

More on the Size of Higman-Haines Sets: Effective Constructions*

Hermann Gruber

Institut für Informatik, Ludwig-Maximilians-Universität München

Oettingenstr. 67, 80538 München, Germany

gruberh@tcs.ifi.lmu.de

Markus Holzer

Institut für Informatik, Technische Universität München

Boltzmannstr. 3, 85748 Garching bei München, Germany

holzer@in.tum.de

Martin Kutrib

Institut für Informatik, Universität Giessen

Arndtstr. 2, 35392 Giessen, Germany

kutrib@informatik.uni-giessen.de

Abstract. A not so well-known result in formal language theory is that the Higman-Haines sets for *any* language are regular [11, Theorem 4.4]. It is easily seen that these sets *cannot* be effectively computed in general. The Higman-Haines sets are the languages of all scattered subwords of a given language as well as the sets of all words that contain some word of a given language as a scattered subword. Recently, the exact level of unsolvability of Higman-Haines sets was studied in [8]. Here we focus on language families whose Higman-Haines sets are effectively constructible. In particular, we study the size of descriptions of Higman-Haines sets for the lower classes of the Chomsky hierarchy, namely for the family of regular, linear context-free, and context-free languages. We prove upper and lower bounds on the size of descriptions of these sets for general and unary languages.

*This paper is completely revised and expanded version of a paper presented at the 5th International Conference on Machines, Computations and Universality (MCU) held in Orléans, France, September 10–13, 2007.

1. Introduction

Higman's Lemma [11] and its generalization, namely Kruskal's Tree Theorem [14], can be used to show that certain rewriting systems terminate. Nevertheless, the result of Higman is not so well known and was frequently rediscovered in the literature, see, e.g., [9, 16, 17]. Although Higman's result appears to be only of theoretical interest, it has some nice applications in formal language theory. It seems that one of the first applications has been given by Haines in [9, Theorem 3], where it is shown that the set of all scattered subwords, that is, the *Higman-Haines set* $\text{DOWN}(L) = \{v \in A^* \mid \text{there exists } w \in L \text{ such that } v \leq w\}$, and the set of all words that contain some word of a given language, that is, the *Higman-Haines set* $\text{UP}(L) = \{v \in A^* \mid \text{there exists } w \in L \text{ such that } w \leq v\}$, are both regular for *any* language $L \subseteq A^*$. Here, \leq refers to the scattered subword relation. As pointed out in [9], this is an exceptional property, which is quite unexpected. Further applications and generalizations of Higman's result can be found in, e.g., [5, 6, 13, 16].

It is worth mentioning that $\text{DOWN}(L)$ and $\text{UP}(L)$ cannot be obtained constructively in general. This is obvious, because L is empty if and only if $\text{DOWN}(L)$ and $\text{UP}(L)$ are empty, but the question whether or not a language is empty is undecidable for recursively enumerable languages and decidable for regular ones. Thus, as expected, for the family of recursively enumerable languages the Higman-Haines sets are not constructible, while it is not hard to see that for regular languages the construction becomes effective. But where exactly is the borderline between language families with non-constructive and constructive Higman-Haines sets? One might expect that, e.g., the family of context-free languages has non-constructive Higman-Haines sets, but surprisingly this is not the case, as has been shown in [17]. On the other hand, recently it was shown in [8] that, in general, the family of Church-Rosser languages has non-constructive Higman-Haines sets. This language family lies in between the regular languages and the growing context-sensitive languages, but is incomparable to the family of context-free languages [1]. Moreover, in [8] the exact level of unsolvability of the Higman-Haines sets for certain language families was studied. Further, recursion theoretic results on the down-set of certain language families can be found in the recent paper [4]. Thus, the non-constructive side of Higman-Haines sets is well studied. But besides the results in [17] there is only little known about constructibility issues. Except for some results about regular languages accepted by nondeterministic finite automata in [8] the same is true for descriptive complexity issues. This is the starting point of our investigations on Higman-Haines sets whose sizes of description are effectively computable. In particular we consider the problem of computing the Higman-Haines sets induced by the families of regular, context-free, and linear context-free languages. For the size of the Higman-Haines sets generated by regular languages upper and lower bounds are presented. That is, we prove that an exponential blow-up is sufficient and necessary in the worst case for a deterministic finite automaton to accept the Higman-Haines set $\text{DOWN}(L)$ or $\text{UP}(L)$ generated by some language that is represented by another deterministic finite automaton. This nicely contrasts the result about nondeterministic finite automata where a matching upper and lower bound of the same size as the given automaton has been derived in [8]. Furthermore, we investigate the descriptive complexity of the Higman-Haines sets when the underlying device is a context-free or linear context-free grammar, where we obtain results on general and unary languages. We obtain non-trivial upper and lower bounds for these problems.

The paper is organized as follows. The next section contains preliminaries and basics about Higman-Haines sets. In Section 3 we first summarize the known upper and lower bounds for nondeterministic finite automata [8]. Then we study the sizes of description of the Higman-Haines set for regular lan-

languages in terms of deterministic finite automata. The effect of changing the size measure to the number of transitions of the nondeterministic finite automata is briefly discussed. Furthermore, Higman-Haines sets induced by context-free and linear context-free languages are investigated. In most cases we obtain matching upper and lower bounds in the order of magnitude. Finally, we conclude with some open problems.

2. Preliminaries

We denote the set of *non-negative integers* by \mathbb{N} . The *powerset* of a set S is denoted by 2^S . For an alphabet A , let A^+ be the set of non-empty words w over A . If the *empty word* λ is included, then we use the notation A^* . For the *length* of w we write $|w|$. For the *number of occurrences* of a symbol a in w we use the notation $|w|_a$. Set inclusion is denoted by \subseteq , and strict set inclusion by \subset . Let $v, w \in A^*$ be words over alphabet A . We write $v \leq w$ if there are words v_1, v_2, \dots, v_k and w_1, w_2, \dots, w_{k+1} , for some $k \geq 1$, $v_i \in A^*$, $w_i \in A^*$, such that $v = v_1 v_2 \dots v_k$ and $w = w_1 v_1 w_2 v_2 \dots w_k v_k w_{k+1}$. In case of $v \leq w$ we say that v is a scattered subword of w . Let L be a language over alphabet A . Then

$$\text{DOWN}(L) = \{v \in A^* \mid \text{there exists } w \in L \text{ such that } v \leq w\}$$

and

$$\text{UP}(L) = \{v \in A^* \mid \text{there exists } w \in L \text{ such that } w \leq v\}$$

are the *Higman-Haines sets* generated by L .

The next theorem is the surprising result of Haines. It has been shown about half a century ago. Actually, it is a corollary of Higman's work, but let us state it as a theorem.

Theorem 2.1. (**[9, 11]**) Let L be an arbitrary language. Then both $\text{DOWN}(L)$ and $\text{UP}(L)$ are regular.

In order to talk about the economy of descriptions we first have to define what is meant by the size of automata and grammars. In general, we are interested to measure the length of the string that defines an automaton or grammar. In particular, we sometimes use more convenient and common size measures, if there is a recursive upper bound for the length of the defining string dependent on the chosen size measure. For example, for context-free grammars M , the size $|M|$ equals the total number of occurrences of terminal and non-terminal symbols in the productions. For deterministic and nondeterministic finite automata M , the size $|M|$ equals the product of the number of states and the number of input symbols.

