Compressibility of Finite Languages by Grammars

Sebastian Eberhard and Stefan Hetzl(✉)

Institute of Discrete Mathematics and Geometry, Vienna University of Technology, Wiedner Hauptstraße 8-10, 1040 Wien, Austria
sebastian.eberhard84@gmail.com, stefan.hetzl@tuwien.ac.at

Abstract. We consider the problem of simultaneously compressing a finite set of words by a single grammar. The central result of this paper is the construction of an incompressible sequence of finite word languages. This result is then shown to transfer to tree languages and (via a previously established connection between proof theory and formal language theory) also to formal proofs in first-order predicate logic.

1 Introduction

In grammar-based compression, context-free grammars that generate exactly one word are used for representing the input text. The smallest grammar problem asks for the smallest context-free grammar that generates a given word. Its decision version is known to be NP-complete [28]. However, there is a number of fast algorithms which are practically useful [16,17,19] or achieve good approximation ratios [5,14,15,23,24]. Grammar-based compression also has the considerable practical advantage that many operations can be performed directly on the compressed representation; see [18].

In this paper we consider the problem of simultaneously compressing a finite set of words by a single grammar. Our motivation for investigating this problem is rooted in proof theory and automated deduction: as shown in [7] there is an intimate relationship between a certain class of formal proofs (those with Π_1-cuts) in first-order predicate logic and a certain class of grammars (totally rigid acyclic tree grammars). In particular, the number of production rules in the grammar is a lower bound on the length of the proof. This relationship has been exploited in a method for proof compression whose central combinatorial step is a grammar-based compression of a finite tree language [8–10].

The proof-theoretic application of our work entails a shift of emphasis w.r.t. traditional grammar-based compression in the following respects: first, we do not have any freedom of choice regarding the type of grammar. Totally rigid acyclic tree grammars have to be used because they can be translated to proofs afterwards. Secondly, we are looking for a minimal grammar G s.t. $L(G) \supseteq L$ where L is the finite input language. This is the case because L describes a disjunction which is required to be a tautology (a so-called Herbrand-disjunction,

Research supported by the Vienna Science Fund (WWTF) project VRG12-04.

J. Shallit and A. Okhotin (Eds.): DCFS 2015, LNCS 9118, pp. 93–104, 2015.
DOI: 10.1007/978-3-319-19225-3_8

see [2,6]) and if $L' \supseteq L$ then L' also describes a tautology. This condition is similar to (but different from) the one imposed on cover automata [3,4]: there an automaton A is sought s.t. $L(A) \supseteq L$, but in addition it is required that $L(A) \setminus L$ consists only of words longer than any word in L. And thirdly, the complexity measure we aim at minimising is not the number of symbols in the grammar, but the number of production rules of the grammar. This is due to the fact that the number of production rules corresponds to the number of certain logical inferences in a formal proof.

Along the lines of descriptional complexity measures such as automatic complexity [26] and automaticity [25] one can consider the size of the smallest grammar that covers a language L in the above sense as the complexity of L. Then the result of [7] shows that this complexity measure is a lower bound on the length of a proof with Π_1-cuts of the Herbrand-disjunction described by L.

The central result of this paper is the construction of an incompressible sequence of finite (word) languages. This result extends to tree languages in a straightforward way, and is then used to obtain an incompressibility result for proofs with Π_1-cuts in first-order predicate logic. The length of proofs with cuts is notoriously difficult to control (for propositional logic this is considered the central open problem in proof complexity [22]). Theorem 5 below is, to the best of our knowledge, the first such incompressiblity result in proof theory.

2 Grammar-Based Compression of Finite Languages

Definition 1. *A* context-free grammar (CFG) *is a* 4*-tuple* $G = (N, \Sigma, P, S)$ *where* N *is a finite set of nonterminals,* Σ *is a finite alphabet,* $S \in N$ *is the starting symbol and* P *is a finite set of productions of the form* $A \to w$ *where* $A \in N$ *and* $w \in (\Sigma \cup N)^*$.

As usual, the one-step derivation relation $\Longrightarrow_G$ of G is defined by $u \Longrightarrow_G v$ iff there is a production $A \to w$ in G s.t. v is obtained from u by replacing an occurrence of A by w. The derivation relation $\Longrightarrow_G^*$ is the reflexive and transitive closure of $\Longrightarrow_G$ and the language of G is $L(G) = \{w \in \Sigma^* \mid S \Longrightarrow_G^* w\}$. We omit the subscript G if the grammar is clear from the context.

Definition 2. *A* right-linear grammar *is a context-free grammar* (N, Σ, P, S) *s.t. all productions in* P *are of the form* $A \to vB$ *or* $A \to v$ *for* $A, B \in N$ *and* $v \in \Sigma^*$.

It is well-known, see e.g., [11], that the languages of right-linear grammars are exactly the regular languages.

