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 [2, 4] 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．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 [5, 15] ．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，提案されているさまざまな判定方法 [6, 7, 8, 9, 10, 11, 13, 16, 18, 21, 22, 23, 24, 26, 27, 28, 29, 30] 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．

本論文は全 7 章で構成される．第 2 章では項書き換えシステムの基本的な概念について説明する．第 3 章では合流性基本判定条件を用いた合流性の自動判定について説明する．第 4 章では直和性，第 5 章では可換性を用いた項書き換えシステムの自動分解について説明する．第 6 章で，項書き換えシステムの自動分解と合流性の自動判定を組み合わせた合流性自動判定システムの手続きを示し，合流性自動判定の実験結果を報告する．第 7 章では本研究のまとめと今後の課題について検討する．

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 [2, 4] . 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 R 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．

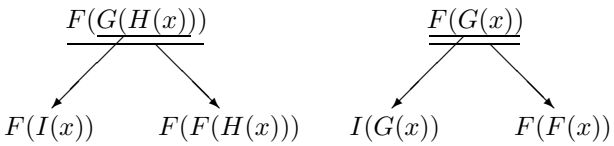
† 東北大学 電気通信研究所

‡ 本論文の一部は研究速報として文献 [31] で出版されている．

危険対 [2, 4] 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 R を考える．

$$R = \begin{cases} F(x) & \to & I(x) \\ G(H(x)) & \to & I(x) \\ F(G(x)) & \to & F(F(x)) \end{cases}$$

ここで，2 つの書き換え規則の左辺に重なりが存在するとき，左辺を重ね合わせて得られる最も一般的な項 $F(G(H(x)))$ や $F(G(x))$ からは，それぞれの書き換え規則によって以下の 2 通りの書き換えが可能である．



ここで，項 $F(G(H(x)))$ から得られた項の対 $\langle F(I(x)), F(F(H(x)))\rangle$ 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 [26]．一方，項 $F(G(x))$ から得られた項の対 $\langle I(G(x)), F(F(x))\rangle$ 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 [26]．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 [6, 7, 8, 9, 10, 11, 13, 16, 18, 21, 22, 23, 24, 26, 27, 28, 29, 30]．本研究では，以下の判定条件を用いて合流性の自動判定を行う．

定理 1 (Knuth-Bendix の合流性判定条件 [18]) 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 の各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反例が得られる．

以下の例では，定理 1 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．

$$R = \begin{cases} W(B(x)) & \to & W(x) \\ B(I(x)) & \to & J(x) \\ W(I(x)) & \to & W(J(x)) \end{cases}$$

R は停止性をみたしており，ただ 1 つの危険対 $\langle W(J(x)), W(I(x))\rangle$ の要素は同じ正規形をもつので，定理 1 より R は合流性をみたす．

例 2 R を以下の項書き換えシステムとする．

$$R = \begin{cases} W(B(x)) & \to & I(x) \\ B(S(x)) & \to & S(x) \\ W(x) & \to & I(x) \end{cases}$$

R は停止性をみたしているが，1 番目と 3 番目の規則から得られる危険対 $\langle I(x), I(B(x))\rangle$ の要素が同じ正規形をもたないので，定理 1 より R の非合流性が示される．

項書き換えシステム R の並行書き換え $s \multimap_R t$ とは，項 s の中の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 [21, 30]．ここで，独立して書き換え可能とは，対応する書き換え規則の左辺の関数記号が重ならずに出現していることである．たとえば，次の項書き換えシステムを考える．

$$R = \begin{cases} F(x) & \to & G(x,x) \\ I(x) & \to & J(x) \end{cases}$$

このとき，項 $F(I(A))$ で書き換え可能な部分項 $F(I(A))$ と $I(A)$ は，対応する書き換え規則の左辺 $F(x)$ と $I(x)$ の関数記号 F と I が項 $F(I(A))$ 上で重ならずに出現しているので，独立して書き換え可能である．そこで，両者を同時に書き換えると $F(I(A)) \multimap G(J(A), J(A))$ が得られる．