3. Effective Higman-Haines Set Sizes

This section is three-fold. First we turn to the family of regular languages and then to the family of context-free languages, whose Higman-Haines sets can effectively be constructed [17]. Finally we consider the special case of unary languages. In these subsections we are interested in the constructions itself as well as in the sizes of description of the Higman-Haines sets.

3.1. Regular Languages

Let $M = (S, A, \delta, s_0, F)$ be a nondeterministic finite automaton (NFA), where S is the finite set of *internal states*, A is the finite set of *input symbols*, $s_0 \in S$ is the *initial state*, $F \subseteq S$ is the set of

accepting states, and $\delta : S \times (A \cup \{\lambda\}) \rightarrow 2^S$ is the *partial transition function*. An NFA is deterministic (DFA) if and only if $|\delta(s, a)| \leq 1$, $|\delta(s, \lambda)| \leq 1$, and $|\delta(s, a)| = 1 \iff |\delta(s, \lambda)| = 0$, for all $s \in S$ and $a \in A$.

Without loss of generality, we assume that the finite automata are always *reduced*. This means that there are no unreachable states and that from any state an accepting state can be reached.

Concerning the size of an NFA accepting $\text{DOWN}(L(M))$ or $\text{UP}(L(M))$ of a given NFA language, one finds the following situation, which was proven in [8].

Lemma 3.1. Let M be an NFA of size $n \geq 1$. Then size n is sufficient and necessary in the worst case for an NFA M' to accept $\text{DOWN}(L(M))$ or $\text{UP}(L(M))$. The NFA M' can effectively be constructed.

The tight bounds shown for the sizes are not too complicated. So, the natural question for bounds based on different reasonable measures raises immediately. It turns out that the situation is different, if the number of defined transitions, i.e., the number of edges in the transition graph, is used to measure the size of NFAs. Recall the construction of the NFA for the language $\text{DOWN}(L(M))$. At first the transition function δ of M is replaced by δ_1 , where δ_1 provides all transitions of δ and, in addition, λ -transitions whenever δ provides a non- λ -transition:

$$\forall s \in S, a \in A : \delta_1(s, a) = \delta(s, a) \quad \text{and} \quad \forall s \in S : \delta_1(s, \lambda) = \delta(s, \lambda) \cup \bigcup_{a \in A} \delta(s, a).$$

So, given an input v from $\text{DOWN}(L(M))$ such that $v \leq w$ and $w \in L(M)$, the new NFA simulates M on w in such a way that it guesses the missing input symbols and performs the corresponding transitions of M as λ -transitions. Moreover, since the NFA is still reduced, there is an accepting λ -path from every state. Therefore, we can define any state to be an accepting state. Moreover, we safely may delete all λ -transitions from a state to itself. It is easy to see that the new NFA accepts $\text{DOWN}(L(M))$.

A closer look reveals that the size can be optimized. If there appears a cycle in M , then there appears a cycle of λ -transitions in the new NFA. In this case all states on the cycle can be merged into one state which allows all outgoing transitions of the merged states and gets all incoming transitions to the merged states. On the other hand, if there appears an accepting state without outgoing transitions, the λ -transitions to that state can be omitted, since all states are accepting ones. The accepted language is not changed by the optimizations, but regardless of whether there appears a cycle or a state without outgoing transitions, at least one transition is omitted.

The next lemma gives an upper bound due to the above construction and its optimization.

Lemma 3.2. For any NFA M with $n \geq 1$ transitions, one can effectively construct an NFA accepting $\text{DOWN}(L(M))$ with at most $2n - 1$ transitions.

Proof:

We consider the above construction of an NFA accepting the language $\text{DOWN}(L(M))$. The first construction step inserts a λ -transition for each existing transition. So, the number of transitions is increased from n to $2n$. The optimizations reduce this number at least by one. \square

For the special case of finite unary languages with endmarker, that is, languages of the form $L\{b\}$, where $L \subseteq \{a\}^*$ is finite, the upper bound is much better and, in particular, is shown to be tight.

Theorem 3.1. Let M be an NFA with $n \geq 1$ transitions accepting a finite unary language with end-marker. Then $n + \lceil \log(n) \rceil$ transitions are sufficient and necessary in the worst case for an NFA M' to accept $\text{DOWN}(L(M))$. The NFA M' can effectively be constructed.

Proof:

In order to prove the lower bound we use the finite languages $L_n = \{a^{n-1}b\}$ with $n \geq 1$ as witnesses. Clearly, L_n is accepted by some $(n + 1)$ -state NFA M whose transition function defines n transitions. It remains to be shown that any NFA M' which accepts $\text{DOWN}(L_n)$ needs at least $n + \lceil \log(n) \rceil$ transitions. Since $\text{DOWN}(L_n)$ is finite, automaton M' has no cycles. We consider accepting computations for the scattered subwords $a^i b$, for $0 \leq i \leq n - 1$. For each of these subwords, automaton M performs at least $i + 1$ different transitions, say $t_1^i, t_2^i, \dots, t_{i+1}^i$. Let s_i be the state in which M' reads the last letter b from its input.

Now we will show that for each two scattered subwords $a^i b$ and $a^j b$ with $i < j$ automaton M' has to use at least one transition which is not in $t_1^j, t_2^j, \dots, t_{j+1}^j$ in order to accept $a^i b$. We consider two cases: (1) If the states s_i and s_j are equal, then on input prefix a^i automaton M' must have used at least one transition, say t' , not appearing in $t_1^j, t_2^j, \dots, t_j^j$. Otherwise, there exists a cycle in $t_1^j, t_2^j, \dots, t_j^j$. Moreover, transition t' is not equal to t_{j+1}^j , since the latter reads an input symbol b . (2) If the states s_i and s_j are different, we denote the transition from s_i which reads the last input letter b by t' and observe that t' is different from t_{j+1}^j since the latter is defined for state $s_i \neq s_j$. Furthermore, it is different from $t_1^j, t_2^j, \dots, t_j^j$ since the latter do not read the input symbol b . In both cases, t' is a new transition.

Altogether, M' performs at least n different transitions to accept $a^{n-1}b$. It is easy to see that, in addition, there are at least $\lceil \log(n) \rceil$ more transitions in order to meet the shown condition, i.e., to distinguish between the n scattered subwords $a^i b$, for $0 \leq i \leq n - 1$.

In order to prove the upper bound, we turn to the construction of M' with $n + \lceil \log(n) \rceil$ transitions. Let $s_0, s_1, \dots, s_{n-1}, s_n$ be the sequence of states passed through during an accepting computation on input $a^{n-1}b$. This gives $n - 1$ a -transitions and one b -transition. Now we add λ -transitions from s_{2^i-1} to $s_{2^{i+1}-1}$, for all $0 \leq i \leq \lceil \log(n) \rceil - 1$. That is, from s_0 to s_1 , from s_1 to s_3 , from s_3 to s_7 , and so on. If $n - 1$ is not of the form $2^k - 1$, we continue the path by adding another λ -transition to s_{n-1} . Altogether, this gives $\lceil \log(n) \rceil$ transitions in addition. Finally, we distinguish s_0, s_{n-1} , and s_n to be accepting states. It is easy to see that M' accepts $\text{DOWN}(L_n)$. \square

In the remainder of this subsection we consider DFAs. First observe that the constructions presented so far heavily rely on nondeterminism. Even when starting with a DFA, the resulting automata accepting $\text{DOWN}(L(M))$ or $\text{UP}(L(M))$ are nondeterministic in general. So, applying Lemma 3.1 and the well-known powerset construction gives an upper bound on the size of an equivalent DFA.

