Pumping lemmas for classes of languages generated by folding systems

Jorge C. Lucero[*]

December 14, 2018

Abstract

Geometric folding processes are ubiquitous in natural systems ranging from protein biochemistry to patterns of insect wings and leaves. In a previous study, a folding operation between strings of formal languages was introduced as a model of such processes. The operation was then used to define a folding system (F-system) as a construct consisting of a core language, containing the strings to be folded, and a folding procedure language, which defines how the folding is done. This paper reviews main definitions associated with F-systems and next it determines necessary conditions for a language to belong to classes generated by such systems. The conditions are stated in the form of pumping lemmas and four classes are considered, in which the core and folding procedure languages are both regular, one of them is regular and the other context-free, or both are context-free. Full demonstrations of the lemmas are provided, and the analysis is illustrated with examples.

1 Introduction

In a recent paper, Sburlan (2011) introduced a folding operation for strings of symbols of a given formal language. The operation was inspired in actual geometric folding processes that occur in, e.g., protein biochemistry (Dobson, 2003), in-vitro DNA shaping (Rothemund, 2006), and even origami (the Japanese art of paper folding; Demaine and O'Rourke, 2007), and it was proposed as a restricted computational model of such processes.

The folding operation was applied to define folding systems (F-systems) of the form $\Phi = (L_1, L_2)$, where L_1 is the language that contains the strings to be folded (the core language), and L_2 is the language that contains strings defining how the folding must be performed (the folding procedure language). Then, the computing power of F-systems was investigated by comparison with standard language classes from the Chomsky hierarchy; i.e., regular, context-free, context-sensitive, recursive and recursively enumerable languages.

The paper fitted well within a growing body of applications of geometric folding to science and technology which have surged in recent years; in, e.g., aerospace and automotive technology (Cipra, 2001), civil engineering (Filipov, Tachi, and Paulino, 2015), biology (Mahadevan and Rica, 2005), and robotics (Felton, Tolley, Demaine, Rus, and Wood,

---

[*]Dept. Computer Science, University of Brasília, Brazil. E-mail: lucero@unb.br

2014). Also, a number of theoretical studies have considered algebraic models, algorithmic complexity, and other mathematical and computational aspects of folding(e.g., Akitaya, Demaine, and Ku, 2017; Alperin, 2000; Ida, Fleuriot, and Ghourabi, 2016).

F-systems might find relevant applications to DNA computing and related areas of natural computing (Kari and Rozenberg, 2008). Thus, the general purpose of the present paper is to further explore capabilities and limitations of such systems, and it will consider the classes of languages generated when the core and the folding procedure languages are regular or context-free. Necessary conditions for a language to belong to some of those classes were presented by Sburlan (2011) in the form of pumping lemmas, similar to the well known pumping lemmas for regular and context-free languages (Hopcroft, Motwani, and Ullman, 2001). His analysis considered the cases in which both the core and the folding procedure language are regular, and in which the core language is context-free and the folding procedure language is regular. Here, the two cases will be revised, in order to solve detected inconsistencies (see footnote 2 to Lemma 1). The lemmas will be restated in a weaker form (as consequence of the revision) and full proofs will be provided. Also, it will be shown that the same lemma for the case in which the core language is context-free and the folding procedures language is regular also applies to the case in which the class attribution is reversed. Finally, a lemma for the remaining case in which both the core and the folding procedure languages are context-free will be presented and proved.

2   Folding systems

For clarity of the analysis, let us review main definitions associated to folding operations and systems.

Let Σ be an alphabet, $\Gamma = \{\mathtt{u}, \mathtt{d}\}$, and $f : \Sigma^* \times \Sigma \times \Gamma \to \Sigma^*$ a function such that

$$f(w, a, b) = \begin{cases} aw, & \text{if } b = \mathtt{u}, \\ wa, & \text{if } b = \mathtt{d}. \end{cases} \tag{1}$$

Then, the folding function $h : \Sigma^* \times \Gamma^* \to \Sigma^*$ is a partial function defined by

$$h(w, v) = \begin{cases} f(f(\dots f(\varepsilon, a_1, b_1) \dots, a_{k-1}, b_{k-1}), a_k, b_k), & \text{if } |w| = |v| > 0 \\ \varepsilon, & \text{if } |w| = |v| = 0, \\ \text{undefined}, & \text{if } |w| \neq |v|. \end{cases} \tag{2}$$
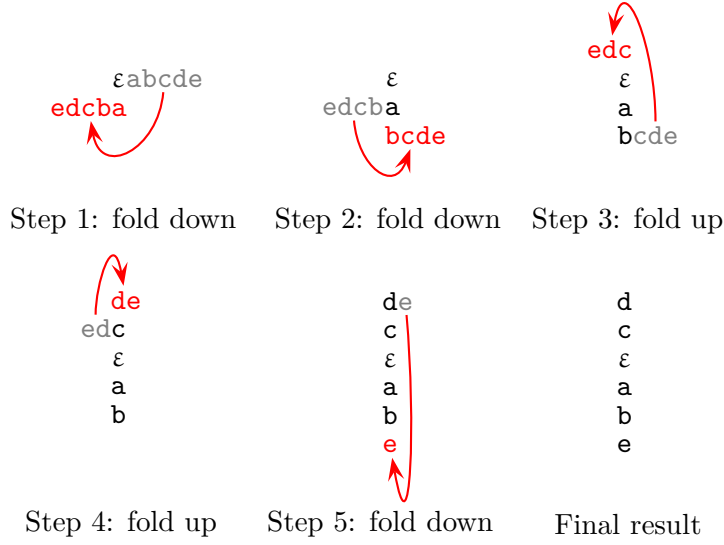
where $w = a_1 a_2 \dots a_k$, $v = b_1 b_2 \dots b_k$, $a_i \in \Sigma$ and $b_i \in \Gamma$ for $i = 1, 2, \cdots, k$, and ε is the empty string.[1]

