Confluence of Length Preserving String Rewriting Systems is Undecidable

Yi Wang, Masahiko Sakai, Naoki Nishida,
Toshiki Sakabe, Keiichiro Kusakari

Graduate School of Information Science, Nagoya University
{wangyi@trs.cm, sakai@, nishida@, sakabe@, kusakari@}is.nagoya-u.ac.jp

Abstract

This paper shows the undecidability of confluence for length preserving string rewriting systems. It is proven by reducing the Post’s correspondence problem (PCP), which is known to be undecidable, to confluence problem for length preserving string rewriting systems. More precisely, we designed a reduction algorithm having the property that the existence of a solution for a given instance of PCP coincides with the non-confluence of the string rewriting system obtained from the reduction algorithm.

Keywords Post’s Correspondence Problem

1 Introduction

String rewriting systems (SRSs) are said to be length preserving if the left-hand side and the right-hand side of each rule have the same length. Caron showed that its termination is an undecidable property[1]. This paper shows its confluence is also an undecidable property, although both of the reachability problem and confluence of given strings are easily known to be decidable.

Confluence is generally undecidable for term rewriting systems (TRSs) and for string rewriting systems. Hence several decidable classes on confluence have been studied: terminating TRSs[6], ground TRSs[9], linear shallow TRSs[3], shallow right-linear TRSs[4]. There are also results on undecidable classes on confluence: semi-constructor TRSs[7] and flat TRSs[5, 8].

In this paper, we show the undecidability of confluence for length preserving SRSs and prove it by reducing the Post’s correspondence problem (PCP), which is known to be undecidable, to confluence problem for length preserving string rewriting systems. More precisely, we designed a reduction algorithm having the property that the existence of a solution for a given instance of PCP coincides with the non-confluence of the SRS obtained from the reduction algorithm.

2 Preliminaries

Let $\Sigma$ be an alphabet. A string rewrite rule is a pair of strings $l, r \in \Sigma^*$, denoted by $l \rightarrow r$. A finite set of string rewrite rules is called a string rewriting system (SRS). An SRS $\mathcal{R}$ induces a rewrite step relation $\rightarrow_{\mathcal{R}}$ defined by $s \rightarrow_{\mathcal{R}} t$ if there are $u, v \in \Sigma^*$ and a rule $l \rightarrow r$ in $\mathcal{R}$ such that $s = ulv$ and $t = urv$. We use $\rightarrow^+_{\mathcal{R}}$ for the transitive closure of $\rightarrow_{\mathcal{R}}$ and $\rightarrow^*_{\mathcal{R}}$ for the
reflexive-transitive closure of $\rightarrow_{R}$. We say that strings $s_{1}$ and $s_{2}$ are joinable if $s_{1} \xrightarrow{R}s_{2}$ for some $s$, denoted by $s_{1} \rightarrow_{R}s_{2}$. A string $s$ is confluent if $s_{1} \rightarrow_{R}s_{2}$ for any $s_{1} \xrightarrow{R}s_{2}$. An SRS $R$ is confluent if all strings are confluent.

In this paper, the notation $|u|$ represents the length of string $u$. The notation $a \cdots a$ denotes the string that consists of $m$ symbols of $a$. We refer $\{r \rightarrow l \mid l \rightarrow r \in R\}$ by $R^{-1}$.

3 Length preserving SRSs and undecidability of their confluence

**Definition 3.1** An SRS $R$ is said to be length preserving if $|l| = |r|$ for every rule $l \rightarrow r$ in $R$.

Since rules are finite, symbols appearing in rules are also finite. Hence strings composed of $n$ such symbols are finite. Thus the decidability of the following problems for length preserving SRSs are trivially follows:

1. **Reachability problem** is a problem to decide $s \xrightarrow{R} t$ for given strings $s$ and $t$ and an SRS $R$.

2. **String-confluence problem** is a problem to decide confluence of $s$ for a given string $s$ and a SRS $R$.

Now we recall Post's correspondence problem, which is known to be undecidable.

**Definition 3.2** An instance of PCP is a set $P \subseteq A^{*} \times A^{*}$ of finite pairs of strings over an alphabet $A$ with at least two symbols. A solution of $P$ is a string $w$ such that

$$w = u_{1} \cdots u_{k} = v_{1} \cdots v_{k}$$

for some $(u_{i}, v_{i}) \in P$. The Post's correspondence problem (PCP) is a problem to decide whether such a solution exists or not.