Corollary 3.1. For any DFA M of size $n \geq 1$, one can effectively construct a DFA accepting $\text{UP}(L(M))$ or $\text{DOWN}(L(M))$ whose size is at most 2^n .

For the next two theorems we need some more notations. Let $L \subseteq A^*$ be an arbitrary language. Then the *Myhill-Nerode* equivalence relation \equiv_L is defined as follows: For $u, v \in A^*$ let $u \equiv_L v$ if and only if $uw \in L \iff vw \in L$, for all $w \in A^*$. It is well known that the number of states of the minimal deterministic finite automaton accepting the language $L \subseteq A^*$ equals the cardinality of the set of equivalence classes induced by the Myhill-Nerode relation.

We continue our investigations by proving a non-trivial lower bound for DFAs accepting the language $\text{DOWN}(L(M))$, for some given DFA M . The lower bound is quite close to the upper bound of the previous corollary.

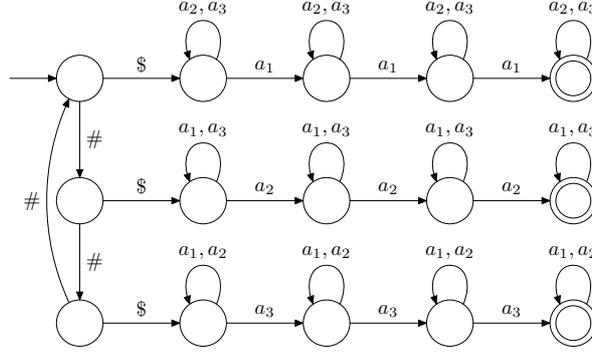


Figure 1. A DFA of size $5 \cdot 16$ accepting $\text{DOWN}(L_3)$ —the non-accepting sink state is not depicted.

Theorem 3.2. For every $n \geq 1$, there exists a language L_n over an $(n + 2)$ -letter alphabet accepted by a DFA of size $(n + 2)(n + 1)^2$, such that any DFA accepting $\text{DOWN}(L_n)$ is at least of size $2^{\Omega(n \log n)}$.

Proof:

Let $A = \{a_1, a_2, \dots, a_n\}$ and $\#, \$ \notin A$. Consider the language $L_n \subseteq (A \cup \{\#, \$\})^*$ defined as

$$L_n = \{ \#^j \$ w \mid w \in A^*, j \geq 0, i = j \bmod n, |w|_{a_{i+1}} = n \}.$$

A DFA accepting language L_3 is depicted in Figure 1. It is easy to see that any DFA accepting L_n needs $n + 1$ states for each letter a_i to count up to n . Moreover, for the $\#$ -prefix n states are used, and finally one non-accepting sink state is needed. This results in $n(n + 1) + n + 1$ states, which gives size $(n + 2)(n^2 + 2n + 1) = (n + 2)(n + 1)^2$. It is not hard to verify that the DFA is minimal.

It remains to be shown that the minimal DFA accepting $\text{DOWN}(L_n)$ has at least $(n + 2)^n + 1$ states, where

$$\text{DOWN}(L_n) = \{ \#^j a w \mid w \in A^*, j \geq 0, a \in \{ \$, \lambda \} \text{ and } \bigvee_{i=1}^n |w|_{a_i} \leq n \}.$$

First we consider any two different words of the form $w_{i_1, i_2, \dots, i_n} = \$ a_1^{i_1} a_2^{i_2} \dots a_n^{i_n}$ with $0 \leq i_j \leq n + 1$ and $1 \leq j \leq n$, and show that they are non-equivalent with respect to the Myhill-Nerode relation $\equiv_{\text{DOWN}(L_n)}$. Let w_{i_1, i_2, \dots, i_n} and $w_{i'_1, i'_2, \dots, i'_n}$ be two different words. Then $i_k \neq i'_k$, for some $1 \leq k \leq n$. Without loss of generality we assume that $i_k < i'_k$. Then the word

$$w_{i_1, i_2, \dots, i_n} \cdot a_1^{n+1} a_2^{n+1} \dots a_{k-1}^{n+1} a_k^{(n+1)-i'_k} a_{k+1}^{n+1} \dots a_n^{n+1}$$

belongs to $\text{DOWN}(L_n)$, because letter a_k appears at most n times. On the other hand,

$$w_{i'_1, i'_2, \dots, i'_n} \cdot a_1^{n+1} a_2^{n+1} \dots a_{k-1}^{n+1} a_k^{(n+1)-i'_k} a_{k+1}^{n+1} \dots a_n^{n+1}$$

does not belong to $\text{DOWN}(L_n)$ since all letters a_j , for $1 \leq j \leq n$, appear at least $n+1$ times. Hence, there are $(n+2)^n$ different equivalence classes induced by the words w_{i_1, i_2, \dots, i_n} . Moreover, the empty word λ is non-equivalent to all the other words. This is seen by concatenating the words with $\$$. Therefore, we have obtained at least $(n+2)^n + 1$ equivalence classes. In fact, one can construct a DFA with this number of states accepting $\text{DOWN}(L_n)$. Therefore, $2^{\Omega(n \log n)}$ is a lower bound on the size of any DFA accepting $\text{DOWN}(L_n)$. \square

Now we turn to deduce a lower bound on the size of any DFA accepting $\text{UP}(L(M))$, for a given DFA M .

Theorem 3.3. For every $n \geq 1$, there exists a language L_n over an $(n+2)$ -letter alphabet accepted by a DFA of size $(n+2)(n+1)^2$, such that any DFA accepting $\text{UP}(L_n)$ is at least of size $2^{\Omega(n \log n)}$.

Proof:

Again, we use the language L_n from the proof of Theorem 3.2. So, we already know that there is a DFA of size $(n+2)(n+1)^2$ accepting L_n . The description of the language $\text{UP}(L_n)$ is more involved compared with $\text{DOWN}(L_n)$, since $\text{UP}(L_n)$ has the following representation as a finite union of languages:

$$\text{UP}(L_n) = \bigcup_{j=1}^n P_j \$ S_j,$$

where for $1 \leq j < n$,

$$P_j = \{ w \in (A \cup \{\$\})^* \mid |w|_{\#} = j - 1 \} \quad \text{and} \quad P_n = \{ w \in (A \cup \{\$\})^* \mid |w|_{\#} \geq n - 1 \},$$

and for $1 \leq j \leq n$,

$$S_j = \{ w \in (A \cup \{\#, \$\})^* \mid \bigvee_{i=1}^j |w|_{a_i} \geq n \}.$$

