On the index of Simon’s congruence for piecewise testability

This item was submitted to Loughborough University’s Institutional Repository by the/an author.


Additional Information:

- This paper was accepted for publication in the journal Information Processing Letters and the definitive published version is available at https://doi.org/10.1016/j.ipl.2014.11.008

Metadata Record: https://dspace.lboro.ac.uk/2134/31967

Version: Accepted for publication

Publisher: © Elsevier

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: https://creativecommons.org/licenses/by-nc-nd/4.0/

Please cite the published version.
On the index of Simon’s congruence for piecewise testability

P. Karandikar\textsuperscript{a,b,1,2}, M. Kufleitner\textsuperscript{c,3}, Ph. Schnoebelen\textsuperscript{a,2}

\textsuperscript{a}Lab. Specification & Verification, CNRS UMR 8643 & ENS Cachan, France
\textsuperscript{b}Chennai Mathematical Institute, Chennai, India
\textsuperscript{c}Institut für Formale Methoden der Informatik, University of Stuttgart, Germany

Abstract

Simon’s congruence, denoted ∼\textsubscript{n}, relates words having the same subwords of length up to \textit{n}. We show that, over a \textit{k}-letter alphabet, the number of words modulo ∼\textsubscript{n} is in 2\textsuperscript{k(n−1) log \textit{n}}.

\textbf{Keywords:} Combinatorics of words; Piecewise testable languages; Subwords and subsequences.

1. Introduction

Piecewise testable languages, introduced by Imre Simon in the 1970s, are a family of star-free regular languages that are definable by the presence and absence of given (scattered) subwords \textsuperscript{1}. Formally, a language \(L \subseteq A^*\) is \textit{n}-piecewise testable if \(x \in L\) and \(x \sim \textsubscript{n} y\) imply \(y \in L\), where \(x \sim \textsubscript{n} y\) iff \(x\) and \(y\) have the same subwords of length at most \(n\) (see next section for all definitions missing in this introduction). Piecewise testable languages are important because they are the languages defined by \(B\Sigma_1\) formulae, a simple fragment of first-order logic that is prominent in database queries \textsuperscript{4}. They also occur in learning theory \textsuperscript{3}, computational linguistics \textsuperscript{2}, etc.

It is easy to see that \(\sim\textsubscript{n}\) is a congruence with finite index and Sakarovitch and Simon raised the question of how to better characterize or evaluate this number \textsuperscript{2} p. 110. Let us write \(C_{k}(n)\) for the number of \(\sim\textsubscript{n}\) classes over \(k\) letters, i.e., when \(|A| = k\). It is clear that \(C_{k}(n) \geq k^n\) since two words \(x, y \in A^*\) (i.e., of length at most \(n\)) are related by \(\sim\textsubscript{n}\) only if they are equal. In fact, this reasoning gives

\[
C_{k}(n) \geq k^n + k^{n-1} + \cdots + k + 1 = \frac{k^{n+1} - 1}{k - 1} \quad (1)
\]

assuming \(k \neq 1\). On the other hand, any congruence class in \(\sim\textsubscript{n}\) is completely characterized by a set of subwords in \(A^\leq n\), hence

\[
C_{k}(n) \leq 2^{\frac{k^{n+1} - 1}{k - 1}}. \quad (2)
\]

Estimating the size of \(C_{k}(n)\) has applications in descriptive complexity, for example for estimating the number of \(n\)-piecewise testable languages (over a given alphabet), or for bounding the size of canonical automata for \(n\)-piecewise testable languages \textsuperscript{2,3,4}.

Unfortunately the above bounds, summarized as \(k^n \leq C_{k}(n) \leq 2^{k(n-1)\log n}\), leave a large (“exponential”) gap and it is not clear towards which side is the actual value leaning. Eq. \textsuperscript{4} gives a lower bound that is obviously very naive since it only counts the simplest classes. On the other hand, Eq. \textsuperscript{2} too makes wide simplifications since not every subset of \(A^\leq n\) corresponds to a congruence class. For example, if \(aa\) and \(bb\) are subwords of some \(x\) then necessarily \(x\) also has \(ab\) or \(ba\) among its length 2 subwords.