定理 2 (Oostrom の合流性条件 [30]) 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．

1. $\forall\langle c_1, c_2\rangle \in CP_{out}(R), \exists s, c_1 \multimap s \overset{*}{\leftarrow} c_2$
2. $\forall\langle c_1, c_2\rangle \in CP_{in}(R), c_1 \multimap c_2$

なお，条件 1. の $s \overset{*}{\leftarrow} c_2$ は決定不能な条件であるため，本論文では決定可能性を保証するために，$s \overset{*}{\leftarrow} c_2$ を決定可能な条件 $s \leftarrow\!\!\circ\!\!- c_2$ に弱めたものを Oostrom の合流性条件として用いる．この場合，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にすぎないため，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．

以下の例では，定理 2 を用いて合流性を判定する．

例 3 R を以下の項書き換えシステムとする．

$$R = \begin{cases} F(H(x), y) & \to & F(H(x), I(I(y))) \\ F(x, G(y)) & \to & F(I(x), G(y)) \\ I(x) & \to & x \end{cases}$$

R は停止性をみたさないため Knuth-Bendix 条件は適用できないが，R は左線形であり，1 番目と 2 番目の書き換え規則から得られる外側危険対 $\langle F(H(x), I(I(G(y)))), F(I(H(x)), G(y))\rangle$ の要素に対してそれぞれ 1 回の並行書き換えを行うと合流するので，定理 2 より合流性が示される．