In order to obtain the $2^{\Omega(n \log n)}$ lower bound on the size of any DFA accepting $\text{UP}(L_n)$, it suffices to show that $\text{UP}(L_n)$ induces at least n^n equivalence classes with respect to the Myhill-Nerode relation $\equiv_{\text{UP}(L_n)}$. Let

$$w_{i_1, i_2, \dots, i_n} = \#^n \$ a_1^{i_1} a_2^{i_2} \dots a_n^{i_n}$$

with $0 \leq i_j < n$ and $1 \leq j \leq n$. Note that $w_{i_1, i_2, \dots, i_n} \cdot a_k^{n-i_k}$ belongs to the language $P_n \$ S_n$. Any two different words w_{i_1, i_2, \dots, i_n} and $w_{i'_1, i'_2, \dots, i'_n}$ do not belong to the same equivalence class. Since both words are different, we have $i_k \neq i'_k$, for some $1 \leq k \leq n$. Assume without loss of generality that $i_k < i'_k$. Then it is easy to see that

$$w_{i_1, i_2, \dots, i_n} \cdot a_k^{n-i'_k} \notin \text{UP}(L_n) \quad \text{but} \quad w_{i'_1, i'_2, \dots, i'_n} \cdot a_k^{n-i'_k} \in \text{UP}(L_n).$$

In the former case $\bigvee_{i=1}^n |w_{i_1, i_2, \dots, i_n} \cdot a_k^{n-i'_k}|_{a_i} \geq n$ is false, whereas $\bigvee_{i=1}^n |w_{i'_1, i'_2, \dots, i'_n} \cdot a_k^{n-i'_k}|_{a_i} \geq n$ becomes true in the latter case, since the word under consideration contains exactly n symbols a_k . Therefore, any DFA accepting the language $\text{UP}(L_n)$ must have at least n^n states. \square

In fact, with a more careful analysis one obtains that any DFA accepting $\text{UP}(L_n)$ must have at least $n^n + n^{n-1} + \dots + n + 1$ states. To this end, one shows that the languages $P_i \$ S_i$, for $1 \leq i \leq n$, induce n^i pairwise different equivalence classes.

Finally, it is worth to mention that the lower bounds of the previous two theorems slightly improve when the number of states is used to measure the size of DFAs. The next theorem summarizes the lower bounds.

Theorem 3.4. For every $n \geq 1$, there exists a language L_n over an $(n + 2)$ -letter alphabet accepted by a DFA with $(n + 1)^2$ states, such that $2^{\Omega(n \log n)}$ states are necessary for any DFA accepting $\text{DOWN}(L_n)$. A similar statement is valid for $\text{UP}(L_n)$.

After the conference version of this paper appeared, the precise number of required states for both $\text{DOWN}(L)$ and $\text{UP}(L)$ was determined in [15]. The exact bounds are given as functions in the number of states, irrespective of alphabet size. The witness languages use larger alphabets than the languages used in this paper; therefore the mentioned work also discusses the effect of alphabet size on the required number of states and establishes roughly exponential lower bounds also for alphabets of constant size. Part of these results improve the lower bounds obtained here, see [15] for details.

3.2. Context-Free and Linear Context-Free Languages

In this subsection we are interested in the size of NFAs accepting the Higman-Haines sets of languages generated by context-free and linear context-free grammars. Recall that we use the total number of occurrences of terminal and nonterminal symbols in the productions as size measure for grammars. Let $G = (N, T, P, S)$ be a context-free grammar, where N is the finite set of *nonterminals*, T is the finite set of *terminals*, $P \subseteq N \times (N \cup T)^*$ is the finite set of productions, and $S \in N$ is the *axiom*. A context-free grammar $G = (N, T, P, S)$ is *linear context free* if $P \subseteq N \times T^*(N \cup \{\lambda\})T^*$. Without loss of generality, we assume that the context-free grammars are always *reduced*, which means that there are no unreachable or unproductive nonterminals.

As in the previous subsection we first show how to construct an NFA for $\text{DOWN}(L(G))$. In order to simplify the analysis we assume that the right hand-sides of the productions are described by NFAs with input alphabet $N \cup T$. We refer to such a grammar as an *extended* (linear) context-free grammar. Note, that one can assume that for each extended context-free grammar there is exactly one NFA for each nonterminal appearing at a right-hand side. The following theorem is a detailed analysis of the inductive construction presented in [17].

Theorem 3.5. Let G be a context-free grammar of size $n \geq 1$. Then size $O(n \cdot 2^{\sqrt{2^n} \log n})$ is sufficient for an NFA M' to accept $\text{DOWN}(L(G))$. The NFA M' can effectively be constructed.

Proof:

First, the context-free grammar $G = (N, T, P, S)$ is transformed into an extended context-free grammar G' —the details are left to the reader. Second, we observe that each nonterminal appears at the left-hand-side of at least one production, and at least one nonterminal is rewritten by some terminal symbol. Therefore, the number of nonterminals is at most $\lfloor \frac{n}{2} \rfloor$. Next, we inductively proceed as in [17]. For a nonterminal $A \in N$ we set the alphabet $T_A = (N \setminus \{A\}) \cup T$, and define the extended context-free grammar $G_A = (\{A\}, T_A, P_A, A)$ with $P_A = \{A \rightarrow M \mid (A \rightarrow M) \in P\}$, where M in $(A \rightarrow M) \in P$

refers to the NFA of the right hand-side of the production. Further set $L_A = L(G_A)$. Observe that G_A is an extended context-free grammar with only *one* nonterminal, and thus one can obtain an NFA M_A describing $\text{DOWN}(L(G_A))$ over the alphabet T_A by a subroutine to be detailed below. Then the induction is as follows: Let $G_0 = G'$. If A is not the axiom S of G_0 , we can replace each A -transition occurring in the right-hand-side automata of non- A -productions of G_0 with a copy of M_A to obtain an extended grammar G_1 having one nonterminal less than G_0 , and $\text{DOWN}(L(G_1)) = \text{DOWN}(L(G_0))$. This construction step can be iterated for at most $\lfloor \frac{n}{2} \rfloor - 1$ times, yielding extended context-free grammars $G_2, G_3, \dots, G_{\lfloor \frac{n}{2} \rfloor - 1}$, satisfying $\text{DOWN}(L(G_i)) = \text{DOWN}(L(G_{i+1}))$, for $0 \leq i < \lfloor \frac{n}{2} \rfloor$, where in the latter grammar $G_{\lfloor \frac{n}{2} \rfloor - 1}$ the only remaining nonterminal is the axiom S of G . Finally, we apply the mentioned subroutine to construct the NFA M' which results in the finite automaton accepting the language $\text{DOWN}(L(G))$.

It remains to describe the above mentioned subroutine and deduce an upper bound on the size of the automaton M' . The subroutine works for an extended grammar $G_A = (\{A\}, T_A, \{A \rightarrow M\}, A)$ with only *one* nonterminal. We distinguish two cases:

1. The production set given by $L(M)$ is linear, i.e., $L(M) \subseteq T_A^* \{A, \lambda\} T_A^*$, or
2. the production set given by $L(M)$ is nonlinear.

In the first case, we construct an NFA M_T with $L(M_T) = L(M) \cap T_A^*$, which is obtained by removing all A -transitions from M . Similarly, we build NFAs M_P and M_S for the quotients

$$\begin{aligned} L(M_P) &= \{x \in T_A^* \mid xAz \in L(M) \text{ for some } z \in (T_A \cup \{A\})^*\} \text{ and} \\ L(M_S) &= \{z \in T_A^* \mid xAz \in L(M) \text{ for some } x \in (T_A \cup \{A\})^*\}. \end{aligned}$$

Then it is straightforward to construct an NFA M_A having a single initial state and a single accepting state with $L(M_A) = \text{DOWN}(L(M_P)^* \cdot L(M_T) \cdot L(M_S)^*) = \text{DOWN}(L(G_A))$. The number of non- λ -transitions, in M_A is at most three times that of M .

