The smallest grammar problem revisited

Danny Hucke, Markus Lohrey, and Carl Philipp Reh

University of Siegen, Germany
{hucke,lohrey,reh}@eti.uni-siegen.de

Abstract. In a seminal paper of Charikar et al. on the smallest grammar problem, the authors derive upper and lower bounds on the approximation ratios for several grammar-based compressors, but in all cases there is a gap between the lower and upper bound. Here we close the gaps for LZ78 and BISECTION by showing that the approximation ratio of LZ78 is $\Theta((n/\log n)^{2/3})$, whereas the approximation ratio of BISECTION is $\Theta((n/\log n)^{1/2})$. We also derive a lower bound for a smallest grammar for a word in terms of its number of LZ77-factors, which refines existing bounds of Rytter. Finally, we improve results of Arpe and Reischuk relating grammar-based compression for arbitrary alphabets and binary alphabets.

1 Introduction

The idea of grammar-based compression is based on the fact that in many cases a word w can be succinctly represented by a context-free grammar that produces exactly w. Such a grammar is called a *straight-line program* (SLP) for w. In the best case, one gets an SLP of size $O(\log n)$ for a word of length n, where the size of an SLP is the total length of all right-hand sides of the rules of the grammar. A grammar-based compressor is an algorithm that produces for a given word w an SLP $\mathbb{A}$ for w, where, of course, $\mathbb{A}$ should be smaller than w. Grammar-based compressors can be found at many places in the literature. Probably the best known example is the classical LZ78-compressor of Lempel and Ziv [17]. Indeed, it is straightforward to transform the LZ78-representation of a word w into an SLP for w. Other well-known grammar-based compressors are BISECTION [9], SEQUITUR [13], and RePair [10], just to mention a few.

One of the first appearances of straight-line programs in the literature are [2, 5], where they are called *word chains* (since they generalize addition chains from numbers to words). In [2], Berstel and Brlek prove that the function $g(k,n) = \max\{g(w) \mid w \in \{1,\ldots,k\}^n\}$, where $g(w)$ is the size of a smallest SLP for the word w, is in $\Theta(n/\log_k n)$. Note that $g(k,n)$ measures the worst case SLP-compression over all words of length n over a k-letter alphabet. The first systematic investigations of grammar-based compressors are [4, 8]. Whereas in [8], grammar-based compressors are used for universal lossless compression (in the information-theoretic sense), Charikar et al. study in [4] the worst case approximation ratio of grammar-based compressors. For a given grammar-based compressor $\mathcal{C}$ that computes from a given word w an SLP $\mathcal{C}(w)$ for w one defines the approximation ratio of $\mathcal{C}$ on w as the quotient of the size of $\mathcal{C}(w)$ and the size $g(w)$ of a smallest SLP for w. The approximation ratio $\alpha_{\mathcal{C}}(n)$ is the maximal approximation ratio of $\mathcal{C}$ among all words of length n over any alphabet. In [4] the authors compute upper and lower bounds for the approximation ratios of several grammar-based compressors (among them are the compressors mentioned above), but for none of the compressors the lower and upper bounds match. Our first main contribution (Section 3) closes the gaps for $\mathsf{LZ78}$ and $\mathsf{BISECTION}$. For this we improve the corresponding lower bounds from [4] and obtain the approximation ratios $\Theta((n/\log n)^{1/2})$ for $\mathsf{BISECTION}$ and $\Theta((n/\log n)^{2/3})$ for $\mathsf{LZ78}$. For $\mathsf{BISECTION}$ (resp., $\mathsf{LZ78}$), we prove this lower bound for a binary (resp., ternary) alphabet.

In Section 4 we compare the size of a smallest SLP for a word w with the number of factors of the $\mathsf{LZ77}$-factorization of w (we denote the latter with $g_{\mathsf{LZ77}}(w)$). Rytter [14] proved for every word w of length n the following bounds on the size $g(w)$ of a smallest SLP for w: $g(w) \geq g_{\mathsf{LZ77}}(w)$ and $g(w) \in O(g_{\mathsf{LZ77}}(w) \cdot \log n)$. This leads to the question whether the upper bound $g(w) \in O(g_{\mathsf{LZ77}}(w) \cdot \log n)$ on $g(w)$ can be improved. This would have immediate consequences for grammar-based compression: If one could construct in polynomial time an SLP of size $o(g_{\mathsf{LZ77}}(w) \cdot \log n)$ for a given word w, then one would obtain a grammar-based compressor with an approximation ratio of $o(\log n)$. Currently, the theoretically best grammar-based compressors (which all work in linear time) achieve an approximation ratio in $O(\log(n/g(w)))$ [4, 7, 14], and a polynomial time grammar-based compressor with an approximation ratio in $o(\log n/\log\log n)$ would imply a spectacular breakthrough on a long standing open problem on approximating addition chains [4]. Here, we partially answer the above question whether the bound $g(w) \in O(g_{\mathsf{LZ77}}(w) \cdot \log n)$ is sharp. Using a Kolmogorov complexity argument we construct a sequence of words w_n for which $g(w_n) \in \Omega(g_{\mathsf{LZ77}}(w_n) \cdot \log |w_n| / \log\log |w_n|)$.

Our last contribution deals with the hardness of the smallest grammar problem for words over a binary alphabet. The smallest grammar problem is the problem of computing a smallest grammar for a given input word. Storer and Szymanski [15] and Charikar et al. [4] proved that the smallest grammar problem cannot be solved in polynomial time unless $\mathsf{P} = \mathsf{NP}$. Even worse, unless $\mathsf{P} = \mathsf{NP}$ one cannot compute in polynomial time for a given word w an SLP of size $< 8569/8568 \cdot g(w)$ [4]. The construction in [4] uses an alphabet of unbounded size, and it was open whether this complexity lower bound also holds for words over a fixed alphabet. In [4] it is remarked that the construction in [15] shows that the smallest grammar problem for words over a ternary alphabet cannot be solved in polynomial time unless $\mathsf{P} = \mathsf{NP}$. But this is not clear at all, see the recent paper [3] for a detailed explanation. In the same paper [3] it was shown that the smallest grammar problem for an alphabet of size 24 cannot be solved in polynomial time unless $\mathsf{P} = \mathsf{NP}$ using a rather complicated construction [3]. It is far from clear whether this construction can be adapted so that it works also for a binary alphabet. Another idea is to reduce the smallest grammar problem for unbounded alphabets to the smallest grammar problem for a binary alphabet. This route was investigated in [1], where the following result was shown: If there is a polynomial time grammar-based compressor with approximation ratio c (a constant) on binary words, then there is a polynomial time grammar-based compressor with approximation ratio $24c + \varepsilon$ for every $\varepsilon > 0$ on arbitrary words. The construction in [1] uses a quite technical block encoding of arbitrary alphabets into a binary alphabet. Here, we present a very simple construction that encodes the i-th alphabet symbol by $a^i b$, which yields the same result as [1] but with $24c + \varepsilon$ replaced by 6.