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 R 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
```

なお，項書き換えシステムの停止性は一般には決定不能であり [2, 4]**，本アルゴリズムでは辞書式経路順序に基づく停止性の十分条件の判定のみを行っている．つまり，**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** [14] **の方法で行う．さらに，**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** [1, 12, 19, 20, 25, 26, 28]**．**

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 [25]**)** R_1, R_2 **を互いに素な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 4 R **を以下の項書き換えシステムとする．**

$$R = \begin{cases} F(x, A(G(x))) & \to & G(F(x,x)) \\ F(x, G(x)) & \to & G(F(x,x)) \\ A(x) & \to & x \\ H(x) & \to & H(B(H(x))) \end{cases}$$

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．**

$$R_1 = \begin{cases} F(x, A(G(x))) & \to & G(F(x,x)) \\ F(x, G(x)) & \to & G(F(x,x)) \\ A(x) & \to & x \end{cases}$$

$$R_2 = \begin{cases} H(x) & \to & H(B(H(x))) \end{cases}$$

このとき，R_1 **は停止性をみたすので** Knuth-Bendix **条件，**R_2 **は左線形なので** 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．**

例 5 R **を以下の項書き換えシステムとする．**

$$R = \begin{cases} I(x) & \to & I(B(x)) \\ F(E(x), x) & \to & G(x) \\ E(x) & \to & x \end{cases}$$

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．

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

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

このとき，R_1 に対しては Oostrom 条件より合流性が示されるが，R_2 に対しては Knuth-Bendix 条件より非合流性が示されるので，定理 3 より R の非合流性が示される．

項書き換えシステム R を $R = R_1 \oplus R_2 \oplus \cdots \oplus R_n (n \geq 2)$ 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び，特にどの成分もこれ以上直和分解できないとき，最小の直和分解成分とよぶ．ここで，最小の直和分解は一意に得られることに注意する．また，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分割統治法に基づく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 [26]．

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，$t_1 \stackrel{*}{\leftarrow}_{R_1} t \stackrel{*}{\rightarrow}_{R_2} t_2$ ならば $t_1 \stackrel{*}{\rightarrow}_{R_2} s \stackrel{*}{\leftarrow}_{R_1} t_2$ 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 [26]) 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 6 R を以下の項書き換えシステムとする．

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

ここで，R には 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．しかし，後ほど説明するように R は以下のように可換分解可能である．

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

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

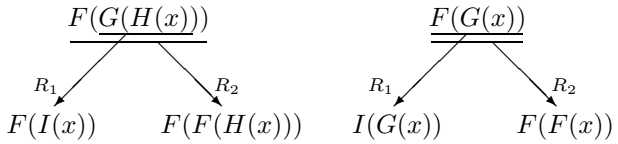
このとき，R_1 に対しては Knuth-Bendix 条件，R_2 に対しては Oostrom 条件が適用できるので，定理 4 より R の合流性が示される．

項書き換えシステム R_1 と R_2 の間の危険対は，R_1 と R_2 の可換性判定に重要である．たとえば，次の 2 つの項書き換えシステムを考える．

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

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

このとき，R_1 の書き換え規則の左辺と R_2 の書き換え規則の左辺の重なりから得られる項 $F(G(H(x)))$ や $F(G(x))$ からは，R_1 と R_2 の書き換え規則によって以下の 2 通りの書き換えが可能である．



ここで，項 $F(G(H(x)))$ から得られた項の対 $\langle F(I(x)), F(F(H(x)))\rangle$ のように，R_1 で真部分項，R_2 で項全体をそれぞれ書き換えて得られた項の対を R_1 と R_2 の間の内側危険対という [26]．一方，項 $F(G(x))$ から得られた項の対 $\langle I(G(x)), F(F(x))\rangle$ のように，R_1 と R_2 でそれぞれ項全体を書き換えて得られた項の対を R_1 と R_2 の間の外側危険対という [26]．R_1 と R_2 の間の内側危険対の集合を $CP_{in}(R_1, R_2)$，R_1 と R_2 の間の外側危険対の集合を $CP_{out}(R_1, R_2)$ と記す．R_1 と R_2 の間の危険対の集合を $CP(R_1, R_2) = CP_{in}(R_1, R_2) \cup CP_{out}(R_1, R_2)$ とする．このとき，文献 [26] の可換性条件を拡張した以下の定理が成立する．