Example 1. Let $w = \mathtt{abcde}$ and $v = \mathtt{dduud}$. Then,
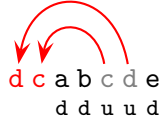
$$\begin{aligned} h(w, v) &= f(f(f(f(f(\varepsilon, \mathtt{a}, \mathtt{d}), \mathtt{b}, \mathtt{d}), \mathtt{c}, \mathtt{u}), \mathtt{d}, \mathtt{u}), \mathtt{e}, \mathtt{d}), \\ &= f(f(f(f(\mathtt{a}, \mathtt{b}, \mathtt{d}), \mathtt{c}, \mathtt{u}), \mathtt{d}, \mathtt{u}), \mathtt{e}, \mathtt{d}), \\ &= f(f(f(\mathtt{ab}, \mathtt{c}, \mathtt{u}), \mathtt{d}, \mathtt{u}), \mathtt{e}, \mathtt{d}), \\ &= f(f(\mathtt{cab}, \mathtt{d}, \mathtt{u}), \mathtt{e}, \mathtt{d}), \\ &= f(\mathtt{dcab}, \mathtt{e}, \mathtt{d}), \\ &= \mathtt{dcabe}. \end{aligned}$$

---

[1] Sburlan's (2011) original definition has been modified in order to include the case in which both w and v are empty strings.

Figure 1: Computation of $h(\texttt{abcde}, \texttt{dduud}) = \texttt{dcabe}$ as a sequence of folding operations.

Figure 2: Computation of $h(\texttt{abcde}, \texttt{dduud}) = \texttt{dcabe}$ using Eq. (3).

The computation of $h(w, v)$ is represented graphically in Fig. 1. As shown there, each application of function f may be regarded as a folding operation that arranges the symbols of w in a stack. Strings over Γ describe how each folding must be performed, where symbol $\texttt{u}$ represents a "folding up" action and symbol $\texttt{d}$ represents a "folding down" action. The final result is the created stack, read from top to bottom.

Another way to see the folding operation is illustrated in Fig. 2. The folded string may be written as

$$h(w, v) = w_{\texttt{u}}^{\mathcal{R}} w_{\texttt{d}}, \tag{3}$$

where $w_{\texttt{u}}$ is the sequence of symbols in w that are folded up, $\mathcal{R}$ denotes the reverse order operator, and $w_{\texttt{d}}$ is the sequence of symbols in w that are folded down. Letting $w = xy$, $v = st$, with $|x| = |s|$ and $|y| = |t|$, we obtain the identity

$$h(xy, st) = y_{\texttt{u}}^{\mathcal{R}} h(x, s) y_{\texttt{d}}. \tag{4}$$

A folding system (F-system) is defined as a pair $\Phi = (L_1, L_2)$, where $L_1 \subseteq \Sigma^*$ is the core language, and $L_2 \subseteq \Gamma^*$ is the folding procedure language. The language of Φ is

$$L(\Phi) = \{h(w, v) | w \in L_1, v \in L_2, |w| = |v|\}. \tag{5}$$

The class of all languages generated by F-systems with core languages of a class $\mathcal{C}$ and folding procedure languages of a class $\mathcal{H}$ is defined as

$$\mathcal{F}(\mathcal{C}, \mathcal{H}) = \{L(\Phi) | \Phi = (L_1, L_2), L_1 \in \mathcal{C}, L_2 \in \mathcal{H}\}. \tag{6}$$

3

3 Pumping lemmas

The first lemma states necessary conditions for a language to belong to class $\mathcal{F}(\mathsf{REG}, \mathsf{REG})$, where REG is the class of regular languages.

Lemma 1. *Let $L \in \mathcal{F}(REG, REG)$ be an infinite language over an alphabet Σ. Then, there are strings $u, v, x, y, z \in \Sigma^*$, with $|vy| > 0$, such that $uv^i xy^i z \in L$ for all $i \geq 0$.*[2]

Proof. Let $L_1 \subseteq \Sigma^*$, $L_2 \subseteq \{\mathtt{u}, \mathtt{d}\}^*$ and define the F-system $\Phi = (L_1, L_2)$. If $L = L(\Phi)$ is infinite, then L_1 and L_2 are also infinite. Set $p = \max(p_1, p_2)$, where p_1 and p_2 are pumping lengths for L_1 and L_2 (from the pumping lemma for regular languages), respectively, and choose any string $w \in L$ such that $|w| \geq p$. Then, there are strings $r \in L_1$ and $s \in L_2$ such that $w = h(r, s)$, with $|r| = |s| = |w| \geq p$.

According to the pumping lemma for regular languages, r may be written as $r = x_r y_r z_r$, with $|y_r| > 0$, such that $x_r y_r^j z_r \in L_1$ for any $j \geq 0$. Similarly, s may be written as $s = x_s y_s z_s$, with $|y_s| > 0$, such that $x_s y_s^j z_s \in L_2$ for any $j \geq 0$.

Consider the sequences $\{r_j\}$ and $\{s_j\}$ defined by

$$r_j = x_r y_r^{|y_s|j+1} z_r, \tag{7}$$

$$s_j = x_s y_s^{|y_r|j+1} z_s, \tag{8}$$

for any $j \geq 0$. Clearly, $r_j \in L_1$, $s_j \in L_2$; further, $|r_j| = |s_j|$ and so each of the strings in $\{r_j\}$ may be folded according with the corresponding string in $\{s_j\}$. For clarity, let us arrange r_j and s_j in the form of a double stranded structure $\left[\frac{r_j}{s_j}\right]$, illustrated with an example in Fig. 3. Note that, for j large enough, repetitions of substring y_r overlap with repetitions of substring y_s. Therefore, we may seek expressions of r_j and s_j of the form

$$r_j = \xi_1 \xi_2^{j-j_0} \xi_3, \tag{9}$$

$$s_j = \mu_1 \mu_2^{j-j_0} \mu_3, \tag{10}$$

with substrings ξ_k and μ_k such that $|\xi_k| = |\mu_k|$ for $k = 1, 2, 3$, and for $j \geq j_0$, where j_0 is a constant to be determined.