**Example 3.3** $P = \{(aba, a), (aab, abab), (bb, babba)\}$ is an instance of PCP. $P$ has a solution $ababbaaababa$ with $(u_{1}, v_{1}) = (aba, a)$, $(u_{2}, v_{2}) = (bb, babba)$, $(u_{3}, v_{3}) = (aab, abab)$ and $(u_{4}, v_{4}) = (aba, a)$.

**Theorem 3.4** ([10]) PCP is undecidable.

As a preparation of the algorithm that transform an instance of PCP to an SRS, we introduce a kind of null symbol $-\cdot$ and an equal length representation of each pair in instances of PCP. Let $P = (u_{1}, v_{1}), \ldots , (u_{n}, v_{n})$ be an instance of PCP over $A$.

$$\overline{P} = \{(u, v \cdots \cdot m) \mid (u, v) \in P \text{ and } |u| - |v| = m \geq 0\}$$

$$\cup \{(u \cdots \cdot m, v) \mid (u, v) \in P \text{ and } |u| - |v| = m < 0\}$$

We use $\overline{A}$ for $A \cup \{-\}.$

**Example 3.5** For instance $P = \{(ab, a), (a, ba)\}$ of PCP over $\{a, b\}$, we have

$$\overline{P} = \{(ab, a-), (a-, ba)\}$$
We use symbols like $X_{a}^{b}$. For an easy handling of strings that consist of such symbols, we introduce a notation defined as follows:

$$(X_{1} \cdots X_{n})_{a_{1} \cdots a_{n}}^{b_{1} \cdots b_{n}} = X_{1}^{a_{1}} \cdots X_{n}^{a_{n}}_{b_{1} \cdots b_{n}}$$

For example $(eed)_{c_{123}, d_{123}}^{b_{123}}$ denotes $e_{c_{1}}^{b_{1}}, e_{c_{2}}^{b_{2}}, e_{c_{3}}^{b_{3}}$. Note that the length of the strings in the superscripts and subscripts are the same when we use this notation. We call $X$ the tag of $X_{a}^{b}$.

**Definition 3.6** Let $P$ be an instance of PCP over $A$. The SRS $\mathcal{R}_{P}$ over $\Sigma$ obtained from $P$ is defined as follows:

$$\Sigma = \{\Xi_{0}, \Xi_{1}, \Xi_{2}, \Psi_{0}, \Psi_{1}, \Psi_{2}\} \cup \Sigma_{c}$$

$$\Sigma_{c} = \{e_{x_{1}^{1}}^{x_{1}}| x_{i} \in A\}$$

$$\mathcal{R}_{P} = \Theta \cup \Theta^{-1} \cup \Phi$$

$$\Theta = \delta \cup \alpha_{1} \cup \beta_{1} \cup \alpha_{2} \cup \beta_{2} \cup \delta_{1}$$

$$\delta = \{c_{x_{1}^{1}}^{x_{1}} \rightarrow c_{x_{1}^{1} x_{2}^{1}}^{x_{1}}, c_{x_{2}^{1}}^{x_{1} x_{2}^{1}} \rightarrow c_{x_{2}^{1} x_{3}^{1}}^{x_{1}}| x_{j}, y_{j} \in A, z \in A\}$$

$$\alpha_{1} = \{c_{x_{1}^{1} x_{2}^{1}}^{x_{1}} \rightarrow c_{x_{1}^{1} x_{2}^{1} x_{3}^{1}}^{x_{1}}| (u, v) \in P\}$$

$$\beta_{1} = \{c_{x_{1}^{1} x_{2}^{1} x_{3}^{1}}^{x_{1}} \rightarrow (p_{x_{1}^{1}}^{x_{1}} \delta_{x_{1}^{1}}^{x_{1}})_{x_{1}^{1} x_{2}^{1} x_{3}^{1}}^{x_{1}}| (u, v) \in P, x_{j} \in A\}$$

$$\gamma_{1} = \{e_{x_{1}^{1}}^{x_{1}} \rightarrow \Xi_{0} p_{x_{1}^{1}}^{x_{1}}| x_{j} \in A\}$$

$$\alpha_{2} = \{c_{x_{1}^{1} x_{2}^{1}}^{x_{1}} \rightarrow e_{x_{1}^{1}}^{x_{1}}| x_{j}, y_{j} \in A\}$$

$$\beta_{2} = \{c_{x_{1}^{1} x_{2}^{1}}^{x_{1}} \rightarrow e_{x_{1}^{1} x_{2}^{1}}^{x_{1}}| x_{j}, y_{j} \in A\}$$

$$\gamma_{2} = \{e_{x_{1}^{1}}^{x_{1}} \rightarrow \Xi_{0} e_{x_{1}^{1}}^{x_{1}}| x_{j} \in A\}$$