Definition 3. *Let* $G = (N, \Sigma, P, S)$ *be a context-free grammar. The ordering* $<_G^1$ *of* N *is defined as follows:* $A <_G^1 B$ *iff there is a production* $A \to w$ *in* P *s.t.* B *occurs in* w. *The ordering* $<_G$ *is defined as the transitive closure of* $<_G^1$. *We say that* G *is cyclic (respectively acyclic) iff* $<_G$ *is.*

We abbreviate "right-linear acyclic grammar" as "RLAG". Let $A \in N$; then a production whose left hand side is A is called A-production. We write P_A for the set of A-productions in P. For $N' \subseteq N$ we define $P_{N'} = \bigcup_{A \in N'} P_A$. For a language L and a CFG G we say that G covers L if $L(G) \supseteq L$. The size of a CFG $G = (N, \Sigma, P, S)$ is defined as $|G| = |P|$. The length of a right-linear production rule $A \to wB$ or $A \to w$ for $w \in \Sigma^*$ is defined as $|w|$.

Definition 4. *A finite language L is called* compressible *if there is a RLAG G which covers L and satisfies $|G| < |L|$. It is called* incompressible *otherwise.*

A variant of this problem, the equality formulation, consists in asking for a grammar G with $L(G) = L$ and $|G| < |L|$. As explained in the introduction, the cover formulation is motivated by our proof-theoretic application, see Sect. 5. However, our main result on incompressibility also applies to the equality formulation; see Corollary 1.

The choice of RLAGs for the compression of finite languages is quite natural in view of the fact that right-linear grammars generate exactly the regular languages and the observation that a right-linear grammar where every nonterminal is accessible and which does not contain trivial productions generates a finite language iff it is acyclic.

Definition 5. *A sequence $(L_n)_{n \geq 1}$ of finite languages is called* incompressible *if there is an $M \in \mathbb{N}$ s.t. for all $n \geq M$ the language L_n is incompressible. A sequence $(L_n)_{n \geq 1}$ of finite languages is called* compressible *if for every $M \in \mathbb{N}$ there is an $n \geq M$ s.t. L_n is compressible.*

Note that it is trivial to construct an incompressible sequence of languages of small size, e.g., $L_n = \{a\}$ for a letter a. It is also trivial to construct a sequence of incompressible languages in an unbounded signature, e.g., $L_n = \{a_1, \ldots, a_n\}$ for letters $a_1, a_2, \ldots$. Consequently, in this paper we will construct an incompressible sequence of languages of unbounded size over a constant alphabet.

3 Incompressible Languages

3.1 Reduced Grammars

In this section we will make some preparatory observations on the structure of RLAGs which compress finite languages, leading to the notion of strong compressibility.

Definition 6. *Let $G = (N, \Sigma, P, S)$ be a RLAG. Then a rule $A \to w$ is called* trivial *if $A = S$ and $w \in \Sigma^*$. We define $G_{\mathrm{t}} = (N, \Sigma, P_{\mathrm{t}}, S)$ where P_{t} is the set of trivial rules of G.*

We say that a word u is a *subword* of a word w if there are words v_1, v_2 s.t. $w = v_1 u v_2$. We say that a word u is a *prefix* of a word w if there is a word v s.t. $w = uv$.

Definition 7. *Let L be a finite language and G be a RLAG that covers L. Then G is called* reduced w.r.t. *L if for every non-trivial production rule $A \to wB$ or $A \to w$ of G there are distinct $u, v \in L \setminus L(G_t)$ s.t. w is a subword of both u and v.*

If the language is clear from the context we will say "reduced" instead of "reduced w.r.t. L". Intuitively, in a grammar which is reduced w.r.t. L all production rules are either trivial or useful for the compression of L. The following lemma shows that for questions of compressibility, it is sufficient to consider reduced RLAGs.

Lemma 1. *Let L be a finite language and G be a RLAG which covers L. Then there is a reduced RLAG G^* which covers L and satisfies $|G^*| \leq |G|$.*

Proof Sketch. Replace each non-trivial production rule that is only used for deriving a single word w by the trivial production $S \to w$.

Definition 8. *A language L is called* strongly compressible *if there is a reduced RLAG G without trivial rules s.t. G covers L and $|G| < |L|$. A sequence $(L_n)_{n \geq 1}$ of finite languages is called* strongly compressible *if for every $M \in \mathbb{N}$ there is an $n \geq M$ s.t. L_n is strongly compressible.*

Lemma 2. *Let L be a compressible language, then there is a language $L' \subseteq L$ which is strongly compressible.*

Proof Sketch. After obtaining a reduced RLAG G' which compresses L from Lemma 1 let L' be all words in L derivable from the non-trivial part of G'.

3.2 Segmented Languages

From this section on we will often use the alphabet $\Sigma = \{\mathbf{0}, \mathbf{1}, \mathbf{s}\}$. The letters **0** and **1** will be used for the binary representation of natural numbers, while the letter **s** will serve as a separator. The incompressible sequence of languages used for the main result of this paper will be a sequence of segmented languages, a notion which we define now and study in this section.

Definition 9. *Let $\Sigma = \{\mathbf{0}, \mathbf{1}, \mathbf{s}\}$. A word $w \in \Sigma^*$ s.t. $w = (\mathbf{s}v)^k$ for some $k \geq 1$ and some $v \in \{\mathbf{0}, \mathbf{1}\}^+$ is called* segmented word*. The word v is called the* building block *of w. Occurrences of v in w are called* segments*. A segmented word $(\mathbf{s}v)^k$ where $|v| = l$ is called a* (k, l)-segmented word*. A language consisting of (k, l)-segmented words is called a* (k, l)-segmented language*.*

The following lemma states the key property of segmented languages: long rules are not useful for compression.