---

[2]Sburlan's (2011) original version of the lemma states that there exists a positive constant p such that any string $w \in L$, with $|w| \geq p$, can be written as $w = uvxyz$ satisfying $uv^i xy^i z \in L$ for each $i \geq 0$, $|vy| \geq 1$, and $|vxy| \leq p$. In his proof, he set $p = \max(p_1, p_2)$ where p_1 and p_2 are the pumping lengths for the core and the folding procedure languages. However, it can be shown that such a value of p does not always work. As a simple example, set $\Phi = (L_1, L_2)$ with $L_1 = \mathtt{aaaab}^*$ and $L_2 = (\mathtt{uu})^*\mathtt{ddd}$. Then, $p = 5$. However, $L(\Phi) = \mathtt{aaaab} \cup (\mathtt{bb})^*\mathtt{aaaabbb}$, and note that string $\mathtt{aaaab}$ has length 5 but it can not be pumped as indicated by the lemma. Although the original lemma may still hold for some other value of p ($p = 9$, in case of the example, with $u = x = y = \varepsilon$, $v = \mathtt{bb}$ and $z = \mathtt{aaaabbb}$), a full demonstration was not provided.

Other inconsistencies have been detected in the proof. For example, the proof is based on constructing a double stranded structure $\left[\frac{r_j}{s_j}\right]$, and the technique is also used here. However, instead of Eqs. (7) and (8), the previous proof sets $r_j = x_r y_r^{\frac{\mathrm{lcm}(|y_r|, |y_s|)}{|y_r|}j} z_r$ and $s_j = x_s y_s^{\frac{\mathrm{lcm}(|y_r|, |y_s|)}{|y_s|}j} z_s$, where lcm denotes the least common multiple. Since the stranded construction requires $|r_j| = |s_j|$, it follows that $|x_r z_r| = |x_s z_s|$ and hence $|y_r| = |y_s|$. However, those conditions do not hold for arbitrary regular languages L_1 and L_2 (note that, in the above example, $|y_r| = |\mathtt{b}| = 1$ whereas $|y_s| = |\mathtt{uu}| = 2$).

Similar problems have been found also in the original version of the lemma for the case of $L \in \mathcal{F}(\mathsf{CF}, \mathsf{REG})$.
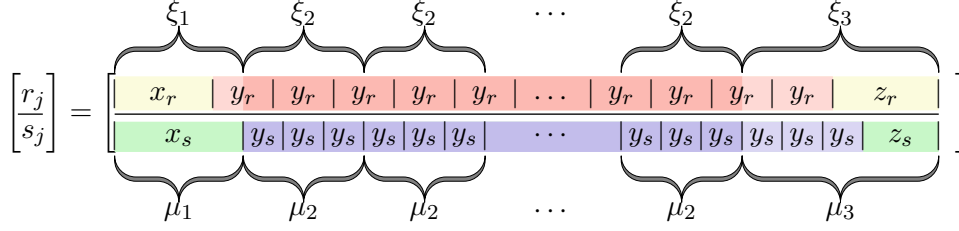
Figure 3: Double stranded structure built with strings $r_j \in L_1$ and $s_j \in L_2$, in the case in which both L_1 and L_2 are regular languages.

In Eqs. (7) and (8), the iterated portions may be decomposed as

$$y_r^{|y_s|j+1} = \alpha_1 \xi_2^{j-j_0} \alpha_2, \tag{11}$$

$$y_s^{|y_r|j+1} = \beta_1 \mu_2^{j-j_0} \beta_2, \tag{12}$$

with

$$\alpha_1 \alpha_2 = y_r^{j_0|y_s|+1}, \tag{13}$$

$$\beta_1 \beta_2 = y_s^{j_0|y_r|+1}. \tag{14}$$

Consequently, ξ_2 and μ_2 will be formed by repetitions of strings y_r and y_s, respectively, with

$$
\begin{aligned}
|\xi_2^{j-j_0}| &= |y_r^{|y_s|j+1}| - |\alpha_1 \alpha_2|, \\
&= |y_r|(|y_s|j+1) - |y_r|(j_0|y_s|+1) \\
&= |y_r||y_s|(j-j_0),
\end{aligned}
\tag{15}
$$

and the same result for $|\mu_2^{j-j0}|$. Consequently, $|\xi_2| = |\mu_2| = |y_r||y_s|$.

Next, we set $\xi_1 = x_r \alpha_1$, $\xi_3 = \alpha_2 z_r$, $\mu_1 = x_s \beta_1$, $\mu_3 = \beta_2 z_s$. Substrings α_1, α_2, β_1, β_2 are computed so that $|\xi_1| = |\mu_1|$ and $|\xi_3| = |\mu_3|$, with

$$
\begin{cases}
\beta_1 = \varepsilon, |\alpha_1| = |x_s| - |x_r|, & \text{if } |x_s| \geq |x_r|, \\
\alpha_1 = \varepsilon, |\beta_1| = |x_r| - |x_s|, & \text{if } |x_s| < |x_r|.
\end{cases}
\tag{16}
$$

and using Eqs. (13) and (14). During the calculations, constant j_0 is chosen large enough so as to produce $|\alpha_2| \geq 0$ and $|\beta_2| \geq 0$. result.

Finally, $h(r_j, s_j)$ is computed by using Eqs. (9), (10), and the identity in Eq. (4). Note that ξ_1, ξ_2 and ξ_3 are folded as determined by μ_1, μ_2, and μ_3, respectively. Then, letting $i = j - j_0$,

$$h(\xi_1 \xi_2^i \xi_3, \mu_1 \mu_2^i \mu_3) = (\xi_3)_{\mathtt{u}}^{\mathcal{R}} [(\xi_2)_{\mathtt{u}}^{\mathcal{R}}]^i (\xi_1)_{\mathtt{u}}^{\mathcal{R}} (\xi_1)_{\mathtt{d}} (\xi_2)_{\mathtt{d}}^i (\xi_3)_{\mathtt{d}}. \tag{17}$$

The condition stated by the lemma is obtained from Eq. (17) with $u = (\xi_3)_{\mathtt{u}}^{\mathcal{R}}$, $v = (\xi_2)_{\mathtt{u}}^{\mathcal{R}}$, $x = (\xi_1)_{\mathtt{u}}^{\mathcal{R}} (\xi_1)_{\mathtt{d}}$, $y = (\xi_2)_{\mathtt{d}}$, $z = (\xi_3)_{\mathtt{d}}$. Also, $|vy| = |\xi_2| = |y_r||y_s| > 0$. $\qquad \square$