In the second case, i.e., $L(M)$ is nonlinear, we construct automata M_P, M_T, M_S , and M_I , where the former three NFAs are as in the previous case, and M_I accepts the quotient

$$L(M_I) = \{y \in T_A^* \mid xAyAz \in L(M) \text{ for some } x, z \in (T_A \cup \{A\})^*\}.$$

Again, it is not hard to construct an NFA M_A with a single initial and a single accepting state accepting $L(M_A) = \text{DOWN}((L(M_T) \cup L(M_P) \cup L(M_I) \cup L(M_S))^*) = \text{DOWN}(L(G_A))$ with no more than four times as many non- λ -transitions as M .

The upper bound on the size of an NFA accepting $\text{DOWN}(L(G))$ is deduced as follows: For an extended context-free grammar G , let $|G|_t$ denote the sum of the number of non- λ -transitions in the right-hand-side automata in the productions of G . With this notation we obtain the recurrence

$$|G_k|_t \leq 4 \cdot (|G_{k-1}|_t)^2, \quad \text{for } 1 \leq k < \lfloor \frac{n}{2} \rfloor,$$

describing the substitution step in the k th iteration of the construction of G_k from G_{k-1} . Taking logarithms and setting $H_k = \log |G_k|_t$, we obtain a linear recurrence $H_k \leq 2 \cdot H_{k-1} + 2$. Solving the linear recurrence, we obtain the inequality $H_k \leq 2^k \cdot H_0 + 2^{k+1} - 2$. Since $|G_0|_t \leq n$, we have

$$H_{\lfloor \frac{n}{2} \rfloor - 1} \leq 2^{\lfloor \frac{n}{2} \rfloor - 1} \cdot H_0 + 2^{\lfloor \frac{n}{2} \rfloor} - 2 \leq 2^{\lfloor \frac{n}{2} \rfloor - 1} \cdot \log n + 2^{\lfloor \frac{n}{2} \rfloor} - 2.$$

When replacing the axiom in $G_{\lfloor \frac{n}{2} \rfloor - 1}$ in the final step, the number of non- λ -transitions is increased at most by a factor of four, which results in

$$|G_{\lfloor \frac{n}{2} \rfloor}|_t \leq 2^{2^{\lfloor \frac{n}{2} \rfloor - 1} \cdot \log n + 2^{\lfloor \frac{n}{2} \rfloor}} \leq 2^{2^{\lfloor \frac{n}{2} \rfloor - 1} \cdot \log n + 2^{\lfloor \frac{n}{2} \rfloor - 1} \cdot \log n} \leq 2^{2^{\lfloor \frac{n}{2} \rfloor} \log n} \leq 2^{\sqrt{2^n} \log n},$$

for all $n \geq 4$. It remains to be shown that for every NFA with n non- λ -transitions, there is an equivalent NFA with at most $O(n)$ states. An easy construction can be used to remove all non-initial states having neither ingoing nor outgoing alphabetical transitions after adding some extra λ -transitions where necessary. By a simple counting argument, we find that the latter automaton has at most $2n + 1$ states. Hence, this shows that the NFA M' accepting $\text{DOWN}(L(G))$ has a size of at most $O(n \cdot 2^{\sqrt{2^n} \log n})$. \square

For the lower bounds we obtain:

Theorem 3.6. For every $n \geq 1$, there is a language L_n over a unary alphabet generated by a context-free grammar of size $3n + 2$, such that size $2^{\Omega(n)}$ is necessary for any NFA accepting $\text{DOWN}(L(G))$ or $\text{UP}(L(G))$.

Proof:

For every $n \geq 1$, consider the finite unary languages $L_n = \{a^{2^n}\}$ generated by the context-free grammar $G = (\{A_1, A_2, \dots, A_{n+1}\}, \{a\}, P, A_1)$ with the productions $A_i \rightarrow A_{i+1}A_{i+1}$, for $1 \leq i \leq n$, and $A_{n+1} \rightarrow a$. Obviously, grammar G has size $3n + 2$. The word a^{2^n} is the longest word in $\text{DOWN}(L(G))$ and the shortest word in $\text{UP}(L(G))$. In both cases, any finite automaton accepting the language takes at least as many states as the length of the word. So, it takes at least 2^n states and, thus, has at least size 2^n . \square

Now we turn our attention to the construction of an NFA accepting $\text{UP}(L(G))$, for a context-free grammar G . To this end, we call a word $w \in L$ *minimal* in L if and only if there is no different $v \in L$ with $v \leq w$. The set of minimal elements in L is called a *basis of the language* $\text{UP}(L)$. Observe that any shortest word in L is minimal in L , and any such word must therefore be part of the basis. In fact, Higman's Lemma [11] says that for any arbitrary language L there exists a natural number n , which depends only on L , such that $\text{UP}(L) = \bigcup_{1 \leq i \leq n} \text{UP}(\{w_i\})$, for some words $w_i \in L$ with $1 \leq i \leq n$. Sometimes the result is called the *finite basis property*. For the construction of an NFA accepting $\text{UP}(L(G))$, where G is a context-free grammar with terminal alphabet A , we proceed as follows:

1. Determine the basis $B \subseteq A^*$ of the language $\text{UP}(L(G))$ with the algorithm presented in [17].
2. Construct an NFA M accepting language B , and apply the construction given in the previous subsection to obtain an NFA M' accepting $\text{UP}(B)$, which equals the language $\text{UP}(L(G))$ by the finite basis property.

The first step basically consists of inductively computing B starting from $B_0 = \emptyset$. Language B_{i+1} is obtained by extending B_i by a shortest word w in $L(G) \setminus \text{UP}(B_i)$, i.e., setting $B_{i+1} = B_i \cup \{w\}$. This process is repeated as long as $(L(G) \setminus \text{UP}(B_i)) \neq \emptyset$. If this condition is met, the set B equals the last extended B_i . Since context-free languages are closed under set difference with regular sets, the set B can effectively be constructed. Taking this approach we would end up with a double exponential upper

bound for the NFA accepting the up-set of a context-free language. Nevertheless, in the next theorem we show that we can do much better.

In the proof to come we require that the context-free grammar is in 2-normalform, i.e., the productions are of the form $P \subseteq N \times (N^2 \cup T^2 \cup NT \cup TN \cup N \cup T \cup \{\lambda\})$. This is no restriction, since in [10] it was shown that for a given context-free grammar of size n one can effectively construct an equivalent context-free grammar in 2-normalform of size at most $3 \cdot n$.

Theorem 3.7. Let G be a context-free grammar of size $n \geq 1$. Then size $2^{O(n)}$ is sufficient for an NFA M to accept $\text{UP}(L(G))$. The NFA M can effectively be constructed.

