適用できない複雑な項書き換えシステムを自動分解し，分解された部分システムに対して適切な合流性判定条件を適用することによって，全体の合流性を自動判定する．したがって，本システムでは，これまで判定困難であった広いクラスの項書き換えシステムの合流性自動判定が可能である．

2. 項書き換えシステム

項書き換えシステムは書き換え規則の有限集合である [1]．自然数 $0, 1, 2, \ldots$ を $0, S(0), S(S(0)), \ldots$ と表現すると，自然数上での加算は以下の項書き換えシステムで与えられる．

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

たとえば，$1 + 1 = 2$ の計算は，R では $S(0) + S(0) \to S(S(0) + 0) \to S(S(0))$ なる書き換えで実現できる．このとき，項 $S(S(0))$ のようにこれ以上書き換えできない項を正規形という．

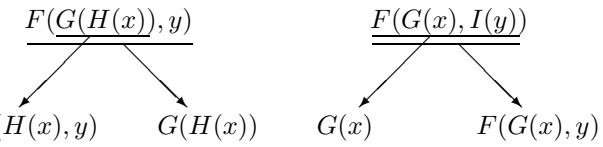
項 s から t に 0 回以上の書き換えで到達できるとき $s \xrightarrow{*} t$ と記す．任意の項 t, t_1, t_2 について，$t_1 \xleftarrow{*} t \xrightarrow{*} t_2$ ならばある項 s が存在して $t_1 \xrightarrow{*} s \xleftarrow{*} t_2$ となるとき，項書き換えシステム R は合流性 (Church-Rosser 性) をもつといい $CR(R)$ と記す．無限列 $t_0 \to t_1 \to t_2 \to \cdots$ が存在しないならば，R は停止性をもつという．どの変数も 1 回しか現れない項を線形であるという．すべての書き換え規則の左辺が線形ならば R は左線形であるという．

危険対 [1] は，項書き換えシステムの合流性判定に重要である．たとえば，次の項書き換えシステムを考える．

$$R = \left\{ \begin{array}{lcl} F(G(x), y) & \to & G(x) \\ G(H(x)) & \to & H(x) \\ F(x, I(y)) & \to & F(x, y) \end{array} \right.$$

†東北大学 電気通信研究所

このとき，項 $F(G(H(x)), y)$ や $F(G(x), I(y))$ からは以下のように 2 通りの書き換えが可能である．



ここで，項 $F(G(H(x)), y)$ から得られた項の対 $\langle F(H(x), y), G(H(x)) \rangle$ のように，項全体と真部分項をそれぞれ書き換えて得られた項の対を内側危険対という [5]．一方，項 $F(G(x), I(y))$ から得られた項の対 $\langle G(x), F(G(x), y) \rangle$ のように，項全体を書き換えて得られた項の対を外側危険対という [5]．R の内側危険対の集合を $CP_{in}(R)$，外側危険対の集合を $CP_{out}(R)$ と記す．R の危険対の集合を $CP(R) = CP_{in}(R) \cup CP_{out}(R)$ とする．

3. 合流性基本判定条件

項書き換えシステムの合流性は一般的には決定不能であるが，さまざまな決定可能条件や十分条件が知られている [1, 6]．本研究では，以下の判定条件を用いて合流性の判定を行う．

定理 1 (Knuth-Bendix の合流性判定条件 [1]) 停止性をみたす項書き換えシステム R が合流性をみたすための必要十分条件は R の危険対の要素が同じ正規形をもつことである．

項書き換えシステムが停止性をみたすとき，Knuth-Bendix の合流性判定条件は決定可能であり，非合流の場合には危険対から得られた反例が求まる．

項書き換えシステム R の並行書き換え $s \multimap\!\!\to_R t$ とは，独立して書き換え可能な部分項を任意個選び，同時に書き換えて t が得られることである [7]．

定理 2 (Oostrom の合流性条件 [7]) 以下の条件をみたす左線形項書き換えシステム R は合流性をみたす．

- $\forall \langle c_1, c_2 \rangle \in CP_{out}(R), \exists s, c_1 \multimap\!\!\to s \leftarrow\!\!\multimap c_2$
- $\forall \langle c_1, c_2 \rangle \in CP_{in}(R), c_1 \multimap\!\!\to c_2$

