Abstract

The language MIX consists of all strings over the three-letter alphabet \{a, b, c\} that contain an equal number of occurrences of each letter. We prove Joshi’s (1985) conjecture that MIX is not a tree-adjoining language.

1 Introduction

The language

$$\text{MIX} = \{ w \in \{a, b, c\}^* \mid |w|_a = |w|_b = |w|_c \}$$

has attracted considerable attention in computational linguistics.\(^1\) This language was used by Bach (1981) in an exercise to show that the permutation closure of a context-free language is not necessarily context-free.\(^2\) MIX may be considered a prototypical example of free word order language, but, as remarked by Bach (1981), it seems that no human language “has such complete freedom for order”, because “typically, certain constituents act as ‘boundary domains’ for scrambling”. Joshi (1985) refers to MIX as representing “an extreme case of the degree of free word order permitted in a language”, which is “linguistically not relevant”. Gazdar (1988) adopts a similar position regarding the relation between MIX and natural languages, noting that “it seems rather unlikely that any natural language will turn out to have a MIX-like characteristic”.

It therefore seems natural to assume that languages such as MIX should be excluded from any class of formal languages that purports to be a tight formal characterization of the possible natural languages. It was in this spirit that Joshi et al. (1991) suggested that MIX should not be in the class of so-called mildly context-sensitive languages:

“[mildly context-sensitive grammars] capture only certain kinds of dependencies, e.g., nested dependencies and certain limited kinds of cross-serial dependencies (for example, in the subordinate clause constructions in Dutch or some variations of them, but perhaps not in the so-called MIX (or Bach) language) . . .”

Mild context-sensitivity is an informally defined notion first introduced by Joshi (1985); it consists of the three conditions of limited cross-serial dependencies, constant growth, and polynomial parsing. The first condition is only vaguely formulated, but the other two conditions are clearly satisfied by tree-adjoining grammars. The suggestion of Joshi et al. (1991) was that MIX should be regarded as a violation of the condition of limited cross-serial dependencies.

Joshi (1985) conjectured rather strongly that MIX is not a tree-adjoining language: “TAGs cannot generate this language, although for TAGs the proof is not in hand yet”. An even stronger conjecture was made by Marsh (1985), namely, that MIX is not an
indexed language. It is known that the indexed languages properly include the tree- adjoining languages.) Joshi et al. (1991), however, expressed a more pessimistic view about the conjecture:

“It is not known whether TAG . . . can generate MIX. This has turned out to be a very difficult problem. In fact, it is not even known whether an IG [(indexed grammar)] can generate MIX.”

This open question has become all the more pressing after a recent result by Salvati (2011). This result says that MIX is in the class of multiple context-free languages (Seki et al., 1991), or equivalently, languages of linear context-free rewriting systems (Vijay-Shanker et al., 1987; Weir, 1988), which has been customarily regarded as a formal counterpart of the informal notion of a mildly context-sensitive language.

It means that either we have to abandon the identification of multiple context-free languages with mildly context-sensitive languages, or we should revise our conception of limited cross-serial dependencies and stop regarding MIX-like languages as violations of this condition. Surely, the resolution of Joshi’s (1985) conjecture should crucially affect the choice between these two alternatives.

In this paper, we prove that MIX is not a tree- adjoining language. Our proof is cast in terms of the formalism of head grammar (Pollard, 1984; Roach, 1987), which is known to be equivalent to TAG (Vijay-Shanker and Weir, 1994). The key to our proof is the notion of an n-decomposition of a string over \( \{a, b, c\} \), which is similar to the notion of a derivation in head grammars, but independent of any particular grammar. The parameter \( n \) indicates how unbalanced the occurrence counts of the three letters can be at any point in a decomposition. We first show that if MIX is generated by some head grammar, then there is an \( n \) such that every string in MIX has an \( n \)-decomposition. We then prove that if every string in MIX has an \( n \)-decomposition, then every string in MIX must have a 2-decomposition. Finally, we exhibit a particular string in MIX that has no 2-decomposition. The length of this string is 87, and the fact that it has no 2-decomposition was verified by a computer program accompanying this paper. We include here a rigorous, mathematical proof of this fact not relying on the computer verification.

2 Head Grammars

A head grammar is a quadruple \( G = (N, \Sigma, P, S) \), where \( N \) is a finite set of nonterminals, \( \Sigma \) is a finite set of terminal symbols (alphabet), \( S \) is a distinguished element of \( N \), and \( P \) is a finite set of rules. Each nonterminal is interpreted as a binary predicate on strings in \( \Sigma^* \). There are four types of rules:

\[
\begin{align*}
A(x_1, x_2, y_1, y_2) & \leftarrow B(x_1, x_2), C(y_1, y_2) \\
A(x_1, x_2, y_1, y_2) & \leftarrow B(x_1, x_2), C(y_1, y_2) \\
A(x_1, y_1, y_2, x_2) & \leftarrow B(x_1, x_2), C(y_1, y_2) \\
A(w_1, w_2) & \leftarrow
\end{align*}
\]

Here, \( A, B, C \in N \), \( x_1, x_2, y_1, y_2 \) are variables, and \( w_1, w_2 \in \Sigma \cup \{\varepsilon\} \). Rules of the first three types are binary rules and rules of the last type are terminating rules. This definition of a head grammar actually corresponds to a normal form for head grammars that appears in section 3.3 of Vijay-Shanker and Weir’s (1994) paper.