Lemma 3. *Let L be a finite (k, l)-segmented language and G be a reduced RLAG that covers L. Then every non-trivial rule of G has length at most l.*

Proof Sketch. If a rule has length $l + 1$, it contains a whole building block and hence fixes a word of L.

Lemma 4. *Let L be a finite (k,l)-segmented language that is strongly compressible. Then $k < |L| - 1$.*

Proof. Note that we have $|w| = k(l+1)$ for all $w \in L$. Let G be a reduced RLAG which compresses L. Then by Lemma 3 every rule in G has length at most l. Hence for deriving a single $w \in L$ the grammar G needs at least $\frac{|w|}{l} > \frac{|w|}{l+1} = k$ rules. Since L is compressible, it is non-empty and hence $k < |G| < |L|$.

Lemma 5. *Let $(L_n)_{n\geq 1}$ be a compressible sequence of finite languages s.t. L_n is (k_n, l_n)-segmented and $(k_n)_{n\geq 1}$ is unbounded. Then there is a sequence of finite languages $(L'_n)_{n\geq 1}$ s.t. 1) $L'_n \subseteq L_n$ for all $n \geq 1$, 2) $(L'_n)_{n\geq 1}$ is strongly compressible, and 3) $(|L'_n|)_{n\geq 1}$ is unbounded.*

Proof. The pointwise application of Lemma 2 to an infinite subsequence of $(L_n)_{n\geq 1}$ that consists of compressible languages only yields a sequence $(L'_n)_{n\geq 1}$ satisfying 1 and 2. By Lemma 4 we have $k_i < |L'_i|$ for infinitely many $i \in \mathbb{N}$ which together with the assumption that $(k_n)_{n\geq 1}$ is unbounded entails 3.

The following Lemma 6 applies the uselessness of long rules for compression to provide an upper bound on the number of segments which a strongly compressing RLAG covers. This upper bound is a key ingredient of the proof of our main result.

Definition 10. *Let $G = (N, \Sigma, P, S)$ be a RLAG. Let $w \in L(G)$ be a (k,l)-segmented word with building block v and let $i \in \{1, \ldots, k\}$. Then $w = w_0 \mathbf{s} v w_1$ for $w_0 = (\mathbf{s}v)^{i-1}$ and $w_1 = (\mathbf{s}v)^{k-i}$. Let δ be a derivation of w w.r.t. G; then it is of the form*

$$S \Longrightarrow^* w'_0 A_1 \Longrightarrow w_0 \mathbf{s} v' A_2 \Longrightarrow \cdots w_0 \mathbf{s} v'' A_n \Longrightarrow w_0 \mathbf{s} v w'_1 A_{n+1} \Longrightarrow^* w$$

for some $A_1, \ldots, A_n, A_{n+1} \in N$ with v', v'' being prefixes of v, w'_0 a prefix of w_0 and w'_1 a prefix of w_1. We define $\mathrm{nonterms}(w, i, \delta) = \{A_j \mid 1 \leq j \leq n\}$.

Lemma 6. *Let L be a finite (k,l)-segmented language that is strongly compressed by a RLAG $G = (N, \Sigma, P, S)$. For each $w \in L$ fix a derivation δ_w of w w.r.t. G. Let $N_0 \subseteq N$, let $P_0 = P_{N_0}$ and let $S_0 = \{(w,i) \in L \times \{1, \ldots, k\} \mid \mathrm{nonterms}(w, i, \delta_w) \subseteq N_0\}$. Then we have $|S_0| \leq 2^{|P_0|} \cdot |P_0|$.*

Proof. For $w \in L$ define $S_{w,0} = \{i \in \{1, \ldots, k\} \mid \mathrm{nonterms}(w, i, \delta_w) \subseteq N_0\}$. By Lemma 3 every rule of G has length at most l. Due to acyclicity, each $A \in N_0$ can be used at most once in a derivation. Therefore by using all $A \in N_0$ in a derivation one can generate at most $|N_0| \cdot l$ terminal symbols, and hence at most $|N_0|$ segments. We thus obtain $|S_{w,0}| \leq |N_0|$.

Furthermore, define $L_0 = \{w \in L \mid \exists i \text{ s.t. } (w,i) \in S_0\}$. Let $P^* \subseteq P_0$ s.t. P^* contains exactly one production for each nonterminal of N_0 and note that there are at most $2^{|P_0|}$ such P^*. If P^* permits deriving a word that contains a subword $v \in \{\mathbf{0}, \mathbf{1}\}^l$, then the choice of P^* uniquely determines a word $w \in L$. If P^* does not allow deriving such a word, then P^* may be used in the derivations δ_w of several $w \in L$; however, it does not contribute to any of its $S_{w,0}$. Therefore we have $|L_0| \leq 2^{|P_0|}$. Putting these two results together, we obtain $|S_0| = \sum_{w \in L_0} |S_{w,0}| \leq |L_0| \cdot |N_0| \leq 2^{|P_0|} \cdot |P_0|$.

3.3 Ordered Grammars

In a RLAG G the ordering $<_G$ is acyclic but, in general, not linear. For technical purposes it will be useful to fix a linearisation of $<_G$ and a corresponding linear order of the productions of G. To that aim we introduce the notion of ordered grammar.