与えられた項から並行書き換えで得られる項は有限個なので，Oostrom の合流性条件は決定可能となる．なお，Oostrom の合流性条件は十分条件であるため，この条件をみたさない場合でも，非合流と判定することはできない．

本論文で用いる合流性基本判定アルゴリズムは，与えられた項書き換えシステムの停止性を最初に調べ，停止性をみたす場合には Knuth-Bendix 条件で合流・非合流

を決定する．停止性判定に失敗した場合，Oostrom 条件に基づいて合流性の判定を試みる．この判定に失敗した場合は合流・非合流は判定不能である．したがって，合流性基本判定アルゴリズムは以下の 3 つの結果を返す．

CR	R は合流．
NONCR	R は非合流．
UNKNOWN	R の合流・非合流は判定不能．

合流性基本判定アルゴリズム crcheck を以下に示す．

```
fun crcheck rs =
    if (lpocheck rs) then
        if (wcrcheck rs) then CR else NONCR
    else
        if (oostromcheck rs) then CR else UNKNOWN
```

このアルゴリズムにおいて，lpocheck は入力された項書き換えシステム rs に対して辞書式経路順序に基づく自動停止性判定を文献 [8] に基づいた方法で行う．wcrcheck は危険対の要素が同じ正規形をもつか否かを調べ，Knuth-Bendix 条件による自動判定を行い，非合流ならば反例を出力する．oostromcheck は Oostrom 条件による自動判定を行う．

4. 項書き換えシステムの直和分解

項書き換えシステム R の合流性判定が困難な場合に，R の部分システムの合流性から R 全体の合流性を導く方法が知られている [4, 5, 6]．

定義 1 (直和) 項書き換えシステム R_1, R_2 に含まれる関数記号の集合が互いに素であるとき，R_1 と R_2 は直和であるといい，$R_1 \cup R_2$ を $R_1 \oplus R_2$ と表す．

定理 3 (直和システムの合流性 [4]) R_1, R_2 を項書き換えシステムとする．このとき，$CR(R_1) \wedge CR(R_2) \Leftrightarrow CR(R_1 \oplus R_2)$．

以下の例では，定理 3 を用いて合流性を判定する．

例 1 R を以下の項書き換えシステムとする．

$$R = \left\{ \begin{array}{lcl} F(x, A(G(x))) & \to & G(F(x,x)) \\ F(x, G(x)) & \to & G(F(x,x)) \\ A(x) & \to & x \\ H(x) & \to & H(B(H(x))) \end{array} \right.$$

R は停止性をみたさず左線形でもないため，Knuth-Bendix 条件も Oostrom 条件も適用できない．ここで，R を以下のように直和分解する．

$$R_1 = \left\{ \begin{array}{lcl} F(x, A(G(x))) & \to & G(F(x,x)) \\ F(x, G(x)) & \to & G(F(x,x)) \\ A(x) & \to & x \end{array} \right.$$

$$R_2 = \left\{ \begin{array}{lcl} H(x) & \to & H(B(H(x))) \end{array} \right.$$

このとき，R_1 に対しては Knuth-Bendix 条件，R_2 に対しては Oostrom 条件が適用可能となり，定理 3 より R の合流性が示される．

項書き換えシステム R を $R = R_1 \oplus R_2 \oplus \cdots \oplus R_n$ に直和分解したとき，$R_1, R_2, \ldots, R_n$ をそれぞれ直和分解成分とよぶ．このとき，直和分解成分が小さいほど合流性判定法の適用が容易となる．最小の直和分解成分は以下のアルゴリズムによって得られる．

```
fun dj_decompose rs =
    let fun dj_merge rss =
        if ∃ rs1 rs2 ∈ rss, not(djcheck rs1 rs2)
            then dj_merge ((rss \ {rs1,rs2}) ∪ {rs1∪rs2})
        else rss
    in
        dj_merge (map (fn r ⇒ {r}) rs)
```

ここで，djcheck rs1 rs2 は 項書き換えシステム rs1 と rs2 の直和性を判定する．

直和分解に基づく合流性判定アルゴリズム dj_crcheck を以下に示す．

```
fun dj_crcheck rs =
    let
        val status = map crcheck (dj_decompose rs)
    in
        if (∀ result ∈ status, result = CR)
            then CR
        else if (∃ result ∈ status, result = NONCR)
            then NONCR
        else
            UNKNOWN
```