The rules of head grammars are interpreted as implications from right to left, where variables can be instantiated to any terminal strings. Each binary relation of MIX with indexed languages is also of interest in combinatorial group theory. Gilman (2005) remarks that “it does not . . . seem to be known whether or not the word problem of \( Z \times Z \) is indexed”, alluding to the language \( O_2 = \{w \in \{a, b, \bar{a}, \bar{b}\}^* \mid |w_{\bar{a}}| = |w_{\bar{b}}| = |w_{b}| = |w_{\bar{b}}|\} \). Since \( O_2 \) and MIX are rationally equivalent, \( O_2 \) is indexed if and only if MIX is indexed (Salvati, 2011).

Joshi et al. (1991) presented linear context-free rewriting systems as mildly context-sensitive grammars. Groenink (1997) wrote “The class of mildly context-sensitive languages seems to be most adequately approached by LCFRS.”

\[A \rightarrow C_{2,2}(B, C)\]
\[A \rightarrow C_{1,3}(B, C)\]
\[A \rightarrow W(B, C)\]
\[A \rightarrow C_{1,1}(w_1 \uparrow w_2)\]
rule involves an operation that combines two pairs of strings to form a new pair. The operation involved in the third rule is known as wrapping; the operations involved in the first two rules we call left concatenation and right concatenation, respectively. If \( G = (N, \Sigma, P, S) \) is a head grammar, \( A \in N, \) and \( w_1, w_2 \in \Sigma^* \), then we say that a fact \( A(w_1, w_2) \) is derivable and write \( \vdash_G A(w_1, w_2) \), if \( A(w_1, w_2) \) can be inferred using the rules in \( P \). More formally, we have \( \vdash_G A(w_1, w_2) \) if one of the following conditions holds:

- \( A(w_1, w_2) \) is a terminating rule in \( P \).
- \( \vdash_G B(u_1, u_2) \), \( \vdash_G C(v_1, v_2) \), and there is a binary rule \( A(\alpha_1, \alpha_2) \leftarrow B(x_1, x_2), C(y_1, y_2) \) in \( P \) such that \( (w_1, w_2) \) is the result of substituting \( u_1, u_2, v_1, v_2 \) for \( x_1, x_2, y_1, y_2 \), respectively, in \( (\alpha_1, \alpha_2) \).

The language of \( G \) is

\[
L(G) = \{ w_1w_2 \mid \vdash_G S(w_1, w_2) \}.
\]

**Example 1.** Let \( G = (N, \Sigma, P, S) \), where \( N = \{ S, A, A', C, D, E, F \} \), \( \Sigma = \{ a, \tilde{a}, \# \} \), and \( P \) consists of the following rules:

\[
\begin{align*}
S(x_1y_1, y_2x_2) & \leftarrow D(x_1, x_2), C(y_1, y_2) \\
C(e, \#) & \leftarrow \\
D(e, e) & \leftarrow \\
D(x_1y_1, y_2x_2) & \leftarrow F(x_1, x_2), D(y_1, y_2) \\
F(x_1y_1, y_2x_2) & \leftarrow A(x_1, x_2), E(y_1, y_2) \\
A(a, a) & \leftarrow \\
E(x_1y_1, y_2x_2) & \leftarrow D(x_1, x_2), A'(y_1, y_2) \\
A'(\tilde{a}, \tilde{a}) & \leftarrow
\end{align*}
\]

We have \( L(G) = \{ w\#w^R \mid w \in D_{[a,\tilde{a}]} \} \), where \( D_{[a,\tilde{a}]} \) is the Dyck language over \( \{ a, \tilde{a} \} \) and \( w^R \) is the reversal of \( w \). All binary rules of this grammar are wrapping rules.

If \( \vdash_G A(w_1, w_2) \), a derivation tree for \( A(w_1, w_2) \) is a finite binary tree whose nodes are labeled by facts that are derived during the derivation of \( A(w_1, w_2) \). A derivation tree for \( A(w_1, w_2) \) represents a “proof” of \( \vdash_G A(w_1, w_2) \), and is formally defined as follows:

- If \( A(w_1, w_2) \) is a terminating rule, then a tree with a single node labeled by \( A(w_1, w_2) \) is a derivation tree for \( A(w_1, w_2) \).
- If \( \vdash_G B(u_1, u_2) \), \( \vdash_G C(v_1, v_2) \), and there is a binary rule \( A(\alpha_1, \alpha_2) \leftarrow B(x_1, x_2), C(y_1, y_2) \) in \( P \) such that \( (w_1, w_2) \) is the result of substituting \( u_1, u_2, v_1, v_2 \) for \( x_1, x_2, y_1, y_2 \), respectively, in \( (\alpha_1, \alpha_2) \), then a tree with two nodes labeled by \( B(u_1, u_2) \) and \( C(v_1, v_2) \) is a derivation tree for \( A(w_1, w_2) \).

If \( w \in L(G) \), a derivation tree for \( w \) is a derivation tree for some \( S(w_1, w_2) \) such that \( w_1w_2 = w \).

