On the state complexity of closures and interiors of regular languages with subwords

P. Karandikar[1,2]* and Ph. Schnoebelen[2]**

[1] Chennai Mathematical Institute
[2] LSV, ENS Cachan, CNRS

Abstract. We study the state complexity of the set of subwords and superwords of regular languages, and provide new lower bounds in the case of languages over a two-letter alphabet. We also consider the dual interior sets, for which the nondeterministic state complexity has a doubly-exponential upper bound. We prove a matching doubly-exponential lower bound for downward interiors in the case of an unbounded alphabet.

1 Introduction

Quoting from [1], "*State complexity problems are a fundamental part of automata theory that has a long history. [. . .] However, many very basic questions, which perhaps should have been solved in the sixties and seventies, have not been considered or solved.*"

In this paper, we are concerned with *(scattered) subwords* and the associated operations on regular languages: computing closures and interiors (see definitions in Section 2). Our motivations come from automatic verification of channel systems, see, e.g., [2, 3]. Other applications exist in data processing or bioinformatics [4]. Closures and interiors wrt subwords and superwords are very basic operations, and the above quote certainly applies to them.

It has been known since [5] that $\downarrow L$ and $\uparrow L$, the downward closure and, respectively, the upward closure, of a language $L \subseteq \Sigma^*$, are regular for any L.

In [6], Gruber et al. explicitely raised the issue of the state complexity of downward and upward closures of regular languages (less explicit precursors exist, e.g. [7]). Given a n-state automaton A, constructing an automaton A' for $\downarrow L(A)$ or for $\uparrow L(A)$ can be done by simply adding extra transitions to A. However, when A is a DFA, the resulting A' is in general not deterministic, and determinization of A' may entail an exponential blowup in general. Gruber et al. proved a $2^{\Omega(\sqrt{n}\log n)}$ lower bound on the number of states of any DFA for $\downarrow L(A)$ or $\uparrow L(A)$, to be compared with the 2^n upper bound that comes from the simple closure+determinization algorithm.

Okhotin improved on these results by showing a $2^{\frac{n}{2}-2}$ and a $2^{n-2}+1$ lower bound for $\downarrow L(A)$ and, respectively, $\uparrow L(A)$ (again for an unbounded alphabet).

* Partially funded by Tata Consultancy Services.
** Supported by Grant ANR-11-BS02-001.

The second bound is known to be tight [8, 9]. However, all these lower bounds assume an unbounded alphabet.

Okhotin also considered the case of languages over a *fixed alphabet* with $|\Sigma| = 3$ letters, in which case he demonstrated an exponential $2^{\sqrt{2n+30}-6}$ and $\frac{1}{5}4^{\sqrt{n/2}}n^{-\frac{3}{4}}$ lower bound for $\downarrow L(A)$ and, respectively, $\uparrow L(A)$ [8]. The construction and the proof are quite involved, and they leave open the case where $|\Sigma| = 2$ (the 1-letter case is trivial). It turns out that, in the 2-letter case, Héam had already proved a $\Omega(r^{\sqrt{n}})$ lower bound for $\uparrow L(A)$, here with $r = (\frac{1+\sqrt{5}}{2})^{\frac{1}{\sqrt{2}}}$ [10], so that the main remaining question is whether $\downarrow L(A)$ may require an exponential number of states even when $|\Sigma| = 2$.

Dual to closures are *interiors*. The *upward interior* and *downward interior* of a language L, denoted $\circlearrowleft L$ and $\circlearrowright L$, are the largest upward-closed and, resp., downward-closed, sets included in L. Building closures and interiors are essential operations when reasoning with subwords, e.g., when model-checking lossy channel systems [11]. The state complexity of interiors has not yet been considered in the literature. When working with DFAs, computing interiors reduces to computing closures, thanks to duality. However, when working with NFAs, the simple complement+closure+complement algorithm only yields a quite large 2^{2^n} upper-bound on the number of states of an NFA for $\circlearrowleft L(A)$ or $\circlearrowright L(A)$ —it actually yields DFAs— and one would like to improve on this, or to prove a matching lower bound.

Our contribution. Regarding closures, we prove in Section 3 an exponential lower bound on $\downarrow L(A)$ in the case of a two-letter alphabet, answering the open question raised above. We also give some new proofs for known results, usually relying on simpler examples demonstrating hard cases. For example, we prove a tighter 2^{n-1} lower bound for $\downarrow L(A)$ when the alphabet is unbounded.

Regarding interiors on NFAs, we show in Section 4 a doubly-exponential lower bound for downward interiors when the alphabet is not bounded. In the case of upward interiors, or in the case of fixed alphabets, we are left with an exponential gap between lower bounds and upper bounds. A partial result is a doubly-exponential lower bound for a restricted version of these problems. Table 1 shows a summary of the known results.