このアルゴリズムでは，dj_decompose を用いて求められた最小の直和分解成分に対して合流性基本判定 crcheck を行う．直和分解成分がすべて合流ならば R は合流，1 つでも非合流ならば，R は非合流と判定して反例を出力する．それ以外の場合は，合流・非合流は判定不能である．ここで，定理 3 より，直和分解成分の反例は全体の反例となっていることに注意する．

5. 項書き換えシステムの可換分解

項書き換えシステム R が直和分解できない場合，R を部分システムに分解して合流性を判定する別の方法として可換分解が知られている [5]．

定義 2 (可換) 項書き換えシステム R_1, R_2 が可換であるとは，$t_1 \overset{*}{\leftarrow}_{R_1} t \overset{*}{\rightarrow}_{R_2} t_2$ ならば $t_1 \overset{*}{\rightarrow}_{R_2} s \overset{*}{\leftarrow}_{R_1} t_2$ となる s が存在することである．このとき，$R_1 \cup R_2$ を $R_1 \sqcup R_2$ と記す．

定理 4 (可換システムの合流性 [5]) R_1, R_2 を項書き換えシステムとする．このとき，$CR(R_1) \wedge CR(R_2) \Rightarrow CR(R_1 \sqcup R_2)$.

以下の例では，定理 4 を用いて合流性を判定する．

例 2 R を以下の項書き換えシステムとする．

$$R = \left\{ \begin{array}{lcl} W(B(x)) & \to & W(x) \\ B(I(x)) & \to & J(x) \\ W(I(x)) & \to & W(J(x)) \\ F(H(x), y) & \to & F(H(x), G(y)) \\ F(x, I(y)) & \to & F(G(x), I(y)) \\ G(x) & \to & x \end{array} \right.$$

R は停止性をみたさず左線形でもないため，Knuth-Bendix 条件も Oostrom 条件も適用できない．また，R を直和に分解することもできない．ここで，R を以下のように可換分解する．

$$R_1 = \left\{ \begin{array}{lcl} W(B(x)) & \to & W(x) \\ B(I(x)) & \to & J(x) \\ W(I(x)) & \to & W(J(x)) \end{array} \right.$$

$$R_2 = \left\{ \begin{array}{lcl} F(H(x),y) & \to & F(H(x),G(y)) \\ F(x,I(y)) & \to & F(G(x),I(y)) \\ G(x) & \to & x \end{array} \right.$$

このとき，R_1 に対しては Knuth-Bendix 条件，R_2 に対しては Oostrom 条件が適用できるので，定理 4 より R の合流性は示される．

項書き換えシステム R を $R = R_1 \sqcup R_2 \sqcup \cdots \sqcup R_n$ に可換分解したとき，$R_1, R_2, \ldots, R_n$ をそれぞれ可換分解成分とよぶ．本研究では，以下の定理を用いて可換分解を行う．

定理 5 ([5]) 左線形項書き換えシステム R_1，R_2 において，R_1 の書き換え規則と R_2 の書き換え規則の間に危険対が存在しないとき，R_1 と R_2 は可換である．

定理 5 に基づく最小の可換分解成分を求めるアルゴリズムを以下に示す．

```
fun nolp_decompose rs =
  let fun nolp_merge rss =
      if ∃ rs1 rs2 ∈ rss, not(nolpcheck rs1 rs2)
       then nolp_merge ((rss \ {rs1,rs2}) ∪ {rs1∪rs2})
      else rss
  in
    nolp_merge (map (fn r ⇒ {r}) rs)
```

ここで，nolpcheck rs1 rs2 は rs1 の書き換え規則と rs2 の書き換え規則の間に危険対が存在しないとき真となる．

上記で求めた最小の可換分解成分に対して，直和分解と同様に合流性基本判定 crcheck を行い，全体の合流性の判定を試みると，以下のような問題が生じる．

例 3 以下の項書き換えシステム R を考える．

$$R = \left\{ \begin{array}{lcl} F(H(x),y) & \to & G(H(x)) \\ H(I(x)) & \to & I(x) \\ F(I(x),y) & \to & G(I(x)) \end{array} \right.$$

Knuth-Bendix 条件から R は合流性をもつ．一方，上記の手続きを R に適用すると，以下の R_1 と R_2 に可換分解される．

$$R_1 = \left\{ \begin{array}{lcl} F(H(x),y) & \to & G(H(x)) \\ H(I(x)) & \to & I(x) \end{array} \right.$$

$$R_2 = \left\{ \begin{array}{lcl} F(I(x),y) & \to & G(I(x)) \end{array} \right.$$