The next lemma considers the two cases in which the core and the folding procedure languages belong to different classes, regular or context-free. Its demonstration follows similar steps as in Lemma 1.
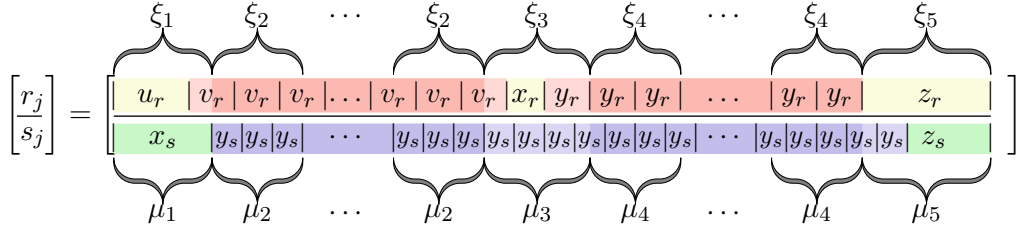
Figure 4: Double stranded structure built with strings $r_j \in L_1$ and $s_j \in L_2$, in the case in which L_1 is a context-free language and L_2 is a regular language.

Lemma 2. *Let $L \in \mathcal{F}(\mathsf{CF}, \mathsf{REG})$ or $L \in \mathcal{F}(\mathsf{REG}, \mathsf{CF})$ be an infinite language over an alphabet Σ. Then, there are strings $w_1, w_2, \ldots, w_9 \in \Sigma^*$, with $|w_2 w_4 w_6 w_8| > 0$, such that $w_1 w_2^i w_3 w_4^i w_5 w_6^i w_7 w_8^i w_9 \in L$ for all $i \geq 0$.*

Proof. Consider first the case $L \in \mathcal{F}(\mathsf{CF}, \mathsf{REG})$ and proceed similarly as in Lemma 1, except that the pumping lemma for context-free languages is used for string $r \in L_1$. Thus, r is written as $r = u_r v_r x_r y_r z_r$, with $|v_r y_r| > 0$ and $u_r v_r^j x_r y_r^j z_r \in L_1$, for any $j \geq 0$. Strings $r_j \in L_1$ and $s_j \in L_2$ are now defined as

$$r_j = u_r v_r^{|y_s|j+1} x_r y_r^{|y_s|j+1} z_r, \tag{18}$$

$$s_j = x_s y_s^{|v_r y_r|j+1} z_s, \tag{19}$$

for any $j \geq 0$, and their double stranded arrangement is illustrated in Fig. 4.

For j large enough, repetitions of y_s overlap with repetitions of v_r (if $|v_r| > 0$) and y_r (if $|y_r| > 0$). Then, we may seek expressions of r_j and s_j of the form

$$r_j = \xi_1 \xi_2^{j-j_0} \xi_3 \xi_4^{j-j_0} \xi_5, \tag{20}$$

$$s_j = \mu_1 \mu_2^{j-j_0} \mu_3 \mu_4^{j-j_0} \mu_5, \tag{21}$$

with substrings ξ_k and μ_k such that $|\xi_k| = |\mu_k|$ for $k = 1, 2, \ldots, 5$, and for $j \geq j_0$, where j_0 is a constant to be determined.

In Eqs. (18) and (19), the iterated portions are decomposed as

$$v_r^{|y_s|j+1} = \alpha_1 \xi_2^{j-j_0} \alpha_2, \tag{22}$$

$$y_r^{|y_s|j+1} = \beta_1 \xi_4^{j-j_0} \beta_2, \tag{23}$$

$$y_s^{|v_r y_r|j+1} = \gamma_1 \mu_2^{j-j_0} \gamma_2 \mu_4^{j-j_0} \gamma_3, \tag{24}$$

with

$$\alpha_1 \alpha_2 = v_r^{j_0|y_s|+1}, \tag{25}$$

$$\beta_1 \beta_2 = y_r^{j_0|y_s|+1}, \tag{26}$$

$$\gamma_1 \gamma_2 \gamma_3 = y_s^{j_0|v_r y_r|+1}, \tag{27}$$

Consequently, ξ_2 and μ_2 will be formed by repetitions of strings v_r and y_s, respectively, with $|\xi_2| = |\mu_2| = |v_r||y_s|$. Also, ξ_4 and μ_4 will be formed by repetitions of strings y_r and y_s, respectively, with $|\xi_4| = |\mu_4| = |y_r||y_s|$.

Next, we set $\xi_1 = u_r\alpha_1$, $\xi_3 = \alpha_2 x_r\beta_1$, $\xi_5 = \alpha_2 z_r$, $\mu_1 = x_s\gamma_1$, $\mu_3 = \gamma_2$, $\mu_5 = \gamma_3 z_s$. Substrings α_1, α_2, β_1, β_2, γ_1, γ_2, and γ_3 are selected so that $|\xi_1 = \mu_1|$, $|\xi_3 = \mu_3|$, and $|\xi_5 = \mu_5|$, with

$$\begin{cases} \gamma_1 = \varepsilon, |\alpha_1| = |x_s| - |u_r|, & \text{if } |x_s| \geq |u_r|, \\ \alpha_1 = \varepsilon, |\gamma_1| = |u_r| - |x_s|, & \text{if } |x_s| < |u_r|, \end{cases} \tag{28}$$

$$\begin{cases} \gamma_3 = \varepsilon, |\beta_2| = |z_s| - |z_r|, & \text{if } |z_s| \geq |z_r|, \\ \beta_2 = \varepsilon, |\gamma_3| = |z_r| - |z_s|, & \text{if } |z_s| < |z_r|, \end{cases} \tag{29}$$

and using Eqs. (25) to (27) for a value of j_0 large enough.