定理 5 以下の条件をみたす左線形項書き換えシステム R_1, R_2 は可換である．

1. $\forall\langle c_1, c_2\rangle \in CP(R_1, R_2), \exists s, c_1 \multimap\!\!\rightarrow_{R_2} s \overset{*}{\leftarrow}_{R_1} c_2$
2. $\forall\langle c_2, c_1\rangle \in CP_{in}(R_2, R_1), c_2 \multimap\!\!\rightarrow_{R_1} c_1$

(略証) 文献 [26] の定理 3.1 の並列書き換え $\multimap\!\!\!\!\!\!\rightarrow$ を並行書き換え $\multimap\!\!\rightarrow$ に置き換えると，文献 [30] の定理 3.3, 定理 3.5 の証明と同様な方法で証明できる．□

なお，本論文では決定可能性を保証するために，条件 1. の $s \overset{*}{\leftarrow}_{R_1} c_2$ を Oostrom の合流性条件と同様に決定可能な条件 $s \leftarrow\!\!\circ\!\!-_{R_1} c_2$ に弱めた可換性条件を用いる．

例 7 R_1, R_2 を以下の左線形項書き換えシステムとする．

$$R_1 = \left\{ \begin{array}{lcl} I(x) & \rightarrow & I(J(x)) \\ J(x) & \rightarrow & J(K(J(x))) \end{array} \right.$$

$$R_2 = \left\{ \begin{array}{lcl} H(I(x)) & \rightarrow & K(J(x)) \\ J(x) & \rightarrow & K(J(x)) \end{array} \right.$$

ここで，R_1 と R_2 の間の危険対 $\langle H(I(J(x))), K(J(x))\rangle$，$\langle J(K(J(x))), K(J(x))\rangle$ は定理 5 の可換性条件をみたしており，R_1 と R_2 は可換となる．

我々の先行研究 [31] では，左線形項書き換えシステム R_1 と R_2 の間に危険対が存在しないという可換性条件を用いて可換分解を試みた．しかし，この可換性条件は R_1 と R_2 の間に危険対が存在する上記の例には適用できない．本論文の定理 5 で示した可換性条件は，先行研究 [31] の可換性条件を含んでおり，従来知られている可換性条件 [26] よりも強力である．実際，例 7 に文献 [26] の可換性条件は適用することができない．

また，直和である 2 つの項書き換えシステムが必ずしも可換であるとは限らないことに注意する．

例 8 R_1, R_2 を以下の項書き換えシステムとする．

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

$$R_2 = \left\{ \begin{array}{lcl} A & \rightarrow & B \end{array} \right.$$

R_1 と R_2 は直和であるが，可換ではない．なぜなら，$G(A) \leftarrow_{R_1} F(A,A) \rightarrow_{R_2} F(B,A)$ となるが，$G(A) \rightarrow_{R_1} s \leftarrow_{R_2} F(B,A)$ となる項 s は存在しないからである．実際，$G(A) \rightarrow_{R_2} G(B) \leftarrow_{R_1} F(B,B) \leftarrow_{R_2} F(B,A)$ とはなるが，$G(B) \overset{*}{\leftarrow}_{R_1} F(B,A)$ とはならない．

項書き換えシステム R を $R = R_1 \sqcup R_2 \sqcup \cdots \sqcup R_n (n \geq 2)$ 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び，特にどの成分もこれ以上可換分解できないとき，極小の可換分解成分とよぶ．ここで，極小の可換分解成分は直和分解の場合と異なり一意には求められない．

例 9 R を以下の左線形項書き換えシステムとする．

$$R = \left\{ \begin{array}{lcl} I(x) & \rightarrow & I(J(x)) \\ J(x) & \rightarrow & J(K(J(x))) \\ H(I(x)) & \rightarrow & K(J(x)) \\ J(x) & \rightarrow & K(J(x)) \end{array} \right.$$

R の可換分解を試みると，$R = R_{11} \sqcup R_{12}$，$R = R_{21} \sqcup R_{22} \sqcup R_{23}$ という２通りの極小の可換分解成分を得る．

$$R_{11} = \left\{ \begin{array}{lcl} I(x) & \to & I(J(x)) \\ J(x) & \to & J(K(J(x))) \end{array} \right.$$

$$R_{12} = \left\{ \begin{array}{lcl} H(I(x)) & \to & K(J(x)) \\ J(x) & \to & K(J(x)) \end{array} \right.$$

ここで，例 7 より R_{11} と R_{12} は可換である．

$$R_{21} = \left\{ \begin{array}{lcl} I(x) & \to & I(J(x)) \\ H(I(x)) & \to & K(J(x)) \end{array} \right.$$

$$R_{22} = \left\{ \begin{array}{lcl} J(x) & \to & J(K(J(x))) \end{array} \right.$$

$$R_{23} = \left\{ \begin{array}{lcl} J(x) & \to & K(J(x)) \end{array} \right.$$

ここで，R_{21} と R_{22}，R_{21} と R_{23} はそれぞれ互いの間に危険対が存在しないので可換である．また，R_{22} と R_{23} の間の危険対 $\langle J(K(J(x))), K(J(x)) \rangle$ は定理 5 の可換性条件をみたしており，R_{22} と R_{23} は可換である．

定理 5 に基づく極小の可換分解成分を全て求めるアルゴリズムを以下に示す．