このとき，$F(I(x),y) \leftarrow F(H(I(x)),y) \to G(H(I(x)))$ なる R_1 の書き換えを，R_1 で合流させることはできない．これを合流させるためには R_2 の書き換え規則が必要である．したがって，最小の可換分解成分を直接用いると，合流性判定に失敗する場合がある．

我々のアルゴリズムでは，この問題を解決するため，最小の可換分解成分から始めて，各成分が合流性をみたすまで可換分解成分の合併を繰り返す．以下に可換分解に基づく合流性判定アルゴリズムを示す．

```
fun com_crcheck rs =
  if (llcheck rs) then
      let fun com_crcheck1 { } = UNKNOWN
            | com_crcheck1 (rss::rsss) =
              if ∃ rs ∈ rss, (crcheck rs ≠ CR) then
              let
              val rsss1 = map
                (fn ⟨rss1,rss2⟩ ⇒ { rs ∪ ⋃ rss1 } ∪ rss2)
                { ⟨rss1,rss2⟩ | rss1 ⊎ rss2 = rss \ { rs },
                                rss1 ≠ ∅ }
              in
                 com_crcheck1 (rsss ∪ rsss1)
              else CR
      in
        com_crcheck1 {(nolp_decompose rs)}
  else UNKNOWN
```

手続き llcheck は左線形性を調べ，左線形でない場合は可換分解を行わない．rsss1 $\subseteq \{\{R_1, \ldots, R_n\} | R = R_1 \sqcup \cdots \sqcup R_n\}$ は可換分解候補の集合であり，すべての成分が合流性をみたす可換分解候補がある場合は R は合流，そのような可換分解候補がない場合は R の合流・非合流は判定不能である．

6. 合流性自動判定システムの実装と実験

これまで説明した合流性基本判定 crcheck，直和分解に基づく判定 dj_crcheck，可換分解に基づく判定 com_crcheck を組み合わせた合流性自動判定システムの構成を図 1 に示す．本システムの実装は関数型言語 SML を用いた．

本システムでは，入力された項書き換えシステム R に対して最初に合流性基本判定 crcheck を行い，適用が失敗した場合は，R を直和分解しそれぞれの成分 $R_1, \ldots, R_n$ に対して合流性基本判定 crcheck を再び試みる．成分がすべて合流ならば R は合流，1 つでも非合流ならば R は非合流と判断する．合流・非合流が判定不能な成分 R_i が左線形の場合には，R_i の可換分解を行い，それぞれの成分 $R_{i1} \ldots R_{im}$ に対して合流性基本判定 crcheck を行う．成分がすべて合流ならば R_i は合流と判断する．

例 4 以下の項書き換えシステム R を合流性自動判定システムで判定する．

$$R = \left\{ \begin{array}{lcl} F(H(x),y) & \to & F(H(x),G(y)) \\ F(x,I(y)) & \to & F(G(x),I(y)) \\ G(x) & \to & x \\ W(W(x)) & \to & W(x) \\ B(I(x)) & \to & W(x) \\ W(B(x)) & \to & B(x) \\ S(x,T(x)) & \to & T(x) \end{array} \right.$$

R に対しては Knuth-Bendix 条件も Oostrom 条件も適用できない．また，直和分解あるいは可換分解のみでは

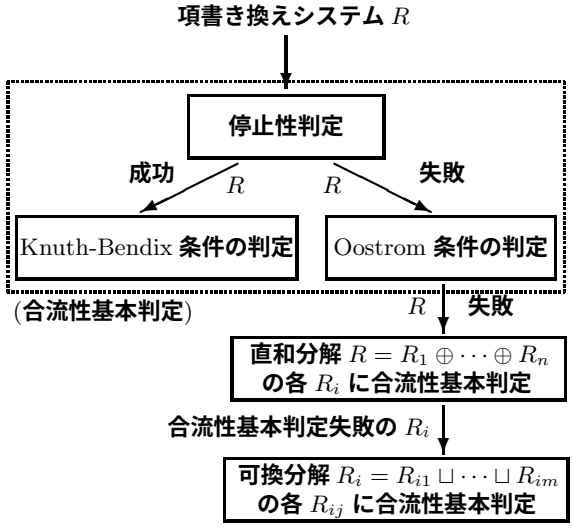


図 1: **合流性自動判定システムの構成**

合流性を判定できない．しかし，我々の自動判定システムでは，R を自動的に以下の 3 成分に分解し，合流性判定に成功する (図 2)．

$$R_1 = \left\{ \begin{array}{lcl} F(H(x),y) & \to & F(H(x),G(y)) \\ F(x,I(y)) & \to & F(G(x),I(y)) \\ G(x) & \to & x \end{array} \right.$$

