A-034

基底項書き換え系の合流性自動判定
Automating Confluence Check of Ground Term Rewriting Systems

村井 正勝 † 青戸 等人 † 外山 芳人 †
Masakatsu Murai Takahito Aoto Yoshihito Toyama

1. はじめに

合流性は，項書き換え系の重要な性質であるが，一般的には決定不能であることが知られている．しかし，いくつかのクラスに対しては合流性判定が決定可能である [1, 2, 3]．また，基底項書き換え系や線形シャロー項書き換え系の合流性判定を多項式時間で実行するアルゴリズムも提案されている [2, 3]．これらのアルゴリズムに共通しているのは，項書き換え系の書き換え規則をフラットな形に変換して適当な閉包操作を行う点である．しかし，フラット変換は書き換え規則数を大幅に増やすため効率的な合流性条件の検証を困難にする場合がある．

本研究では，基底項書き換え系に焦点を絞り，フラット変換を改良した効率的な合流性判定アルゴリズムを提案し，実験結果を報告する．

2. 項書き換え系

項書き換え系は書き換え規則の有限集合である．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系で与えられる．

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

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，$S(0)$ のように変数を含まない項を基底項といい，基底項のみで構成される書き換え系を基底項書き換え系という．

項書き換え系 R で s から t に 0 回以上の書き換えで到達できるとき $s \to_R^* t$ （R が自明のとき，$s \to^* t$ と略記）と記す．任意の項 t, t_1, t_2 について，$t_1 \stackrel{*}{\leftarrow} t \stackrel{*}{\to} t_2$ ならば，ある項 s が存在して $t_1 \stackrel{*}{\to} s \stackrel{*}{\leftarrow} t_2$ となるとき，項書き換え系 R は合流性をもつという．

3. 基底項書き換え系の合流性判定

基底項書きえ系の合流性判定アルゴリズム [3] は，基底項書き換え系 R を入力し，合流または非合流を判定する (図 1)．このアルゴリズムの特徴は，3 ステップの変換をしてから合流性条件の検証を行うことである．合流性条件の検証時間は R^p の書き換え規則数によって大きく影響を受ける．R の書き換え規則数を $|R|$ で表すと，図 1 の変換過程での書き換え規則数は一般に $|R| = |R^c| \ll |R^f| \leq |R^p|$ となる．ここでは，フラット変換を改良することで，$|R^f|$ を小さくし，効率的な合流性判定アルゴリズムへの改良を試みる．

4. フラット変換

フラット変換とは，項書き換え系の各書き換え規則の深さを 2 以下に制限する変換のことで，以下のように定義される．ここで，項 u が真部分項 s をもつとき $u[s]$ と記す．

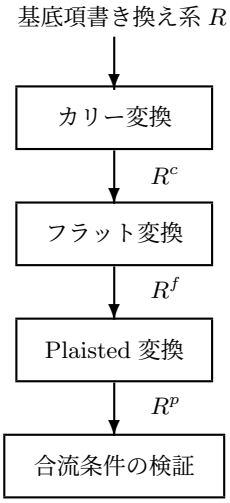

図 1: 基底項書き換え系の合流性判定

定義 1 (フラット変換 [3])

$$\frac{u[s_0] \to t}{u[C] \to t, C \leftrightarrow s_0} \qquad \frac{s \to u[t_0]}{s \to u[C], C \leftrightarrow t_0}$$

上記左側の変換では，項 $u[s_0]$ の真部分項 s_0 が深さ 2 以上のとき， 新しい定数記号 C を導入して $s_0 \to C$ と $C \to s_0$ の 2 つの書き換え規則 ($s_0 \leftrightarrow C$ と略記) を追加し，さらに $u[s_0] \to t$ を $u[C] \to t$ に置き換える．(上記右側の変換についても同様．) フラット変換は，項書き換え系の全ての書き換え規則の深さが 2 以下になるまで行う．なお，フラット変換を行うと書き換え規則が増加するため，後の合流性条件の検証を効率的に行うことが難しくなる．

例 1 (フラット変換) ここではカリー変換実行後の以下の R^c について考える．

$$R^c = \left\{ \begin{array}{l} F(F(F(P, A), B), C) \to F(F(F(Q, A), B), C) \\ F(R, F(F(F(P, A), B), C)) \to \\ \qquad F(F(F(P, F(R, A)), B), C) \end{array} \right.$$

R^c をフラット変換すると以下の R^f が得られる．

† 東北大学 電気通信研究所

$$R^f = \begin{cases} F(R,C_3) \to F(C_4,C) \\ F(C_1,C) \to F(C_2,C) \\ F(P,C_8) \leftrightarrow C_5 \\ F(R,A) \leftrightarrow C_8 \\ F(C_7,B) \leftrightarrow C_1 \\ F(P,A) \leftrightarrow C_7 \\ F(C_6,B) \leftrightarrow C_2 \\ F(Q,A) \leftrightarrow C_6 \\ F(C_1,C) \leftrightarrow C_3 \\ F(C_5,B) \leftrightarrow C_4 \end{cases}$$