2 Straight-line Programs

Let $w = a_1 \cdots a_n$ ($a_1, \ldots, a_n \in \Sigma$) be a *word* over an *alphabet* Σ. The length $|w|$ of w is n and we denote by ε the word of length 0. Let $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$ be the set of nonempty words. For $w \in \Sigma^+$, we call $v \in \Sigma^+$ a *factor* of w if there exist $x, y \in \Sigma^*$ such that $w = xvy$. If $x = \varepsilon$ (respectively $y = \varepsilon$) then we call v a *prefix* (respectively *suffix*) of w. A factorization of w is a decomposition $w = f_1 \cdots f_\ell$ into factors $f_1, \ldots, f_\ell$. For words $w_1, \ldots, w_n \in \Sigma^*$, we further denote by $\prod_{i=j}^{n} w_i$ the word $w_j w_{j+1} \cdots w_n$ if $j \leq n$ and ε otherwise.

A *straight-line program*, briefly SLP, is a context-free grammar that produces a single word $w \in \Sigma^+$. Formally, it is a tuple $\mathbb{A} = (N, \Sigma, P, S)$, where N is a finite set of nonterminals with $N \cap \Sigma = \emptyset$, $S \in N$ is the start nonterminal, and P is a finite set of productions (or rules) of the form $A \to w$ for $A \in N$, $w \in (N \cup \Sigma)^+$ such that: (i) For every $A \in N$, there exists exactly one production of the form $A \to w$, and (ii) the binary relation $\{(A, B) \in N \times N \mid (A \to w) \in P,\ B \text{ occurs in } w\}$ is acyclic. Every nonterminal $A \in N$ produces a unique string $\mathrm{val}_{\mathbb{A}}(A) \in \Sigma^+$. The string defined by $\mathbb{A}$ is $\mathrm{val}(\mathbb{A}) = \mathrm{val}_{\mathbb{A}}(S)$. We omit the subscript $\mathbb{A}$ when it is clear from the context. The *size* of the SLP $\mathbb{A}$ is $|\mathbb{A}| = \sum_{(A \to w) \in P} |w|$. We will use the following lemma which summarizes known results about SLPs.

Lemma 1. *Let Σ be a finite alphabet.*

1. *For every word $w \in \Sigma^+$ of length n, there exists an SLP $\mathbb{A}$ of size $O(n/\log n)$ such that* $\mathrm{val}(\mathbb{A}) = w$.
2. *For an SLP $\mathbb{A}$ and a number $n > 0$, there exists an SLP $\mathbb{B}$ of size $|\mathbb{A}| + O(\log n)$ such that* $\mathrm{val}(\mathbb{B}) = \mathrm{val}(\mathbb{A})^n$.
3. *For SLPs $\mathbb{A}_1$ and $\mathbb{A}_2$ there exists an SLP $\mathbb{B}$ of size $|\mathbb{A}_1| + |\mathbb{A}_2|$ such that* $\mathrm{val}(\mathbb{B}) = \mathrm{val}(\mathbb{A}_1)\mathrm{val}(\mathbb{A}_2)$.
4. *For given words $w_1, \ldots, w_n \in \Sigma^*$, $u \in \Sigma^+$ and SLPs $\mathbb{A}_1, \mathbb{A}_2$ with $\mathrm{val}(\mathbb{A}_1) = u$ and $\mathrm{val}(\mathbb{A}_2) = w_1 x w_2 x \cdots w_{n-1} x w_n$ for a symbol $x \notin \Sigma$, there exists an SLP $\mathbb{B}$ of size $|\mathbb{A}_1| + |\mathbb{A}_2|$ such that* $\mathrm{val}(\mathbb{B}) = w_1 u w_2 u \cdots w_{n-1} u w_n$.

Statement 1 can be found for instance in [2]. Statements 2 and 3 are shown in [4]. The proof of 4 is straightforward: Simply replace in the SLP $\mathbb{A}_2$ every occurrence of the terminal x by the start nonterminal of $\mathbb{A}_1$ and add all rules of $\mathbb{A}_1$ to $\mathbb{A}_2$.

We denote by $g(w)$ the size of a smallest SLP producing the word $w \in \Sigma^+$. The maximal size of a smallest SLP for all words of length n over an alphabet of size k is

$$g(k, n) = \max\{g(w) \mid w \in [1, k]^n\},$$

where $[1, k] = \{1, \ldots, k\}$. By point 1 of Lemma 1 we have $g(k, n) \in O(n/\log_k n)$. In fact, Berstel and Brlek proved in [2] that $g(k, n) \in \Theta(n/\log_k n)$. The following result provides further information about the function $g(k, n)$:

Proposition 2. *Let $n_k = 2k^2 + 2k + 1$ for $k > 0$. Then (i) $g(k, n) < n$ for $n > n_k$ and (ii) $g(k, n) = n$ for $n \leq n_k$.*

Proof. Let $\Sigma_k = \{a_1, \ldots, a_k\}$ and let $M_{n,\ell} \subseteq \Sigma_k^*$ be the set of all words w where a factor v of length ℓ occurs at least n times without overlap. It is easy to see that $g(w) < |w|$ if and only if $w \in M_{3,2} \cup M_{2,3}$. Hence, we have to show that every word $w \notin M_{3,2} \cup M_{2,3}$ has length at most $2k^2 + 2k + 1$. Moreover, we present words $w_k \in \Sigma_k^*$ of length $2k^2 + 2k + 1$ such that $w_k \notin M_{3,2} \cup M_{2,3}$.

Let $w \notin M_{3,2} \cup M_{2,3}$. Consider a factor $a_i a_j$ of length two. If $i \neq j$ then this factor does not overlap itself, and thus $a_i a_j$ occurs at most twice in w. Now consider $a_i a_i$. Then w contains at most four (possibly overlapping) occurrence of $a_i a_i$, because five occurrences of $a_i a_i$ would yield at least three non-overlapping occurrences of $a_i a_i$. It follows that w has at most $2(k^2 - k) + 4k$ positions where a factor of length 2 starts, which implies $|w| \leq 2k^2 + 2k + 1$.

Now we create a word $w_k \notin M_{3,2} \cup M_{2,3}$ which realizes the above maximal occurrences of factors of length 2:

$$w_k = \left(\prod_{i=1}^{k} a_{k-i+1}^5 \right) \prod_{i=1}^{k-1} \left(\prod_{i+2}^{j=k} (a_j a_i)^2 \right) a_{i+1} a_i a_{i+1}$$

For example we have $w_3 = a_3^5 a_2^5 a_1^5 (a_3 a_1)^2 a_2 a_1 a_2 a_3 a_2 a_3$. One can check that $|w_k| = 2k^2 + 2k + 1$ and $w_k \notin M_{3,2} \cup M_{2,3}$. □

3 Approximation ratio

As mentioned in the introduction, there is no polynomial time algorithm that computes a smallest SLP for a given word, unless $\mathsf{P} = \mathsf{NP}$ [4, 15]. This result motivates approximation algorithms which are called *grammar-based compressors*. A grammar-based compressor $\mathcal{C}$ computes for a word w an SLP $\mathcal{C}(w)$ such that $\mathrm{val}(\mathcal{C}(w)) = w$. The *approximation ratio* $\alpha_{\mathcal{C}}(w)$ of $\mathcal{C}$ for an input w is defined as $|\mathcal{C}(w)|/g(w)$. The worst-case approximation ratio $\alpha_{\mathcal{C}}(k, n)$ of $\mathcal{C}$ is the maximal approximation ratio over all words of length n over an alphabet of size k:

$$\alpha_{\mathcal{C}}(k, n) = \max\{\alpha_{\mathcal{C}}(w) \mid w \in [1, k]^n\} = \max\{|\mathcal{C}(w)|/g(w) \mid w \in [1, k]^n\}$$

If the alphabet size is unbounded, i.e. we allow alphabets of size $|w|$, then we write $\alpha_{\mathcal{C}}(n)$ instead of $\alpha_{\mathcal{C}}(n, n)$. This is the definition of the worst-case approximation ratio in [4]. The grammar-based compressors studied in our work are BISECTION [9] and LZ78 [17]. We will abbreviate the approximation ratio of BISECTION (respectively LZ78) by α_{BI} (respectively α_{LZ78}). The families of words which we will use to prove new lower bounds for $\alpha_{\mathsf{BI}}(n)$ and $\alpha_{\mathsf{LZ78}}(n)$ are inspired by the constructions in [4].