**Example 1 (continued).** Figure 1 shows a derivation tree for \( aa\tilde{a}a\tilde{a}aa\tilde{a}aa\).

The following lemma should be intuitively clear from the definition of a derivation tree:

**Lemma 1.** Let \( G = (N, \Sigma, P, S) \) be a head grammar and \( A \) be a nonterminal in \( N \). Suppose that \( w \in L(G) \) has a derivation tree in which a fact \( A(v_1, v_2) \) appears as a label of a node. Then there are strings \( z_0, z_1, z_2 \) with the following properties:

(i) \( w = z_0v_1z_1v_2z_2 \), and

(ii) \( \vdash_G A(u_1, u_2) \) implies \( z_0u_1z_1u_2z_2 \in L(G) \).

**Proof.** We can prove by straightforward induction on the height of derivation trees that whenever \( A(v_1, v_2) \) appears on a node in a derivation tree for \( B(w_1, w_2) \), then there exist \( z_0, z_1, z_2, z_3 \) that satisfy one of the following conditions:

(a) \( w_1 = z_0v_1z_1v_2z_2, w_2 = z_3, \) and \( \vdash_G A(u_1, u_2) \) implies \( \vdash_G B(z_0u_1z_1u_2z_2, z_3) \).

(b) \( w_1 = z_0, w_2 = z_1v_1z_2v_2z_3, \) and \( \vdash_G A(u_1, u_2) \) implies \( \vdash_G B(z_0, z_1u_2z_2u_3) \).
(c) \( w_1 = z_0v_1z_1, w_2 = z_2v_2z_3 \), and \( \vdash_G A(u_1, u_2) \) implies \( \vdash_G B(z_0u_1z_1, z_2u_2z_3) \).

We omit the details. \( \square \)

We call a nonterminal \( A \) of a head grammar \( G \) useless if \( A \) does not appear in any derivation trees for strings in \( L(G) \). Clearly, useless nonterminals can be eliminated from any head grammar without affecting the language of the grammar.

3 Decompositions of Strings in MIX

Henceforth, \( \Sigma = \{a, b, c\} \). Let \( \mathbb{Z} \) denote the set of integers. Define functions \( \psi_1, \psi_2 : \Sigma^* \rightarrow \mathbb{Z}, \psi : \Sigma^* \rightarrow \mathbb{Z} \times \mathbb{Z} \) by

\[
\psi_1(w) = |w|_a - |w|_b,
\psi_2(w) = |w|_b - |w|_c,
\psi(w) = (\psi_1(w), \psi_2(w)).
\]

Clearly, we have \( \psi(a) = (1, 0), \psi(b) = (0, 1), \psi(c) = (-1, -1) \), and

\[ w \in \text{MIX} \iff \psi(w) = (0, 0). \]

Note that for all strings \( w_1, w_2 \in \Sigma^* \), \( \psi(w_1w_2) = \psi(w_1) + \psi(w_2) \). In other words, \( \psi \) is a homomorphism from the free monoid \( \Sigma^* \) to \( \mathbb{Z} \times \mathbb{Z} \) with addition as the monoid operation and \( (0, 0) \) as identity.

**Lemma 2.** Suppose that \( G = (N, \Sigma, P, S) \) is a head grammar without useless nonterminals such that \( L(G) \subseteq \text{MIX} \). There exists a function \( \Psi_G : N \rightarrow \mathbb{Z} \times \mathbb{Z} \) such that \( \vdash_G A(u_1, u_2) \) implies \( \psi(u_1u_2) = \Psi_G(A) \).

**Proof.** Since \( G \) has no useless nonterminals, for each nonterminal \( A \) of \( G \), there is a derivation tree for some string in \( L(G) \) in which \( A \) appears in a node label. By **Lemma 1**, there are strings \( z_0, z_1, z_2 \) such that \( \vdash_G A(u_1, u_2) \) implies \( z_0u_1z_1u_2z_2 \in L(G) \). Since \( L(G) \subseteq \text{MIX} \), we have \( \psi(z_0u_1z_1u_2z_2) = (0, 0) \), and hence

\[ \psi(u_1u_2) = \Psi_G(A) = \psi(z_0z_1z_2). \]

A decomposition of \( w \in \Sigma^* \) is a finite binary tree satisfying the following conditions:

- the root is labeled by some \((w_1, w_2)\) such that \( w = w_1w_2 \),
- each internal node whose left and right children are labeled by \((u_1, u_2)\) and \((v_1, v_2)\), respectively, is labeled by one of \((u_1u_2v_1, v_2), (u_1u_2v_1, v_2u_2)\), \((u_1v_1, v_2u_2)\),
- each leaf node is labeled by some \((s_1, s_2)\) such that \( s_1s_2 \in \{b, c\}^* \cup \{a, c\}^* \cup \{a, b\}^* \).

Thus, the label of an internal node in a decomposition is obtained from the labels of its children by left concatenation, right concatenation, or wrapping. It is easy to see that if \( G \) is a head grammar over the alphabet \( \Sigma \), any derivation for \( w \in L(G) \) induces a decomposition of \( w \). (Just strip off nonterminals.) Note that unlike with derivation trees, we have placed no bound on the length of a string that may appear on a leaf node of a decomposition. This will be convenient in some of the proofs below.