Finally, we compute $h(r_j, s_j)$ using Eqs. (20), (21), and the identity in Eq. (4). Then, letting $i = j - j_0$,

$$h(\xi_1\xi_2^i\xi_3\xi_4^i\xi_5, \mu_1\mu_2^i\mu_3\mu_4^i\mu_5) =$$
$$(\xi_5)_{\mathtt{u}}^{\mathcal{R}}[(\xi_4)_{\mathtt{u}}^{\mathcal{R}}]^i(\xi_3)_{\mathtt{u}}^{\mathcal{R}}[(\xi_2)_{\mathtt{u}}^{\mathcal{R}}]^i(\xi_1)_{\mathtt{u}}^{\mathcal{R}}(\xi_1)_{\mathtt{d}}(\xi_2)_{\mathtt{d}}^i(\xi_3)_{\mathtt{d}}(\xi_4)_{\mathtt{d}}^i(\xi_5)_{\mathtt{d}}. \tag{30}$$

The condition stated by the lemma is obtained from Eq. (30) with $w_1 = (\xi_5)_{\mathtt{u}}^{\mathcal{R}}$, $w_2 = (\xi_4)_{\mathtt{u}}^{\mathcal{R}}$, $w_3 = (\xi_3)_{\mathtt{u}}^{\mathcal{R}}$, $w_4 = (\xi_2)_{\mathtt{u}}^{\mathcal{R}}$, $w_5 = (\xi_1)_{\mathtt{u}}^{\mathcal{R}}(\xi_1)_{\mathtt{d}}$, $w_6 = (\xi_2)_{\mathtt{d}}$, $w_7 = (\xi_3)_{\mathtt{d}}$, $w_8 = (\xi_4)_{\mathtt{d}}$, $w_9 = (\xi_5)_{\mathtt{d}}$. Also, $|w_2 w_4 w_6 w_8| = |\xi_2\xi_4| = |v_r y_r||y_s| > 0$.

Consider next the case $L \in \mathcal{F}(\mathsf{REG}, \mathsf{CF})$. This case is treated as the previous one, except that r and s are decomposed following the pumping lemmas for regular languages and context-free languages, respectively. Thus, strings r_j and s_j are now defined as

$$r_j = x_r y_r^{|v_s y_s|j+1} z_r, \tag{31}$$

$$s_j = u_s v_s^{|y_r|j+1} x_s y_s^{|y_r|j+1} z_s, \tag{32}$$

with $|y_r| > 0$, $|v_s y_s| > 0$, and for any $j \geq 0$.

In Eqs. (31) and (32), the iterated portions are decomposed as

$$y_r^{|v_s y_r|s+1} = \alpha_1\xi_2^{j-j_0}\alpha_2\xi_4^{j-j_0}\alpha_3, \tag{33}$$

$$v_s^{|y_r|j+1} = \beta_1\mu_2^{j-j_0}\beta_2, \tag{34}$$

$$y_s^{|y_r|j+1} = \gamma_1\mu_4^{j-j_0}\gamma_2, \tag{35}$$

with

$$\alpha_1\alpha_2\alpha_3 = y_r^{j_0|v_s y_s|+1}, \tag{36}$$

$$\beta_1\beta_2 = v_s^{j_0|y_r|+1}, \tag{37}$$

$$\gamma_1\gamma_2 = y_s^{j_0|y_r|+1}. \tag{38}$$

The demonstration then follows similar steps as in the previous case: r_j and s_j are expressed as in Eqs. (20) and (21) with $\xi_1 = x_r\alpha_1$, $\xi_3 = \alpha_2$, $\xi_5 = \alpha_3 z_r$, $\mu_1 = u_s\beta_1$, $\mu_3 = \beta_2 x_s\gamma_1$, $\mu_5 = \gamma_2 z_s$. $\qquad\square$

The last lemma considers the case in which both the core and the folding procedure languages are context-free. It has a longer demonstration; nevertheless, it follows the same technique as the previous ones.
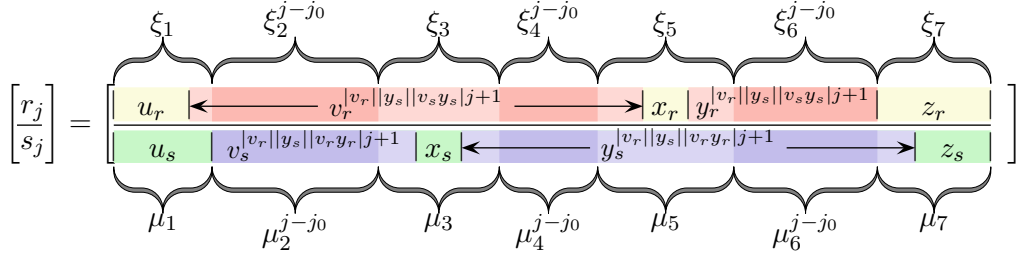
Figure 5: Double stranded structure built with strings $r_j \in L_1$ and $s_j \in L_2$, in the case in which both L_1 and L_2 are context-free languages and $|v_r||y_s| > |v_s||y_r|$.

Lemma 3. *Let $L \in \mathcal{F}(\mathsf{CF}, \mathsf{CF})$ be an infinite language over an alphabet Σ. Then, there are strings $w_1, w_2, \ldots, w_{13} \in \Sigma^*$, with $|w_2 w_4 w_6 \cdots w_{12}| > 0$, such that $w_1 w_2^i w_3 w_4^i w_5 \cdots w_{11} w_{12}^i w_{13} \in L$ for all $i \geq 0$.*

Proof. We proceed as in the previous lemmas, except that the pumping lemma for context-free languages is used for both strings $r \in L_1$ and $s \in L_2$. Thus, r is written as $r = u_r v_r x_r y_r z_r$, with $|v_r y_r| > 0$ and $r_j = u_r v_r^j x_r y_r^j z_r \in L_1$ for any $j \geq 0$, and s is written as $s = u_s v_s x_s y_s z_s$, with $|v_s y_s| > 0$ and $s_j = u_s v_s^j x_s y_s^j z_s \in L_2$ for any $j \geq 0$.

Consider first the case in which $|v_r||y_s| > |v_s||y_r|$. Strings r_j and s_j are defined as

$$r_j = u_r v_r^{|v_r||y_s||v_s y_s|j+1} x_r y_r^{|v_r||y_s||v_s y_s|j+1} z_r, \tag{39}$$

$$s_j = u_s v_s^{|v_r||y_s||v_r y_r|j+1} x_s y_s^{|v_r||y_s||v_r y_r|j+1} z_s, \tag{40}$$

for any $j \geq 0$, and their double stranded arrangement is illustrated in Fig. 5.