Proof:

Let $G = (N, T, P, S)$ be the context-free grammar. Without loss of generality, we may assume that G is in 2-normalform. If $L(G)$ is empty, the statement is obviously true. Thus, assume $L(G) \neq \emptyset$ for the rest of the proof. Let B denote the basis of $L = L(G)$. Then we argue as follows: Essentially the same argument as the one used in the pumping lemma shows that each word z in B admits a derivation tree that is *acyclic*, in the sense that on each path from the root, each nonterminal can occur at most once—in fact, this might not be true for all such trees, since the 2-normalform of G allows cyclic chains of unit productions. Otherwise we could decompose $z = uvwxy$ such that its proper subword uwv would also be in L , contradicting $z \in B$. In particular, word z has a derivation tree where S occurs only at the root. Now let G' be the context-free grammar obtained from G by removing all rules in

$$P_{rhs,S} = \{ A \rightarrow \alpha \mid A \rightarrow \alpha \in P \text{ and } S \text{ appears in } \alpha \},$$

that is, $G' = (N, T, P \setminus P_{rhs,S}, S)$. By the above given argument, this does not change the basis of the generated language, i.e., both languages $L(G)$ and $L(G')$ have the same basis B .

Now define

$$P_{lhs,S} = \{ S \rightarrow \alpha \mid (S \rightarrow \alpha) \in P \}$$

and for each symbol $A \in N \cup T$ that appears at the right-hand-side of a rule in $P_{lhs,S}$ define the context-free language L_A to be generated by the context-free grammar $G_A = (N, T, P_A, A)$, where $P_A = P \setminus (P_{lhs,S} \cup P_{rhs,S})$, if $A \in N$, and $L_A = \{A\}$, if $A \in T$. It should be clear that the basis of $L(G')$, or equivalently of L , is obtained by computing the basis of

$$\bigcup_{\substack{(S \rightarrow XY) \in P_{lhs,S} \\ X, Y \in N \cup T \cup \{\lambda\}}} L_X \cdot L_Y, \quad (1)$$

where $L_X = \{\lambda\}$, if $X = \lambda$. Clearly, the basis of $L_X \cdot L_Y$ is a subset of the basis of L_X concatenated with the basis of L_Y . Since the number of productions used to generate L_X or L_Y by a context-free grammar has decreased by at least $|P_{lhs,S}|$, this gives a terminating recursive algorithm for computing the basis B of L from the context-free grammar G .

Next we show how to construct an NFA M having at most $2^{|P|} + 1$ states accepting the basis B of L , using the above described recursive algorithm. The prove is done by induction on the number of productions of G . Let $n = |P|$. In the base case $n = 1$, the language generated by G contains at most one word of length at most 2, and thus, the statement clearly holds in this case. To do the induction step, we use the fact that the basis computation of L based on Equation (1) and the comment given afterwards can

be implemented using standard NFA constructions for concatenation and union of finite languages [12], giving

$$\text{nsc}(L) \leq \sum_{\substack{(S \rightarrow XY) \in P_{l_{hs}, S} \\ X, Y \in N \cup T \cup \{\lambda\}}} (\text{nsc}(L_X) + \text{nsc}(L_Y) - 1) - 2(|P_{l_{hs}, S}| - 1)$$

as an upper bound on the number of states of M . Here $\text{nsc}(L)$ denotes the number of states of an NFA accepting the basis of L . Since each language L_X and L_Y can be generated by a context-free grammar with at most $\max\{1, n-r\}$ productions, where $r = |P_{l_{hs}, S}|$, for $r < n$ we obtain by induction hypothesis

$$\text{nsc}(L) \leq r \cdot (2 \cdot (2^{n-r} + 1) - 1) - 2(r - 1) = r \cdot 2^{1-r} \cdot 2^n - r + 2.$$

For $r \geq 1$, the right-hand-side is at most $2^n + 1$. It remains to prove the statement for $r = n$. In this case, the only productions in G' are that of $P_{l_{hs}, S}$. But then it is easy to construct an NFA with $n + 2 \leq 2^n + 1$ states accepting the basis B of L . This completes the proof and shows that the number of states of M is at most $2^{|P|} + 1$. Thus, the *size* of M in terms of the size n of G is bounded above by $n \cdot (2^n + 1)$, which is of order $2^{O(n)}$ as stated. \square

In the remainder of this section we concentrate on linear context-free languages. As in the previous proof, we make use of the notion of acyclic derivations, that is, on the path from the root to the leaves, each nonterminal can occur at most once. For derivations induced by linear context-free grammars this implies that no nonterminal occurs more than once in the derivation. Note that even if the sequences of nonterminals derived in acyclic derivations are equal, the derivation may be different. The following lemma gives an upper bound on the cardinality of acyclic derivations of linear context-free grammars.

Lemma 3.3. Let $G = (N, T, P, S)$ be a linear context-free grammar with m productions. Then

$$|\mathcal{A}(G)| \leq 2^{m-1},$$

where $\mathcal{A}(G)$ denotes the set of all acyclic derivations $S \Rightarrow^* w$ in G , for $w \in T^*$.

Proof:

Each production of a linear context-free grammar may appear at most once in any acyclic derivation. Moreover, the set of applied productions is naturally ordered by the nonterminals on the right-hand and left-hand-sides. So, each acyclic derivation corresponds to an (ordered) subset of P , that contains exactly one production with the axiom at its left-hand-side. In total we obtain at most 2^{m-1} different acyclic derivations. \square

Now we are prepared for our first result on linear context-free grammars.

Theorem 3.8. Let G be a linear context-free grammar of size $n \geq 1$. Then size $2^{O(n)}$ is sufficient for an NFA M to accept $\text{DOWN}(L(G))$. The NFA M can effectively be constructed.

Proof:

Let $G = (N, T, P, A_1)$ be a linear context-free grammar. The basic idea for the construction of M is to inspect the derivation trees of G and to modify the underlying grammar such that any self-embedding derivation of the form $A \Rightarrow^* xAz$, for some $A \in N$ and $x, z \in T^*$, is replaced by a derivation $A \Rightarrow^* xA$

and $A \Rightarrow^* Az$, while the respective generated languages have the same down-sets. In other words, the derivation that produces the “coupled” terminal words x and z is made “uncoupled” by a right-linear and a left-linear derivation. In order to make the construction work, one has to take care about these self-embedded derivation parts in an appropriate manner. For a formal treatment of the construction we need some notation.

Let $S = A_1 \Rightarrow^* w$ be a derivation of $w \in T^*$. Then the inner nodes of the derivation tree form a path $p = A_1 A_2 \dots A_k$, for some $k \geq 1$. We can group the inner nodes as follows: We call a subpath of p that represents a self-embedded derivation with nonterminal A , i.e., which begins and ends with the same nonterminal A , an A -block—trivial derivations of the form $A \Rightarrow^* A$ also count as blocks. Now we collapse blocks in order to obtain an acyclic derivation as follows. We consider the path p from left to right. If there is a nonterminal, say A_i , that appears at least twice, then the leftmost and rightmost occurrence defines a block that is collapsed. This means, the corresponding subpath from the first A_i to the last A_i is deleted. The initial subpath from the axiom to the first A_i is kept, and the new derivation continues as from the last A_i . This process is repeated until each nonterminal appears at most once (see Figure 2). In this way, each derivation corresponds to an acyclic one and *vice versa*.