When \( p \) and \( q \) are integers, we write \([p, q]\) for the set \([r \in \mathbb{Z} | p \leq r \leq q]\). We call a decomposition of \( w \) an \( n \)-decomposition if each of its nodes is labeled by some \((v_1, v_2)\) such that \( \psi(v_1v_2) \in [-n, n] \times [-n, n] \).

**Lemma 3.** If \( \text{MIX} = L(G) \) for some head grammar \( G = (\Sigma, N, P, S) \), then there exists an \( n \) such that each \( w \in \text{MIX} \) has an \( n \)-decomposition.

**Proof.** We may suppose without loss of generality that \( G \) has no useless nonterminal. Since \( \text{MIX} = L(G) \), there is a function \( \Psi_G \) satisfying the condition of **Lemma 2**. Since the set \( N \) of nonterminals of \( G \) is finite, there is an \( n \) such that \( \Psi_G(A) \in [-n, n] \times [-n, n] \) for all \( A \in N \). Then it is clear that a derivation tree for \( w \in L(G) \) induces an \( n \)-decomposition of \( w \). \( \square \)

If \( w = d_1 \ldots d_m \in \Sigma^m \), then for \( 0 \leq i \leq j \leq m \), we write \( w[i, j] \) to refer to the substring \( d_i \ldots d_j \) of \( w \). (As a special case, we have \( w[i, i] = \varepsilon \).) The following is a key lemma in our proof:

**Lemma 4.** If each \( w \in \text{MIX} \) has an \( n \)-decomposition, then each \( w \in \text{MIX} \) has a 2-decomposition.

**Proof.** Assume that each \( w \in \text{MIX} \) has an \( n \)-decomposition. Define a homomorphism \( \gamma_n : \Sigma^* \rightarrow \Sigma^* \) by

\[
\gamma_n(a) = a^n,
\gamma_n(b) = b^n,
\gamma_n(c) = c^n.
\]
Clearly, $\gamma_n$ is an injection, and we have $\psi(\gamma_n(v)) = n \cdot \psi(v)$ for all $v \in \Sigma^*$.

Let $w \in \text{MIX}$ with $|w| = m$. Then $w' = \gamma_n(w) \in \text{MIX}$ and $|w'| = mn$. By assumption, $w'$ has an $n$-decomposition $\mathcal{D}$. We assign a 4-tuple $(i, j, k, l)$ of natural numbers to each node of $\mathcal{D}$ in such a way that $(w'[i, j], w'[k, l])$ equals the label of the node. This is done recursively in an obvious way, starting from the root. If the root is labeled by $(w_1, w_2)$, then it is assigned $(0, |w_1|, |w_1|, |w_1w_2|)$. If a node is assigned a tuple $(i, j, k, l)$ and has two children labeled by $(u_1, u_2)$ and $(v_1, v_2)$, respectively, then the 4-tuples assigned to the children are determined according to how $(u_1, u_2)$ and $(v_1, v_2)$ are combined at the parent node:

\[
\begin{array}{cccc}
i & u_1 & u_2 & j \\
i + |u_1| & i + |u_1u_2| & v_1 & k + |u_2v_1| \\
i & u_1 & j & k + |u_2| \\
i + |u_1| & i + |u_1u_2| & v_1 & v_2 \\
\end{array}
\]

Now define a function $f : [0, mn] \to \{ kn \mid 0 \leq k \leq m \}$ by

\[
f(i) = \begin{cases} 
  i & \text{if } n \text{ divides } i, \\
  n \cdot \lfloor i/n \rfloor & \text{if } n \text{ does not divide } i \text{ and } w'[i - 1, i] \in [a, b], \\
  n \cdot \lceil i/n \rceil & \text{if } n \text{ does not divide } i \text{ and } w'[i - 1, i] = c.
\end{cases}
\]

Clearly, $f$ is weakly increasing in the sense that $i \leq j$ implies $f(i) \leq f(j)$. Let $\mathcal{D}'$ be the result of replacing the label of each node in $\mathcal{D}$ by

\[
(w'[f(i), f(j)], w'[f(k), f(l)]),
\]

where $(i, j, k, l)$ is the 4-tuple of natural numbers assigned to that node by the above procedure. It is easy to see that $\mathcal{D}'$ is another decomposition of $w'$. Note that since each of $f(i), f(j), f(k), f(l)$ is an integral multiple of $n$, we always have

\[
(w'[f(i), f(j)], w'[f(k), f(l)]) = (\gamma_n(u), \gamma_n(v))
\]

for some substrings $u, v$ of $w$. This implies that for $h = 1, 2$,

\[
\psi_h(w'[f(i), f(j)]w'[f(k), f(l)])
\]

is an integral multiple of $n$.