Since the question of estimating \(C_{k}(n)\) was raised in \textsuperscript{2} (and to the best of our knowledge) no progress has been made on the question, until Kátaí-Urbán et al. proved the following bounds:

\textbf{Theorem 1.1 (Kátaí-Urbán et al. \textsuperscript{10})}. For all \(k > 1\),

\[
\frac{k^n}{3^n} \log k \leq \log C_{k}(n) < 3^n k^n \log k \quad \text{if } n \text{ is even},
\]

\[
\frac{k^n}{3^n} < \log C_{k}(n) < 3^n k^n \quad \text{if } n \text{ is odd}.
\]

The proof is based on two reductions, one showing \(C_{k+1}(n + 2) \geq C_{k}^{k+2}(n)\) for proving lower bounds, and one showing \(C_{k}(n + 2) \leq (k + 1)^2 C_{k}^{2k-1}(n)\) for proving upper bounds. For fixed \(n\), Theorem \textsuperscript{11} allows to estimate the asymptotic value of \(\log C_{k}(n)\) as a function of \(k\): it is in \(\Theta(k^n)\) or \(\Theta(k^n \log k)\) depending on the parity of \(n\). However, these bounds do not say how, for fixed \(k\), \(C_{k}(n)\) grows as a function of \(n\), which is a more natural question in settings where the alphabet is fixed, and where \(n\) comes from, e.g., the number of variables in a \(B\Sigma_1\) formula. In particular, the lower bound is useless for \(n \geq k\) since in this case \(k^n/3^n < 1\).

\textsuperscript{4}Comparing the bounds from Eqs. \textsuperscript{1} and \textsuperscript{2} with actual values does not bring much light here since the magnitude of \(C_{k}(n)\) makes it hard to compute beyond some very small values of \(k\) and \(n\), see Table \textsuperscript{5,1}.

\textsuperscript{1}Partially supported by Tata Consultancy Services.
\textsuperscript{2}Supported by ANR grant 11-BS02-001-01.
\textsuperscript{3}Supported by DFG grant DI 435/5-2.
Our contribution. In this article, we provide the following bounds:

**Theorem 1.2.** For all $k, n > 1$,

$$\left(\frac{n}{k}\right)^{k-1} \log_2 \left(\frac{n}{k}\right) < \log_2 C_k(n) < k \left(\frac{n + 2k - 3}{k - 1}\right)^{k-1} \log_2 n \log_2 k.$$

Thus, for fixed $k$, $\log C_k(n)$ is in $\Theta(n^{k-1} \log n)$. Compared with Theorem 1.1, our bounds are much tighter for fixed $k$ (and much wider for fixed $n$).

The proof of Theorem 1.2 relies on two new reductions that allows us to relate $C_k(n)$ with $C_{k-1}$ instead of relating it with $C_k(n-2)$ as in [10]. Our contribution is as follows. Section 2 recalls the necessary notations and definitions; the lower bound is proved in Section 3 while the upper bound is proved in Section 4. An appendix lists the exact values of $C_k(n)$ for small $n$ and $k$ that we managed to compute.

2. Basics

We consider words $x, y, w, \ldots$ over a finite $k$-letter alphabet $A_k = \{a_1, \ldots, a_k\}$ sometimes written more simply $A = \{a, b, \ldots\}$. The empty word is denoted $\epsilon$, concatenation is denoted multiplicatively. Given a word $x \in A^*$ and a letter $a \in A$, we write $|x|$ and $|x|_a$ for, respectively, the length of $x$, and the number of occurrences of $a$ in $x$.