3.1 BISECTION

The BISECTION algorithm [9] first splits an input word w with $|w| \geq 2$ as $w = w_1 w_2$ such that $|w_1| = 2^j$ for the unique number $j \geq 0$ with $2^j < |w| \leq 2^{j+1}$. This process is recursively repeated with w_1 and w_2 until we obtain words of length 1. During the process, we introduce a nonterminal for each distinct factor of length at least two and create a rule with two symbols on the right-hand side corresponding to the split. Note that if $w = u_1 u_2 \cdots u_k$ with $|u_i| = 2^n$ for all $i, 1 \leq i \leq k$, then the SLP produced by BISECTION contains a nonterminal for each distinct word u_i $(1 \leq i \leq k)$.

Example 3. BISECTION constructs an SLP for $w = ababbbaabbaaab$ as follows:

- $w = w_1 w_2$ with $w_1 = ababbbaa$, $w_2 = bbaaab$
  Introduced rule: $S \to W_1 W_2$
- $w_1 = x_1 x_2$ with $x_1 = abab$, $x_2 = bbaa$, and $w_2 = x_2 x_3$ with $x_3 = ab$
  Introduced rules: $W_1 \to X_1 X_2,\ W_2 \to X_2 X_3,\ X_3 \to ab$
- $x_1 = x_3 x_3$, $x_2 = y_1 y_2$ with $y_1 = bb$ and $y_2 = aa$
  Introduced rules: $X_1 \to X_3 X_3,\ X_2 \to Y_1 Y_2,\ Y_1 \to bb,\ Y_2 \to aa$

BISECTION performs asymptotically optimal on unary words a^n since it produces an SLP of size $O(\log n)$. Therefore $\alpha_{\mathsf{BI}}(1, n) \in \Theta(1)$. The following bounds on the approximation ratio for alphabets of size at least two are proven in [4, Thm. 5 and 6]:

$$\alpha_{\mathsf{BI}}(2, n) \in \Omega(\sqrt{n}/\log n) \tag{1}$$

$$\alpha_{\mathsf{BI}}(n) \in O(\sqrt{n/\log n}) \tag{2}$$

We improve the lower bound (1) so that it matches the upper bound (2):

Theorem 4. *For every $k, 2 \leq k \leq n$ we have $\alpha_{\mathsf{BI}}(k, n) \in \Theta(\sqrt{n/\log n})$.*

Proof. The upper bound (2) implies that $\alpha_{\mathsf{BI}}(k, n) \in O(\sqrt{n/\log n})$ for all $k, 2 \leq k \leq n$. So it suffices to show $\alpha_{\mathsf{BI}}(2, n) \in \Omega(\sqrt{n/\log n})$. We first show that $\alpha_{\mathsf{BI}}(3, n) \in \Omega(\sqrt{n/\log n})$. In a second step, we encode a ternary alphabet into a binary alphabet while preserving the approximation ratio.

For every $k \geq 2$ let $\mathrm{bin}_k : \{0, 1, \ldots, k-1\} \to \{0, 1\}^{\lceil \log_2 k \rceil}$ be the function where $\mathrm{bin}_k(j)$ $(0 \leq j \leq k-1)$ is the binary representation of j filled with leading zeros (e.g. $\mathrm{bin}_9(3) = 0011$). We further define for every $k \geq 2$ the word

$$u_k = \left(\prod_{j=0}^{k-2} \mathrm{bin}_k(j) a^{m_k} \right) \mathrm{bin}_k(k-1),$$

where $m_k = 2^{k - \lceil \log_2 k \rceil} - \lceil \log_2 k \rceil$. For instance $k = 4$ leads to $m_k = 2$ and $u_4 = 00aa01aa10aa11$. We analyse the approximation ratio $\alpha_{\mathsf{BI}}(s_k)$ for the word

$$s_k = \left(u_k a^{m_k + 1} \right)^{m_k} u_k.$$

Claim 1. The SLP produced by BISECTION on input s_k has size $\Omega(2^k)$.

If s_k is split into non-overlapping factors of length $m_k + \lceil \log_2 k \rceil = 2^{k - \lceil \log_2 k \rceil}$, then the resulting set F_k of factors is

$$F_k = \{a^i \mathrm{bin}_k(j) a^{m_k - i} \mid 0 \leq j \leq k-1,\ 0 \leq i \leq m_k\}.$$

For example s_4 consecutively consists of the factors $00aa$, $01aa$, $10aa$, $11aa$, $a00a$, $a01a$, $a10a$, $a11a$, $aa00$, $aa01$, $aa10$ and $aa11$. The size of F_k is $(m_k + 1) \cdot k \in \Theta(2^k)$, because all factors are pairwise different and $m_k \in \Theta(2^k/k)$. It follows that the SLP produced by BISECTION on input s_k has size $\Omega(2^k)$, because the length of each factor in F_k is a power of two and thus BISECTION creates a nonterminal for each distinct factor in F_k.

Claim 2. A smallest SLP producing s_k has size $O(k)$.

There is an SLP of size $O(\log m_k) = O(k)$ for the word a^{m_k} by Lemma 1 (point 2). This yields an SLP for u_k of size $O(k) + g(u'_k)$ by Lemma 1 (point 4), where $u'_k = (\prod_{i=0}^{k-2} \mathrm{bin}_k(i)x)\mathrm{bin}_k(k-1)$ is obtained from u_k by replacing all occurrences of a^{m_k} by a fresh symbol x. The word u'_k has length $\Theta(k \log k)$. Applying point 1 of Lemma 1 (note that u'_k is a word over a ternary alphabet) it follows that

$$g(u'_k) \in O\left(\frac{k \log k}{\log(k \log k)}\right) = O\left(\frac{k \log k}{\log k + \log\log k}\right) = O(k).$$

Hence $g(u_k) \in O(k)$. Finally, the SLP of size $O(k)$ for u_k yields an SLP of size $O(k)$ for s_k again using Lemma 1 (points 2 and 3).

In conclusion: We showed that a smallest SLP for s_k has size $O(k)$, while BISECTION produces an SLP of size $\Omega(2^k)$.This implies $\alpha_{\mathsf{BI}}(s_k) \in \Omega(2^k/k)$. Let $n = |s_k|$. Since s_k is the concatenation of $\Theta(2^k)$ factors of length $\Theta(2^k/k)$, we have $n \in \Theta(2^{2k}/k)$ and thus $\sqrt{n} \in \Theta(2^k/\sqrt{k})$. This yields $\alpha_{\mathsf{BI}}(s_k) \in \Omega(\sqrt{n/k})$. Together with $k \in \Theta(\log n)$ we obtain $\alpha_{\mathsf{BI}}(3, n) \in \Omega(\sqrt{n/\log n})$.

