An ACCL which is not a CRCL

Colm Ó Dúnlaing

Mathematics, Trinity College, Dublin 2, Ireland

December 3, 2014

Abstract

It is fairly easy to show that every regular set is an almost-confluent congruential language (ACCL), and it is known [3] that every regular set is a Church-Rosser congruential language (CRCL). Whether there exists an ACCL, which is not a CRCL, seems to remain an open question. In this note we present one such ACCL.

1 Introduction

\(\Sigma^*\) denotes the set of ‘strings’ over an alphabet \(\Sigma\) — \(\Sigma\) can be any finite set; strings over \(\Sigma\) are finite sequences drawn from \(\Sigma\). \(\Sigma^*\) is a monoid (with identity \(\lambda\), the empty string) under string concatenation. The length of a string \(x\) is denoted \(|x|\) (\(|\lambda| = 0\)). If \(x \in \Sigma^*\) and \(a \in \Sigma\) then

\[|x|_a\]

is the number of occurrences of \(a\) in \(x\), so

\[\sum_{a \in \Sigma} |x|_a = |x|.

(1.1) Definition A Thue system over a finite alphabet \(\Sigma\) is a set of ordered pairs \((u, w)\) of strings in \(\Sigma^*\). In this note only finite Thue systems are considered.

If \(T\) is a Thue system, then we call the pairs \((u, w)\) in \(T\) its rules, sometimes written \(\leftrightarrow T\).

A congruence on \(\Sigma^*\) (or any semigroup) is an equivalence relation \(\equiv\) such that for all \(u, v, x, y \in \Sigma^*\),

\[x \equiv y \implies uxv \equiv uyw\]

The equivalence classes can be multiplied and thus there is a quotient monoid

\[\Sigma^* / \equiv\]

If \(\equiv\) is a congruence and \(x\) a string, we write

\[[x]_\equiv\]

for the congruence class of \(x\) modulo \(\equiv\).

*e-mail: odunlain@maths.tcd.ie. Mathematics department website: http://www.maths.tcd.ie.
Given \(x, y \in \Sigma^*\), we write

\[x \leftrightarrow_T y\]

if there exist strings \(t, u, v, w\), such that \(x = tuv\), \(y = twv\), and either \((u, w) \in T\) or \((w, u) \in T\).

This relation is symmetric, and its reflexive transitive closure

\[\leftrightarrow^*\]

is a congruence on \(\Sigma^*\). The notation for congruence class is simplified as follows.

\[[x]_T = (\text{def}) \ [x]_{\leftrightarrow^*_T}.

Emphasis is placed on the relative lengths of strings in rules of \(T\).

If \(x \leftrightarrow_T y\) and in addition \(|x| > |y|\), \(|x| \geq |y|\), or \(|x| = |y|\), respectively, write

\[x \rightarrow_T y, \quad \text{or} \quad x \leftrightarrow_T y, \quad \text{or} \quad x \sqsubseteq_T y,\]

respectively.

Since the relation \(\leftrightarrow_T\) is symmetric, we can assume that for any \((u, w) \in T\),

\[|u| \geq |w|\]

(1.2) Definition When \(x = tuv \rightarrow_T twv = y\), so \(|u| > |w|\), we call \(u\) the redex and \(w\) the reduct.

(1.3) Definition A Thue system \(T\) is, respectively, (i) Church-Rosser, (ii) almost confluent, (iii) preperfect, (see [1]), if whenever \(x \leftrightarrow^*_T y\),

(i) there exists a string \(z\) such that \(x \rightarrow^*_T z\) and \(y \rightarrow^*_T z\);

(ii) there exist strings \(z_1\) and \(z_2\) such that \(x \rightarrow^*_T z_1\), \(y \rightarrow^*_T z_2\), and \(z_1 \sqsubseteq^*_T z_2\);

(iii) there exists a string \(z\) such that \(x \rightarrow^*_T z\) and \(y \rightarrow^*_T z\).

(1.4) Definition If \(T\) is a Church-Rosser Thue system, then for any string \(x\), every string \(y\) in \([x]_T\) reduces (modulo \(T\)) to the same irreducible string; we call this string

\(\text{irr}_T(x)\).

The word problem for Church-Rosser systems is in linear time, and for the other two kinds it is PSPACE complete; testing for the Church-Rosser property is tractable; testing for almost confluence is in PSPACE; it is undecidable whether a Thue system is preperfect [1].