We write $x \preceq y$ to denote that a word $x$ is a subsequence of $y$, also called a (scattered) subword. Formally, $x \preceq y$ if $x = x_1 \cdots x_r$ and there are words $y_0, y_1, \ldots, y_r$ such that $y = y_0 x_1 y_1 \cdots x_r y_r$. It is well-known that $\preceq$ is a partial ordering and a monoid precongruence.

For any $n \in \mathbb{N}$, we write $x \sim_n y$ when $x$ and $y$ have the same subsequences of length $\leq n$. For example $x \overset{\text{def}}{=} \text{abacbab} \sim_2 y \overset{\text{def}}{=} \text{baaaacbb}$ since both words have $\{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc, cb\}$ as subsequences of length $\leq 2$. However $x \not\sim_3 y$ since $x \ni a \not\ni b$. Note that $\sim_0 \supseteq \sim_1 \supseteq \sim_2 \supseteq \cdots$, and that $x \sim_0 y$ holds trivially. It is well-known (and easy to see) that each $\sim_n$ is a congruence since the subsequences of some $xy$ are the concatenations of a subword of $x$ and a subword of $y$. Simon defined a piecewise testable language as any $L \subseteq A^*$ that is closed by $\sim_n$ for some $n \in \mathbb{N}$ [11]. These are exactly the languages definable by $BS_\Sigma^+((<, a, b, \ldots))$ formulae [8], i.e., by Boolean combinations of existential first-order formulae with monadic predicates of the form $a(i)$, stating that the $i$-th letter of a word is $a$. For example, $L = A^*aA^*bA^* = \{x \in A^* \mid ab \preceq x\}$ is definable with the following $\Sigma_1$ formula:

$$\exists i : \exists j : i < j \land a(i) \land b(j).$$

The index of $\sim_n$. Since there are only finitely many words of length $\leq n$, the congruence $\sim_n$ partitions $A_k^n$ in finitely many classes, and we write $C_k(n)$ for the number of such classes, i.e., the cardinal of $A_k^n / \sim_n$.

The following is easy to see:

$$C_1(n) = n + 1, \quad C_k(0) = 1, \quad C_k(1) = 2^k.$$

Indeed, for words over a single letter $a$, $x \sim_n y$ iff $|x| = |y| < n$ or $|x| \geq n \leq |y|$, hence the first equality. The second equality restates that $\sim_1$ is trivial, as noted above. For the third equality, one notes that $x \sim_1 y$ if, and only if, the same set of letters is occurring in $x$ and $y$, and that there are $2^n$ such sets of occurring letters.

3. Lower bound

The first half of Theorem 1.2 is proved by first establishing a combinatorial inequality on the $C_k(n)$’s (Proposition 3.3) and then using it to derive Proposition 3.4.

Consider two words $x, y \in A^*$ and a letter $a \in A$.

**Lemma 3.1.** If $x \sim_n y$, then $\min(|x|_a, n) = \min(|y|_a, n)$.

**Proof (Sketch).** If $|x|_a = p < n$ then $a^p \preceq x \preceq a^{p+1}$. From $x \sim_n y$ we deduce $a^p \preceq y \preceq a^{p+1}$, hence $|y|_a = p$. □

Fix now $k \geq 2$, let $A = A_k = \{a_1, \ldots, a_k\}$ and assume $x \sim_n y$. If $|x|_{a_k} = p < n$, then $x$ is some $x_0 a_k x_1 \cdots a_k x_p$ with $x_i \in A_k^{p-1}$ for $i = 0, \ldots, p$. By Lemma 3.1 $y$ too is some $y_0 a_k y_1 \cdots a_k y_p$ with $y_i \in A_k^{p-1}$.