Definition 11. *A* right-linear ordered grammar (RLOG) *is a tuple* $G = (N, \Sigma, P, A_1)$ *where* N *is a list* $A_1, \dots, A_n$ *of nonterminals,* P *is a list* $p_1, \dots, p_m$ *of productions s.t.*

1. $G' = (\{A_1, \dots, A_n\}, \Sigma, \{p_1, \dots, p_m\}, A_1)$ *is a RLAG,*
2. *if* $A_i <_{G'} A_j$ *then* $i < j$*, and*
3. $p_1, \dots, p_m = q_{1,1}, \dots, q_{1,k_1}, \dots, q_{n,1}, \dots, q_{n,k_n}$ *where* $\{q_{i,1}, \dots, q_{i,k_i}\} = P_{A_i}$ *for all* $i \in \{1, \dots, n\}$.

We say that an RLOG compresses a language L, is reduced w.r.t. L, etc., if the underlying RLAG fulfils the respective property.

Definition 12. *Let* $G = ((A_1, \dots, A_n), \Sigma, P, S)$ *be a RLOG. Let* $w \in L(G)$ *be a* (k, l)*-segmented word, let* $i \in \{1, \dots k\}$ *and let* δ *be a derivation of* w *w.r.t.* G*. Let* $m_1 = \min\{j \in \{1, \dots, n\} \mid A_j \in \operatorname{nonterms}(w, i, \delta)\}$ *and* $m_2 = \max\{j \in \{1, \dots, n\} \mid A_j \in \operatorname{nonterms}(w, i, \delta)\}$ *and define* $\operatorname{cost}(w, i, \delta) = \sum_{j=m_1}^{m_2} |P_{A_j}|$.

Note that the cost of the i-th segment of a word w also takes those nonterminals into account which are not used in the derivation δ of w. The following lemma shows that in a strongly compressed segmented language, many segments are cheap.

Lemma 7. *Let* L *be a finite* (k, l)*-segmented language and let* G *be a RLOG that strongly compresses* L*. Let* $w \in L$ *and* δ *be a derivation of* w *w.r.t.* G*. Then for at least half of the* $i \in \{1, \dots, k\}$*, we have* $\operatorname{cost}(w, i, \delta) < \frac{4|L|}{k}$.

Proof. As G compresses L, it covers L, so by Lemma 3 every rule of G has length at most l. Hence each rule of G can contribute to the costs of at most two segments of w, so we have $2|G| \geq \sum_{i=1}^{k} \operatorname{cost}(w, i, \delta)$. Now suppose that $\lceil \frac{k}{2} \rceil$ segments of w have cost at least $\frac{4|L|}{k}$ each, then $\sum_{i=1}^{k} \operatorname{cost}(w, i, \delta) \geq \lceil \frac{k}{2} \rceil \cdot \frac{4|L|}{k} \geq 2|L|$, which is a contradiction to $|G| < |L|$.

Definition 13. *Let* $G = (N, \Sigma, (p_1, \dots, p_m), A_1)$ *be a RLOG and let* $s < m$*. For* $A \in N$ *define* $\operatorname{pmin}(A) = \min\{j \mid p_j \in P_A\}$*, and* $\operatorname{pmax}(A) = \max\{j \mid p_j \in P_A\}$*. Furthermore, for* $j \in \{1, \dots, \lceil \frac{m}{s} \rceil - 1\}$ *define* $N_j = \{A \in N \mid (j-1)s \leq \operatorname{pmin}(A)$ *and* $\operatorname{pmax}(A) < (j+1)s\}$*. We say that* $(N_j)_{j=1}^{\lceil \frac{m}{s} \rceil - 1}$ *is the* s-covering of G.

Note that N_j and N_{j+1} can overlap, but N_j and N_{j+2} can not. Furthermore, note that $|P_{N_j}| \leq 2s$ for all $j \in \{1, \dots, \lceil \frac{m}{s} \rceil - 1\}$. The following lemma applies Lemma 7 to obtain a lower bound on the number of segments covered by the productions of a single N_j.

Lemma 8. *Let L be a finite (k,l)-segmented language, let G be a RLOG which strongly compresses L and let $|G| > s \geq \frac{4|L|}{k}$. Let $N_1, \ldots, N_{\lceil \frac{|G|}{s} \rceil - 1}$ be the s-covering of G. Let $w \in L$ and δ be a G-derivation of w. Then for at least half of the $i \in \{1, \ldots, k\}$ there is a $j \in \{1, \ldots, \lceil \frac{|G|}{s} \rceil - 1\}$ s.t.* $\mathrm{nonterms}(w, i, \delta) \subseteq N_j$.