For j large enough, and since $|v_r||y_s| > |v_s||y_r|$, then repetitions of v_r overlap with repetitions of both v_s (if $|v_s| > 0$) and y_s, and repetitions of y_s overlap with repetitions of both v_r and y_r (if $|y_r| > 0$). Then, we may seek expressions of r_j and s_j of the form

$$r_j = \xi_1 \xi_2^{j-j_0} \xi_3 \xi_4^{j-j_0} \xi_5 \xi_6^{j-j_0} \xi_7, \tag{41}$$

$$s_j = \mu_1 \mu_2^{j-j_0} \mu_3 \mu_4^{j-j_0} \mu_5 \mu_6^{j-j_0} \mu_7, \tag{42}$$

with substrings ξ_k and μ_k such that $|\xi_k| = |\mu_k|$ for $k = 1, 2, \ldots, 7$, and for $j \geq j_0$, where j_0 is a constant to be determined.

In Eqs. (39) and (40), the iterated portions may be decomposed as

$$v_r^{|v_r||y_s||v_s y_s|j+1} = \alpha_1 \xi_2^{(j-j_0)} \alpha_2 \xi_4 (j - j_0) \alpha_3, \tag{43}$$

$$y_r^{|v_r||y_s||v_s y_s|j+1} = \beta_1 \xi_6^{(j-j_0)} \beta_2, \tag{44}$$

$$v_s^{|v_r||y_s||v_r y_r|j+1} = \gamma_1 \mu_2^{(j-j_0)} \gamma_2, \tag{45}$$

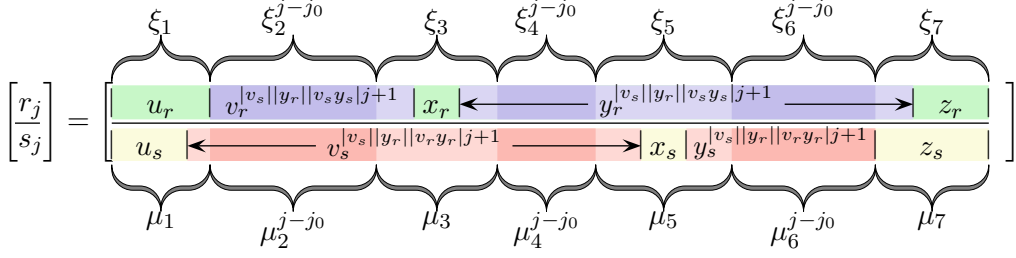$$y_s^{|v_r||y_s||v_r y_r|j+1} = \delta_1 \mu_4^{(j-j_0)} \delta_2 \mu_6^{(j-j_0)} \delta_3, \tag{46}$$

with

$$\alpha_1 \alpha_2 \alpha_3 = v_r^{j_0|v_r||y_s||v_s y_s|+1}, \tag{47}$$

$$\beta_1 \beta_2 = y_r^{j_0|v_r||y_s||v_s y_s|+1}, \tag{48}$$

$$\gamma_1 \gamma_2 = v_s^{j_0|v_r||y_s||v_r y_r|+1} \tag{49}$$

$$\delta_1 \delta_2 \delta_3 = v_s^{j_0|v_r||y_s||v_r y_r|+1}. \tag{50}$$

8

Figure 6: Double stranded structure built with strings $r_j \in L_1$ and $s_j \in L_2$, in the case in which both L_1 and L_2 are context-free languages and $|v_r||y_s| < |v_s||y_r|$.

Consequently, ξ_2 and μ_2 will be formed by repetitions of strings v_r and v_s, respectively, with $|\xi_2| = |\mu_2| = |v_r||v_s||y_s||v_r y_r|$; ξ_4 and μ_4 will be formed by repetitions of strings v_r and y_s, respectively, with $|\xi_4| = |\mu_4| = |v_r||y_s|(|v_r||y_s| - |v_s||y_r|)$; and ξ_6 and μ_6 will be formed by repetitions of strings y_r and y_s, respectively, with $|\xi_6| = |\mu_6| = |v_r||y_r||y_s||v_s y_s|$. Next, we set $\xi_1 = u_r \alpha_1$, $\xi_3 = \alpha_2$, $\xi_5 = \alpha_3 x_r \beta_1$, $\xi_7 = \beta_2 z_r$, $\mu_1 = u_s \gamma_1$, $\mu_3 = \gamma_2 x_s \delta_1$, $\mu_5 = \delta_2$, $\mu_7 = \delta_3 z_s$.

Substrings α_1, α_2, α_3, β_1, β_2, γ_1, γ_2, δ_1, δ_2 and δ_3 are selected so that $|\xi_1 = \mu_1|$, $|\xi_3 = \mu_3|$, $|\xi_5 = \mu_5|$, $|\xi_7 = \mu_7|$, with $\delta_1 = \varepsilon$, $|\alpha_2| = |\gamma_2| + |x_s|$,

$$\begin{cases} \gamma_1 = \varepsilon, |\alpha_1| = |u_s| - |u_r|, & \text{if } |u_s| \geq |u_r|, \\ \alpha_1 = \varepsilon, |\gamma_1| = |u_r| - |u_s|, & \text{if } |u_s| < |u_r|, \end{cases} \tag{51}$$

$$\begin{cases} \delta_3 = \varepsilon, |\beta_2| = |z_s| - |z_r|, & \text{if } |z_s| \geq |z_r|, \\ \beta_2 = \varepsilon, |\delta_3| = |z_r| - |z_s|, & \text{if } |z_s| < |z_r|, \end{cases} \tag{52}$$

and using Eqs. (47) to (50) for a value of j_0 large enough.