**Lemma 3.2.** $x_i \sim_{n-p} y_i$ for all $i = 0, \ldots, p$.

**Proof.** Suppose $w \preceq x_i$ and $|w| \leq n-p$. Let $w' \overset{\text{def}}{=} a_k w a_k^{-1}$. Clearly $w' \preceq x$ and thus $w' \preceq y$ since $x \sim_n y$ and $|w'| \leq n$. Now $w' = a_k w a_k^{-1} \preceq y$ entails $w \preceq y$.

With a symmetric reasoning we show that every subword of $y_i$ having length $\leq n-p$ is a subword of $x_i$ and we conclude $x_i \sim_{n-p} y_i$. □

**Proposition 3.3.** For $k \geq 2$, $C_k(n) \geq \sum_{p=0}^n C_{k-1}^{p+1}(n-p)$.

**Proof.** For words $x = x_0 a_k x_1 \cdots x_{p-1} a_k x_p$ with exactly $p \leq n$ occurrences of $a_k$, we have $C_{k-1}(n-p)$ possible choices of $\sim_{n-p}$ equivalence classes for each $x_i$ ($i = 0, \ldots, p$). By Lemma 3.2 all such choices will result in $n$ words, hence there are exactly $C_{k-1}^{p+1}(n-p)$ classes of words with $p \leq n$ occurrences of $a_k$. By Lemma 3.1 these classes are disjoint for different values of $p$, hence we can add the $C_{k-1}^{p+1}(n-p)$’s. There remain words with $p \geq n$ occurrences of $a_k$, accounting for at least 1, i.e., $C_{k-1}^{n+1}(0)$, additional class. □

**Proposition 3.4.** For all $k, n > 0$:

$$\log_2 C_k(n) > \left(\frac{n}{k}\right)^{k-1} \log_2 \left(\frac{n}{k}\right).$$
Proof. Eq. (4) holds trivially when \( \log_2 \left( \frac{p}{k} \right) \leq 0 \). Hence there only remains to consider the cases where \( n > k \). We reason by induction on \( k \). For \( k = 1 \), Eq. (3) yields \( \log_2 C_1(n) = \log_2(n+1) > \log_2 n = \left( \frac{n}{k} \right) \log_2 \left( \frac{n}{k} \right) \). For the inductive case, Proposition 4.3 yields \( \log_2 C_{k+1}(n) \geq \log_2 C_k(n-p) \) for all \( p \in \{0, \ldots, n\} \). For \( p = \left\lfloor \frac{n}{k+1} \right\rfloor \) this yields

\[
\log_2 C_{k+1}(n) \geq (p+1) \log_2 C_k(n-p) \geq (p+1) \left( \frac{n-p}{k} \right)^{-1} \log_2 \left( \frac{n-p}{k} \right) \geq n \left( \frac{n}{k+1} \right) ^{-1} \log_2 \left( \frac{n}{k+1} \right)
\]

by ind. hyp., noting that \( n > k > 0 \), since \( \frac{n-p}{k+1} \geq 1 \),

\[
= \left( \frac{n}{k+1} \right) \log_2 \left( \frac{n}{k+1} \right)
\]
as desired. \( \Box \)

4. Upper bound

The second half of Theorem 4.2 is again by establishing a combinatorial inequality on the \( C_k(n) \)'s (Proposition 4.3) and then using it to derive Proposition 4.4.

Fix \( k > 0 \) and consider words in \( A_k^n \). We say that a word \( x \) is rich if all the \( k \) letters of \( A_k \) occur in it, and that it is poor otherwise. For \( \ell > 0 \), we further say that \( x \) is \( \ell \)-rich if it can be written as a concatenation of \( \ell \) rich factors (by extension \( x \) is \( 0 \)-rich if \( x \) is poor). The richness of \( x \) is the largest \( \ell \in \mathbb{N} \) such that \( x \) is \( \ell \)-rich. Note that \( \forall a \in A_k : |x|_a \geq \ell \) does not imply that \( x \) is \( \ell \)-rich. We shall follow the easy following result:

**Lemma 4.1.** If \( x_1 \) and \( x_2 \) are respectively \( \ell_1 \)-rich and \( \ell_2 \)-rich, then \( y \sim_n y' \) implies \( x_1 y_2 \sim_\ell x_1 + \ell_1 x_1 y' \).