Proof. By Lemma 7 at least half of the $i \in \{1, \ldots, k\}$ have $\mathrm{cost}(w, i, \delta) < \frac{4|L|}{k}$. Let i be s.t. $\mathrm{cost}(w, i, \delta) < \frac{4|L|}{k}$, then $\mathrm{cost}(w, i, \delta) < s$. Let $A_0 \in N$ be the nonterminal used for entering the i-th segment of w in δ and let $j_0 = \max\{j \in \{1, \ldots, \lceil \frac{|G|}{s} \rceil - 1\} \mid A_0 \in N_j\}$. If $j_0 = \lceil \frac{|G|}{s} \rceil - 1$, then $\mathrm{nonterms}(w, i, \delta) \subseteq N_{j_0}$ because $(N_j)_{j=1}^{\lceil \frac{|G|}{s} \rceil - 1}$ covers all nonterminals and N_{j_0} is the last element of this list. If $j_0 < \lceil \frac{|G|}{s} \rceil - 1$, then $\mathrm{pmin}(A_0) < j_0 s$, for if $\mathrm{pmin}(A_0) \geq j_0 s$, then $A_0 \in N_{j_0+1}$. Therefore $\mathrm{pmin}(A_0) + \mathrm{cost}(w, i, \delta) < \mathrm{pmin}(A_0) + s < (j_0 + 1)s$; hence $\mathrm{nonterms}(w, i, \delta) \subseteq N_{j_0}$.

3.4 The Main Result

For $n \geq 1$ and $k \in \{0, \ldots, 2^n - 1\}$ we write $\mathrm{b}_n k \in \{\mathbf{0}, \mathbf{1}\}^n$ for the n-bit binary representation of k.

Definition 14 (Incompressible sequence). *For all $n \geq 1$ define $l(n) = \lceil \log(n) \rceil$, $k(n) = \lceil \frac{9n}{l(n)+1} \rceil$, and $L_n = \{(\mathbf{s}\mathrm{b}_{l(n)} i)^{k(n)} \mid 0 \leq i \leq n-1\}$.*

Note that $l(n)$ is the number of bits required to represent all elements of $\{0, \ldots, n-1\}$ in binary. Note furthermore that for every $n \geq 1$, we have $|L_n| = n$ and all words in L_n have the same length $k(n)(l(n)+1)$. The number of segments $k(n)$ has been chosen s.t. $k(n)(l(n)+1)$ is $9n$ padded up to the next multiple of $l(n)+1$; hence the length of the words in L_n grows linearly in n.

Example 1. For $n = 5$ we have $l(5) = 3$, $k(5) = 12$ and $L_5 = \{(\mathbf{s000})^{12}, (\mathbf{s001})^{12}, (\mathbf{s010})^{12}, (\mathbf{s011})^{12}, (\mathbf{s100})^{12}\}$.

Theorem 1. *$(L_n)_{n \geq 1}$ is incompressible.*

The proof strategy for this theorem is as follows: both Lemmas 6 and 8 assume a strongly compressed segmented language. But while Lemma 6 states an upper bound on the number of segments covered by a certain part of a strongly compressing grammar, Lemma 8 provides a lower bound on the number of segments covered by the productions of a single N_j. The following proof will show these two bounds to be inconsistent, thus deriving the incompressibility of $(L_n)_{n \geq 1}$.

Proof. Suppose that $(L_n)_{n \geq 1}$ is compressible. Then by Lemma 5 there is a sequence $(L'_n)_{n \geq 1}$ which is strongly compressibly by a sequence $(G_n)_{n \geq 1}$ of RLAGs which we consider as RLOGs $(G'_n)_{n \geq 1}$ by fixing an arbitrary linear order satisfying Definition 11. Let us fix for every $n \geq 1$ and every $w \in L'_n$ a derivation δ_w of w w.r.t. G'_n. This is well-defined, since the L'_n are disjoint, and hence δ_w does not depend on n.

First note that for all $n \geq 1$ we have $k(n) = \lceil \frac{9n}{\lceil \log(n) \rceil + 1} \rceil \geq \frac{9n}{\log(n)+2}$, and since $n \geq |L'_n|$ we have

$$k(n) \geq \frac{9|L'_n|}{\log(|L'_n|) + 2}. \tag{1}$$

Therefore $\frac{4|L'_n|}{k(n)} \leq \frac{4}{9}(\log(|L'_n|)+2) =: s_n$. Let $N_1, \dots N_{\lceil \frac{|G'_n|}{s_n} \rceil - 1}$ be the s_n-covering of G'_n and define $U_n := |\{(w,i) \in L'_n \times \{1,\dots,k(n)\} \mid \exists j \text{ s.t. nonterms}(w,i,\delta_w) \subseteq N_j\}|$. By Lemma 8 we have $U_n \geq \frac{|L'_n| \cdot k(n)}{2}$, which, together with Theorem (1), entails

$$U_n \geq \frac{9|L'_n|^2}{2(\log(|L'_n|) + 2)}. \tag{2}$$

On the other hand, applying Lemma 6 to all N_j for $j = 1, \dots, \lceil \frac{|G'_n|}{s_n} \rceil - 1$ and summing up yields $U_n \leq \sum_{j=1}^{\lceil \frac{|G'_n|}{s_n} \rceil - 1} 2^{|P_{N_j}|} \cdot |P_{N_j}| \leq (\lceil \frac{|G'_n|}{s_n} \rceil - 1) \cdot 2^{2s_n} \cdot 2s_n$. We have $2^{2s_n} \cdot 2s_n \leq C|L'_n|^{\frac{8}{9}}(\log(|L'_n|)+2)$ for some $C \in \mathbb{N}$ and $\lceil \frac{|G'_n|}{s_n} \rceil - 1 \leq \frac{|L'_n|}{s_n} = \frac{9|L'_n|}{4(\log(|L'_n|)+2)}$ and therefore

$$U_n \leq D|L'_n|^{\frac{17}{9}} \text{ for some } D \in \mathbb{N}. \tag{3}$$

Putting Theorem (2) and (3) together we obtain

$$|L'_n|^2 \leq E|L'_n|^{\frac{17}{9}}(\log(|L'_n|) + 2) \text{ for some } E \in \mathbb{N}. \tag{4}$$

But by Lemma 5 the function $n \mapsto |L'_n|$ is unbounded. Hence there is an $M \in \mathbb{N}$ s.t. for all $n \geq M$ the inequality Theorem (4) is not satisfied. This is a contradiction.