Finally, we compute $h(r_j, s_j)$ using Eqs. (41), (42), and the identity in Eq. (4). Then, letting $i = j - j_0$,

$$h(\xi_1 \xi_2^i \xi_3 \xi_4^i \xi_5 \xi_6^i \xi_7, \mu_1 \mu_2^i \mu_3 \mu_4^i \mu_5 \mu_6^i \mu_7) =$$
$$(\xi_7)_\mathsf{u}^\mathcal{R} [(\xi_6)_\mathsf{u}^\mathcal{R}]^i (\xi_5)_\mathsf{u}^\mathcal{R} [(\xi_4)_\mathsf{u}^\mathcal{R}]^i (\xi_3)_\mathsf{u}^\mathcal{R} [(\xi_2)_\mathsf{u}^\mathcal{R}]^i (\xi_1)_\mathsf{u}^\mathcal{R} (\xi_1)_\mathsf{d} (\xi_2)_\mathsf{d}^i (\xi_3)_\mathsf{d} (\xi_4)_\mathsf{d}^i (\xi_5)_\mathsf{d} (\xi_6)_\mathsf{d}^i (\xi_7)_\mathsf{d}. \tag{53}$$

The condition stated by the lemma is obtained from Eq. (53) with $w_1 = (\xi_7)_\mathsf{u}^\mathcal{R}$, $w_2 = (\xi_6)_\mathsf{u}^\mathcal{R}$, $w_3 = (\xi_5)_\mathsf{u}^\mathcal{R}$, $w_4 = (\xi_4)_\mathsf{u}^\mathcal{R}$, $w_5 = (\xi_3)_\mathsf{u}^\mathcal{R}$, $w_6 = (\xi_2)_\mathsf{u}^\mathcal{R}$, $w_7 = (\xi_1)_\mathsf{u}^\mathcal{R}(\xi_1)_\mathsf{d}$, $w_8 = (\xi_2)_\mathsf{d}$, $w_9 = (\xi_3)_\mathsf{d}$, $w_{10} = (\xi_4)_\mathsf{d}$, $w_{11} = (\xi_5)_\mathsf{d}$, $w_{12} = (\xi_6)_\mathsf{d}$, $w_{13} = (\xi_7)_\mathsf{d}$. Also, $|w_2 w_4 w_6 w_8 w_{10} w_{12}| = |\xi_2 \xi_4 \xi_6| = |v_r||y_s||v_r y_r||v_s y_s| > 0$.

Consider next the case in which $|v_r||y_s| < |v_s||y_r|$, and define

$$r_j = u_r v_r^{|v_s||y_r||v_s y_s|j+1} x_r y_r^{|v_s||y_r||v_s y_s|j+1} z_r, \tag{54}$$

$$s_j = u_s v_s^{|v_s||y_r||v_r y_r|j+1} x_s y_s^{|v_s||y_r||v_r y_r|j+1} z_s, \tag{55}$$

for any $j \geq 0$, with the double stranded arrangement illustrated in Fig. 6.

We seek expressions of r_j and s_j of the form in Eqs. (41) and (42), and decompose

the iterated portions in Eqs. (54) and (55) as

$$v_r^{|v_s||y_r||v_sy_s|j+1} = \alpha_1 \xi_2^{(j-j_0)} \alpha_2, \tag{56}$$

$$y_r^{|v_s||y_r||v_sy_s|j+1} = \beta_1 \xi_4^{(j-j_0)} \beta_2 \xi_6^{(j-j_0)} \beta_3, \tag{57}$$

$$v_s^{|v_s||y_r||v_ry_r|j+1} = \gamma_1 \mu_2^{(j-j_0)} \gamma_2 \mu_4^{(j-j_0)} \gamma_3, \tag{58}$$

$$y_s^{|v_s||y_r||v_ry_r|j+1} = \delta_1 \mu_6^{(j-j_0)} \delta_2, \tag{59}$$

with

$$\alpha_1 \alpha_2 = v_r^{j_0|v_s||y_r||v_sy_s|+1}, \tag{60}$$

$$\beta_1 \beta_2 \beta_3 = y_r^{j_0|v_s||y_r||v_sy_s|+1}, \tag{61}$$

$$\gamma_1 \gamma_2 \gamma_3 = v_s^{j_0|v_s||y_r||v_ry_r|+1}, \tag{62}$$

$$\delta_1 \delta_2 = v_s^{j_0|v_s||y_r||v_ry_r|+1}. \tag{63}$$

Consequently, ξ_2 and μ_2 will be formed by repetitions of strings v_r and v_s, respectively, with $|\xi_2| = |\mu_2| = |v_r||v_s||y_r||v_sy_s|$; ξ_4 and μ_4 will be formed by repetitions of strings y_r and v_s, respectively, with $|\xi_4| = |\mu_4| = |y_r||v_s|(|v_s||y_r| - |v_r||y_s|)$; and ξ_6 and μ_6 will be formed by repetitions of strings y_r and y_s, respectively, with $|\xi_6| = |\mu_6| = |y_s||v_s||y_r||v_ry_r|$. Next, we set $\xi_1 = u_r\alpha_1$, $\xi_3 = \alpha_2 x_r \beta_1$, $\xi_5 = \beta_2$, $\xi_7 = \beta_3 z_r$, $\mu_1 = u_s \gamma_1$, $\mu_3 = \gamma_2$, $\mu_5 = \gamma_3 x_s \delta_1$, $\mu_7 = \delta_2 z_s$.

Substrings α_1, α_2, α_3, β_1, β_2, γ_1, γ_2, δ_1, δ_2 and δ_3 are selected so that $|\xi_1 = \mu_1|$, $|\xi_3 = \mu_3|$, $|\xi_5 = \mu_5|$, $|\xi_7 = \mu_7|$, with $\gamma_3 = \varepsilon$, $|\beta_2| = |\delta_1| + |x_s|$,

$$\begin{cases} \delta_2 = \varepsilon, |\beta_3| = |z_s| - |z_r|, & \text{if } |z_s| \geq |z_r|, \\ \beta_3 = \varepsilon, |\delta_2| = |z_r| - |z_s|, & \text{if } |z_s| < |z_r|, \end{cases} \tag{64}$$

and using Eq. (51) and Eqs. (60) to (63), for a value of j_0 large enough.

The demonstration then follows similar steps as in the previous case.

Finally, consider the remaining case of $|v_r||y_s| = |v_s||y_r|$. If $|v_r||y_s| \neq 0$ and $|v_s||y_r| \neq 0$, then the same constructions of the previous cases work, resulting in $\xi_4 = \mu_4 = \varepsilon$ and therefore $w_4 = w_{10} = \varepsilon$.