**Claim.** $\mathcal{D}'$ is a $2n$-decomposition.

We have to show that every node label $(v_1, v_2)$ in $\mathcal{D}'$ satisfies $\psi(v_1v_2) \in [-2n, 2n] \times [-2n, 2n]$. For $h = 1, 2$, define $\varphi_h : [0, mn] \times [0, mn] \to \mathbb{Z}$ as follows:

\[
\varphi_h(i, j) = \begin{cases} 
  \psi_h(w'[i, j]) & \text{if } i \leq j, \\
  -\psi_h(w'[j, i]) & \text{otherwise}.
\end{cases}
\]

Then it is easy to see that for all $i, j, i', j' \in [0, mn]$,

\[
\varphi_h(i', j') = \varphi_h(i', i) + \varphi_h(i, j) + \varphi_h(j, j').
\]

Inspecting the definition of the function $f$, we can check that

\[
\varphi_h(f(i), i) \in [0, n - 1]
\]

always holds. Suppose that $(i, j, k, l)$ is assigned to a node in $\mathcal{D}$. By assumption, we have

\[
\psi_h(w'[i, j]w'[k, l]) \in [-n, n],
\]

and

\[
\psi_h(w'[f(i), f(j)]w'[f(k), f(l)])
\]

\[
= \psi_h(w'[f(i), f(j)]) + \psi_h(w'[f(k), f(l)])
\]

\[
= \varphi_h(f(i), i) + \varphi_h(f(j), j) + \varphi_h(f(k), k) + \varphi_h(f(l), l)
\]

\[
= \varphi_h(f(i), i) + \varphi_h(w'[i, j]) + \varphi_h(j, f(j)) + \varphi_h(f(k), k) + \varphi_h(w'[k, l]) + \varphi_h(l, l)
\]

\[
= \varphi_h(w'[i, j]w'[k, l]) + \varphi_h(f(i), i) + \varphi_h(f(k), k) + \varphi_h(f(j), j) + \varphi_h(l, l)
\]

\[
eq \psi_h(w'[i, j]w'[k, l]) + \varphi_h(f(i), i) + \varphi_h(f(k), k) + \varphi_h(f(j), j) + \varphi_h(l, l)
\]

\[
\in [p + q_1 + q_2 + r_1 + r_2 | p \in [-n, n], q_1, q_2 \in [0, n - 1], r_1, r_2 \in [-n + 1, 0]}
\]

\[
= [-3n + 2, 3n - 2].
\]

Since $\psi_h(w'[f(i), f(j)]w'[f(k), f(l)])$ must be an integral multiple of $n$, it follows that

\[
\psi_h(w'[f(i), f(j)]w'[f(k), f(l)]) \in [-2n, -n, 0, n, 2n].
\]

This establishes the claim.
We have shown that each node of $D'$ is labeled by a pair of strings of the form $(\gamma_n(u), \gamma_n(v))$ such that

$$\psi(\gamma_n(u)\gamma_n(v)) \in \{-2n, -n, 0, n, 2n\} \times \{-2n, -n, 0, n, 2n\}.$$ 

Now it is easy to see that inverting the homomorphism $\gamma_n$ at each node of $D'$

$$(\gamma_n(u), \gamma_n(v)) \mapsto (u, v)$$

gives a 2-decomposition of $w$. □

4 A String in MIX That Has No 2-Decomposition

By Lemmas 3 and 4, in order to prove that there is no head grammar for MIX, it suffices to exhibit a string in MIX that has no 2-decomposition. The following is such a string:

$$w = a^{5}b^{14}a^{19}c^{29}b^{15}a^{5}.$$ 

In this section, we prove that the string $w$ has no 2-decomposition.\footnote{This fact was first verified by the computer program accompanying this paper. The program, written in C, implements a generic, memoized top-down recognizer for the language $\{ w \in \text{MIX} \mid w \text{ has a 2-decomposition} \}$, and does not rely on any special properties of the string $w$.}

It helps to visualize strings in MIX as closed curves in a plane. If $w$ is a string in MIX, by plotting the coordinates of $\psi(v)$ for each prefix $v$ of $w$, we can represent $w$ by a closed curve $C$ together with a map $t: [0, |w|] \to C$. The representation of the string $z$ is given in Figure 2.

Let us call a string $w \in \{a, b, c\}^*$ such that $\psi(w) \in [-2, 2] \times [-2, 2]$ long if $w$ contains all three letters, and short otherwise. (If $\psi(w) \notin [-2, 2] \times [-2, 2]$, then $w$ is neither short nor long.) It is easy to see that a short string $w$ always satisfies

$$|w|_a \leq 4, \quad |w|_b \leq 4, \quad |w|_c \leq 2.$$ 

The maximal length of a short string is 6. (For example, $a^4c^2$ and $b^4c^2$ are short strings of length 6.) We also call a pair of strings $(v_1, v_2)$ long (or short) if $v_1v_2$ is long (or short, respectively).

According to the definition of an $n$-decomposition, a leaf node in a 2-decomposition must be labeled by a short pair of strings. We call a 2-decomposition normal if the label of every internal node is long. Clearly, any 2-decomposition can be turned into a normal 2-decomposition by deleting all nodes that are descendants of nodes with short labels.