Let us now encode words over $\{0, 1, a\}$ into words over $\{0, 1\}$. Consider the homomorphism $f : \{0, 1, a\}^* \to \{0, 1\}^*$ with $f(0) = 00$, $f(1) = 01$ and $f(a) = 10$. Then we can prove the same approximation ratio of BISECTION for the input $f(s_k) \in \{0, 1\}^*$ that we proved for s_k above: The size of a smallest SLP for $f(s_k)$ is at most twice as large as the size of a smallest SLP for s_k, because an SLP for s_k can be transformed into an SLP for $f(s_k)$ by replacing every occurrence of a symbol $x \in \{0, 1, a\}$ by $f(x)$. Moreover, if we split $f(s_k)$ into non-overlapping factors of twice the length as we considered for s_k, then we obtain the factors from $f(F_k)$, whose length is again a power of two. Since f is injective, we have $|f(F_k)| = |F_k| \in \Theta(2^k)$. □

3.2 LZ78

The LZ78 algorithm on input $w \in \Sigma^+$ implicitly creates a list of words $f_1, \ldots, f_\ell$ (which we call the LZ78-*factorization*) with $w = f_1 \cdots f_\ell$ such that the following properties hold, where we set $f_0 = \varepsilon$:

- $f_i \neq f_j$ for all $i, j, 0 \leq i, j \leq \ell - 1$ with $i \neq j$.
- For all $i, 1 \leq i \leq \ell - 1$ there exist $j, 0 \leq j < i$ and $a \in \Sigma$ such that $f_i = f_j a$.
- $f_\ell = f_i$ for some $0 \leq i \leq \ell - 1$.

Note that the LZ78-factorization is unique for each word w. To compute it, the LZ78 algorithm needs ℓ steps performed by a single left-to-right pass. In the k^{th} step ($1 \leq k \leq \ell - 1$) it chooses the factor f_k as the shortest prefix of the unprocessed suffix $f_k \cdots f_\ell$ such that $f_k \neq f_i$ for all $i < k$. If there is no such prefix, then the end of w is reached and the algorithm sets f_ℓ to the (possibly empty) unprocessed suffix of w.

The factorization $f_1, \dots, f_\ell$ yields an SLP for w of size at most 3ℓ as described in the following example:

Example 5. The $\mathsf{LZ78}$-factorization of $w = aabaaababababaa$ is a, ab, aa, aba, b, $abab$, aa and leads to an SLP with the following rules:

- $S \to F_1 F_2 F_3 F_4 F_5 F_6 F_3$
- $F_1 \to a,\ F_2 \to F_1 b,\ F_3 \to F_1 a,\ F_4 \to F_2 a,\ F_5 \to b,\ F_6 \to F_4 b$

We have a nonterminal F_i for each factor f_i ($1 \le i \le 6$) such that $\mathrm{val}_{\mathbb{A}}(F_i) = f_i$. The last factor aa is represented in the start rule by the nonterminal F_3.

The $\mathsf{LZ78}$-factorization of a^n ($n > 0$) is $a^1, a^2, \dots, a^m, a^k$, where $k \in \{0, \dots, m\}$ such that $n = k + \sum_{i=1}^{m} i$. Note that $m \in \Theta(\sqrt{n})$ and thus $\alpha_{\mathsf{LZ78}}(1, n) \in \Theta(\sqrt{n}/\log n)$. The following bounds for the worst-case approximation ratio of $\mathsf{LZ78}$ were shown in [4, Thm. 3 and 4]:

$$\alpha_{\mathsf{LZ78}}(2, n) \in \Omega(n^{2/3}/\log n) \tag{3}$$

$$\alpha_{\mathsf{LZ78}}(n) \in O((n/\log n)^{2/3}) \tag{4}$$

For ternary (or larger) alphabets, we will improve the lower bound so that it matches the upper bound in (4).

Theorem 6. *For every $k \ge 3$ we have $\alpha_{\mathsf{LZ78}}(k, n) \in \Theta((n/\log n)^{2/3})$.*

Proof. Due to (4) it suffices to show $\alpha_{\mathsf{LZ78}}(3, n) \in \Omega((n/\log n)^{2/3})$. For $k \ge 2, m \ge 1$, let $u_{m,k} = \left(a^k b^m c\right)^{k(m+2)-1}$ and $v_{m,k} = \left(\prod_{i=1}^{m} b^i a^k\right)^{k^2}$. We now analyse the approximation ratio of $\mathsf{LZ78}$ on the words

$$s_{m,k} = a^{k(k+1)/2}\, b^{m(m+1)/2}\, u_{m,k}\, v_{m,k}.$$

For example we have $u_{2,4} = (a^4 b^2 c)^{15}$, $v_{2,4} = (b a^4 b^2 a^4)^{16}$ and $s_{2,4} = a^{10}\, b^3\, u_{2,4}\, v_{2,4}$.

Claim 1. The SLP produced by $\mathsf{LZ78}$ on input $s_{m,k}$ has size $\Theta(k^2 m)$.

We consider the $\mathsf{LZ78}$-factorization $f_1, \dots, f_\ell$ of $s_{m,k}$. The prefix $a^{k(k+1)/2}$ produces the factors $f_i = a^i$ for every $i, 1 \le i \le k$ and the substring $b^{m(m+1)/2}$ produces the factors $f_{k+i} = b^i$ for every $i, 1 \le i \le m$.

We next show that the substring $u_{m,k}$ then produces (among other factors) all factors $a^i b^j$, where $1 \le i \le k$, $1 \le j \le m$. All other factors produced by $u_{m,k}$ contain the letter c and therefore do not affect the factorization of the final suffix $v_{m,k} \in \{a, b\}^*$.

The first factors of $u_{m,k}$ in $s_{m,k}$ are $f_{k+m+1} = a^k b$ and $f_{k+m+2} = b^{m-1} c$, which together form the first occurrence of $a^k b^m c$. The next two factors are $a^k b^2$ and $b^{m-2} c$. This pattern continues and the prefix $(a^k b^m c)^m$ of $u_{m,k}$ yields the next $2m$ factors $f_{k+m+2i-1} = a^k b^i$ and $f_{k+m+2i} = b^{m-i} c$ for every $i, 1 \le i \le m$. The factorization of $u_{m,k}$ continues with $f_{k+3m+1} = a^k b^m c$ followed by $f_{k+3m+2} = a^k b^m ca$. Next, we have $f_{k+3m+3} = a^{k-1} b$ and $f_{k+3m+4} = b^{m-1} ca$, which is the beginning of a similar pattern as we discovered for $(a^k b^m c)^m$. Therefore, the next $2m$ factors are $f_{k+3m+2i+1} = a^{k-1} b^i$ and $f_{k+3m+2i+2} = b^{m-i} ca$ for every $i, 1 \le i \le m$. The next two factors are $f_{k+5m+3} = a^{k-1} b^m c$ followed by $f_{k+5m+4} = a^k b^m ca^2$. The iteration of these arguments yields k (consecutive) blocks of $2m+2$ factors (resp. $2m+1$ in the last block) in $u_{m,k}$:

1st	block:	$\prod_{i=1}^{m} \big($ $a^k b^i$	$b^{m-i} c \big)$	$a^k b^m c$	$a^k b^m ca$
2nd	block:	$\prod_{i=1}^{m} \big($ $a^{k-1} b^i$	$b^{m-i} ca \big)$	$a^{k-1} b^m c$	$a^k b^m ca^2$
...					
$(k-1)$th	block:	$\prod_{i=1}^{m} \big($ $a^2 b^i$	$b^{m-i} ca^{k-2} \big)$	$a^2 b^m c$	$a^k b^m ca^{k-1}$
kth	block:	$\prod_{i=1}^{m} \big($ ab^i	$b^{m-i} ca^{k-1} \big)$	$ab^m c$	

We will show that the remaining suffix $v_{m,k}$ of $s_{m,k}$ produces then the set of factors

$$\left\{a^i b^p a^j \mid 0 \leq i \leq k-1,\ 1 \leq j \leq k,\ 1 \leq p \leq m\right\}.$$

Let $x = k + m + k(2m+2) - 1$ and note that this is the number of factors that we have produced so far. The factorization of $v_{m,k}$ in $s_{m,k}$ slightly differs whether m is even or is odd. We now assume that m is even and explain the difference to the other case afterwards. The first factor of $v_{m,k}$ in $s_{m,k}$ is $f_{x+1} = ba$. We already have produced the factors $a^{k-1}b^i$ for every $i, 1 \leq i \leq m$ and hence $f_{x+i} = a^{k-1}b^i a$ for every $i, 2 \leq i \leq m$ and $f_{x+m+1} = a^{k-1}ba$. The next m factors are $f_{x+m+i} = a^{k-1}b^i a^2$ if i is even, $f_{x+m+i} = a^{k-2}b^i a$ if i is odd ($2 \leq i \leq m$) and $f_{x+2m+1} = a^{k-2}ba$. This pattern continues: The next m factors are $f_{x+2m+i} = a^{k-1}b^i a^3$ if i is even, $f_{x+2m+i} = a^{k-3}b^i a$ if i is odd ($2 \leq i \leq m$) and $f_{x+3m+1} = a^{k-3}ba$ and so on. Hence, we get the following sets of factors for $(\prod_{i=1}^{m} b^i a^k)^k$:

(i) $\{a^{k-i}b^p a \mid 1 \leq i \leq k,\ 1 \leq p \leq m,\ p \text{ is odd}\}$ for $f_{x+1}, f_{x+3}\ldots, f_{x+km-1}$
(ii) $\{a^{k-1}b^p a^j \mid 1 \leq j \leq k,\ 1 \leq p \leq m,\ p \text{ is even}\}$ for $f_{x+2}, f_{x+4}, \ldots, f_{x+km}$

The remaining word then starts with the factor $f_{y+1} = ba^2$, where $y = x + km$. Now the former pattern can be adapted to the next k repetitions of $\prod_{i=1}^{m} b^i a^k$ which gives us the following factors:

(i) $\{a^{k-i}b^p a^2 \mid 1 \leq i \leq k,\ 1 \leq p \leq m,\ p \text{ is odd}\}$ for $f_{y+1}, f_{y+3}\ldots, f_{y+km-1}$
(ii) $\{a^{k-2}b^p a^j \mid 1 \leq j \leq k,\ 1 \leq p \leq m,\ p \text{ is even}\}$ for $f_{y+2}, f_{y+4}, \ldots, f_{y+km}$

The iteration of this process then reveals the whole pattern and thus yields the claimed factorization of $v_{m,k}$ in $s_{m,k}$ into factors $a^i b^p a^j$ for every $i, 0 \leq i \leq k-1, j, 1 \leq j \leq k$ and $p, 1 \leq p \leq m$. If m is odd then the patterns in (i) and (ii) switch after each occurrence of $\prod_{i=1}^{m} b^i a^k$, which does not affect the result but makes the pattern slightly more complicated. But the case that m is even suffices in order to derive the lower bound from the theorem.

We conclude that there are exactly $k + m + k(2m+2) - 1 + k^2 m$ factors (ignoring $f_\ell = \varepsilon$) and hence the SLP produced by LZ78 on input $s_{m,k}$ has size $\Theta(k^2 m)$.

Claim 2. A smallest SLP producing $s_{m,k}$ has size $O(\log k + m)$.

We will combine the points stated in Lemma 1 to prove this claim. Points 2 and 3 yield an SLP of size $O(\log k + \log m)$ for the prefix $a^{k(k+1)/2}\ b^{m(m+1)/2}\ u_{m,k}$ of $s_{m,k}$. To bound the size of an SLP for $v_{m,k}$ note at first that there is an SLP of size $O(\log k)$ producing a^k by point 2 of Lemma 1. Applying point 4 and again point 2, it follows that there is an SLP of size $O(\log k) + g(v'_{m,k})$ producing $v_{m,k}$, where $v'_{m,k} = \prod_{i=1}^{m} b^i x$ for some fresh letter x. To get a small SLP for $v'_{m,k}$, we can introduce m nonterminals $B_1, \ldots, B_m$ producing $b^1, \ldots, b^m$ by adding rules $B_1 \to b$ and $B_{i+1} \to B_i b$ ($1 \leq i \leq m-1$). This is enough to get an SLP of size $O(m)$ for $v'_{m,k}$ and therefore an SLP of size $O(\log k + m)$ for $v_{m,k}$. Together with our first observation and point 3 of Lemma 1 this yields an SLP of size $O(\log k + m)$ for $s_{m,k}$.

Claim 1 and 2 imply $\alpha_{\mathsf{LZ78}}(s_{m,k}) \in \Omega(k^2 m/(\log k + m))$. Let us now fix $m = \lceil \log k \rceil$. We get $\alpha_{\mathsf{LZ78}}(s_{m,k}) \in \Omega(k^2)$. Moreover, for the length $n = |s_{m,k}|$ of $s_{m,k}$ we have $n \in \Theta(k^3 m + k^2 m^2) = \Theta(k^3 \log k)$. We get $\alpha_{\mathsf{LZ78}}(s_{m,k}) \in \Omega((n/\log k)^{2/3})$ which together with $\log n \in \Theta(\log k)$ finishes the proof. $\square$

It remains open whether also $\alpha_{\mathsf{LZ78}}(2, n) \in \Theta((n/\log n)^{2/3})$ holds. In contrast to BISECTION it is not clear how to encode a ternary alphabet into a binary alphabet while preserving the approximation ratio for LZ78.

4 LZ77 and composition systems

The LZ77*-factorization* of a non-empty word $w \in \Sigma^+$ is $w = f_1 f_2 \cdots f_m$, where for every $i, 1 \leq i \leq m$, f_i is (i) the longest non-empty prefix of $f_i f_{i+1} \cdots f_m$ which is a factor of $f_1 f_2 \cdots f_{i-1}$ or (ii) the first symbol of $f_i f_{i+1} \cdots f_m$ if such a prefix does not exist. Let $g_{\mathsf{LZ77}}(w) = m$ be the number of factors in the LZ77-factorization of w.

Example 7. The LZ77-factorization of $w = aabaaabababababaa$ is a, a, b, aa, aba, ba, $baba$, a and we have $g_{\mathsf{LZ77}}(w) = 8$.