(1.5) Definition A language \(L\) is congruential if there exists a congruence \(\equiv\) and a finite set of strings

\[x_1, x_2, \ldots, x_n, \quad \text{such that} \quad L = [x_1]_{\equiv} \cup [x_2]_{\equiv} \cup \ldots \cup [x_n]_{\equiv}\]

If the congruence is generated by a Thue system, i.e., it is \(\leftrightarrow_T\) for some finite Thue system \(T\), and \(T\) is, respectively, Church-Rosser, or almost confluent, or preperfect, then \(L\) is a Church-Rosser, or almost confluent, or preperfect congruential language: CRCL, ACCL, or PPCL.
An interesting and old result is that every regular set is an ACCL. It can be shown as follows: if \( L \) is a regular set then there exists a finite monoid \( M \) and a homomorphism from \( \Sigma^* \) to \( M \) such that \( L \) is a union of \( h^{-1}(g) \) for suitable \( g \) in \( M \). But this partition
\[
\{h^{-1}(g) : g \in M\}
\]
can also be realised by a finite almost-confluent system, namely: let \( N \) be the maximal length of minimal strings in this partition (a string is minimal if whenever \( x \leftrightarrow_T y, |x| \leq |y| \)). Then the system
\[
S = \{(x, y) : x, y \in \Sigma^*, |x| \leq N + 1, x \leftrightarrow_T y, y \text{ minimal}\}
\]
is almost confluent and its congruence classes coincide with the inverse images \( h^{-1}(g) \), as required.

A long-standing open problem was whether every regular set is a CRCL: it was settled in the affirmative a few years ago [3].

That left open the unlikely possibility that every ACCL is a CRCL. This note shows the contrary.

The analysis in this paper is simple and direct. In fact, the problem is not susceptible to more sophisticated methods. As noted in [4], Kolmogorov-complexity-based analyses showing palindromes not to be Church-Rosser also shows them not to be almost confluent. Indeed, in [4] we were only able to show that they are ‘preperfect languages’.

All Church-Rosser monoids are \( \text{FP}_\infty \) [5,2]. On the other hand, if one inspects the group furnished by Squier [5], which is not \( \text{FP}_3 \), it has an obvious presentation as a monoid, but the presentation again turns out to be preperfect rather than almost confluent.

Book’s reduction machine [1] can be used with almost-confluent Thue systems, from which it is follows that ACCLs are linear time recognisable. The word problem for an almost confluent Thue system is PSPACE-complete, but (as is easy to show) if the system presents a group then the word problem is linear time. So there are few complexity-based arguments separating ACCLs from CRCLs.

2 An ACCL which is not a CRCL

We shall introduce an almost confluent Thue system over a 4-letter alphabet \( \Sigma = \{a, b, c, d\} \), and an involution
\[
a \mapsto c \mapsto a, \quad b \mapsto d \mapsto b
\]
or
\[
\overline{a} = c, \overline{c} = a, \overline{b} = d, \overline{d} = b.
\]

Any string in \( \Sigma^* \) can and will be written using \( a, b, \overline{a}, \overline{b} \).

\begin{definition}
We call \( a, b \) positive and \( c, d \) (i.e., \( \overline{a}, \overline{b} \)) negative.

Given a string \( x \) over \( a, b, \overline{a}, \overline{b} \),
\[
|x|_{\text{pos}} = |x|_a + |x|_b, \quad |x|_{\text{neg}} = |x|_{\overline{a}} + |x|_{\overline{b}}
\]
the number of occurrences of positive and negative letters in \( x \).
\end{definition}

\footnote{Church-Rosser languages are a much richer class of languages than Church-Rosser congruential.}
Let \( h : \Sigma^* \rightarrow \mathbb{Z} \) (the additive group of integers) denote the following map:

\[
h(x) = |x|_{\text{pos}} - |x|_{\text{neg}}.
\]

This is a homomorphism, and

\[
a \mapsto 1, \quad b \mapsto 1, \quad a \mapsto -1, \quad b \mapsto -1.
\]

Let \( S \) be the Thue system

\[
aa \rightarrow \lambda, \quad \overline{a}a \rightarrow \lambda, \quad a\overline{b} \rightarrow \lambda, \quad b\overline{a} \rightarrow \lambda, \quad \overline{a}b \rightarrow \lambda, \quad \overline{b}a \rightarrow \lambda, \quad a \overline{b} \rightarrow \lambda, \quad b \overline{a} \rightarrow \lambda.
\]

The map \( h \) preserves both sides of each rule in \( S \), and therefore induces a homomorphism

\[
\Sigma^*/\xrightarrow{\sim} S \rightarrow \mathbb{Z}.
\]

For the rest of this paper, we assume that strings are written in terms of \( a, b, \overline{a}, \overline{b} \).

(2.2) Definition Given a string \( x = a_1 a_2 \ldots a_k \), the string \( \tilde{x} \) is defined as

\[
\tilde{x} = \overline{a}_k \overline{a}_{k-1} \ldots \overline{a}_1.
\]

Clearly \( h(x\tilde{x}) = 0 \) and \([x\tilde{x}]_S = [\tilde{x}x]_S = [\lambda]_S\).