```
fun com_decompose rs =
    let fun com_decompose1 rssss =
            ⋃ { ⋃ { com_decompose2 ⟨rs',rss\{rs'}⟩ | rs' ∈ rss } | rss ∈ rssss }
        and com_decompose2 ⟨rs',rss'⟩ =
            let val rsps = { ⟨rs1,rs2⟩ | rs1 ⊎ rs2 = rs',
                                         rs1, rs2 ≠ ∅,
                                         allcomcheck ({rs1,rs2} ∪ rss') }
            in if rsps = {} then {{rs'} ∪ rss'}
               else com_decompose1 {{rs1,rs2} ∪ rss' | ⟨rs1,rs2⟩ ∈ rsps }
    in com_decompose1 {{rs}}
```

ここで，allcomcheck rss は rss の異なるどの 2 つの要素も定理 5 をみたすとき真となる．

可換分解の場合には，直和分解の場合と異なり，分解成分が小さいほど合流性判定が容易になるとは限らない．たとえば，上記で求めた極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 10 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．

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

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

このとき，$F(I(x), y) \leftarrow F(H(I(x)), y) \rightarrow G(H(I(x)))$ 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，極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．

この問題を解決するために，我々のアルゴリズムでは，全ての極小の可換分解成分の集合から始めて，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ことにより，定理 5 に適用できる可換分解が存在するならば必ず解を得ることができる．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 (com_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．** $\mathrm{rsss1} \subseteq \{\{R_1, \ldots, R_n\} | R = R_1 \sqcup \cdots \sqcup R_n\}$ **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 (約** 1300 **行)．**

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．なお，合流と判定されなかった可換分解成分に対して再び直和分解を試みることも可能ではあるが，今回は実行していない．**

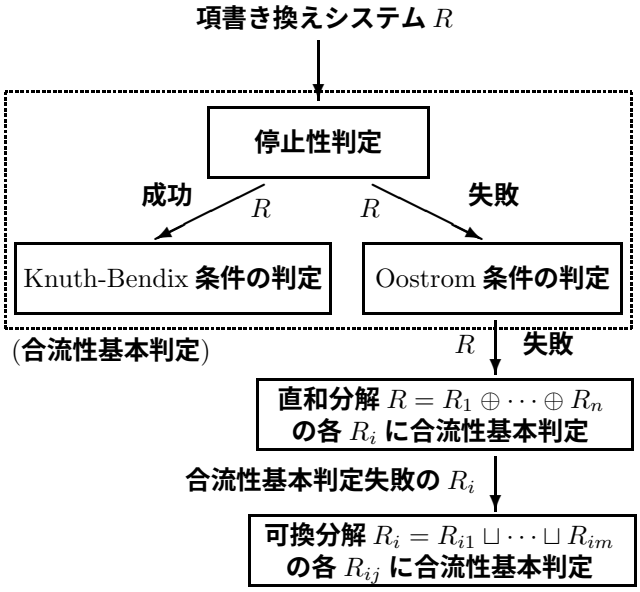


図 1: 合流性自動判定システムの構成

例 11 以下の項書き換えシステム R に対する合流性自動判定システムの実行例 (図 2) を示す．

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

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 $R = R_1 \oplus (R_2 \sqcup R_3) \oplus R_4$ に分解し，合流性判定に成功する．ここで，可換分解 $R_2 \sqcup R_3$ には定理 5 が用いられていることに注意する．文献 [26] の可換分解アルゴリズムでは，このような可換分解は得られない．

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

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

$$R_3 = \left\{ \begin{array}{lcl} I(x) & \to & I(J(x)) \\ J(x) & \to & J(K(J(x))) \end{array} \right.$$

$$R_4 = \left\{ \begin{array}{lcl} S(x, T(x)) & \to & T(x) \end{array} \right.$$

（入力）