Finally, we analyze in Section 5 the computational complexity of deciding whether $L(A)$ is upward or downward-closed for a DFA or a NFA A.

2 Basic notions and results

Fix a finite alphabet $\Sigma = \{a, b, \ldots\}$. We say that a ℓ-letter word $x = a_1\, a_2 \cdots a_\ell$ is a subword of y, written $x \sqsubseteq y$, when $y = y_0\, a_1\, y_1 \cdots y_{\ell-1}\, a_\ell\, y_\ell$ for some factors $y_0, \ldots, y_\ell \in \Sigma^*$, i.e., when there are positions $p_1 < p_2 < \cdots < p_\ell$ s.t. $x[i] = y[p_i]$ for all $1 \leqslant i \leqslant \ell = |x|$. For a language $L \subseteq \Sigma^*$, its downward closure is $\downarrow L \overset{\text{def}}{=} \{x \in \Sigma^* \mid \exists y \in L : x \sqsubseteq y\}$. Symmetrically, we consider an upward closure

Table 1. A summary of the results. Each cell shows (a bound on) the maximum number of states that can result when the operation is applied to an automaton with n states and the output is minimized.

Operation	NFA	DFA
Upward closure	n	$2^{\Theta(n)}$, and $2^{\Omega(n^{1/2})}$ for $\lvert\Sigma\rvert = 2$
Downward closure	n	$2^{\Theta(n)}$, and $2^{\Omega(n^{1/3})}$ for $\lvert\Sigma\rvert = 2$
Upward interior	$\leqslant 2^{2^n}$, $\Omega(2^n)$	same as downward closure
Downward interior	$2^{2^{\Theta(n)}}$	same as upward closure

operation and we let $\uparrow L \overset{\text{def}}{=} \{x \in \Sigma^* \mid \exists y \in L : y \sqsubseteq x\}$. For singletons, we may write $\uparrow x$ and $\downarrow x$ for $\uparrow\{x\}$ and $\downarrow\{x\}$, e.g., $\downarrow a\,b\,b = \{\epsilon, a, b, a\,b, b\,b, a\,b\,b\}$. Closures distribute over union, that is, $\uparrow L = \bigcup_{x\in L} \uparrow x$ and $\downarrow L = \bigcup_{x\in L} \downarrow x$. A language L is *downward-closed* (or *upward-closed*) if $L = \downarrow L$ (respectively, if $L = \uparrow L$). Note that L is downward-closed if, and only if, $\Sigma^* \smallsetminus L$ is upward-closed.

Upward-closed languages are also called *shuffle ideals* since they satisfy $L = L \sqcup\!\sqcup\, \Sigma^*$. They correspond exactly to level $\frac{1}{2}$ of Straubing's hierarchy [12].

Since, by Higman's Lemma, any L has only finitely many minimal elements wrt the subword ordering, one deduces that $\uparrow L$ is regular for any L.

Effective construction of a finite-state automaton for $\downarrow L$ or $\uparrow L$ is easy when L is regular (see Section 3), is possible when L is context-free [13, 14], and is not possible in general since this would allow deciding the emptiness of L.

The *upward interior* of L is $\circlearrowleft L \overset{\text{def}}{=} \{x \in \Sigma^* \mid \uparrow x \subseteq L\}$. Its *downward interior* is $\circlearrowright L \overset{\text{def}}{=} \{x \in \Sigma^* \mid \downarrow x \subseteq L\}$. Alternative characterizations are possible, e.g., by noting that $\circlearrowleft L$ (respectively, $\circlearrowright L$) is the largest upward-closed (respectively, downward-closed) language contained in L, or by using the following dualities:

$$\circlearrowright L = \Sigma^* \smallsetminus \uparrow(\Sigma^* \smallsetminus L)\,, \qquad \circlearrowleft L = \Sigma^* \smallsetminus \downarrow(\Sigma^* \smallsetminus L)\,. \tag{1}$$

If L is regular, one may compute automata for the interiors of L by combining complementations and closures as in Eq. (1).

When considering a finite automaton $A = (\Sigma, Q, \delta, I, F)$, we usually write n for $|Q|$ (the number of states), m for $|\delta|$ (the number of transitions, seeing $\delta \subseteq Q \times \Sigma \times Q$ as a table), and k for $|\Sigma|$ (the size of the alphabet). For a regular language L, $n_{\mathrm{D}}(L)$ and $n_{\mathrm{N}}(L)$ denote the minimum number of states of a DFA (resp., a NFA) that accepts L.