One important property of the string $z$ is the following:

**Lemma 5.** If $z = x_1v_2$ and $\psi(v) \in [-2, 2] \times [-2, 2]$, then either $v$ or $x_1v_2$ is short.

**Proof.** This is easy to see from the graphical representation in Figure 2. If a substring $v$ of $z$ has $\psi(v) \in [-2, 2] \times [-2, 2]$, then the subcurve corresponding to $v$ must have initial and final coordinates whose difference lies in $[-2, 2] \times [-2, 2]$. If $v$ contains all three letters, then it must contain as a substring at least one of $ba^{19}c$, $ac^{29}b$, and $cb^{15}a$. The only way to satisfy both these conditions is to have the subcurve corresponding to $v$ start and end very close to the origin, so that $x_1v_2$ is short. (Note that the distance between the coordinate $(5, 0)$ corresponding to position 5 of $z$ and the diagonal segment corresponding to the substring $c^{29}$ is large enough that it is impossible for $v$ to start at position 5 and end in the middle of $c^{29}$ without violating the condition $\psi(v) \in [-2, 2] \times [-2, 2]$.) □

Lemma 5 leads to the following observation. Let us call a decomposition of a string concatenation-free if each of its non-leaf labels is the wrapping of the labels of the children.
Lemma 6. If $z$ has a 2-decomposition, then $z$ has a normal, concatenation-free 2-decomposition.

Proof. Let $D$ be a 2-decomposition of $z$. Without loss of generality, we may assume that $D$ is normal. Suppose that $D$ contains a node $\mu$ whose label is the left or right concatenation of the labels of its children, $(u_1, u_2)$ and $(v_1, v_2)$. We only consider the case of left concatenation since the case of right concatenation is entirely analogous; so we suppose that the node $\mu$ is labeled by $(u_1u_2v_1, v_2)$.

It follows that $z = x_1u_1u_2x_2$, and by Lemma 5, either $u_1u_2$ or $x_1x_2$ is short. If $u_1u_2$ is short, then the left child of $\mu$ is a leaf because $D$ is normal. We can replace its label by $(u_1u_2, e)$; the label $(u_1u_2v_1, v_2)$ of $\mu$ will now be the wrapping (as well as left concatenation) of the two child labels, $(u_1u_2, e)$ and $(v_1, v_2)$. If $x_1x_2$ is short, then we can combine by wrapping a single node labeled by $(x_1, x_2)$ with the subtree of $D$ rooted at the left child of $\mu$, to obtain a new 2-decomposition of $z$. In either case, the result is a normal 2-decomposition of $z$ with fewer instances of concatenation. Repeating this procedure, we eventually obtain a normal, concatenation-free 2-decomposition of $z$. □

Another useful property of the string $z$ is the following:

Lemma 7. Suppose that the following conditions hold:

(i) $z = x_1u_1v_1y_2u_2x_2$, 
(ii) $x_1y_2$ is a short string, and 
(iii) both $\psi(u_1u_2)$ and $\psi(v_1v_2)$ are in $[-2, 2] 	imes [-2, 2]$.

Then either $(u_1, u_2)$ or $(v_1, v_2)$ is short.

Proof. Suppose $(u_1, u_2)$ and $(v_1, v_2)$ are both long. Since $(u_1, u_2)$ and $(v_1, v_2)$ must both contain $c$, either $u_1$ ends in $c$ and $v_1$ starts in $c$, or else $v_2$ ends in $c$ and $u_2$ starts in $c$.

Case 1. $u_1$ ends in $c$ and $v_1$ starts in $c$. Since $(v_1, v_2)$ must contain at least one occurrence of $a$, the string $v_1y_2v_2$ must contain $cb^{15}a$ as a substring.

Since $x_1y_2x_2$ is short, we have $|y_2| \leq 4$. It follows that $|v_1y_2v_2| \geq 11$. But $v_1y_2v_2$ is a substring of $c^{28}b^{15}a^5$, so $|v_1y_2v_2| \leq 5$. This clearly contradicts $\psi(v_1y_2) \in [-2, 2] \times [-2, 2]$.

Case 2. $v_2$ ends in $c$ and $u_2$ starts in $c$. In this case, $cb^{15}a^5$ is a suffix of $u_2x_2$. Since $x_1y_2x_2$ is short, $|x_2| \leq 4$. This means that $cb^{15}a$ is a substring of $u_2$ and hence $|u_2| = 15$.

On the other hand, since $(v_1, v_2)$ must contain at least one occurrence of $b$, the string $v_1y_2v_2$ must contain $ba^9c$ as a substring. This implies that $|u_1u_2| \leq 10$. But since $|u_2| = 15$, we have $|u_1u_2| \geq 15$. This clearly contradicts $\psi(u_1u_2) \in [-2, 2] \times [-2, 2]$. □

We now assume that $z$ has a normal, concatenation-free 2-decomposition $D$ and derive a contradiction. We do this by following a certain path in $D$. Starting from the root, we descend in $D$, always choosing a non-leaf child, as long as there is one. We show that this path will never terminate.

The $i$-th node on the path will be denoted by $\mu_i$, counting the root as the 0-th node. The label of $\mu_i$ will be denoted by $(w_{i,1}, w_{i,2})$. With each $i$, we associate three strings $x_{i,1}, y_{i,1}, x_{i,2}$ such that $x_{i,1}y_{i,1}w_{i,2}x_{i,2} = z$, analogously to Lemma 1. Since $\psi(w_{i,1}w_{i,2}) \in [-2, 2] \times [-2, 2]$ and $\psi(z) = (0, 0)$, we will always have $\psi(x_{i,1}y_{i,1}x_{i,2}) \in [-2, 2] \times [-2, 2]$.