**Example 3.7** Let $P = \{(a, ba), (ab, a)\}$ be an instance of PCP, where $P$ has a solution $aba$. In $\mathcal{R}_{P}$, rules in $\alpha_{1}$ and $\beta_{1}$ depend on $P$ and the other rules depend only on the signature $A$.

$$\alpha_{1} = \{c_{x_{1}^{1}}^{x_{1}} \rightarrow c_{x_{1}^{1}}^{x_{1}} \Psi_{0} \rightarrow p_{x_{1}^{1}}^{x_{1}} \tilde{c}_{x_{1}^{1}}^{x_{1}} \Psi_{1}, c_{x_{2}^{1}}^{x_{1}} \rightarrow c_{x_{2}^{1}}^{x_{1}} \Psi_{0} \rightarrow p_{x_{2}^{1}}^{x_{1}} \tilde{c}_{x_{2}^{1}}^{x_{1}} \Psi_{1}\}$$

$$\beta_{1} = \{c_{x_{1}^{1} x_{2}^{1}}^{x_{1}} \rightarrow c_{x_{1}^{1} x_{2}^{1} x_{3}^{1}}^{x_{1}}, c_{x_{2}^{1} x_{3}^{1}}^{x_{1} x_{2}^{1} x_{3}^{1}} \rightarrow p_{x_{2}^{1} x_{3}^{1}}^{x_{1} x_{2}^{1} x_{3}^{1}} \tilde{c}_{x_{2}^{1} x_{3}^{1}}^{x_{1}} \Psi_{1}| x_{i} \in A\}$$

$\mathcal{R}_{P}$ is not confluent since we have the following reduction sequences:

$$\lambda_{0}c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \rightarrow \lambda_{0}c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{0} \rightarrow \lambda_{1} \Psi_{0} c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{1} \rightarrow \lambda_{1} \Psi_{0} c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{1} \rightarrow \lambda_{1} \Psi_{0} c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{1},$$

$$\lambda_{0}c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \rightarrow \lambda_{0}c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{0} \rightarrow \lambda_{1} \Psi_{0} c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{1} \rightarrow \lambda_{1} \Psi_{0} c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{1} \rightarrow \lambda_{1} \Psi_{0} c_{x_{1}^{1}}^{x_{1}} c_{x_{2}^{1}}^{x_{1}} \Psi_{1},$$

and their last step by $\gamma_{1}$ rules are one way.
Obviously $\mathcal{R}_P$ is length preserving. The proof of the following main lemma is found in the next section.

**Lemma 3.8** Let $P$ be an instance of PCP. Then, $P$ has a solution if and only if $\mathcal{R}_P$ is not confluent.

**Theorem 3.9** Confluence of length preserving SRSs is undecidable.

**Proof.** We assume that confluence of length preserving SRSs is decidable. Then it follows from Lemma 3.8 that PCP is decidable, which contradicts to Theorem 3.4. □

Strings can be regarded as terms over unary functions and a variable. For example, a string $abc$ corresponds to a term $a(b(c(x)))$. Hence we can consider classes of TRSs that contains all length preserving SRSs. Then, the undecidability of confluence for such classes is a corollary of the above theorem.

Structure preserving TRSs are TRSs in which the left-hand side and right-hand side of each rule have the same tree structure and the same variable occurrences. The tree structures are stable against reductions in this class of TRSs. Hence we have the following corollary.

**Corollary 3.10** Confluence of structure preserving TRSs is undecidable.

## 4 Proof of Lemma 3.8

We use notion of persistency to simplify proofs as done in [2].

**Theorem 4.1** (persistence of confluence[11]) A well-typed (many-sorted) term rewriting system is confluent if and only if its underlined (untyped) term rewriting system is confluent.

Now we apply Theorem 4.1 to $\mathcal{R}_P$.

**Lemma 4.2** Let $\mathcal{R}_P$ be the SRS over $\Sigma = \{\Xi_0, \Xi_1, \Xi_2, \Psi_0, \Psi_1, \Psi_2\} \cup \Sigma_c$ obtained from an instance $P$ of PCP. Then $\mathcal{R}_P$ is confluent if and only if $w$ is confluent for every $w$ in either of the following three forms:

(p1) $\Xi_i \chi$

(p2) $\chi \Psi_j$,

(p3) $\Xi_i \chi \Psi_j$,

where $\chi \in (\Sigma_c)^*$, $i, j \in \{0, 1, 2\}$.