We now illustrate a well-known technique for proving lower bounds on $n_{\mathrm{N}}(L)$:

Lemma 2.1 (Extended fooling set technique, [15]). *Let L be a regular language. Suppose there exists a set of pairs of words $S = \{(x_i, y_i)\}_{1\leqslant i\leqslant n}$ such that for all i, j, $x_i\,y_i \in L$ and at least one of $x_i\,y_j$ and $x_j\,y_i$ is not in L. Then $n_N(L) \geqslant n$.*

Lemma 2.2 (An application of the fooling set technique). *Fix Σ and define the following two languages:*

$$U \overset{\text{def}}{=} \{x \mid \forall a \in \Sigma : \exists i : x[i] = a\}\,, \qquad V \overset{\text{def}}{=} \{x \mid \forall i \neq j : x[i] \neq x[j]\}\,. \tag{2}$$

Then $n_N(U) \geqslant 2^{|\Sigma|}$ and $n_N(V) \geqslant 2^{|\Sigma|}$.

Proof. The same proof applies to U and V: note that U has all words where every letter in Σ appears at least once, while V has all words where no letter appears twice.

With any $\Gamma \subseteq \Sigma$, we associate two words x_Γ and y_Γ, where x_Γ (respectively, y_Γ) has exactly one occurrence of each letter from Γ (respectively, each letter not in Γ). Then $x_\Gamma y_\Gamma$ is in U and V, while for any $\Delta \neq \Gamma$ one of $x_\Gamma y_\Delta$ and $x_\Delta y_\Gamma$ is not in U (and one is not in V). One concludes with Lemma 2.1. □

3 State complexity of closures

3.1 Nondeterministic automata

For a regular language L, an NFA for the upward or downward closure of L is obtained by simply adding transitions to an NFA for L, without increasing the number of states. More precisely, given an NFA A for L, an NFA for $\uparrow L$ is obtained by adding to A self-loops $q \xrightarrow{a} q$ for every state q of A and every letter $a \in \Sigma$. Similarly, an NFA for $\downarrow L$ is obtained by adding to A epsilon transitions $p \xrightarrow{\epsilon} q$ for every transition $p \to q$ of A (on any letter).

3.2 Deterministic automata

Since every DFA is an NFA, Section 3.1 along with the powerset construction shows that if a language has an n-state DFA, then both its upward and downward closures have DFAs with at most 2^n states. An exponential blowup is also necessary as we now illustrate.

Let $\Sigma = \{a_1, \ldots, a_k\}$ and define $L_1 \overset{\text{def}}{=} \{a\,a \mid a \in \Sigma\}$, i.e., L_1 contains all words consisting of two identical letters. The minimal DFA for L_1 has $n = k + 2$ and $m = 2k$, see Fig. 1.

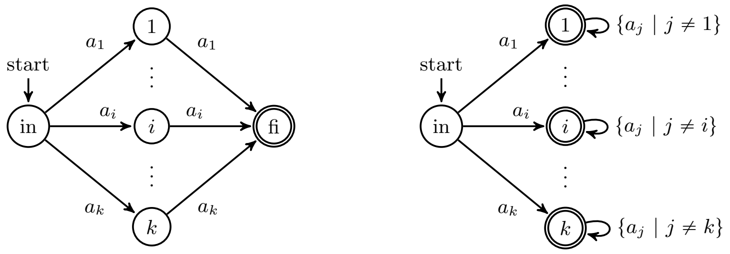

Fig. 1. DFAs for $L_1 = \bigcup_{a\in\Sigma} a\,a$ (left) and $L_2 = \bigcup_{a\in\Sigma} a \cdot (\Sigma \smallsetminus a)^*$ (right).

Now $\uparrow L_1 = \{x \in \Sigma^{\geqslant 2} \mid \exists j > i : x[i] = x[j]\} = \bigcup_{a\in\Sigma} \Sigma^* \cdot a \cdot \Sigma^* \cdot a \cdot \Sigma^*$, i.e., $\uparrow L_1$ has all words where *some letter reappears*, i.e., $\uparrow L_1$ is the complement of V from Lemma 2.2. A DFA for $\uparrow L_1$ has to record all letters previously read: the minimal (complete) DFA has $2^k + 1$ states. Hence $2^{n-2} + 1$ states are sometimes required for the minimal DFA recognizing the upward closure of an n-state DFA.