**Proof.** A subword \( u \) of \( x_1 y_2 x_2 \) can be decomposed as \( u = u_1 v u_2 \) where \( u_1 \) is the largest prefix of \( u \) that is a subword of \( x \) and \( u_2 \) is the largest suffix of the remaining \( u_1 \)-u that is a subword of \( x_2 \). Thus \( u \leq y \) since \( u \leq x_1 y_2 x_2 \). Now, since \( x_1 \) is \( \ell_1 \)-rich, \( |u_1| \geq \ell_1 \) (unless \( u \) is too short), and similarly \( |u_2| \geq \ell_2 \) (unless \( \ldots \). Finally \( |u| \leq n \) when \( |u| \leq \ell_1 + n + \ell_2 \), and then \( u \leq y' \) since \( y \sim_n y' \), entailing \( u \leq x_1 y' x_2 \). A symmetrical reasoning shows that subwords of \( x_1 y' x_2 \) of length \( \leq \ell_1 + n + \ell_2 \) are subwords of \( x_1 y_2 x_2 \) and we are done. \( \Box \)

The rich factorization of \( x \in A_k^n \) is the decomposition \( x = x_1 a_1 \ldots a_m a_{n+1}y \) obtained in the following way: if \( x \) is poor, we let \( m = 0 \) and \( y = x \); otherwise \( x \) is rich, we let \( x_1 a_1 \) (with \( a_1 \in A_k \)) be the shortest prefix of \( x \) that is rich, write \( x = x_1 a_1 x' \) and let \( x_2 a_2 \ldots a_m a_{n+1}y \) be the rich factorization of the remaining suffix \( x' \). By construction \( m \) is the richness of \( x \). E.g., assuming \( k = 3 \), the following is a rich factorization with \( m = 2 \):

\[
\text{bbababccccaaabbbaa} = \text{x}_1 \text{x}_2 \ 	ext{c} \ 	ext{ccaa} \cdot \text{b} \ 	ext{bbaa}
\]

Note that, by definition, \( x_1, \ldots, x_n \) and \( y \) are poor.

**Lemma 4.2.** Consider two words \( x, x' \) of richness \( m \) and with rich factorizations \( x = x_1 a_1 \ldots x_m a_my \) and \( x' = x'_1 a'_1 \ldots x'_{m'} a'_{m'}y' \). Suppose that \( y \sim_n y' \) and that \( x_i \sim_{n+1} x'_i \) for all \( i = 1, \ldots, m \). Then \( x \sim_{n+m} x' \).

**Proof.** By repeatedly using Lemma 4.1 one shows

\[
x_1 a_1 x_2 a_2 \ldots x_m a_m y \sim_{n+m} x'_1 a'_1 x'_2 a'_2 \ldots x'_{m'} a'_{m'} y'
\]

\[
\sim_{n+m} x'_1 a'_1 x'_2 a'_2 \ldots x'_{m'} a'_{m'} y'
\]

using the fact that each factor \( x_i, a_i \) is rich. \( \Box \)

**Proposition 4.3.** For all \( n \geq 0 \) and \( k \geq 2 \),

\[
C_k(n) \leq 1 + \sum_{m=0}^{n-1} k^{m+1} C^m_{k-1}(n-m+1) C_{k-1}(n-m)
\]

Furthermore, for \( k = 2 \),

\[
C_2(n) \leq 2 n^m = \frac{2^{2n-1}}{n-1}.
\]