Initially, $(w_{0,1}, w_{0,2})$ is the label of the root $\mu_0$ and $x_{0,1} = y_0 = x_{0,2} = e$. If $\mu_i$ is not a leaf node, let $(u_{i,1}, u_{i,2})$ and $(v_{i,1}, v_{i,2})$ be the labels of the left and right children of $\mu_i$, respectively. If the left child is not a leaf node, we let $\mu_{i+1}$ be the left child, in which case we have $(w_{i+1,1}, w_{i+1,2}) = (u_{i,1}, u_{i,2})$, $x_{i+1,1} = x_{i,1}$, $x_{i+1,2} = x_{i,2}$, and $y_{i+1} = y_{i,1}y_{i,2}$. Otherwise, $\mu_{i+1}$ will be the right child of $\mu_i$, and we have $(w_{i+1,1}, w_{i+1,2}) = (v_{i,1}, v_{i,2})$, $x_{i+1,1} = x_{i,1}u_{i,1}$, $x_{i+1,2} = u_{i,2}x_{i,2}$, and $y_{i+1} = y_{i,1}$.

The path $\mu_0, \mu_1, \mu_2, \ldots$ is naturally divided into two parts. The initial part of the path consists of nodes where $x_{i,1}y_{i,1}x_{i,2}$ is short. Note that $x_{i,1}y_{i,1}x_{i,2} = e$ is short. As long as $x_{i,1}y_{i,1}x_{i,2}$ is short, $(w_{i,1}, w_{i,2})$ must be long and $\mu_i$ has two children labeled by $(u_{i,1}, u_{i,2})$ and $(v_{i,1}, v_{i,2})$. By Lemma 7, either $(u_{i,1}, u_{i,2})$ or $(v_{i,1}, v_{i,2})$ must be short. Since the length
of $z$ is 87 and the length of a short string is at most 6, exactly one of $(u_{i,1}, u_{i,2})$ and $(v_{i,1}, v_{i,2})$ must be long.

We must eventually enter the second part of the path, where $x_{i,1}y_{i,2}$ is no longer short. Let $\mu_m$ be the first node belonging to this part of the path. Note that at $\mu_m$, we have $\psi(x_{m,1}y_{m}x_{m,2}) = \psi(x_{m-1,1}y_{m-1}x_{m-1,2}) + \psi(v)$ for some short string $v$. (Namely, $v = u_{m-1,1}u_{m-2,1}$ or $v = v_{m-1,1}v_{m-2,1}$.)

**Lemma 8.** If $u$ and $v$ are short strings and $\psi(uv) \in [-2,2] \times [-2,2]$, then $|uv|_d \leq 4$ for each $d \in \{a, b, c\}$.

**Proof.** Since $u$ and $v$ are short, we have $|u_a| \leq 4, |u_b| \leq 4, |u_c| \leq 2$ and $|v_a| \leq 4, |v_b| \leq 4, |v_c| \leq 2$. It immediately follows that $|uv|_c \leq 4$. We distinguish two cases.

**Case 1.** $|uv|_c \leq 2$. Since $\psi(uv) \in [-2,2] \times [-2,2]$, we must have $|u|_a \leq 4$ and $|v|_b \leq 4$.

**Case 2.** $|uv|_c > 2$. Since $|u_a| \leq 2$ and $|v_c| \leq 2$, we must have $|u_c| \geq 1$ and $|v_c| \geq 1$. Also, $\psi(uv) \in [-2,2] \times [-2,2]$ implies that $|uv|_d \geq 1$ and $|uv|_b \geq 1$. Since $u$ and $v$ are short, it follows that one of the following two conditions must hold:

(i) $|u_a| \geq 1, |u_b| = 0$ and $|v_a| = 0, |v_b| \geq 1$.

(ii) $|u_a| = 0, |u_b| \geq 1$ and $|v_a| \geq 1, |v_b| = 0$.

In the former case, $|uv|_a = |u|_a \leq 4$ and $|uv|_b = |v|_b \leq 4$. In the latter case, $|uv|_a = |v|_a \leq 4$ and $|uv|_b = |u|_b \leq 4$.

By Lemma 8, the number of occurrences of each letter in $x_{m,1}y_{m}x_{m,2}$ is in $[1,4]$. This can only be if

\[
x_{m,1}x_{m,2} = a^j, \quad y_{m} = c^b b^l,
\]

for some $j,k,l \in [1,4]$. This means that the string $z$ must have been split into two strings $(w_{i,0,1}, w_{i,0,2})$ at the root of $D$ somewhere in the vicinity of position 67 (see Figure 2).

It immediately follows that for all $i \geq m$, $w_{i,1}$ is a substring of $a^5 b^{14} a^{19} c^{28}$ and $w_{i,2}$ is a substring of $b^{14} a^5$. We show by induction that for all $i \geq m$, the following condition holds:

$$(\dagger) \text{ ba}^{19} c^{17} \text{ is a substring of } w_{i,1}.$$ 

The condition $(\dagger)$ clearly holds for $i = m$. Now assume $(\dagger)$. Then $(w_{i,1}, w_{i,2})$ is long, and $\mu_i$ has left and right children, labeled by $(u_{i,1}, u_{i,2})$ and $(v_{i,1}, v_{i,2})$, respectively, such that $w_{i,1} = u_{i,1}v_{i,1}$ and $w_{i,2} = v_{i,2}u_{i,2}$.