Further define $L_2 \stackrel{\text{def}}{=} \{x \in \Sigma^+ \mid \forall i > 1 : x[i] \neq x[1]\} = \bigcup_{a\in\Sigma} a \cdot \big(\Sigma \smallsetminus \{a\}\big)^*$, i.e., L_2 has words where *the first letter does not reappear*. The minimal DFA for L_2 has $n = k + 1$ and $m = k^2$, see Fig. 1. Now $\downarrow L_2 = \{x \mid \exists a \in \Sigma : \forall i > 1 : x[i] \neq a\} = \epsilon + \Sigma \cdot \bigcup_{a\in\Sigma}\big(\Sigma \smallsetminus \{a\}\big)^*$, i.e., $\downarrow L_2$ has all words x such that the first suffix $x[2,\ldots,\ell]$ does not use all letters. Equivalently $x \in \downarrow L_2$ iff $x \in L_2$ or x does not use all letters, i.e., $\downarrow L_2$ is the union of L_2 and the complement of U from Lemma 2.2. The minimal DFA for $\downarrow L_2$ just records all letters previously encountered *except the first*, hence has exactly 2^k states. Thus 2^{n-1} states may be required for a DFA recognizing the downward closure of an n-state DFA.

The above simple examples use a linear-sized alphabet to establish the lower bounds. This raises the question of whether exponential lower bounds still apply in the case of a fixed alphabet. The 1-letter case is degenerate since then both $n_\mathrm{D}(\uparrow L)$ and $n_\mathrm{D}(\downarrow L)$ are $\leqslant n_\mathrm{D}(L)$. In the 3-letter case, exponential lower bounds are shown in [8].

In the 2-letter case, an exponential lower bound for upward closure is shown with the following witness: For $n > 0$, let $L_n = \{a^i b\, a^{2j} b\, a^i \mid i + j + 1 = n\}$. Then $n_D(L_n) = (n+1)^2$, while $n_D(\uparrow L_n) \geqslant \frac{1}{7}(\frac{1+\sqrt{5}}{2})^n$ for $n \geqslant 4$ [10, Prop. 5.11]. However, the downward closure of these languages does not demonstrate a state blowup, in fact $n_D(\downarrow L_n) = n^2 + 3n - 1$ for $n \geqslant 2$.

We now show an exponential lower bound for downward closures in the case of a two-letter alphabet. Interestingly, the same languages can also serve as hard case for upward closure (but it gives weaker bounds than in [10]).

Theorem 3.1. *The state complexity of computing downward closure for DFAs is in $2^{\Omega(n^{1/3})}$. The same result holds for upward closure.*

We now prove the theorem. Fix a positive integer n. Let

$$S = \{n, n+1, \ldots, 2n\}\ ,$$

and define morphisms $c, h : S^* \to \{a,b\}^*$ with, for any $i \in S$:

$$c(i) \stackrel{\text{def}}{=} a^i\, b^{3n-i}\ , \qquad\qquad h(i) \stackrel{\text{def}}{=} c(i)\, c(i)\ .$$

Note that $c(i)$ always has length $3n$, begins with at least n a's, and ends with at least n b's. If we now let

$$L \stackrel{\text{def}}{=} \{c(i)^n \mid i \in S\}\ ,$$

L is a finite language of $n+1$ words, each of length $3n^2$ so that clearly $n_\mathrm{D}(L)$ is in $3n^3 + O(n^2)$. (In fact, $n_\mathrm{D}(L) = 3n^3 + 1$.) In the rest of this section we show that both $n_\mathrm{D}(\uparrow L)$ and $n_\mathrm{D}(\downarrow L)$ are in $2^{\Omega(n)}$.

Lemma 3.2. *For $i, j \in S$, the longest prefix of $c(i)^\omega$ that embeds in $h(j) = c(j)\, c(j)$ is $c(i)$ if $i \neq j$ and $c(i)\, c(i)$ if $i = j$.*

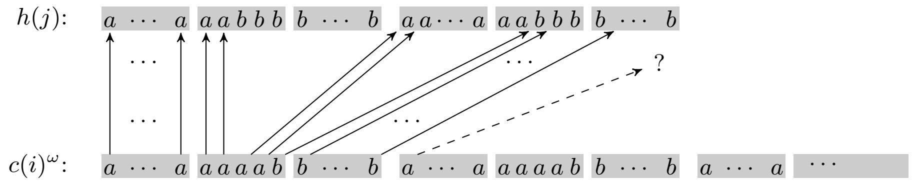

Fig. 2. Case "$i > j$" in Lemma 3.2: here $i = n+4$ and $j = n+2$ for $n = 5$.