We are interested in the following ratios, where $1 \leq k \leq n$:

$$\beta_{\mathsf{LZ77}}(k, n) = \max\{g(w)/g_{\mathsf{LZ77}}(w) \mid w \in [1, k]^n\} \text{ and } \beta_{\mathsf{LZ77}}(n) = \beta_{\mathsf{LZ77}}(n, n).$$

For a word w over a unary alphabet one has $g_{\mathsf{LZ77}}(w) \in \Theta(\log |w|)$ and therefore $\beta_{\mathsf{LZ77}}(1, n) \in \Theta(1)$. Rytter proved that for every word w, $g(w) \geq g_{\mathsf{LZ77}}(w)$ and hence $\beta_{\mathsf{LZ77}}(k, n) \geq 1$ for all $k, 1 \leq k \leq n$ [14].[1] Moreover, in the same paper, he constructed for a word w an SLP of size $O(g_{\mathsf{LZ77}}(w) \cdot \log |w|)$. This yields $\beta_{\mathsf{LZ77}}(n) \in O(\log n)$. Using Kolmogorov complexity we prove the lower bound $\beta_{\mathsf{LZ77}}(2, n) \in \Omega(\log n/\log\log n)$.

For a partial recursive function $\phi : \{0,1\}^* \to \{0,1\}^*$ and a word $w \in \{0,1\}^*$ let $C_\phi(w) = \min\{|p| \mid p \in \{0,1\}^*, \phi(p) = w\}$ (where we define $\min(\emptyset) = \infty$) be the Kolmogorov complexity of w with respect to ϕ. The invariance theorem of Kolmogorov complexity states that there is a partial recursive surjective function $U : \{0,1\}^* \to \{0,1\}^*$ such that for every partial recursive function $\phi : \{0,1\}^* \to \{0,1\}^*$ there is a constant $c \geq 0$ with $C_U(w) \leq C_\phi(w) + c$ for all w. We fix such a function U (it can be obtained from a universal Turing machine) and define the Kolmogorov complexity of w as $C(w) := C_U(w)$. It is well known that for every $n \geq 0$ there is a word $w \in \{0,1\}^n$ with $C(w) \geq n$ (such a word is called Kolmogorov random). See [11] for further details.

Theorem 8. $\beta_{\mathsf{LZ77}}(2, n) \in \Omega(\log n/\log\log n)$.

[1] It is shown in [14] that every SLP in Chomsky normal form for w has at least $g_{\mathsf{LZ77}}(w)$ many nonterminals. But the number of nonterminals in a smallest Chomsky normal form SLP for w is bounded by $g(w)$.

Proof. Let $m \in \mathbb{N}$, $w \in \{0,1\}^*$, $|w| = m^2$ and $C(w) \geq m^2$. We factorize w as $w = w_1 \cdots w_m$ where $|w_i| = m$ for every $i, 1 \leq i \leq m$. We encode every w_i into a binary number of size $\Theta(2^m)$ using the following (ranking) function $p : \{0,1\}^* \to \mathbb{N}$: We define $p(u) = i$ if and only if u is the i^{th} word in the length-lexicographic enumeration of all words from $\{0,1\}^*$ (where $p(\varepsilon) = 0$). This is a computable bijection from $\{0,1\}^*$ to $\mathbb{N}$ such that $p(x) \in \Theta(2^{|x|})$. Let $N_i = p(w_i)$ for every $i, 1 \leq i \leq m$. Thus, we have $N_i \in \Theta(2^m)$. Let $N = \max\{N_1, \ldots, N_m\} \in \Theta(2^m)$ and define the word $v = a^N \# a^{N_1} \# \ldots \# a^{N_m} \#$ over the alphabet $\{a, \#\}$. Let $\mathbb{A}$ be a smallest SLP for v. Note that v and hence $\mathbb{A}$ uniquely encodes the word w. Since an SLP of size k can be encoded by a bit string of size $O(k \log k)$ [16] and $C(w) \geq m^2$, it follows that $|\mathbb{A}| \cdot \log |\mathbb{A}| \in \Omega(m^2)$. Note that this is the point where the Kolmogorov randomness of w is applied. Moreover, there exists an SLP for v of size $O(m \cdot \log N) = O(m^2)$. Thus, $|\mathbb{A}| \in O(m^2)$, which together with $|\mathbb{A}| \cdot \log(|\mathbb{A}|) \in \Omega(m^2)$ implies $|\mathbb{A}| \in \Omega(m^2/\log m)$ and hence $g(v) \in \Omega(m^2/\log m)$. On the other hand, the $\mathsf{LZ77}$-factorization of v has $O(m)$ factors: The prefix $a^N\#$ of v contributes $O(\log N) = O(m)$ factors. Because $N = \max\{N_1, \ldots, N_m\}$, every $a^{N_i}\#$, where $1 \leq i \leq m$, contributes at most one additional factor. Altogether, we get $g_{\mathsf{LZ77}}(v) \in O(m)$. Let $n = |v| \in \Theta(m \cdot 2^m)$, which implies $\log n \in \Theta(m)$. We get

$$\beta_{\mathsf{LZ77}}(2,n) \geq \frac{g(v)}{g_{\mathsf{LZ77}}(v)} \in \Omega\left(\frac{m^2}{m \log m}\right) = \Omega\left(\frac{m}{\log m}\right) = \Omega\left(\frac{\log n}{\log \log n}\right).$$

This concludes the proof. □

It remains open, whether the lower bound in Theorem 8 can be raised to $\Omega(\log n)$.

A common generalization of SLPs and $\mathsf{LZ77}$-factorizations are so called *composition systems* [6] or *Cut-SLP* [12] (briefly CSLP), which we define next. For a word $w = a_1 \cdots a_n \in \Sigma^*$ and $1 \leq i \leq j \leq n$ we define $w[i:j] = a_i \cdots a_j$. A CLSP $\mathbb{C} = (N, \Sigma, P, S)$ is defined analogously to an SLP but in addition may contain rules of the form $A \to B[i:j]$ for $A, B \in N$ and $1 \leq i \leq j \leq |\text{val}(B)|$. We then define $\text{val}(A) = \text{val}(B)[i:j]$. The size of a right-hand side $B[i:j]$ is set to $|B[i:j]| = 1$ and the size of a CSLP is $|\mathbb{C}| = \sum_{(A \to w) \in P} |w|$. We denote by $g_{\mathsf{CSLP}}(w)$ the size of a smallest CSLP $\mathbb{C}$ such that $\text{val}(\mathbb{C}) = w$ and define $\beta_{\mathsf{CSLP}}(k,n) = \max\{g(w)/g_{\mathsf{CSLP}}(w) \mid w \in [1,k]^n\}$ and $\beta_{\mathsf{CSLP}}(n) = \beta_{\mathsf{CSLP}}(n,n)$.

Note that if a non-empty word w has an $\mathsf{LZ77}$-factorization $w = f_1 f_2 \cdots f_m$ of length m then $g_{\mathsf{CSLP}}(w) \leq 3m$: We introduce for every $i, 1 \leq i \leq m$ a nonterminal A_i which evaluates to $f_1 \cdots f_i$. For this, we set $A_1 \to f_1$ (f_1 must be a single symbol). For every $i, 2 \leq i \leq m$ we set $A_i \to A_{i-1} f_i$ if f_i is a single symbol and $A_i \to A_{i-1} B_i$, $B_i \to A_{i-1}[j:k]$ if $f_i = (f_1 \cdots f_{i-1})[j:k]$. Together with Theorem 8 this yields the lower bound in the following theorem. The upper bound follows easily using the techniques from [14].