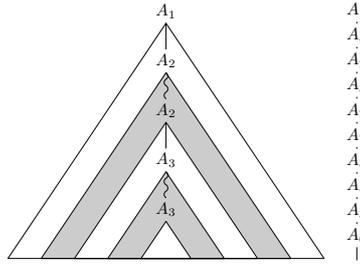


Figure 2. Collapsing the path $p = A_1 A_4 A_2 A_3 A_2 A_7 A_3 A_5 A_3 A_6$ yields the path $A_1 A_4 A_2 A_7 A_3 A_6$ that corresponds to an acyclic derivation. Blocks are gray shaded and their subderivations are drawn by a curled line. Single step subderivations are grouped together. They are depicted white and their subderivations are drawn by a solid line.

Now consider the *acyclic* derivation $D : S = A_1 \Rightarrow^* w$ of $w \in T^*$ induced by the linear context-free rules $A_i \rightarrow x_i A_{i+1} z_i$ with $x_i, z_i \in T^*$ and $A_i \in N$, for $1 \leq i < k$, and $A_k \rightarrow y$ with $y \in T^*$, for some $k \leq |N|$. For each nonterminal $A \in N$ we consider the quotients

$$\begin{aligned} L(M_{A,P}) &= \{x \in T^* \mid A \Rightarrow^* xAz \text{ for some } z \in T^*\} \text{ and} \\ L(M_{A,S}) &= \{z \in T^* \mid A \Rightarrow^* xAz \text{ for some } x \in T^*\}. \end{aligned}$$

By deleting z from any production $A \rightarrow xBz$, for $x, z \in T^*$ and $B \in N$, deleting Az from any production $A \rightarrow xAz$, for $x, z \in T^*$, and erasing each production $A \rightarrow y$, for $y \in T^*$ we obtain a rightlinear grammar for $L(M_{A,P})$. Similarly, we construct a leftlinear grammar for $L(M_{A,S})$. Then it is straightforward to construct NFAs $M_{A,P}$ and $M_{A,S}$ accepting $L(M_{A,P})$ and $L(M_{A,S})$ that have a single initial state and a single accepting state, respectively. Both NFAs can be constructed such that they have at most $|N| \leq n$ states.

For the acyclic derivation D from $\mathcal{A}(G)$ we obtain an NFA M_D accepting the language $L(D_1)$ which

is inductively defined as follows: For $1 \leq i \leq k - 1$ let

$$L(D_i) = L(M_{A_i, P})^* \cdot \{x_i\} \cdot L(D_{i+1}) \cdot \{z_i\} \cdot L(M_{A_i, S})^*$$

and

$$L(D_k) = L(M_{A_k, P})^* \cdot \{y\} \cdot L(M_{A_k, S})^*.$$

Then by our previous investigations one observes that

$$\text{DOWN}(L(G)) = \bigcup_{D \in \mathcal{A}(G)} \text{DOWN}(L(M_D)),$$

since the down-set of the set of all partial derivations corresponding to an A -block equals the language $\text{DOWN}(L(M_{A, P})^* \cdot A \cdot L(M_{A, S})^*)$.

By using standard NFA constructions for concatenation and Kleene star, we obtain for each acyclic derivation D an NFA M_D that has at most $4 \cdot n^2$ states, since all involved languages can be accepted by NFAs of at most n states, and each acyclic derivation has at most $|N| \leq n$ nonterminals. Thus, combining the upper bound on $\mathcal{A}(G)$ of Lemma 3.3 with the fact that the down-operator does not increase the nondeterministic state complexity results in an upper bound of $4n^2 \cdot 2^{n-1} = 2^{O(n)}$ states for an NFA accepting the set $\bigcup_{D \in \mathcal{A}(G)} \text{DOWN}(L(M_D))$ —here the standard NFA construction for union is used to obtain the result. This proves the stated claim. \square

In order to derive a lower bound we use the finite languages $L_n = \{ww^R \mid w \in \{a, b\}^n\}$, which can be generated by linear context-free grammars $G_n = (\{A_i \mid 1 \leq i \leq n\}, \{a, b\}, P, A_1)$ with the productions $A_i \rightarrow aA_{i+1}a$, $A_i \rightarrow bA_{i+1}b$, for $1 \leq i < n$, and $A_n \rightarrow aa$ and $A_n \rightarrow bb$. Since any NFA accepting L_n needs at least 2^n states [7], the next theorem follows. Note that the lower bound also holds for the up-set problem.

Theorem 3.9. For every $n \geq 1$, there is a linear context-free language L_n over a binary alphabet generated by a linear context-free grammar of size $8n - 2$, such that size $2^{\Omega(n)}$ is necessary for any NFA accepting $\text{DOWN}(L(G))$ or $\text{UP}(L(G))$. \square

Since linear context-free grammars are only a special case of context-free grammars, Theorem 3.7 already gives an upper bound for the size of the set $\text{UP}(L(G))$, for some linear context-free grammar G of size n . Therefore, in order of magnitude the derived lower bounds are the best possible.

Theorem 3.10. Let G be a linear context-free grammar of size $n \geq 1$. Then size $2^{O(n)}$ is sufficient for an NFA M to accept $\text{UP}(L(G))$. The NFA M can effectively be constructed. \square

We summarize our bounds on up- and down-sets for linear context-free languages in the following corollary.

Corollary 3.2. Let G be a linear context-free grammar of size n . Then size $2^{\Theta(n)}$ is sufficient and necessary in the worst case for an NFA to accept $\text{DOWN}(L(G))$ or $\text{UP}(L(G))$. The NFA can effectively be constructed. \square

3.3. Unary Regular, Linear Context-Free, and Context-Free Languages

Finally, we consider unary languages in more detail. We start with a complete structural characterization of the Higman-Haines sets of unary languages. We omit the straightforward proofs.

Theorem 3.11. Let L be an arbitrary nonempty unary language over the alphabet $\{a\}$. Then we have $\text{UP}(L) = \{a^n \mid n \geq \min\{|w| \mid w \in L\}\}$. If L is infinite, then $\text{DOWN}(L) = \{a\}^*$, and otherwise $\text{DOWN}(L) = \{a^n \mid n \leq \max\{|w| \mid w \in L\}\}$. \square

Thus, for NFAs accepting unary languages, one obtains trivial upper and matching lower bounds of size n (which, in fact, also holds for the number of states) for accepting the Higman-Haines sets. But what about the bounds for linear context-free or context-free grammars generating unary languages? Unary languages generated by (linear) context-free grammars are regular. For their context-free grammars we find the following situation:

Theorem 3.12. Let G be a context-free grammar of size $n \geq 1$ generating a unary language L . Then size $2^{O(n)}$ is sufficient for an NFA M to accept $\text{DOWN}(L(G))$ or $\text{UP}(L(G))$. The NFA M can effectively be constructed. Moreover, for every $n \geq 1$, there is a context-free language L_n over a unary alphabet generated by a context-free grammar of size $3n + 2$, such that size $2^{\Omega(n)}$ is necessary for any NFA accepting $\text{DOWN}(L_n)$ or $\text{UP}(L_n)$.

Proof:

For the upper bound we argue as follows. Without loss of generality we may assume that the context-free grammar G is in 2-normalform. If G has m nonterminals, the shortest word in $L(G)$ is at most of length 2^{m-1} , and if $L(G)$ is finite, then also the longest word in $L(G)$ is at most of length 2^{m-1} . Since the number of nonterminals is at most $\lceil \frac{n}{2} \rceil$, the upper bound $2^{O(n)}$ follows immediately by Theorem 3.11. Finally, the lower bound is literally that of Theorem 3.6. \square

If we consider the number of nonterminals of a context-free grammar in 2-normalform as a size measure, we obtain matching upper and lower bounds. The argumentation is similar to that of the previous proof. Moreover, we need the fact that any NFA accepting a down-set (up-set) of a unary language takes at least one more state than the length of the longest (shortest) word of that set.