3.5 Remarks

Every sequence of languages which is incompressible in the cover formulation is also incompressible in the (more restricted) equality formulation. Therefore we immediately obtain the following corollary from Theorem 1.

Corollary 1. *There is no sequence $(G_n)_{n\geq 1}$ of RLAGs and $M \in \mathbb{N}$ s.t. $L(G_n) = L_n$ and $|G_n| < |L_n|$ for all $n \geq M$.*

On the other hand, the sequence $(L_n)_{n\geq 1}$ can be compressed by stronger formalisms:

Theorem 2. *There is a sequence $(G_n)_{n\geq 1}$ of acyclic CFGs which compresses $(L_n)_{n\geq 1}$.*

Proof. Let $G_n = (\{S, A_1, \dots, A_{l(n)}\}, \{\mathbf{0}, \mathbf{1}, \mathbf{s}\}, P_n, S)$ where

$$P_n = \{S \to (\mathbf{s}A_1)^{k(n)},\ A_1 \to \mathbf{0}A_2 \mid \mathbf{1}A_2,\ \dots,\ A_{l(n)} \to \mathbf{0} \mid \mathbf{1}\}.$$

Then $L(G_n) \supseteq L_n$ for all $n \geq 1$ and $|G_n| = 2\lceil \log(n) \rceil + 1 < n = |L_n|$ for all $n \geq M$ for a certain M.

The length of the words in L_n grows linearly. Under the condition that $|L_n| = n$ this is the best possible:

Theorem 3. *Let $(L'_n)_{n\geq 1}$ be a sequence of finite languages over a finite alphabet $\Sigma = \{a_1, \dots a_k\}$ s.t. $|L'_n| = n$ and s.t. there is a sublinear function that bounds the maximal length l_n of a word in L'_n. Then $(L'_n)_{n\geq 1}$ is compressible.*

Proof. Let $G_n = (\{A_1, \dots, A_{l_n}\}, \Sigma, P_n, A_1)$ where

$$P_n = \{A_1 \to a_1 A_2 \mid \cdots \mid a_k A_2 \mid A_2,\ \dots,\ A_{l_n} \to a_1 \mid \cdots \mid a_k \mid \varepsilon\}.$$

Then $L(G_n) = \Sigma^{\leq l_n} \supseteq L'_n$ and $|G_n| = (k+1) \cdot l_n$ which, from a certain $M \in \mathbb{N}$ on, is less than $|L'_n| = n$.

4 Application to Tree Languages

In this section we will transfer the main theorem to tree languages. The grammars we will be considering for the compression of finite tree languages are totally rigid acyclic tree grammars. Rigid tree languages were introduced in [12,13] with applications in verification in mind. A presentation of this class of languages via rigid grammars was given in [7].

For a ranked alphabet (i.e., a term signature) Σ we write $\mathcal{T}(\Sigma)$ for the set of all terms built from function and constant symbols of Σ.

Definition 15. *A* regular tree grammar *is a tuple $G = (N, \Sigma, P, S)$ where N is a set of nonterminals of arity 0, Σ is a term signature, $S \in N$ is the starting symbol and P is a finite set of productions of the form $A \to t$ where $A \in N$ and $t \in \mathcal{T}(\Sigma \cup N)$.*

The ordering $<_G$ of nonterminals is defined analogously to the case of word grammars. Hence we can speak about acyclic regular tree grammars. As usual for tree grammars, a derivation is a finite list of terms $t_1, \dots, t_n$ s.t. t_{i+1} is obtained from t_i by applying a production rule to a single position. A derivation w.r.t. a grammar $G = (N, \Sigma, P, S)$ is said to satisfy the rigidity condition if for every nonterminal $A \in N$ it uses at most one A-production.

Definition 16. *A* totally rigid acyclic tree grammar (TRATG) *is an acyclic regular tree grammar $G = (N, \Sigma, P, S)$ whose language $L(G)$ is the set of all $t \in \mathcal{T}(\Sigma)$ that have a derivation from S satisfying the rigidity condition.*

(In)compressibility of tree languages and sequences thereof is defined analogously to the case of word languages replacing RLAGs with TRATGs.