Theorem 9. *We have* $\beta_{\mathsf{CSLP}}(2,n) \in \Omega(\log n/\log\log n)$ *and* $\beta_{\mathsf{CSLP}}(n) \in O(\log n)$.

5 Hardness of grammar-based compression for binary alphabets

The goal of this section is to prove the following result:

Theorem 10. *Let $c \geq 1$ be a constant. If there exists a polynomial time grammar-based compressor $\mathcal{C}$ with $\alpha_{\mathcal{C}}(2,n) \leq c$ then there exists a polynomial time grammar-based compressor $\mathcal{D}$ with $\alpha_{\mathcal{D}}(n) \leq 6c$.*

For a factor $24 + \varepsilon$ (with $\varepsilon > 0$) instead of 6 this result was shown in [1] using a more complicated block encoding.

We split the proof of Theorem 10 in two lemmas that state translations between SLPs over arbitrary alphabets and SLPs over a binary alphabet. For the rest of this section fix the alphabets $\Sigma = \{a_0, \dots, a_{k-1}\}$ and $\Sigma_2 = \{a, b\}$. To translate between these two alphabets, we define an injective homomorphism $\varphi : \Sigma^* \to \Sigma_2^*$ by

$$\varphi(a_i) = a^i b \quad (0 \leq i \leq k-1). \tag{5}$$

Lemma 11. *Let $w \in \Sigma^*$ such that every symbol from Σ occurs in w. From an SLP $\mathbb{A}$ for w one can construct in polynomial time an SLP $\mathbb{B}$ for $\varphi(w)$ of size at most $3 \cdot |\mathbb{A}|$.*

Proof. To translate $\mathbb{A}$ into an SLP $\mathbb{B}$ for $\varphi(w)$, we first add the productions $A_0 \to b$ and $A_i \to aA_{i-1}$ for every $i, 1 \leq i \leq k-1$. Finally, we replace in $\mathbb{A}$ every occurrence of $a_i \in \Sigma$ by A_i. This yields an SLP $\mathbb{B}$ for $\varphi(w)$ of size $|\mathbb{A}| + 2k - 1$. Because $k \leq |\mathbb{A}|$ (since every symbol from Σ occurs in w), we obtain $|\mathbb{B}| \leq 3 \cdot |\mathbb{A}|$. □

Lemma 12. *Let $w \in \Sigma^*$ such that every symbol from Σ occurs in w. From an SLP $\mathbb{B}$ for $\varphi(w)$ one can construct in linear time an SLP $\mathbb{A}$ for w of size at most $2 \cdot |\mathbb{B}|$.*

Proof. A factor of a word from $\varphi(\Sigma^*)$ is of the form $s = a^{i_1}b \cdots a^{i_n}ba^{i_{n+1}}$ for some $n \geq 0$, and $0 \leq i_1, \dots, i_{n+1} \leq k-1$. Take new symbols $\tilde{a}_i$, $0 \leq i \leq k-1$. Intuitively, $\tilde{a}_i$ is an abbreviation for a^i (whereas a_i is an abbreviation for $a^i b$). The symbols $\tilde{a}_i$ are only used during the construction for clarification, and disappear at the end. For the word $s = a^{i_1}b \cdots a^{i_n}ba^{i_{n+1}}$ define $\ell(s) \in \Sigma \cup \{\varepsilon\}$, $m(s) \in \Sigma^*$, and $r(s) \in \{\tilde{a}_i \mid 0 \leq i \leq k-1\}$ as follows:

$$\ell(s) = \begin{cases} a_{i_1} & \text{if } n \geq 1 \\ \varepsilon & \text{if } n = 0 \end{cases} \qquad m(s) = a_{i_2} \cdots a_{i_n} \qquad r(s) = \tilde{a}_{i_{n+1}}$$

Note that $\ell(s) = \varepsilon$ implies that $m(s) = \varepsilon$ as well. Finally, let

$$\psi(s) = \underbrace{a_{i_1}}_{\ell(s)} \underbrace{a_{i_2} \cdots a_{i_n}}_{m(s)} \underbrace{\tilde{a}_{i_{n+1}}}_{r(s)}.$$

Note that for every word $w \in \Sigma^*$ we have $\psi(\varphi(w)) = w\tilde{a}_0$.

Let $w \in \Sigma^*$ and $\mathbb{B} = (N, \Sigma_2, P, S)$ be an SLP for $\varphi(w)$. For a nonterminal $A \in N$ we define $\ell(A), m(A), r(A)$ as $\ell(\mathrm{val}(A)), m(\mathrm{val}(A)), r(\mathrm{val}(A))$. We now define an SLP $\mathbb{A}'$ that contains for every nonterminal $A \in N$ a nonterminal A' such that $\mathrm{val}(A') = m(A)$. Moreover, the algorithm also computes $\ell(A)$ and $r(A)$.

We define the productions of $\mathbb{A}'$ inductively over the structure of $\mathbb{B}$. Consider a production $(A \to \alpha) \in P$, where $\alpha = w_0 A_1 w_1 A_2 \cdots w_{n-1} A_n w_n$ with $n \geq 0$,

$A_1, \ldots, A_n \in N$, and $w_0, w_1, \ldots, w_n \in \Sigma_2^*$. Let $\ell_i = \ell(A_i)$ and $r_i = r(A_i)$. The right-hand side for A' is obtained as follows. We start with the word

$$\psi(w_0)\, \ell_1\, A'_1\, r_1\, \psi(w_1)\, \ell_2\, A'_2\, r_2 \cdots \psi(w_{n-1})\, \ell_n\, A'_n\, r_n\, \psi(w_n). \tag{6}$$

Note that each of the factors $\ell_i A'_i r_i$ produces (by induction) $\psi(\mathrm{val}(A_i))$. Next we remove every A'_i that derives the empty word (which is equivalent to $m(A_i) = \varepsilon$). After this step, every occurrence of a symbol $\tilde{a}_i$ is either the last symbol of the word or it is followed by another symbol $\tilde{a}_j$ or a_j. This allows us to eliminate all occurrences of symbols $\tilde{a}_i$ except for the last symbol using the two reduction rules $\tilde{a}_i\tilde{a}_j \to \tilde{a}_{i+j}$ (which corresponds to $a^i a^j = a^{i+j}$) and $\tilde{a}_i a_j \to a_{i+j}$ (which corresponds to $a^i a^j b = a^{i+j} b$). If we perform these rules as long as possible (the order of applications is not relevant since these rules form a confluent and terminating system), only a single occurrence of a symbol $\tilde{a}_i$ at the end of the right-hand side will remain. The resulting word α' produces $\psi(A)$. Hence, we obtain the right-hand side for the nonterminal A' by removing the first symbol of α' if it is of the form a_i (this symbol is then $\ell(A)$) and the last symbol of α', which must be of the form $\tilde{a}_j$ (this symbol is $r(A)$).

Note that for the start variable S of $\mathbb{B}$ we must have $r(S) = \tilde{a}_0$ since $\mathrm{val}(S)$ belongs to the image of φ. Let $S' \to \sigma$ be the production for S' in $\mathbb{A}'$. We obtain the SLP $\mathbb{A}$ by replacing this production by $S' \to \ell(S)\sigma$. Since $\mathrm{val}_{\mathbb{A}'}(S') = m(S)$ and $\mathrm{val}_{\mathbb{B}}(S) = \varphi(w)$ we have $\mathrm{val}_{\mathbb{A}}(S') = \ell(S)m(S) = w$.