Proof (Sketch). The case $i = j$ is clear. Fig. 2 displays the leftmost embedding of $c(i)^\omega$ in $h(j)$ in a case where $i > j$. The remaining case, $i < j$, is similar. □

For each $i \in S$, let the morphisms $\eta_i, \theta_i : S^* \to (\mathbb{N}, +)$ be defined by

$$\eta_i(j) \overset{\text{def}}{=} \begin{cases} 1 & \text{if } i \neq j\ , \\ 2 & \text{if } i = j\ , \end{cases} \qquad \theta_i(j) \overset{\text{def}}{=} \begin{cases} 2 & \text{if } i \neq j\ , \\ 1 & \text{if } i = j\ . \end{cases}$$

Thus for $\sigma = p_1 p_2 \cdots p_s \in S^*$, $\eta_i(\sigma)$ is s plus the number of occurrences of i in σ, while $\theta_i(\sigma)$ is $2s$ minus the number of these occurrences of i.

Lemma 3.3. *Let $\sigma \in S^*$. The smallest ℓ such that $c(\sigma)$ embeds in $c(i)^\ell$ is $\theta_i(\sigma)$.*

Proof. We write $\sigma = p_1 p_2 \cdots p_s$ and prove the result by induction on s. The $s = 0$ case is trivial. The $s = 1$ case follows from Lemma 3.2, since for any p_1 and i, $c(p_1) \sqsubseteq c(i)$ iff $p_1 = i$, and $c(p_1) \sqsubseteq h(i) = c(i)^2$ always.

Assume now $s > 1$, write $\sigma = \sigma' p_s$ and let $\ell' = \theta_i(\sigma')$. By the induction hypothesis, $c(\sigma') \not\sqsubseteq c(i)^{\ell'-1}$ and $c(\sigma') \sqsubseteq c(i)^{\ell'} = c(i)^{\ell'-1} a^i b^{3n-i}$. Write now $c(i)^{\ell'} = w\, v$ where w is the shortest prefix of $c(i)^{\ell'}$ with $c(\sigma') \sqsubseteq w$. Since $c(\sigma')$ ends with a b that only embeds in the $a^i b^{3n-i}$ suffix of $c(i)^{\ell'}$, v is necessarily b^r for some r. So for all $z \in \{a,b\}^*$, $c(p_s) \sqsubseteq z$ if and only if $c(p_s) \sqsubseteq v\, z$. We have $c(p_s) \sqsubseteq c(i)^{\theta_i(p_s)}$ and $c(p_s) \not\sqsubseteq v\, c(i)^{\theta_i(p_s)-1}$. Noting that $\sigma = \sigma' p_s$, we get $c(\sigma) \sqsubseteq c(i)^{\theta_i(\sigma)}$ and $c(\sigma) \not\sqsubseteq c(i)^{\theta_i(\sigma)-1}$. □

We now derive a lower bound on the number of states in the minimal complete DFA for $\downarrow L$. For every subset X of S of size $n/2$ (assume n is even), let $w_X \in \{a,b\}^*$ be defined as follows: let the elements of X be $p_1 < p_2 < \cdots < p_{n/2}$ and let

$$w_X \overset{\text{def}}{=} c(p_1 p_2 \cdots p_{n/2})\ .$$

Note that $\theta_i(p_1 p_2 \cdots p_{n/2}) = n$ if $i \notin X$ and $\theta_i(p_1 p_2 \cdots p_{n/2}) = n - 1$ if $i \in X$.

Lemma 3.4. *Let X and Y be subsets of S of size $n/2$ with $X \neq Y$. There exists a word $v \in \{a,b\}^*$ such that $w_X v \in \downarrow L$ and $w_Y v \notin \downarrow L$.*

Proof. Let $i \in X \smallsetminus Y$. Let $v = c(i)$. Then

– By Lemma 3.3, $w_X \sqsubseteq c(i)^{n-1}$, and so $w_X v \sqsubseteq c(i)^n$. So $w_X v \in \downarrow L$.
– By Lemma 3.3, the smallest ℓ such that $w_Y v \sqsubseteq c(i)^\ell$ is $n+1$. Similarly, for $j \neq i$, the smallest ℓ such that $w_Y v \sqsubseteq c(j)^\ell$ is at least $n - 1 + 2 = n + 1$ (the w_Y contributes at least $n-1$ and the v contributes 2). So $w_Y v \notin \downarrow L$. □

This shows that for any complete DFA A recognizing $\downarrow L$, the state of A reached from the start state by every word in $\{w_X \mid X \subseteq S, |X| = n/2\}$ is distinct. Thus A has at least $\binom{n+1}{n/2}$ states, which is $\approx \frac{2^{n+3/2}}{\sqrt{\pi n}}$.