Theorem 3.13. Let G be a context-free grammar in 2-normalform with $n \geq 1$ nonterminals generating a unary language L . Then $2^{n-1} + 1$ states are sufficient and necessary in the worst case for an NFA to accept $\text{DOWN}(L(G))$ or $\text{UP}(L(G))$.

Finally, for linear context-free languages we find the following situation:

Theorem 3.14. Let G be a linear context-free grammar of size $n \geq 1$ generating a unary language L . Then size n is sufficient for NFA M to accept $\text{DOWN}(L(G))$ or $\text{UP}(L(G))$. The NFA M can effectively be constructed. Moreover, for every $n \geq 2$ there is a linear context-free language L_n over a unary alphabet generated by a linear context-free grammar of size n , such that size n is necessary for any NFA accepting $\text{DOWN}(L_n)$ or $\text{UP}(L_n)$.

Language L specified as ...		DOWN(L)		UP(L)	
		bounds		bounds	
		lower	upper	lower	upper
general	NFA	n			
	linear CFG	$2^{\Theta(n)}$		$2^{\Theta(n)}$	
	CFG	$2^{\Omega(n)}$	$2^{2^{O(n)}}$		
unary	NFA	n			
	linear CFG	n			
	CFG	$2^{\Theta(n)}$			

Table 1. Summary of upper and lower bounds on NFA size for $\text{DOWN}(L)$ and $\text{UP}(L)$, when L is specified as a nondeterministic finite automaton (NFA), linear context-free grammar (linear CFG), or a context-free grammar (CFG).

Proof:

Excluding only trivial cases for which the statement is readily verified, we always may assume that $\text{DOWN}(L(G)) \neq \{a\}^*$ and $L(G) \neq \emptyset$. In all remaining cases $\text{DOWN}(L(G)) = \text{DOWN}(\{w\})$ and $\text{UP}(L(G)) = \text{UP}(\{w\})$, for some word $w \in \{a\}^*$ that has an acyclic derivation in G . By the characterization given in Theorem 3.11 it can be determined whether a trivial case holds. Otherwise, the word w can be derived. Each linear grammar admitting an acyclic derivation generating w has size at least $|G| \geq |w| + 1$, since the axiom needs to be productive, and the right-hand sides of the rules need to have at least $|w|$ occurrences of terminal symbols. On the other hand, size $|w| + 1 \leq |G|$ is sufficient for a unary NFA to accept $\text{DOWN}(\{w\})$ or $\text{UP}(\{w\})$.

For each $n \geq 2$, the linear context-free grammar $G_n = (\{S\}, \{a\}, \{S \rightarrow a^{n-1}\}, S)$ is a witness for the fact that the bound is tight. Any NFA needs size n in order to accept either $\text{UP}(L(G_n))$ or $\text{DOWN}(L(G_n))$. \square

4. Conclusions

We have studied the size of Higman-Haines sets, which are the sets of all scattered subwords of a given language and the sets of all words that contain some word of a given language as a scattered subword. In particular, we considered the Higman-Haines sets induced by context-free, linear context-free and regular languages. For these language families we showed lower and upper bounds on the size of finite automata accepting the Higman-Haines set. After discussing bounds for different size measures for finite automata, we concentrated on the size of nondeterministic finite automata. For this measure, the results are summarized in the Table 1. Nevertheless, several questions about the size of Higman-Haines sets remain unanswered. In addition to the challenges posed in [15], we suggest to investigate the following:

1. Can one obtain better matching bounds for the down-set of context-free languages? What will Table 1 look like if we measure the size of *deterministic* finite automata accepting the down-set or up-set?

2. There are some other interesting and important subfamilies of the context-free languages, e.g., bounded, deterministic or turn-bounded context-free languages. The sizes of the corresponding Higman-Haines sets are worth studying.
3. Our investigations are based on the special case of the scattered subword relation. Since the result of Higman and Haines only needs a well-partial-order one may ask similar questions for other well-partial-orders as, e.g., for the Parikh subword quasi-order or for monotone well-quasi-orders—see [3, 13] for further results about these well-quasi-orders.

References

- [1] Buntrock, G., Otto, F.: Growing Context-Sensitive Languages and Church-Rosser Languages, *Inform. Comput.*, **141**, 1998, 1–36.
- [2] Dassow, J., Păun, G.: *Regulated Rewriting in Formal Language Theory*, vol. 18 of *EATCS Monographs in Theoretical Computer Science*, Springer, 1989.
- [3] Ehrenfeucht, A., Haussler, D., Rozenberg, G.: On regularity of context-free languages, *Theoret. Comput. Sci.*, **27**, 1983, 311–332.
- [4] Fenner, S., Gasarch, W., Postow, B.: The complexity of finding $\text{SUBSEQ}(A)$, *Theory Comput. Sys.*, to appear.
- [5] Fernau, H., Stephan, F.: Characterizations of Recursively Enumerable Sets by Programmed Grammars With Unconditional Transfer, *J. Autom., Lang. Comb.*, **4**, 1999, 117–152.
- [6] Gilman, R. H.: A shrinking lemma for indexed languages, *Theoret. Comput. Sci.*, **163**, 1996, 277–281.
- [7] Glaister, I., Shallit, J.: A lower bound technique for the size of nondeterministic finite automata, *Inform. Process. Lett.*, **59**, 1996, 75–77.
- [8] Gruber, H., Holzer, M., Kutrib, M.: The Size of Higman-Haines Sets, *Theoret. Comput. Sci.*, **387**, 2007, 167–176.
- [9] Haines, L. H.: On Free Monoids Partially Ordered by Embedding, *J. Combinatorial Theory*, **6**, 1969, 94–98.
- [10] Harrison, M. A., Yehudai, A.: Eliminating Null Rules in Linear Time, *Comput. J.*, **24**, 1981, 156–161.
- [11] Higman, G.: Ordering by Divisibility in Abstract Algebras, *Proc. London Math. Soc.*, **2**, 1952, 326–336.
- [12] Holzer, M., Kutrib, M.: Nondeterministic Descriptive Complexity of Regular Languages, *Int. J. Found. Comput. Sci.*, **14**, 2003, 1087–1102.
- [13] Ilie, L.: *Decision Problems on Orders of Words*, Ph.D. thesis, Department of Mathematics, University of Turku, Finland, 1998.
- [14] Kruskal, J. B.: The Theory of Well-Quasi-Ordering: A Frequently Discovered Concept, *J. Combinatorial Theory*, **13**, 1972, 297–305.
- [15] Okhotin, A.: On the state complexity of scattered substrings and superstrings, *Turku Centre for Computer Science (TUCS) Technical Report No. 849*, Turku, Finland, October 2007.
- [16] van Leeuwen, J.: A Regularity Condition for Parallel Rewriting Systems, *SIAC News*, **8**, 1976, 24–27.
- [17] van Leeuwen, J.: Effective Constructions in Well-Partially-Ordered Free Monoids, *Discrete Mathematics*, **21**, 1978, 237–252.