Definition 17. *For an alphabet Σ define $\Sigma^{\mathrm{T}} = \{f_a/1 \mid a \in \Sigma\} \cup \{e\}$. For $w \in \Sigma^*$ define the term w^{T} recursively by $\varepsilon^{\mathrm{T}} = e$, and $(av)^{\mathrm{T}} = f_a(v^{\mathrm{T}})$. For $L \subseteq \Sigma^*$ define $L^{\mathrm{T}} = \{w^{\mathrm{T}} \mid w \in L\}$.*

Theorem 4. *The sequence of tree languages $(L_n^{\mathrm{T}})_{n\geq 1}$ is incompressible.*

Proof Sketch. Starting from the assumption that $(L_n^{\mathrm{T}})_{n\geq 1}$ is compressible, transform the compressing TRATGs into RLAGs compressing $(L_n)_{n\geq 1}$ hence arriving at a contradiction to Theorem 1.

5 Application to Proof Theory

A *sequent* is an expression of the form $\Gamma \vdash \Delta$ where Γ and Δ are finite sets of formulas in first-order predicate logic. The intended interpretation of a sequent $\Gamma \vdash \Delta$ is the formula $(\bigwedge_{\varphi\in\Gamma} \varphi) \to (\bigvee_{\psi\in\Delta} \psi)$. The logical complexity $\|\Gamma \vdash \Delta\|$ of a sequent is the number of logical connectives it contains.

A *proof* is a tree whose nodes are sequents which is built according to certain logical inference rules. The leaves are of the form $A \vdash A$. An important inference rule is the so-called *cut rule*:

$$\frac{\Gamma \vdash \Delta, A \quad A, \Pi \vdash \Lambda}{\Gamma, \Pi \vdash \Delta, \Lambda}\,\text{cut}$$

For a complete list of inference rules of the sequent calculus, the interested reader is referred to [1]. The length of a proof π, written as $|\pi|$, is the number of inferences in π. The cut rule formalises the use of a lemma in mathematical practice and is of particular importance here because it allows compressing proofs in first-order logic non-elementarily [20,21,27]. A cut is said to be a Π_1-cut if its cut formula A is of the form $\forall x\, B$ for B quantifier-free.

The following result is a proof-theoretic corollary of the incompressiblity theorem proved in Sect. 3 and extended to tree languages in Sect. 4. It should be stressed that the proof-theoretic techniques for deriving it from Theorem 4 are simple standard techniques, the details of which are omitted from this paper for space reasons.

Theorem 5. *There is a sequences of sequents $(S_n)_{n\geq 1}$ such that $\|S_n\|$ is constant, there is a cut-free proof of S_n with $O(2^{2n})$ inferences and there is $M \in \mathbb{N}$ s.t. for all $n \geq M$: every proof with Π_1-cuts of S_n has at least 2^n inferences.*

Proof Sketch. For $i \geq 1$ define the sequent $R_i :=$

$$\forall y\forall v\, P(0,y,e,v), \forall x\forall y\forall u\forall v\,(P(x,f_0(y),u,v) \to P(x,y,f_0(u),v)),$$
$$\forall x\forall y\forall u\forall v\,(P(x,f_1(y),u,v) \to P(x,y,f_1(u),v)),$$
$$\forall x\forall y\forall u\forall v\,(P(x,f_s(y),u,v) \to P(x,y,u,v)), \forall x\forall y\forall v\,(P(x,y,v,v) \to P(s(x),y,e,v)),$$
$$\forall v\,(P(\overline{k(n)},e,e,v) \to Q(\overline{1},v)), \forall x\forall v\,((Q(x,f_0(v)) \wedge Q(x,f_1(v))) \to Q(s(x),v))$$
$$\vdash Q(\overline{l(n)},e)$$

where $\overline{n}$ for $n \in \mathbb{N}$ denotes the term $s^n(0)$. Define $(S_n)_{n\geq 1}$ by $S_n = R_{2^n}$. Then $\|S_n\|$ is constant and S_n has a straightforward cut-free proof with $O(2^{2n})$ inferences. Furthermore, every cut-free proof of S_n must instantiate the quantifier $\forall y$ in $\forall y\forall v\, P(0,y,e,v)$ with all terms in $L^{\mathrm{T}}_{2^n}$. The lower bound on the proofs with cuts then follows immediately from Theorem 4 and a suitable (but straightforward) generalisation of Theorem 22 in [7].

6 Conclusion

We have investigated the problem of simultaneously compressing a finite set of words by a right-linear acyclic grammar. We have constructed an incompressible

sequence of languages and applied it to obtain an incompressibility-result in proof theory.

This problem of simultaneous compression has received only little attention in the literature so far and consequently there is a number of interesting open questions, for example: what is the complexity of the smallest grammar problem in this setting? How difficult is the approximation of the smallest grammar? Can approximation algorithms and techniques be carried over from the case of one word to this setting? How does the situation change when we do not minimise the number of production rules but the symbol complexity of the grammar?

Fast approximation algorithms for computing a minimal RLAG (or TRATG) that covers a given finite input language would also be of high practical value in the cut-introduction method [9,10] and its implementation [8].

Acknowledgments. The authors would like to thank Manfred Schmidt-Schauß for several helpful conversations about the topic of this paper, Werner Kuich for a number of remarks that improved the presentation of the results, and the anonymous reviewers for numerous important comments and suggestions.