```
djcom_crcheck
( IO.rdrules [
 "F(D(x),y) -> F(D(x),G(G(y)))",
 "F(x,E(y)) -> F(G(x),E(y))",
 "G(x) -> x",
 "H(I(x)) -> K(J(x))",
 "J(x) -> K(J(x))",
 "I(x) -> I(J(x))",
 "J(x) -> J(K(J(x)))",
 "S(x,T(x)) -> T(x)"
 ] );
```

（出力）

```
(disjoint)[E,G,D,F]
[ F(D(x), y) -> F(D(x), G(G(y))),
  F(x, E(y)) -> F(G(x), E(y)),
  G(x) -> x ]
'Unknown-Terminating' and 'Left-Linear' and 'Oostrom' :CR

(disjoint)[T,S]
[ S(x, T(x)) -> T(x) ]
'Terminating' and 'WCR' :CR

(disjoint)[J,K,H,I]
(Com)
[ H(I(x)) -> K(J(x)),
  J(x) -> K(J(x)) ]
'Unknown-Terminating' and 'Left-Linear' and 'Oostrom' :CR
(Com)
[ I(x) -> I(J(x)),
  J(x) -> J(K(J(x))) ]
'Unknown-Terminating' and 'Left-Linear' and 'Oostrom' :CR
djResult: CR