To bound the size of $\mathbb{A}$ note that the length of the word in (6) is at most $|\alpha| + 2n$. But when forming the right-hand side of A', all symbols $r_1, \ldots, r_n$ are removed from (6). Hence, $|\mathbb{A}'|$ is bounded by the size of $\mathbb{B}$ plus the total number of occurrences of nonterminals in right-hand sides of $\mathbb{B}$, which is at most $2|\mathbb{B}| - 1$ (there is at least one terminal occurrence in a right-hand side). Since $|\mathbb{A}| = |\mathbb{A}'| + 1$ we get $|\mathbb{A}| \leq 2|\mathbb{B}|$.

It is easy to observe that the runtime of the algorithm is linear. □

Example 13. Consider the production $A \to a^3ba^5A_1a^3A_2a^2b^2A_3a^2$ and assume that $\mathrm{val}(A_1) = a^2$, $\mathrm{val}(A_2) = aba^3ba$ and $\mathrm{val}(A_3) = ba^2ba^3$. Hence, when we produce the right-hand side for A' we have: $\mathrm{val}(A'_1) = \varepsilon$, $\mathrm{val}(A'_2) = a_3$, $\mathrm{val}(A'_3) = a_2$, $\ell_1 = \varepsilon$, $r_1 = \tilde{a}_2$, $\ell_2 = a_1$, $r_2 = \tilde{a}_1$, $\ell_3 = a_0$, $r_3 = \tilde{a}_3$. We start with the word

$$a_3\tilde{a}_5A'_1\tilde{a}_2\tilde{a}_3a_1A'_2\tilde{a}_1a_2a_0\tilde{a}_0a_0A'_3\tilde{a}_3\tilde{a}_2.$$

Then we replace A'_1 by ε and obtain $a_3\tilde{a}_5\tilde{a}_2\tilde{a}_3a_1A'_2\tilde{a}_1a_2a_0\tilde{a}_0a_0A'_3\tilde{a}_3\tilde{a}_2$. Applying the reduction rules finally yields $a_3a_{11}A'_2a_3a_0a_0A'_3\tilde{a}_5$. Hence, we have $\ell(A) = a_3$, $r(A) = \tilde{a}_5$ and the production for A' is $A' \to a_{11}A'_2a_3a_0a_0A'_3$.

Proof of Theorem 10. Let $\mathcal{C}$ be an arbitrary grammar-based compressor working in polynomial time such that $\alpha_{\mathcal{C}}(2, n) \leq c$. The grammar-based compressor $\mathcal{D}$ works for an input word w over an arbitrary alphabet as follows: Let $\Sigma = \{a_0, \ldots, a_{k-1}\}$ be the set of symbols that occur in w and let φ be defined as in (5). Using $\mathcal{C}$, one first computes an SLP $\mathbb{B}$ for $\varphi(w)$ such that $|\mathbb{B}| \leq c \cdot g(\varphi(w))$. Then, using Lemma 12, one computes from $\mathbb{B}$ an SLP $\mathbb{A}$ for w such that $|\mathbb{A}| \leq 2c \cdot g(\varphi(w))$. Lemma 11 implies $g(\varphi(w)) \leq 3 \cdot g(w)$ and hence $|\mathbb{A}| \leq 6c \cdot g(w)$, which proves the theorem. □

6 Open problems

Several open problems arise from this paper. First of all, it would be nice to to prove (or disprove) the lower bound $\Omega((n/\log n)^{2/3})$ for the approximation ratio of $\mathsf{LZ78}$ also for a binary alphabet. Our proof needs a ternary alphabet. Another interesting question arises from the gap between the lower bound $\Omega(\log n/\log\log n)$ and the upper bound $O(\log n)$ for $\beta_{\mathsf{LZ77}}(n)$ (worst case size of a smallest SLP in relation to the number of $\mathsf{LZ77}$-factors). It is open whether the factor $1/\log\log n$ in the lower bound is necessary. Finally, one should try to narrow also the gaps between the lower and upper bounds for the other grammar-based compressors analyzed in [4]. In particular, for the so called global algorithms from [4] these gaps are quite large.

Acknowledgment. The work in this paper was supported by the DFG grant LO 748/10-1.

References

1. J. Arpe and R. Reischuk. On the complexity of optimal grammar-based compression. In *Proc. DCC 2006*, pages 173–182. IEEE Computer Society, 2006.
2. J. Berstel and S. Brlek. On the length of word chains. *Inf. Process. Lett.*, 26(1):23–28, 1987.
3. K. Casel, H. Fernau, S. Gaspers, B. Gras, and M. L. Schmid. On the complexity of grammar-based compression over fixed alphabets. In *Proc. ICALP 2016*, Lecture Notes in Computer Science. Springer, 1996. to appear.
4. M. Charikar, E. Lehman, A. Lehman, D. Liu, R. Panigrahy, M. Prabhakaran, A. Sahai, and A. Shelat. The smallest grammar problem. *IEEE Trans. Inf. Theory*, 51(7):2554–2576, 2005.
5. A. A. Diwan. A new combinatorial complexity measure for languages. Tata Institute, Bombay, India, 1986.
6. L. Gasieniec, M. Karpinski, W. Plandowski, and W. Rytter. Efficient algorithms for Lempel-Ziv encoding (extended abstract). In *Proc. SWAT 1996*, volume 1097 of *Lecture Notes in Computer Science*, pages 392–403. Springer, 1996.
7. A. Jeż. Approximation of grammar-based compression via recompression. In *Proc. CPM 2013*, volume 7922 of *LNCS*, pages 165–176. Springer, 2013.
8. J. C. Kieffer and E.-H. Yang. Grammar-based codes: A new class of universal lossless source codes. *IEEE Trans. Inf. Theory*, 46(3):737–754, 2000.
9. J. C. Kieffer, E.-H. Yang, G. J. Nelson, and P. C. Cosman. Universal lossless compression via multilevel pattern matching. *IEEE Trans. Inf. Theory*, 46(4):1227–1245, 2000.
10. N. J. Larsson and A. Moffat. Offline dictionary-based compression. In *Proc. DCC 1999*, pages 296–305. IEEE Computer Society, 1999.
11. M. Li and P. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, Third Edition*. Springer, 2008.
12. M. Lohrey. *The Compressed Word Problem for Groups*. Springer, 2014.
13. C. G. Nevill-Manning and I. H. Witten. Identifying hierarchical strcture in sequences: A linear-time algorithm. *J. Artif. Intell. Res.*, 7:67–82, 1997.
14. W. Rytter. Application of Lempel-Ziv factorization to the approximation of grammar-based compression. *Theor. Comput. Sci.*, 302(1–3):211–222, 2003.
15. J. A. Storer and T. G. Szymanski. Data compression via textual substitution. *J. ACM*, 29(4):928–951, 1982.
16. Y. Tabei, Y. Takabatake, and H. Sakamoto. A succinct grammar compression. In *Proc. CPM 2013*, volume 7922 of *Lecture Notes in Computer Science*, pages 235–246. Springer, 2013.
17. J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Trans. Inf. Theory*, 24(5):530–536, 1977.