**Proof.** Considering following typing to $R_P$, the lemma follows from the persistency (Theorem 4.1).

$$\begin{align*}
\Psi_i : T'' \to T & \text{ for each } i \in \{0, 1, 2\} \\
X : T \to T & \text{ for each } X \in \Sigma_c \\
\Xi_i : T \to T' & \text{ for each } i \in \{0, 1, 2\}
\end{align*}$$

□
In the sequel, we analyze the confluent property for \( \mathcal{R}_P = \Theta \cup \Theta^{-1} \cup \Phi \) obtained from an instance \( P \) of PCP.

We define an equivalence relation \( \sim \subseteq (\overline{A})^* \times (\overline{A})^* \) as identity relation with ignoring all null symbols \( - \), that is \( u \sim v \) if and only if \( \tilde{u} = \tilde{v} \) where \( \tilde{u} \) and \( \tilde{v} \) denote the strings obtained from \( u \) and \( v \) by omitting all \( - \)'s respectively.

For a string \( \Xi_0(c \cdots c)_v^w \psi_0 \), rules in \( \alpha_1 \) and \( \beta_1 \) are to check that \( u = u' \) and \( v = v' \) and that \( (u, v) \) consists of a list of pairs in \( \overline{P} \). Rules in \( \delta \) gather null symbols in subscripts \( u' \) and \( v' \) backward. From these observation, the following lemma holds.

**Lemma 4.3** Let \( u, u', v, v' \in (\overline{A})^* \). Then, \( u' \sim u = u_1 \cdots u_k \) and \( v' \sim v = v_1 \cdots v_k \) for some \( k \) and \( (u_i, v_i) \in \overline{P} \) if and only if \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P^*}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_1 \) for some \( \chi \in (\Sigma_c)^* \).

**Proof.** (\( \Rightarrow \)) We have a reduction sequence \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_0 \) since \( u \sim u' \) and \( v \sim v' \). As shown in Example 3.5, we have a reduction sequence \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P^*}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_1 \) for some \( x_i \in \overline{A} \) since \( u = u_1 \cdots u_n \) and \( v = v_1 \cdots v_n \) for some \( (u_i, v_i) \in \overline{P} \).

(\( \Leftarrow \)) Let \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_1 \). Then rules \( \alpha_1 \) and \( \beta_1 \) must be used in the reduction and all tags of \( \chi \) are \( p' \) or \( c' \). Since \( u \) and \( v \) cannot be modified by any rule in \( \mathcal{R}_P \) and any possible reductions have no harmful branches from the construction of \( \mathcal{R}_P \), the string \( \Xi_0(c \cdots c)_v^w \psi_0 \) must appear in the reduction. Thus we have \( u = u_1 \cdots u_k \) and \( v = v_1 \cdots v_k \) for some \( k \) and \( (u_i, v_i) \in \overline{P} \) from the construction of \( \alpha_1 \) and \( \beta_1 \) rules. We also have \( u \sim u' \) and \( v \sim v' \) from the construction of \( \delta \) rules.

For a string \( \Xi_0(c \cdots c)_v^w \psi_0 \), rules in \( \alpha_2 \) and \( \beta_2 \) are to check that \( u' = v' \). From this observation, the following lemma holds.

**Lemma 4.4** Let \( u, u', v, v' \in (\overline{A})^* \) such that \( |u| = |u'| = |v| = |v'| \). Then, \( u' \sim v' \) if and only if \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_2 \) for some \( \chi \in (\Sigma_c)^* \).

**Proof.** (\( \Rightarrow \)) We have a reduction sequence \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_0 \) since \( u' \sim v' \). As shown in Example 3.5, we have a reduction sequence \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_2 \) for some \( x_i \in \overline{A} \).

(\( \Leftarrow \)) Let \( \Xi_0(c \cdots c)_v^w \psi_0 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_2 \). Then rules \( \alpha_2 \) and \( \beta_2 \) must be used and all tags of \( \chi \) are \( c' \). Since any possible reductions have no harmful branches from the construction of \( \mathcal{R}_P \), the string \( \Xi_0(c \cdots c)_v^w \psi_0 \) must appear in the reduction for some string \( w \). Thus we have \( u' \sim w \) and \( v' \sim w \) from the construction of \( \delta \) rules.

**Lemma 4.5** Let \( P \) be an instance of PCP.

(a) If \( P \) has a solution, then \( \Xi_0(c \cdots c)_v^w \psi_1 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_2 \) for some \( \chi, \chi' \in (\Sigma_c)^* \).