**Proof.** Consider two words \( x, x' \) and their rich factorization \( x = x_1 a_1 \ldots x_m a_m y \) and \( x' = x'_1 a'_1 \ldots x'_{m'} a'_{m'} y' \). By Lemma 4.1 they belong to the same \( \sim_n \) class if \( m = y \sim_n y' \), and \( a_i = a'_i \) and \( x_i \sim_{n+1} x'_i \) for all \( i = 1, \ldots, m \). Now for every fixed \( m \), there are at most \( k^m \) choices for \( a_i \)'s, \( C^m_{k-1}(n-m+1) \) non-equivalent choices for the \( x_i \)'s, \( C_k(n-m) \) choices for \( y \) and a letter that is missing in it. We only need to consider \( m \) varying up to \( n-1 \) since all words of richness \( \geq n \) are \( \sim_n \)-equivalent, accounting for one additional possible \( \sim_n \) class.

For the second inequality, assume that \( k = 2 \) and \( A_2 = \{a, b\} \). A word \( x \in A_2^n \) can be decomposed as a sequence of \( m \) non-empty blocks of the same letter, of the form, e.g., \( x = a^{k_1} b^{k_2} a^{k_3} b^{k_4} \ldots a^{k_m} \) (this example assumes that \( x \) starts and ends with \( a \), hence \( m \) is odd). If two words like \( x = a^{k_1} b^{k_2} a^{k_3} b^{k_4} \ldots a^{k_m} \) and \( x' = a^{k'_1} b^{k'_2} a^{k'_3} b^{k'_4} \ldots a^{k'_m} \) have the same first letter \( a \), the same alternation depth \( m \), and have min\((\ell_i, n) = \min(\ell_i, n) \) for all \( i = 1, \ldots, m \), then they are \( \sim_n \)-equivalent. For a given \( m > 0 \), there are \( 2 \) possibilities for choosing the first letter and \( n^m \) non-equivalent choices for the \( \ell_i \)'s. Finally, all words with alternation depths \( m \geq 2n \) are \( \sim_n \)-equivalent, hence we can restrict our attention to \( 1 \leq m \leq 2n - 1 \). The extra summand \( 2 n^m \) in Eq. (5) accounts for the single class with \( m \geq 2n \) and the single class with \( m = 0 \). \( \Box \)
Proposition 4.4. For all $k, n > 1$:

$$C_k(n) < 2^k \left( \frac{n+2k-3}{k} \right)^{k-1} \log_2 n \log_2 k.$$ 

PROOF. By induction on $k$. For $k = 2$, Eq. (5) yields:

$$C_2(n) \leq 2^n - 1 < n^{2n+1} - 1$$

since $n \geq 2$,

$$= n^{2n+2} \leq 2^{2(n+1)} \log_2 n$$

$$= 2^k \left( \frac{n+2k-3}{k} \right)^{k-1} \log_2 n \log_2 k.$$ 

For the inductive case, Proposition 4.3 yields:

$$C_{k+1}(n) \leq n \left( k + 1 \right) C_k(n) + \sum_{m=1}^{n-1} (k+1)^{m+1} C_k(n - m)$$

$$= 1 + (k+1) C_k(n)$$

$$+ \sum_{m=1}^{n-1} (k+1)^{m+1} C_k(n - m)$$

$$< (k+1)^n C_k(n) + \sum_{m=1}^{n-1} (k+1)^n C_k^{m+1}(n - m + 1)$$

since $C_k(q) \leq C_k(q+1)$,

$$= (k+1)^n \sum_{m=0}^{n-1} 2^{k(m+1)} \left( \frac{n-m+2k-2}{k} - 1 \right)^{k-1} \log_2 n \log_2 k$$

by in. hyp.,

$$< (k+1)^n \sum_{m=0}^{n-1} 2^{k(m+1)} \left( \frac{n-m+2k-2}{k} - 1 \right)^{k-1} \log_2 n \log_2 k.$$ 