Result: CR
val it = true : bool
Time: 12msec.
```

図 2: **合流性自動判定システムの実行例**

項書き換えシステムの合流性に関連する論文より抜粋したさまざまな例に対して，本システムを用いた合流性自動判定を行った (表 1)．68 例に対して実験を行い，20 例の合流性判定，5 例の非合流性判定に成功した．43 例については判定に失敗した．1 例を除いては 1sec. 以内に結果が得られた．なお，4 例については実行時間が 100msec. 以上であるが，これは可換分解に要した時間が大きかったことが原因であると考えられる．

判定に失敗した 43 例の中には．実装が比較的容易と予想される合流性条件を組み込んだり，停止性判定法を強力にすることにより判定が可能になる例も多い．例えば，[27] 例 2 の以下の項書き

表 1: 合流性自動判定システムの実験結果

例	判定	実行時間	例	判定	実行時間
CL	CR	0	[16] 813 ページ 定義	Can't judge	3
加算 + AC	Can't judge	7	[16] 813 ページ 定義	Can't judge	1
加算 + C	Can't judge	2	[16] 814 ページ 例	CR	113
群論	CR	12	[16] 814 ページ 反例	Can't judge	231
群論 + C	Can't judge	7	[16] 816 ページ	not CR	4
[1] 例 5	Can't judge	10	[17] CL+Dk	Can't judge	1
[3] 204 ページ	Can't judge	1	[17] CL+Ds	Can't judge	1
[3] 209 ページ	Can't judge	0	[17] CL+SP	Can't judge	1
[4] 287 ページ	Can't judge	0	[17] CL+Dh	Can't judge	1
[7] 例 1	Timeout	-	[17] CL+Dh(,)	CR	0
[8] 例 5	CR	2	[17] CL+B	Can't judge	1
[9] 例	Can't judge	1	[17] CL+B(,,)	CR	2
[10] 例 1	Can't judge	0	[19] 例 4.2.2	not CR	0
[11] 例 2	Can't judge	26	[19] 例 4.4.3	CR	0
[11] 例 3	not CR	3	[20] 例 4.4	CR	0
[11] 例 4	not CR	3	[20] 例 5.12	Can't judge	3
[11] 例 6	CR	44	[21] 例 1	Can't judge	3
[11] 例 7	Can't judge	17	[21] 14 ページ R_1	CR	18
[11] 例 8	Can't judge	8	[21] 14 ページ R_2	Can't judge	37
[11] 例 9	Can't judge	1	[21] 14 ページ R_3	CR	0
[12] 例 2.2.28	Can't judge	1	[22] 例 1	CR	375
[12] 28 ページ	Can't judge	2	[22] 例 2	Can't judge	21
[12] 例 3.3.1	Can't judge	1	[23] 例 1	Can't judge	1
[12] 例 3.3.2	Can't judge	283	[23] 例 2	Can't judge	3
[12] 例 3.4.23	CR	2	[23] 例 3	Can't judge	1
[12] 例 3.5.7	CR	7	[24] 例 (abstract)	CR	0
[13] 例 1	Can't judge	13	[24] 例 4.2	CR	2
[13] 例 2	Can't judge	27	[24] 例 5.1	CR	0
[13] 例 3	Can't judge	15	[26] 例 3.3	CR	0
[13] 例 5	Can't judge	19	[27] 例 1	Can't judge	1
[13] 例 6	CR	0	[27] 例 2	Can't judge	0
[16] 811 ページ 例	Can't judge	3	[27] 例 3	Can't judge	1
[16] 811 ページ 例 (a)	not CR	2	[29] 例 5	Can't judge	0
[16] 811 ページ 例 (b)	CR	1	[29] 例 6	Can't judge	0

実行時間単位 (msec.)

換えシステム R は，本システムの辞書式経路順序では停止性が判定できないが，より強力な停止性判定法を用いれば，Knuth-Bendix 条件による合流性判定が可能である．

$$R = \left\{ \begin{array}{lcl} F(F(x)) & \rightarrow & F(G(F(x))) \end{array} \right.$$

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，非常に基本的な Knuth-Bendix 条件と Oostrom 条件のみを用いているにもかかわらず，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, 7, 8, 23, 28] 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性条件 [6, 7, 8, 9, 10, 11, 13, 21, 22, 23, 24, 27, 28, 29] 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．また，本システムでは停止性自動判定を辞書式経路順序に基づく文献 [14] の方法で行ったが，より強力な既存の停止性自動判定システム [5, 15] を利用できるようになれば，さらに強力な合流性自動判定が期待できる．

謝辞

有益なコメントをお寄せ下さった査読者に深く感謝致します．なお，本研究は一部日本学術振興会科学研究費 17700002,19500003 の補助を受けて行われた．

参考文献

[1] T. Aoto and Y. Toyama. On composable properties of term rewriting systems. In *Proceedings of the 6th International Joint Conference ALP'97 – HOA'97*, Vol. 1298 of *LNCS*, pp. 114–128. Springer-Verlag, 1997.

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

[3] N. Dershowitz. Innocuous constructor-sharing combinations. In *Proceedings of the 8th International Conference on Rewriting Technique and Applications (RTA-97)*, Vol. 1232 of *LNCS*, pp. 202–216. Springer-Verlag, 1997.

[4] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, Vol. B, pp. 243–320. North-Holland, 1990.

[5] 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.

[6] G. Godoy and A. Tiwari. Confluence of shallow right-linear rewrite systems. In *Proceedings of the 19th Annual Conference of the European Association for Computer Science Logic (CSL'05)*, Vol. 3634 of *LNCS*, pp. 541–556. Springer-Verlag, 2005.

[7] G. Godoy, A. Tiwari, and R. M. Verma. On the confluence of linear shallow term rewrite systems. In *Proceedings of the24th International Symposium on Theoretical Aspects of Computer Science (STACS 2003)*, Vol. 2607 of *LNCS*, pp. 85–96. Springer-Verlag, 2003.

[8] G. Godoy, A. Tiwari, and R. M. Verma. Confluence by rewrite closure and right ground term rewrite systems. *Applicable Algebra in Engineering, Communication and Computing*, Vol. 15, No. 1, pp. 13–36, 2004.