If $|v_r||y_s| = |v_s||y_r| = 0$, then either $v_r = v_s = \varepsilon$ or $y_r = y_s = \varepsilon$ (recall that the pumping lemma for context-free languages demands $|v_ry_r| > 0$ and $|v_sy_s| > 0$). Assume first $v_r = v_s = \varepsilon$, and define

$$r_j = u_r x_r y_r^{|y_s|j+1} z_r, \tag{65}$$

$$s_j = u_s x_s y_s^{|y_r|j+1} z_s, \tag{66}$$

for any $j \geq 0$. Next, follow the same procedure as in Lemma 1 to express r_j and s_j in the form of Eqs. (9) and (10), with the exception that $\xi_1 = u_r x_r \alpha_1$ and $\mu_1 = u_s x_s \beta_1$ (instead of $\xi_1 = x_r \alpha_1$ and $\mu_1 = x_s \beta_1$, respectively). Then, $h(r_j, s_j)$ is given by Eq. (17) and the condition stated by the present lemma is obtained with $w_1 = (\xi_3)_\mathtt{u}^\mathcal{R}$, $w_2 = (\xi_2)_\mathtt{u}^\mathcal{R}$, $w_3 = w_4 = w_5 = w_6 = \varepsilon$, $w_7 = (\xi_1)_\mathtt{u}^\mathcal{R}(\xi_1)_\mathtt{d}$, $w_8 = w_9 = w_{10} = w_{11} = \varepsilon$, $w_{12} = (\xi_2)_\mathtt{d}$, $w_{13} = (\xi_3)_\mathtt{d}$. Also, $|w_2w_4w_6w_8w_{10}w_{12}| = |\xi_2| = |y_r||y_s| \geq 1$.

If $y_r = y_s = \varepsilon$, proceed as above with

$$r_j = u_r v_r^{|v_s|j+1} x_r z_r, \tag{67}$$

$$s_j = u_s v_s^{|v_r|j+1} x_s z_s, \tag{68}$$

for any $j \geq 0$. $\qquad\square$

4 Final remarks and example

The lemmas only apply to infinite languages. However, any finite language is in class $\mathcal{F}(\mathsf{REG}, \mathsf{REG})$. Let L_1 be any arbitrary language and define $\Phi = (L_1, \mathsf{d}^*)$; then, $L_1 = L(\Phi)$. Therefore, if L_1 finite then it is regular and, consequently, $L_1 \in \mathcal{F}(\mathsf{REG}, \mathsf{REG})$. Further, and since $\mathsf{REG} \subset \mathsf{CF}$, then L_1 also is in $\mathcal{F}(\mathsf{CF}, \mathsf{REG})$, $\mathcal{F}(\mathsf{REG}, \mathsf{CF})$, and $\mathcal{F}(\mathsf{CF}, \mathsf{CF})$.

In spite of the lemmas being weak, they still are useful to prove non membership of some languages in a class, as the following example shows.

Example 2. Consider $L = \{\mathsf{a}^n \,|\, n \text{ is prime }\}$ and assume that L satisfies the lemma. Then, there are strings $w_1, w_2, \ldots, w_{13} \in \mathsf{a}^*$, with $|w_2 w_4 w_6 \cdots w_{12}| > 0$, such that $w_1 w_2^i w_3 w_4^i w_5 \cdots w_{11} w_{12}^i w_{13} \in L$ for all $i \geq 0$. Letting $i = 1$, we have that $w_1 w_2 w_3 w_4 w_5 \cdots w_{11} w_{12} w_{13} = \mathsf{a}^k$ for some prime k. Then, $w_2 w_4 w_6 \cdots w_{12} = \mathsf{a}^\ell$ with $0 < \ell \leq k$. Next, letting $i = k + 1$, we obtain $w_1 w_2^i w_3 w_4^i w_5 \cdots w_{11} w_{12}^i w_{13} = \mathsf{a}^{k(1+\ell)}$. However, this string is not in L_2, because the number of a's is not a prime. The contradiction implies that $L_2 \notin \mathcal{F}(\mathsf{CF}, \mathsf{CF})$.

Acknowledgements

References

Akitaya, H. A., Demaine E. D., and Ku J. S. (2017). "Simple folding is really hard," *J. Inform. Process.* 25, 580–589.

Alperin, R. C. (2000). "A mathematical theory of origami constructions and numbers," *New York J. Math.* 6, 119–133.

Cipra, B. (2001). "In the fold: Origami meets mathematics," *SIAM News* 34, 1–4.

Demaine, E. D. and O'Rourke, J. (2007). *Geometric Folding Algorithms: Linkages, Origami, Polyhedra. Cambridge* (University Press, New York), pp. 167–298.

Dobson, C. M. (2003). "Protein folding and misfolding," *Nature* 426, 884–890.

Felton, S., Tolley, M., Demaine, E., Rus, D., and Wood, R. (2014). "A method for building self-folding machines," *Science* 345, 644–646.

Filipov, E. T., Tachi, T., and Paulino, G. H. (2015). "Origami tubes assembled into stiff, yet reconfigurable structures and metamaterials," *Proc. Natl. Acad. Sci. USA* 112, 12321–12326.

Hopcroft, J. E., Motwani R., and Ullman, J. D. (2001). *Introduction to Automata Theory, Languages and Computation*, 2nd edn. (Addison-Wesley, New York), pp. 126–127 and 275–276.

Ida, T., Fleuriot, J., and Ghourabi, F. (2016). "A new formalization of origami in geometric algebra," in *Proceedings of ADG 2016: 11th International Workshop on Automated Deduction in Geometry*, edited by J. Narboux, P. Schreck, and I. Streinu (Strasbourg, France), pp. 117–136.

Kari, L. and Rozenberg, G. (2008). "The many facets of natural computing," *Commun. ACM* 51, 72–83.

Mahadevan, L. and Rica, S. (2005) . "Self-organized origami," *Science* 307, 1740–1740.

Rothemund, P. W. (2006). "Folding DNA to create nanoscale shapes and patterns," *Nature* 440, 297.

Sburlan, D. (2011). "Computing by folding," *Int. J. Comput. Commun. Controls* 6, 739–748.