Since $(m + 1) \left( \frac{n-m+2k-2}{k} \right)^{k-1} \leq (n+2k-1)^k$ for all $m \in \{0, \ldots, n-1\}$—see Appendix A—, we may proceed with:

$$C_{k+1}(n) < (k+1)^n \sum_{m=0}^{n-1} 2^k \left( \frac{n+2k-1}{k} \right)^k \log_2 n \log_2 k$$

$$= n(k+1)^n 2^k \left( \frac{n+2k-1}{k} \right)^k \log_2 n \log_2 k$$

$$= 2 \log_2 n \log_2 \log_2 (k+1) + k \left( \frac{n+2k-1}{k} \right) \log_2 n \log_2 k$$

$$< 2 \left( \log_2 n + n \log_2 (k+1) \right) \left( \frac{n+2k-1}{k} \right) \log_2 n \log_2 (k+1)$$

$$< (k+1)^n \left( \frac{n+2k-1}{k} \right) \log_2 n \log_2 (k+1)$$

since $\log_2 n + n < \left( \frac{n+2k-1}{k} \right) \log_2 n$ (see below). This is the desired bound.

To see that $\log_2 n + n < \left( \frac{n+2k-1}{k} \right) \log_2 n$, we use

$$\left( \frac{n + 2k - 1}{k} \right)^k > \left( \frac{n + 1}{k} \right)^k = \sum_{j=0}^{k} \binom{k}{j} \left( \frac{n}{k} \right)^j$$

$$= 1 + k \cdot \left( \frac{n}{k} \right) + \cdots + n + 1.$$ 

This completes the proof. □

By combining the two bounds in Propositions 3.4 and 4.3 we obtain Theorem 1.2 and 4.1 implying that $\log C_k(n)$ is in $\Theta\left(n^{k-1} \log n \right)$ for fixed alphabet size $k$.

5. Conclusion

We proved that, over a fixed $k$-letter alphabet, $C_k(n)$ is in $2^\Theta(n^{k-1} \log n)$. This shows that $C_k(n)$ is not doubly exponential in $n$ as Eq. (2) and Theorem 1.1 would allow. It also is not simply exponential, bounded by a term of the form $2^g(k)^{n^c}$ where the exponent $c$ does not depend on $k$.

We are still far from having a precise understanding of how $C_k(n)$ behaves and there are obvious directions for improving Theorem 1.2. For example, its bounds are not monotonic in $k$ (while the bounds in Theorem 1.1 are not monotonic in $n$) and it only partially uses the combinatorial inequalities given by Propositions 3.3 and 4.3.

Acknowledgments. We thank J. Berstel, J.-É. Pin and M. Zeitoun for their comments and suggestions.

References


Appendix A. Additional proofs

We prove that \((m + 1) \left( \frac{n - m + 2k - 2}{k - 1} \right)^{k-1} \leq \left( \frac{n + 2k - 1}{k} \right)^{k}\) for all \(m = 0, \ldots, n - 1\), an inequality that was used to establish Proposition 4.4.

For \(k > 0\) and \(x, y \in \mathbb{R}\), let

\[
F_k(x) \overset{\text{def}}{=} \left( \frac{x + 2k - 1}{k} \right)^{k},
\]

\[
G_{k,x}(y) \overset{\text{def}}{=} (y + 1)(x - y + 2k)^{k}.
\]

Let us check that \(G_{k,x}(\frac{k+x}{k+1}) = F_{k+1}(x)\) for any \(k > 0\) and \(x \geq 0\):

\[
G_{k,x} \left( \frac{k + x}{k + 1} \right) = \left( \frac{k + x}{k + 1} + 1 \right) \frac{1}{k^k} \left( x - \frac{k + x}{k + 1} + 2k \right)^k
\]

\[
= \frac{x + 2k + 1}{k + 1} \frac{1}{k^k} \left( \frac{kx + 2k^2 + k}{k + 1} \right)^k
\]

\[
= \frac{x + 2k + 1}{k + 1} \frac{1}{k^k} \left( \frac{k}{k + 1} \right)^k (x + 2k + 1)^k
\]

\[
= \left( \frac{x + 2k + 1}{k + 1} \right)^{k+1} = F_{k+1}(x). \quad (†)
\]