We consider two cases.

**Case 1.** $u_{i,1}$ contains $c$. Then $ba^{19}c$ is a substring of $w_{i,1}$. Since $u_{i,2}$ is a substring of $b^{14}a^5$, it cannot contain any occurrences of $c$. Since $\psi_4(u_{i,1}u_{i,2}) \in [-2,2]$, it follows that $u_{i,1}$ must contain at least 17 occurrences of $c$; hence $ba^{19}c^{17}$ is a substring of $w_{i,1}$. Since $(u_{i,1}, u_{i,2})$ is long, $(w_{i+1,1}, w_{i+1,2}) = (u_{i,1}, u_{i,2})$.

Therefore, the condition $(\dagger)$ holds with $i + 1$ in place of $i$.

**Case 2.** $u_{i,1}$ does not contain $c$. Then $(u_{i,1}, u_{i,2})$ is short and $(w_{i+1,1}, w_{i+1,2}) = (v_{i,1}, v_{i,2})$. Note that $v_{i,1}$ must contain at least 17 occurrences of $c$, but $v_{i,2}$ is a substring of $b^{14}a^5$ and hence cannot contain more than 14 occurrences of $b$. Since $\psi_4(v_{i,2}) \in [-2,2]$, it follows that $v_{i,1}$ must contain at least one occurrence of $b$. Therefore, $ba^{19}c^{17}$ must be a substring of $v_{i,1} = w_{i+1,1}$, which shows that $(\dagger)$ holds with $i + 1$ in place of $i$.

We have proved that $(\dagger)$ holds for all $i \geq m$. It follows that for all $i, \mu_i$ has two children and hence $\mu_{i+1}$ is defined. This means that the path $\mu_0, \mu_1, \mu_2, \ldots$ is infinite, contradicting the assumption that $D$ is a 2-decomposition of $z$.

We have proved the following:

**Lemma 9.** There is a string in MIX that has no 2-decomposition.

**Theorem 10.** There is no head grammar $G$ such that $L(G) = MIX$.

**Proof.** Immediate from Lemmas 3, 4, and 9. □

**References**

Setsuo Arikawa, Takeshi Shinohara, and Akihiro Yamamoto. 1992. Learning elementary formal systems. *Theoretical Computer Science*, 95(1):97–113.

Emmon Bach. 1981. Discontinuous constituents in generalized categorial grammars. In Victoria Burke and James Pustejovsky, editors, *Proceedings of the 11th Annual Meeting of the North East Linguistic Society*, pages 1–12.

Emmon Bach. 1988. Categorial grammars as theories of language. In Richard T. Oehrle, Emmon Bach, and Deirdre Wheeler, editors, *Categorial Grammars and Natural Language Structures*, pages 17–34. D. Reidel, Dordrecht.

673
Gerald Gazdar. 1988. Applicability of indexed grammars to natural languages. In U. Reyle and C. Rohrer, editors, *Natural Language Parsing and Linguistic Theories*, pages 69–94. D. Reidel Publishing Company, Dordrecht.

Robert Gilman. 2005. Formal languages and their application to combinatorial group theory. In Alexandre V. Borovik, editor, *Groups, Languages, Algorithms*, number 378 in Contemporary Mathematics, pages 1–36. American Mathematical Society, Providence, RI.

Annius V. Groenink. 1997. Mild context-sensitivity and tuple-based generalizations of context-free grammar. *Linguistics and Philosophy*, 20:607–636.

Aravind K. Joshi, Vijay K. Shanker, and David J. Weir. 1991. The convergence of mildly context-sensitive grammar formalisms. In Peter Sells, Stuart M. Shieber, and Thomas Wasow, editors, *Foundational Issues in Natural Language Processing*, pages 31–81. The MIT Press, Cambridge, MA.

Aravind K. Joshi. 1985. Tree-adjointing grammars: How much context sensitivity is required to provide reasonable structural descriptions? In David Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge.

Markus Kracht. 2003. *The Mathematics of Language*, volume 63 of *Studies in Generative Grammar*. Mouton de Gruyter, Berlin.

William Marsh. 1985. Some conjectures on indexed languages. Paper presented to the Association for Symbolic Logic Meeting, Stanford University, July 15–19. Abstract appears in *Journal of Symbolic Logic* 51(3):849 (1986).

Carl J. Pollard. 1984. *Generalized Phrase Structure Grammars, Head Grammars, and Natural Language*. Ph.D. thesis, Department of Linguistics, Stanford University.

Kelly Roach. 1987. Formal properties of head grammars. In Alexis Manaster-Ramer, editor, *Mathematics of Language*, pages 293–347. John Benjamins, Amsterdam.

Sylvain Salvati. 2011. MIX is a 2-MCFL and the word problem in $\mathbb{Z}^2$ is captured by the IO and the OI hierarchies. Technical report, INRIA.

Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context free grammars. *Theoretical Computer Science*, 88(2):191–229.

Raymond M. Smullyan. 1961. *Theory of Formal Systems*. Princeton University Press, Princeton, NJ.

K. Vijay-Shanker and D. J. Weir. 1994. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:511–546.