(2.3) Definition A string \( x \) is mixed if it contains both positive (\( a \) or \( b \)) and negative (\( \overline{a} \) or \( \overline{b} \)) letters. Else it is unmixed. Unmixed strings can be empty, positive, or negative, in the obvious sense.

If \( x \) is mixed, then it contains an adjacent pair of positive and negative letters which can be reduced (modulo \( S \)). Thus mixed strings are reducible. Unmixed strings are irreducible.

Thus every string \( x \) can be reduced to a positive or negative string. If \( x \) is positive then \( h(x) = |x| \).

If \( x \) is negative then \( h(x) = -|x| \).

(2.4) Lemma If \( x \) and \( y \) are both positive strings, or both negative, and \(|x| = |y|\), then \( x \xrightarrow{\sim} S y \).

(2.5) Corollary \( S \) is almost confluent and \( h \) induces an isomorphism of \( \Sigma^*/\xrightarrow{\sim} S \) with \( \mathbb{Z} \).

Proof. Suppose \( h(x) = h(y) \).

Reduce \( x \) and \( y \) (modulo \( S \)) to irreducible strings \( x' \) and \( y' \). Then \( h(x') = h(x) = h(y) = h(y') \), and \( x' \) and \( y' \) are unmixed.

If \( h(x) = 0 \), then \( x' = y' = \lambda \). If \( h(x) > 0 \), then \( x' \) and \( y' \) are entirely positive, \(|x'| = |y'|\), and \( x' \xrightarrow{\sim} S y' \).

Similarly if \( h(x) < 0 \).

We have shown that if \( h(x) = h(y) \) then there exist irreducible strings \( x' \) and \( y' \) such that \( x \xrightarrow{\sim} S x' \), \( y \xrightarrow{\sim} S y' \), and \( x' \xrightarrow{\sim} S y' \).
In particular, \( x \leftrightarrow_S y \). Conversely, as has been noted, if \( x \leftrightarrow_S y \) then \( h(x) = h(y) \): \( h \) induces an isomorphism of \( \Sigma^* / \leftrightarrow_S \) with its image, \( \mathbb{Z} \).

Finally, if \( x \leftrightarrow_S y \), then \( h(x) = h(y) \), so there exist strings \( x', y' \) so

\[
\begin{align*}
x \mathrel{\leftrightarrow_S} x' & \mathrel{\leftrightarrow_S} y' \mathrel{\leftrightarrow_S} y
\end{align*}
\]

so \( S \) is almost confluent. \( \blacksquare \)

(2.6) **Definition**

\[ L = [\lambda]_S = h^{-1}(0). \]

This is our candidate for a non-CRCL.

(2.7) **Corollary** \( L \) is an ACCL. \( \blacksquare \)

(2.8) **Theorem** \( L \) is not a CRCL.

We prove this by contradiction. Otherwise there exists a Church-Rosser Thue system \( T \) and a list of irreducible strings

\[ u_1, \ldots, u_n \]

in \( L \) such that

(2.9)

\[ L = [\lambda]_S = [u_1]_T \cup \ldots \cup [u_n]_T \]

or equivalently

\[ x \in L \iff \text{irr}_T(x) \in \{u_1, \ldots, u_n\}. \]

Associated with \( T \) and the strings \( u_j \), we define the following constants:

(2.10) **Definition**

\[ Q = \max_{(\ell, r) \in T} |\ell| \quad \text{and} \quad R = \max_{1 \leq j \leq n} |u_j|_{\text{neg}}. \]

\( Q \) is the maximum length of redexes in \( T \).

(2.11) **Lemma** If such a Thue system \( T \) exists, then \( T \) refines \( S \) (in the sense that \( x \mathrel{\leftrightarrow_T} y \implies x \mathrel{\leftrightarrow_S} y \)).

**Proof.** It is enough to show that whenever

\[ x \rightarrow_T y, \]

\[ [x]_S = [y]_S. \] Clearly

\[ x \bar{x} \rightarrow_T y \bar{x} \]

But \( x \bar{x} \in [\lambda]_S \), which is a union of congruence class modulo \( T \), so \( y \bar{x} \in [\lambda]_S \). Then \( [y \bar{x}]_S = [\lambda x]_S = [x]_S \). But \( [y \lambda\bar{x}]_S = [y\lambda]_S = [y]_S \), so \( [x]_S = [y]_S \), as required. \( \blacksquare \)

(2.12) **Corollary** If \( x \) is unmixed, then \( x \) is irreducible (modulo \( T \)).
Proof: $x$ is irreducible (modulo $S$) and $T$ refines $S$. 