$$R_2 = \left\{ \begin{array}{lcl} W(W(x)) & \to & W(x) \\ B(I(x)) & \to & W(x) \\ W(B(x)) & \to & B(x) \end{array} \right.$$

$$R_3 = \left\{ \begin{array}{lcl} S(x,T(x)) & \to & T(x) \end{array} \right.$$

7. おわりに

本論文では，項書き換えシステムの合流性自動判定システムを実現した．本システムの特徴は，与えられた項書き換えシステムを適切な部分システムに自動分解し，それぞれの部分システムに適した合流性判定条件を適用することで，全体の合流性を自動判定する点にある．このようなアプローチに基づく合流性自動判定アルゴリズムは，我々の知る限りこれまで提案されていない．さまざまな判定条件を組み合わせた強力な合流性判定を実現するためには，本システムのアプローチが極めて有効であると考えられる．実際，我々の実装したシステム上での実験では，従来の合流性判定方法が適用困難な項書き換えシステムに対しても，適切な部分システムに分解することにより，合流性の自動判定に成功した．合流性が決定可能なクラス [6] に対する自動判定や，これまで提案されているさまざまな合流条件 [6] の自動判定を本システムに組み込み，さらに強力な判定システムを実現することは今後の課題である．

(入力)

```
djcom_crcheck
( IO.rdrules [
 "F(H(x),y) -> F(H(x),G(y))",
 "F(x,I(y)) -> F(G(x),I(y))",
 "G(x) -> x",
 "W(W(x)) -> W(x)",
 "B(I(x)) -> W(x)",
 "W(B(x)) -> B(x)",
 "S(x,T(x)) -> T(x)"
]);
```

(出力)

```
(disjoint)[W,B,I,G,H,F]
(Com)
[ W(W(x)) -> W(x),
  W(B(x)) -> B(x),
  B(I(x)) -> W(x) ]
'Terminating' and 'WCR' :CR
(Com)
[ F(H(x), y) -> F(H(x), G(y)),
  F(x, I(y)) -> F(G(x), I(y)),
  G(x) -> x ]
'Unknown-Terminating' and 'Left-Linear'
 and 'Oostrom' :CR
djResult: CR
(disjoint)[T,S]
[ S(x, T(x)) -> T(x) ]
'Terminating' and 'WCR' :CR

Result: CR
val it = true : bool
```

図 2: **合流性自動判定システムの実行例**

参考文献

[1] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, Cambridge, 1998.

[2] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Mechanizing and improving dependency pairs. *Journal of Automated Reasoning*, Vol. 37, No. 3, pp. 155–203, 2006.

[3] N. Hirokawa and A. Middeldorp. Tyrolean termination tool: Techniques and features. *Information and Computation*, Vol. 205, No. 4, pp. 474–511, 2007.

[4] Y. Toyama. On the Church-Rosser property for the direct sum of term rewriting systems. *Journal of ACM*, Vol. 34, No. 1, pp. 128–143, 1987.

[5] Y. Toyama. Commutativity of term rewriting systems. In K. Fuchi and L. Kott, editors, *Programming of future generation computers II*, pp. 393–407. North-Holland, Amsterdam, 1988.

[6] Y. Toyama. Confluent term rewriting systems (invited talk). In *Proc. 16th RTA, LNCS 3467*, p. 1, 2005. Slides are available from `http://www.nue.riec.tohoku.ac.jp/user/toyama/slides/toyama-RTA05.pdf`.

[7] V. van Oostrom. Developing Developments. *Theoretical Computer Science*, Vol. 175, No. 1, pp. 159–181, 1997.

[8] N. Hirokawa and A. Middeldorp. Tsukuba Termination Tool. In *Proc. 14th RTA, LNCS 2706*, pp. 311–320, 2003.