References

1. Buss, S.: An Introduction to proof theory. In: Buss, S. (ed.) The Handbook of Proof Theory, pp. 2–78. Elsevier, Amsterdam (1998)
2. Buss, S.R.: On Herbrand's theorem. In: Leivant, D. (ed.) LCC'94. LNCS, vol. 960, pp. 195–209. Springer, Heidelberg (1995)
3. Câmpeanu, C., Sântean, N., Yu, S.: Minimal cover-automata for finite languages. In: Champarnaud, J.-M., Maurel, D., Ziadi, D. (eds.) WIA 1998. LNCS, vol. 1660, pp. 43–56. Springer, Heidelberg (1999)
4. Câmpeanu, C., Santean, N., Yu, S.: Minimal cover-automata for finite languages. Theoret. Comput. Sci. **267**(1–2), 3–16 (2001)
5. Charikar, M., Lehman, E., Liu, D., Panigrahy, R., Prabhakaran, M., Sahai, A., Shelat, A.: The smallest grammar problem. IEEE Trans. Inf. Theory **51**(7), 2554–2576 (2005)
6. Herbrand, J.: Recherches sur la théorie de la démonstration. Ph.D. thesis, Université de Paris (1930)
7. Hetzl, S.: Applying tree languages in proof theory. In: Dediu, A.-H., Martín-Vide, C. (eds.) LATA 2012. LNCS, vol. 7183, pp. 301–312. Springer, Heidelberg (2012)
8. Hetzl, S., Leitsch, A., Reis, G., Tapolczai, J., Weller, D.: Introducing quantified cuts in logic with equality. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) IJCAR 2014. LNCS, vol. 8562, pp. 240–254. Springer, Heidelberg (2014)
9. Hetzl, S., Leitsch, A., Reis, G., Weller, D.: Algorithmic introduction of quantified cuts. Theoret. Comput. Sci. **549**, 1–16 (2014)
10. Hetzl, S., Leitsch, A., Weller, D.: Towards algorithmic cut-introduction. In: Bjørner, N., Voronkov, A. (eds.) LPAR-18 2012. LNCS, vol. 7180, pp. 228–242. Springer, Heidelberg (2012)
11. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, Cambridge (1979)

12. Jacquemard, F., Klay, F., Vacher, C.: Rigid tree automata. In: Dediu, A.H., Ionescu, A.M., Martín-Vide, C. (eds.) LATA 2009. LNCS, vol. 5457, pp. 446–457. Springer, Heidelberg (2009)
13. Jacquemard, F., Klay, F., Vacher, C.: Rigid tree automata and applications. Inf. Comput. **209**, 486–512 (2011)
14. Jeż, A.: Approximation of grammar-based compression via recompression. In: Fischer, J., Sanders, P. (eds.) CPM 2013. LNCS, vol. 7922, pp. 165–176. Springer, Heidelberg (2013)
15. Jeż, A.: A *really* simple approximation of smallest grammar. In: Kulikov, A.S., Kuznetsov, S.O., Pevzner, P. (eds.) CPM 2014. LNCS, vol. 8486, pp. 182–191. Springer, Heidelberg (2014)
16. Kieffer, J.C., Yang, E.H.: Grammar-based codes: a new class of universal lossless source codes. IEEE Trans. Inf. Theory **46**(3), 737–754 (2000)
17. Larsson, N.J., Moffat, A.: Offline dictionary-based compression. In: Data Compression Conference (DCC 1999). pp. 296–305. IEEE Computer Society (1999)
18. Lohrey, M.: Algorithmics on SLP-compressed strings: a survey. Groups Complex. Cryptol. **4**(2), 241–299 (2012)
19. Nevill-Manning, C.G., Witten, I.H.: Identifying hierarchical structure in sequences: a linear-time algorithm. J. Artif. Intell. Res. **7**, 67–82 (1997)
20. Orevkov, V.: Lower bounds for increasing complexity of derivations after cut elimination. Zapiski Nauchnykh Seminarov Leningradskogo Otdeleniya Matematicheskogo Instituta **88**, 137–161 (1979)
21. Pudlák, P.: The Lengths of proofs. In: Buss, S. (ed.) Handbook of Proof Theory, pp. 547–637. Elsevier, Amsterdam (1998)
22. Pudlák, P.: Twelve problems in proof complexity. In: Hirsch, E.A., Razborov, A.A., Semenov, A., Slissenko, A. (eds.) Computer Science – Theory and Applications. LNCS, vol. 5010, pp. 13–27. Springer, Heidelberg (2008)
23. Rytter, W.: Application of Lempel-Ziv factorization to the approximation of grammar-based compression. Theoret. Comput. Sci. **302**(1–3), 211–222 (2003)
24. Sakamoto, H.: A fully linear-time approximation algorithm for grammar-based compression. J. Discrete Algorithms **3**(2–4), 416–430 (2005)
25. Shallit, J., Breitbart, Y.: Automaticity I: properties of a measure of descriptional complexity. J. Comput. Syst. Sci. **53**, 10–25 (1996)
26. Shallit, J., Wang, M.W.: Automatic complexity of strings. J. Automata, Lang. Comb. **6**(4), 537–554 (2001)
27. Statman, R.: Lower bounds on Herbrand's theorem. Proc. Am. Math. Soc. **75**, 104–107 (1979)
28. Storer, J.A., Szymanski, T.G.: The macro model for data compression (extended abstract). In: Proceedings of the Tenth Annual ACM Symposium on Theory of Computing (STOC '78). pp. 30–39. ACM, New York (1978)