フラット変換前と変換後の項書き換え系 R^c,R^f を比較すると，$|R^c| = 2$，$|R^f| = 18$ となり，大幅にサイズが増加している．

合流性判定アルゴリズムでは，R^f から Plaisted 変換で R^p を求め，合流条件の検証を行う．R^p は以下の項書き換え系となる．
$R^p = \{s \to t | s,t$ は $s \to^*_{R^f} t$ をみたすR^f の両辺の部分項$\}$
このとき，Plaisted 変換でフラット変換が本質的に必要とされるのは，R^f の深さ２以上の左辺がそのまま出現している真部分項のみである．それ以外の部分項については，フラット変換を行わなくても，Plaisted 変換後の合流条件の検証は同様に可能である．さらに，フラット変換を行う回数が少なくなると，$|R^f|$ が小さくなるため，$|R^p|$ も小さくなる．

以上の考察から，書き換え規則の深さ２以上の真部分項が R^c の左辺の場合のみフラット変換を行う以下の改良フラット変換が考えられる．

定義 2 (改良フラット変換)

$$\frac{u[s_0] \to t}{u[C] \to t, C \leftrightarrow s_0} \qquad \frac{s \to u[t_0]}{s \to u[C], C \leftrightarrow t_0}$$

ここで，s_0, t_0 は u の真部分項として出現している深さ２以上の R^c の左辺の項

例 2 (改良フラット変換) 例１と同じ R^c に対して改良フラット変換を行うと，以下のような $R^{f'}$ が得られる．

$$R^{f'} = \begin{cases} F(F(F(P,A),B),C) \to F(F(F(Q,A),B),C) \\ F(R,C_1) \to F(F(F(P,F(R,A)),B),C) \\ F(F(F(P,A),B),C) \leftrightarrow C_1 \end{cases}$$

例１と比較すると $|R^f| = 18$，$|R^{p'}| = 4$ となり，サイズの増加は改良フラット変換の方が小さい．さらに，$R^f, R^{f'}$ から Plaisted 変換で R^p,$R^{p'}$ を求めると $|R^p| = 44$，$|R^{p'}| = 23$ となり，合流性条件の検証をするための書き換え規則数も改良フラット変換の方が少ない．

なお，合流性判定アルゴリズムのフラット変換を改良フラット変換に変更した場合，合流性条件の検証もそれにあわせて修正が必要である．しかし，この修正による実行時間の増加はほとんどない．

例	Flat	改良 Flat	判定結果
$R10$	216 [31]	216 [19]	合流
$R11$	16 [30]	20 [18]	合流
$R12$	2676 [83]	964 [39]	合流
$R13$	24 [83]	24 [38]	非合流
$R14$	24 [50]	20 [25]	非合流
$R15$	2940 [85]	984 [41]	合流
$R16$	11913 [131]	4976 [66]	合流
$R17$	19057 [171]	6437 [76]	合流
$R18$	1168 [161]	384 [64]	非合流

表 1: 合流性判定アルゴリズムの実行時間 (msec)

5. 合流性自動判定アルゴリズムの実装と実験

フラット変換と改良フラット変換に基づいた合流性判定アルゴリズム **Flat** と**改良 Flat** を SML/NJ で実装した．さらに，９例の基底項書き換え系に対して，実行時間と $|R^p|$ を比較した (表１)．なお，表中の 216 [31] 等は実行時間が 216msec で $|R^p| = 31$ を表す．

実験結果では，$|R^p|$ は **Flat** よりも**改良 Flat** の方が約 50 %小さくなっている．実行時間に関しては，**Flat** の実行時間が 1000msec 以上の場合に，**改良 Flat** による実行時間の短縮が著しい．これは，合流条件の検証において合流と判定するためには，$|R^p|$ から構成される有限個の反例候補をすべて調べる必要があり，その結果 $|R^p|$ の大小が実行時間に直接影響するためであると考えられる．一方，合流条件の検証で非合流と判定されるのは，反例が検出された場合であるので，すべての候補を調べる必要はない．したがって，非合流の判定時間に関しては，$|R^p|$ の大小が直接影響しない．

6. おわりに

本研究で，書き換え規則数の増加がフラット変換よりも少ない改良フラット変換を提案し，それを用いて効率的な基底項書き換え系の合流性判定アルゴリズムを実現した．

基底項書き換え系以外にも同様にフラット変換に基づく合流性判定アルゴリズムが存在する [1, 2]．これらに対しても，改良フラット変換を拡張することで，効率的なアルゴリズムを提案することは今後の課題である．また，フラット変換をまったく用いない基底項書き換え系の合流性判定アルゴリズムの可能性についても今後検討していく．

参考文献

[1] G. Godoy, A. Tiwari, *Confluence of Shallow Right-Linear Rewrite Systems. LNCS 3634*, pp.541-556, 2005.

[2] G. Godoy, A. Tiwari, R. Verma, Deciding confluence of certain term rewriting system in polynomial time, *Annual of Pure and Apllied Logic130(1-3)*, pp.33-59, 2004.

[3] H. Comon, G. Godoy, The Confluence of Ground Term Rewrite Systems is Decidable in Polynomial Time, *42th Annual IEEE symposium of Foundadation of computer science(FOCS)*,pp.298–307, 2001.