For $n_{\mathrm{D}}(\uparrow L)$, the reasoning is similar:

Lemma 3.5. *Let $\sigma \in S^*$. For all $i \in S$, the longest prefix of $c(i)^\omega$ that embeds in $h(\sigma)$ is $c(i)^{\eta_i(\sigma)}$.*

Proof. By induction on the length of σ and applying Lemma 3.2. □

For every subset X of S of size $n/2$ (assume n is even), let $w'_X \in \{a,b\}^*$ be defined as follows: let the elements of X be $p_1 < p_2 < \cdots < p_{n/2}$ and let

$$w'_X \stackrel{\text{def}}{=} h(p_1 p_2 \cdots p_{n/2}) = c(p_1 p_1 p_2 p_2 \cdots p_{n/2} p_{n/2}) \,.$$

Lemma 3.6. *Let X and Y be subsets of S of size $n/2$ with $X \neq Y$. There exists a word $v \in \{a,b\}^*$ such that $w'_X v \in \uparrow L$ and $w'_Y v \notin \uparrow L$.*

Proof. Let $i \in X \smallsetminus Y$. Let $v = c(i)^{n-(n/2+1)} = c(i)^{n/2-1}$.

- By Lemma 3.5, $c(i)^{n/2+1} \sqsubseteq w'_X$, thus $c(i)^n \sqsubseteq w'_X v$, hence $w'_X v \in \uparrow L$.
- By Lemma 3.5, the longest prefix of $c(i)^n$ that embeds in $w'_Y v$ is at most $c(i)^\ell$ where $\ell = n/2 + n/2 - 1 = n - 1$. The longest prefix of $c(j)^n$ that embeds in $w'_Y v$ for $j \neq i$ is at most $c(j)^\ell$ where

$$\ell = \frac{n}{2} + 1 + \left\lceil \frac{n/2 - 1}{2} \right\rceil \leqslant n - 1$$

Therefore $c(j)^n \not\sqsubseteq w'_Y v$ when $j = i$ and also when $j \neq i$. Thus $w'_Y v \notin \uparrow L$. □

With Lemma 3.6 we reason exactly as we did for $n_{\mathrm{D}}(\downarrow L)$ after Lemma 3.4 and conclude that $n_{\mathrm{D}}(\uparrow L) \geqslant \binom{n+1}{n/2}$ here too.

4 State complexity of interiors

Recall Eq. (1) that expresses interiors with closures and complements. Since complementation of DFAs does not increase the number of states, the bounds on interiors are the same as the bounds on closures in the case of DFAs.

For NFAs, Eq. (1) provides an obvious 2^{2^n} upper bound on the NFA state complexity of both the upward and the downward interior, simply by combining the powerset construction for complementation and the results of Section 3.1. (Alternatively, it is possible to design a "powerset-like construction" that directly builds a DFA for the interior, upward or downward, of a language recognized by a DFA: this returns the same DFA as with the complement+closure+complement procedure.) Note that both procedures yield DFAs for the interiors while we are looking for better bounds on their NFA state complexity.

Proposition 4.1. *The NFA state complexity of the downward interior is in $2^{2^{\Theta(n)}}$ (assuming an unbounded alphabet).*

Proof. Let ℓ be a positive integer, and let $\Sigma = \{0,1\}^\ell$, so that $k = |\Sigma| = 2^\ell$. Let

$$L \stackrel{\text{def}}{=} \Sigma^* \smallsetminus \{a\,a \mid a \in \Sigma\} = \{w \mid |w| \neq 2\} \cup \{a\,b \mid a, b \in \Sigma, a \neq b\} .$$

Two letters in Σ, viewed as ℓ-bit sequences, are distinct if and only if they differ in at least one bit. An NFA can check this by guessing the position in which they differ and checking that the letters indeed differ in this position. Fig. 3 shows an NFA for $\{a\,b \mid a \neq b\}$ with $2\ell + 2$ states.

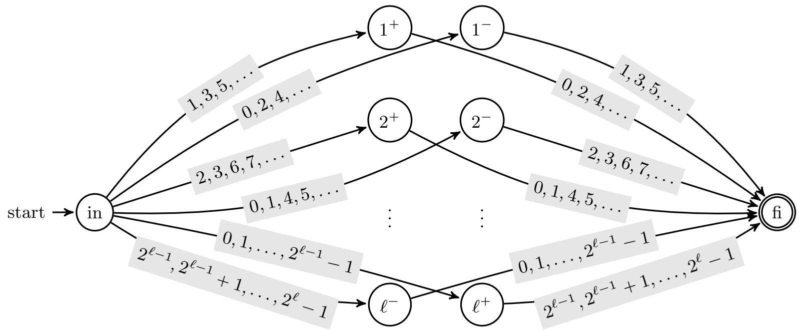

Fig. 3. DFA for $\{a\,b \mid a, b \in \Sigma, a \neq b\}$ with $2\ell + 2$ states and $\ell 2^\ell$ transitions.