[9] H. Gomi, M. Oyamaguchi, and Y. Ohta. On the Church-Rosser property of non-e-overlapping and strongly depth-preserving term rewriting systems. *Transactions of IPSJ*, Vol. 37, No. 12, pp. 2147–2160, 1996.

[10] H. Gomi, M. Oyamaguchi, and Y. Ohta. On the Church-Rosser property of non-E-overlapping and strongly depth-preserving term rewriting systems. *Transactions of IPSJ*, Vol. 39, No. 4, pp. 992–1005, 1998.

[11] B. Gramlich. Confluence without termination via parallel critical pairs. In *Proceedings of the 21st colloquium on Trees in Algebra and Programming (CAAP'96)*, Vol. 1059 of *LNCS*, pp. 211–225. Springer-Verlag, 1996.

[12] B. Gramlich. *Termination and Confluence Properties of Structured Rewrite Systems*. PhD thesis, Universität Kaiserslautern, 1996.

[13] B. Gramlich and S. Lucas. Generalizing Newman's lemma for left-linear rewrite systems. In *Proceedings of the 17th International Conference on Rewriting Technique and Applications (RTA 2006)*, Vol. 4098 of *LNCS*, pp. 187–201. Springer-Verlag, 2006.

[14] N. Hirokawa and A. Middeldorp. Tsukuba termination tool. In *Proceedings of the 14th International Conference on Rewriting Techniques and Applications*, Vol. 2706 of *LNCS*, pp. 311–320. Springer-Verlag, 2003.

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

[16] G. Huet. Confluent reductions: abstract properties and applications to term rewriting systems. *Journal of the Association for Computing Machinery*, Vol. 27, No. 4, pp. 797–821, 1980.

[17] J. W. Klop. *Combinatory Reduction Systems*, Vol. 127 of *Mathematical Centre Tracts*. CWI, Amsterdom, Holland, 1980.

[18] D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational problems in abstract algebra*, pp. 263–297. Pergamon Press, 1970.

[19] E. Ohlebusch. *Modular properties of composable term rewriting systems*. PhD thesis, Universität Bielefeld, 1994.

[20] E. Ohlebusch. On the modularity of confluence of constructor-sharing term rewriting systems. In *Proceedings of the 19th colloquium on Trees in Algebra and Programming (CAAP'94)*, Vol. 787 of *LNCS*, pp. 261–275. Springer-Verlag, 1994.

[21] S. Okui. Simultaneous critical pairs and Church-Rosser property. In *Proceedings of the 9th International Conference on Rewriting Technique and Applications (RTA-98)*, Vol. 1379 of *LNCS*, pp. 2–16. Springer-Verlag, 1998.

[22] M. Oyamaguchi and Y. Ohta. A new parallel closed condition for Church-Rosser of left-linear TRS's. In *Proceedings of the 8th International Conference on Rewriting Technique and Applications (RTA-97)*, Vol. 1232 of *LNCS*, pp. 187–201. Springer-Verlag, 1997.

[23] A. Tiwari. Deciding confluence of certain term rewriting systems in polynomial time. In *Proceedings of the 17th IEEE Symposium on Logic in Computer Science (LICS 2002)*, pp. 447–458. IEEE Computer Society Press, 2002.

[24] Y. Toyama. On the Church-Rosser property of term rewriting systems. Technical Report 17672, NTT ECL, 1981. *In Japanese.*

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

[26] 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, 1988.

[27] Y. Toyama. Labeling technique for term rewriting systems. In *LA Symposium 98-7*, 1998. *In Japanese.*

[28] Y. Toyama. Confluent term rewriting systems (invited talk). In *Proceedings of the 16th International Conference on Rewriting Techniques and Applications*, Vol. 3467 of *LNCS*, p. 1. Springer-Verlag, 2005. Slides are available from `http://www.nue.riec.tohoku.ac.jp/user/toyama/slides/toyama-RTA05.pdf`.

[29] Y. Toyama and M. Oyamaguchi. Church-Rosser property and unique normal form property of non-duplicting term rewriting systems. In *Proceedings of the 4th International Workshop on Conditional (and Typed) Rewriting Systems (CTRS-94)*, Vol. 968 of *LNCS*, pp. 316–331. Springer-Verlag, 1994.

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

[31] **吉田順一**, **青戸等人**, **外山芳人**. **項書き換えシステムの合流性自動判定**. **情報技術レターズ** (FIT 2007), Vol. 6, pp. 31–34, 2007.