(2.13) **Lemma** Suppose that $xy \rightarrow_T z$ where $y$ is unmixed (and $|z| \geq Q$). Then $z$ can be factored as $xy'$ where $y'$ is unmixed and $|y'| > |y| - Q$ \((\ref{eq:2.10})\).

**Proof.** The redex in $xy$ cannot be entirely in $y$ since $y$ is irreducible. Therefore the redex is in $xs$ where $|s| < Q$ (possibly $s = \lambda$). Setting $xy = xsy'$, $y'$ is a suffix of $z$, $y'$ is unmixed, and $|y'| > |y| - Q$. 

(2.14) **Lemma** Suppose $x \rightarrow_T y$. Then $|x|_{\text{pos}} > |y|_{\text{pos}}$ and $|x|_{\text{neg}} > |y|_{\text{neg}}$.

**Proof** Since $h(x) = h(y)$, $|x|_{\text{neg}} - |y|_{\text{neg}} = |x|_{\text{pos}} - |y|_{\text{pos}}$, so the number of positive and negative letters is reduced by the same amount, namely, $(|x| - |y|)/2$. 

(2.15) **Corollary** For any positive integer $k$, if $y$ is positive of length $QR + k$ \((\ref{eq:2.10})\), then for $1 \leq i \leq n$,

$$y \quad \text{and} \quad \text{irr}_T(u_iy)$$

agree on their rightmost $k$ letters.

**Proof.** Lemma \((\ref{eq:2.13})\) can be extended inductively so that if $u_iy$ is reduced $t$ times, then the reduced string agrees with $y$ on their rightmost $|y| - tQ$ letters. By Lemma \((\ref{eq:2.14})\), $u_iy$ can be reduced at most $|u_iy|_{\text{neg}}$ times. But $|u_iy|_{\text{neg}} = |u_i|_{\text{neg}}$ and $|u_i|_{\text{neg}} \leq R$, so $y$ and $\text{irr}_T(u_iy)$ agree on their rightmost $|y| - QR$ letters; and $|y| - QR = k$. 

**Proof of Theorem \((\ref{eq:2.3})\)** Let $k = \lceil \log_2(n+1) \rceil$ and let $x$ be a positive string of length $QR + k$. For any positive string $y$ of the same length as $x$, $x \overset{*}{\leftrightarrow}_S y$.

Let $u_i = \text{irr}_T(x\hat{x})$ (noting that $x\hat{x} \in L$). For any positive string $y$ with $|y| = |x|$, $x\overset{\lambda}{\leftrightarrow}_S y$ so $\hat{x}\hat{x}\overset{\lambda}{\leftrightarrow}_S \hat{x}y$. But $\hat{x}\hat{x}\overset{\lambda}{\leftrightarrow}_S L$ and $\text{irr}_T(\hat{x}y) = u_j$ for some $j$. Therefore $[\hat{x}y]_T = [u_j]_T$ and $[x\hat{x}\hat{x}]_T = [zu_j]_T$. But $u_i = \text{irr}_T(x\hat{x})$, so, for every positive $y$ with $|y| = |x|$,

$$[u_iy]_T = [zu_j]_T$$

(2.16)

for some $j$. Let $\{y_q\}$ be an enumeration of all positive strings $y$ of length $|x|$ which agree with $x$ on their first $QR$ letters. There are $2^k$ such strings. By Corollary \((\ref{eq:2.15})\) for each string $y_q$,

$$y_q \quad \text{and} \quad \text{irr}_T(u_iy_q)$$

agree on their rightmost $k$ letters. The irreducible strings belong to different congruence classes. Therefore there are at least $2^k$ congruence classes fitting the left-hand side of equation \((\ref{eq:2.16})\) and there are at most $n$ classes matching the right-hand side. Since $2^k > n$, we have a contradiction: $L$ is not a CRCL. 

**3 Acknowledgement**

The author is grateful to Friedrich Otto for some corrections and helpful suggestions.
4 References

1. Ronald V. Book and Friedrich Otto (1993). String-rewriting systems. Springer texts and monographs in computer science.

2. Daniel E. Cohen (1997). String rewriting and homology of monoids. Math. Structures in Computer Science 7:3, 207–240.

3. Volker Diekert, Manfred Kufleitner, Klaus Reinhardt, and Tobias Walter (2012). Regular languages are Church-Rosser congruential. Proc. 39th. ICALP II, Springer LNCS 7392, 177–188.

4. Colm Ó Dúnlaing and Natalie Schluter (2010). A shorter proof that palindromes are not a Church-Rosser language, with extensions to almost-confluent and preperfect Thue systems. Theoretical Computer Science 411, 677–690.

5. Craig C. Squier (1987). Word problems and a homological finiteness condition for monoids. J. Pure and Applied Algebra 49, 201–217.