Since now $\{w \mid |w| \neq 2\}$ is recognized by an NFA with 4 states, L is recognized by an NFA with $n = 2\ell + 6$ states.

Finally, $\Cap L$ consists of all words where every letter is distinct (equivalently, no letter appears more than once), a language called V in Eq. (2). We conclude with Lemma 2.2 showing $n_{\mathrm{N}}(V) \geqslant 2^{|\Sigma|} = 2^{2^\ell} = 2^{2^{n/2-3}}$. □

Proposition 4.2. *The NFA state complexity of the upward interior is in $\Omega(2^n)$ (assuming an unbounded alphabet).*

Proof. For Σ a k-letter alphabet we consider $L_3 \stackrel{\text{def}}{=} \Sigma^* \smallsetminus L_2$ with the same L_2 used earlier, see Fig. 1 in Section 3.2. Thus L_3 contains all words where the first letter reappears. (It also contains the empty word). By complementing the DFA for L_2, one sees that a minimal DFA for L_3 has $n = k + 2$ states.

We noted in Section 3.2 that $\downarrow L_2 = L_2 \cup (\Sigma^* \smallsetminus U)$, where U is the language of all words where each letter from Σ occurs at least once. Hence $\circlearrowleft L_3 = \Sigma^* \smallsetminus \downarrow L_2 = (\Sigma^* \smallsetminus L_2) \cap U = L_3 \cap U$.

Observe now that for any $a \in \Sigma$ and $w \in \Sigma^*$, $a\,w \in L_3 \cap U$ iff $w \in U$. Thus any NFA for $L_3 \cap U$ can be transformed into an NFA for U by simply changing the initial states, and so a state lower bound for U implies the same lower bound for $L_3 \cap U$. With Lemma 2.2 we get $n_{\mathrm{N}}(\circlearrowleft L_3) = n_{\mathrm{N}}(L_3 \cap U) \geqslant n_{\mathrm{N}}(U) \geqslant 2^k = 2^{n-2}$, witnessing the required exponential lower bound. □

The above results leave us with an exponential gap between lower and upper bounds for $n_{\mathrm{N}}(\circlearrowleft L)$ —and even for $n_{\mathrm{D}}(\circlearrowleft L)$— when L is given by a NFA. We have not been able to close this gap and we do not yet feel able to formulate a conjecture on whether an exponential $2^{n^{O(1)}}$ bound exists or not. Trying to find hard cases by exhaustive or heuristic search is difficult because the search space is huge even for small n, and for most languages the upward interior is trivial. For NFAs with $n = 3$ states and with $|\Sigma| = k = 3$ letters, a worst case example is $L = \big((a+b)(a+b+c)^*(a+b) + (b+c)(a+b+c)^*(a+c)\big)^*$. Here $n_{\mathrm{N}}(L) = 3$ and $n_{\mathrm{N}}(\circlearrowleft L) = 10$, which is well below the 2^{2^n} upper bound.

In the rest of this section, we establish a doubly exponential lower bound for a more general construction called *restricted interior.*

Let Σ be an alphabet and let $X \subseteq \Sigma$. For words u, v, we write $u \sqsubseteq_X v$ if u is obtained from v by deleting some (occurrences of) letters in X, necessarily keeping letters in $\Sigma \setminus X$ intact. For example, $a\,b\,b\,a \sqsubseteq_{\{b,c\}} a\,b\,c\,b\,c\,b\,c\,a\,c$, but $b\,b\,a \not\sqsubseteq_{\{b,c\}} a\,b\,c\,b\,c\,b\,c\,a\,c$. Closures and interiors are defined as one would expect:

$$\downarrow_X L \stackrel{\text{def}}{=} \{w \mid \exists v \in L : w \sqsubseteq_X v\}, \quad \circlearrowright_X L \stackrel{\text{def}}{=} \{w \mid \downarrow_X\{w\} \subseteq L\} = \Sigma^* \setminus \uparrow_X(\Sigma^* \setminus L),$$
$$\uparrow_X L \stackrel{\text{def}}{=} \{w \mid \exists v \in L : v \sqsubseteq_X w\}, \quad \circlearrowleft_X L \stackrel{\text{def}}{=} \{w \mid \uparrow_X\{w\} \subseteq L\} = \Sigma^* \setminus \downarrow_X(\Sigma^* \setminus L).$$

Theorem 4.3. *The NFA state complexity of the restricted upward interior is* $\leqslant 2^{2^n}$ *and in* $2^{2^{\Omega(\sqrt{n})}}$*. The lower bound holds with a 3-letter alphabet.*

As with $\downarrow L$, one can obtain an NFA for $\downarrow_X L$ from an NFA for L by simply adding transitions, without adding new states. Hence the upper bound is clear in Theorem 4.3, and we only need to prove the lower bound.