We now claim that \(G_{k,x}(y) \leq F_{k+1}(x)\) for all \(y \in [0, x]\). For \(n, k \geq 2\), the claim entails \(G_{k-1,n}(m) \leq F_k(m)\), i.e. \((m+1) \left( \frac{n - m + 2k - 2}{k - 1} \right)^{k-1} \leq \left( \frac{n + 2k - 1}{k} \right)^{k}\), for \(m = 0, \ldots, n - 1\) as announced.

**Proof (of the claim).** Let \(y_{\text{max}} \overset{\text{def}}{=} \frac{k + x}{k + 1}\). We prove that \(G_{k,x}(y) \leq G_{k,x}(y_{\text{max}})\) and conclude using Eq. (†): \(G_{k,x}\) is well-defined and differentiable over \(\mathbb{R}\), its derivative is

\[
G'_{k,x}(y) = \frac{(x - y + 2k)^k - (y + 1)k(x - y + 2k)^{k-1}}{k^k}
\]

\[
= \frac{(x - y + 2k)^{k-1}}{k^k} ((x - y + 2k) - (y + 1)k)
\]

\[
= \frac{(x - y + 2k)^{k-1}}{k^k} (x + k - y(k + 1)).
\]

Thus \(G'_{k,x}(y)\) is 0 for \(y = y_{\text{max}}\), is strictly positive for \(0 \leq y < y_{\text{max}}\), and strictly negative for \(y_{\text{max}} < y \leq x\). Hence, over \([0, x]\), \(G_{k,x}\) reaches its maximum at \(y_{\text{max}}\). \(\square\)

Appendix B. First values for \(C_k(n)\)

We computed the first values of \(C_k(n)\) by a brute-force method that listed all minimal representatives of \(\sim_n\) equivalence classes over a \(k\)-letter alphabet. Here \(x\) is minimal if \(x \sim_n y\) implies \((|x| < |y|)\) or \((|x| = |y|\) and \(x \leq_{\text{lex}} y\)).

Every equivalence class has a unique minimal representative. Note that if a concatenation \(xx'\) is minimal then both \(x\) and \(x'\) are. Therefore, when listing the minimal representatives in order of increasing length, it is possible to stop when, for some length \(\ell\), one finds no minimal representatives. In that case we know that there cannot exist minimal representatives of length \(>\ell\).

The cells left blank in the table were not computed for lack of memory.
<table>
<thead>
<tr>
<th>$n$</th>
<th>$k = 1$</th>
<th>$k = 2$</th>
<th>$k = 3$</th>
<th>$k = 4$</th>
<th>$k = 5$</th>
<th>$k = 6$</th>
<th>$k = 7$</th>
<th>$k = 8$</th>
<th>$k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$2^k$</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>$2^k$</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>16</td>
<td>152</td>
<td>2 326</td>
<td>52 132</td>
<td>1 602 420</td>
<td>64 529 264</td>
<td>$\geq 173 \cdot 10^7$</td>
<td>$10^7$</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>68</td>
<td>5 312</td>
<td>1 395 588</td>
<td>1 031 153 002</td>
<td>$\geq 23 \cdot 10^7$</td>
<td>$10^7$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>312</td>
<td>334 202</td>
<td>$\geq 73 \cdot 10^7$</td>
<td>$10^7$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>1 560</td>
<td>38 450 477</td>
<td>$10^7$</td>
<td>$10^7$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8 528</td>
<td>$\geq 39 \cdot 10^7$</td>
<td>$10^7$</td>
<td>$10^7$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>50 864</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>329 248</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>2 298 592</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>17 203 264</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>12</td>
<td>137 289 920</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table B.1: Computed values for $C_k(n)$