(b) If \( \Xi_0(c \cdots c)_v^w \psi_1 \stackrel{R_P}{\rightarrow} \Xi_0(c \cdots c)_v^w \psi_2 \) for some \( \chi, \chi' \in (\Sigma_c)^* \), then \( P \) has a solution.

**Proof.** (a) Let \( P \) has a solution. Then we have \( u = u_1 \cdots u_k \sim v_1 \cdots v_k = v \) for some \( k \) and \( (u_i, v_i) \in \overline{P} \). Hence the claim follows from Lemma 4.3, Lemma 4.4.
(b) Let \( \Xi_0 \stackrel{f_1}{\gamma_1} \chi \Psi_1 \stackrel{f_2}{\gamma_2} \Xi_0 e_{\gamma_0} \). Then it is easy to see that a string \( \Xi_0 (c \cdots c) \gamma_0 \Psi_0 \) must appear in this reduction. From Lemma 4.3 and Lemma 4.4, we have \( u = u_1 \cdots u_k \sim v \) for some \( k \) and \( (u_i, v_i) \in P \), which means \( P \) has a solution \( \hat{u} \).

The following proposition obviously follows since \( (\mathcal{R}_P \backslash (\gamma_1 \cup \gamma_2))^{-1} \subset \mathcal{R}_P \).

**Proposition 4.6** Let \( S_1, S_2 \in \Sigma^* \). Then,

1. \( S_1 \stackrel{\mathcal{R}_P \backslash \gamma_1}{\rightarrow} S_2 \) implies \( S_1 \downarrow \mathcal{R}_P S_2 \), and
2. \( S_1 \stackrel{\mathcal{R}_P \backslash \gamma_2}{\rightarrow} S_2 \) implies \( S_1 \downarrow \mathcal{R}_P S_2 \).

**Proof.** Proof by induction on the number of reductions by \( \gamma \) rules.

**Proof for Lemma 3.8**

\( \Rightarrow \): Let \( P \) has a solution. Then we have \( \Xi_0 \stackrel{f_1}{\gamma_1} \chi \Psi_1 \stackrel{f_2}{\gamma_2} \Xi_0 (c \cdots c) \gamma_0 \Psi_0 \stackrel{f_2}{\gamma_2} \Xi_0 e_{\gamma_0} \chi' \Psi_2 \) for some \( \chi, \chi' \in (\Sigma_c)^* \) by Lemma 4.5(a). Hence we have \( \Xi_1 \stackrel{f_1}{\gamma_1} \chi \Psi_1 \stackrel{f_2}{\gamma_2} \Xi_0 (c \cdots c) \gamma_0 \Psi_0 \stackrel{f_2}{\gamma_2} \Xi_2 e_{\gamma_0} \chi' \Psi_2 \) by using rules \( \gamma_1 \) and \( \gamma_2 \), which leads non-confluence of \( \mathcal{R}_P \).

\( \Leftarrow \): Let \( P \) has no solution. Let's show that \( \mathcal{R}_P \) is confluent. Let \( S_1 \stackrel{\mathcal{R}_P}{\rightarrow} S_0 \stackrel{\mathcal{R}_P}{\rightarrow} S_2 \). From Lemma 4.2, it is enough to consider three kinds of forms \( (p1), (p2) \) and \( (p3) \) as \( S_0 \).

- Consider the case that \( S_0 \) starts with \( \Xi_0 \) and ends with \( \Psi_i \) for some \( i \in \{0, 1, 2\} \). Assume that both of \( \gamma_1 \) and \( \gamma_2 \) are applied in the reduction sequence. Then \( P \) must have a solution by Lemma 4.5(b), which is a contradiction. Hence at least one of \( \gamma_1 \) or \( \gamma_2 \) rules cannot be applied in the reduction sequence.

- In either of the other cases:
  - The case that \( S_0 \) ends with \( \Psi_i \) for some \( i \in \{0, 1, 2\} \) and all other symbols are of \( \Sigma_c \).
  - The case that \( S_0 \) starts with \( \Xi_0 \) of \( \Xi_2 \), and
  - The case that \( S_0 \) starts with \( \Xi_0 \) and all other symbols are of \( \Sigma_c \).

It is easy to see that at least one of \( \gamma_1 \) or \( \gamma_2 \) rules cannot be applied in the reduction sequence.

In any of the above cases, we have \( S_1 \downarrow \mathcal{R}_P S_2 \) by Proposition 4.6.

**References**