Fix $n \in \mathbb{N}$. Let $\Sigma = \{0, 1, \#\}$, and $\Sigma_{01} = \{0, 1\}$. Define the following languages:

– N is the set of all words over Σ in which the sum of the number of 0s and the number of 1s is divisible by n;
– $B = ((\epsilon + \#)(0+1)^n)^*$. Note that $B \subseteq N$;
– H_2 is the set of all words over Σ with exactly two occurrences of #.

Let $L \subseteq \Sigma^*$ consists of all the following words:

– words in $(N \setminus H_2) \cup (N \setminus B)$;
– words in $H_2 \cap B$ such that the factors of length n immediately following the two occurrences of # are distinct.

Both $N \setminus H_2$ and $N \setminus B$ are recognized by NFAs with $O(n)$ states. The second summand of L is recognized by an NFA with $O(n^2)$ states, as the n-length factors immediately following the two occurrences of # being unequal can be checked by guessing a position at which they differ. So L is recognized by an NFA with $O(n^2)$ states. Note that $L \subseteq N$.

Consider $\circlearrowleft_{\{\#\}} L$. This is the set of all words in L such that no matter how we insert occurrences of #, the resulting word remains in L.

Let $\Gamma = \{0,1\}^n$, considered as an alphabet. Define the homomorphism $h : \Gamma^* \to \Sigma^*$, as $h(x) = x$ for all x. As in Lemma 2.2, let $V \subseteq \Gamma^*$ consist of all words over Γ in which no letter appears twice, and define $V' \subseteq \Sigma^*$ as $h(V)$. Note that $V' \subseteq N \cap \Sigma_{01}^*$.

Lemma 4.4. $V' = (\circlearrowleft_{\{\#\}} L) \cap \Sigma_{01}^*$.

Proof. ($\subseteq$:) Let $w \in V'$ and v be such that $w \sqsubseteq_{\{\#\}} v$. Since $V' \subseteq N = \uparrow_{\{\#\}}(N)$, we know that $v \in N$. If $v \notin H_2 \cap B$, then $v \in L$. Otherwise, if $v \in H_2 \cap B$, then by the definition of V, it is easy to see that $v \in L$.

($\supseteq$:) Conversely, let $w \in (\circlearrowleft_{\{\#\}} L) \cap \Sigma_{01}^*$. In particular, $w \in L$, and so $w \in N$. $|w|$ is divisible by n, and so $w = h(x)$ for some $x \in \Gamma^*$. By inserting two copies of # at suitable positions in w, and using the fact that the resulting word belongs to L, one concludes that $x \in V$, and so $w \in V'$. □

Lemma 4.5. $2^{2^n} \leqslant n_N(V) \leqslant n_N(V') \leqslant n_N(\circlearrowleft_{\{\#\}} L)$.

Proof. Lemma 2.2 gives $2^{2^n} \leqslant n_{\mathrm{N}}(V)$. Then $V = h^{-1}(V')$ gives $n_{\mathrm{N}}(V) \leqslant n_{\mathrm{N}}(V')$ since it is easy to transform an NFA for V' into an NFA for $h^{-1}(V')$ (for any morphism h, in fact) with the same number of states. Finally, $V' = (\circlearrowleft_{\{\#\}} L) \cap \Sigma_{01}^*$ gives $n_{\mathrm{N}}(V') \leqslant n_{\mathrm{N}}(\circlearrowleft_{\{\#\}} L)$ since an NFA for V' can be obtained from an NFA for $\circlearrowleft_{\{\#\}} L$ by deleting all transitions labelled by #. □

Since $n_{\mathrm{N}}(L)$ is in $O(n^2)$, Lemma 4.5 concludes the proof of Theorem 4.3.

5 Complexity of decision problems on subwords

In automata-based procedures for logic and verification, the state complexity of automata constructions is not always the best measure of computational complexity. In this section we give elementary proofs showing that the problem of deciding whether $L(A)$ is upward-closed, or downward-closed, is unsurprisingly PSPACE-complete for NFAs, and NL-complete for DFAs. (For upward-closedness, this is already shown in [10], and quadratic-time algorithms that decide upward-closedness of $L(A)$ for a DFA A already appear in [16, 12].)

Proposition 5.1. *Deciding whether $L(A)$ is upward-closed or downward-closed is* PSPACE*-complete when A is a NFA, even in the 2-letter alphabet case.*

Proof (Sketch). A PSPACE algorithm simply tests for inclusion between two automata, A and its closure. PSPACE-hardness can be shown by adapting the proof for hardness of universality. Let R be a length-preserving semi-Thue system and x, x' two strings of same length. It is PSPACE-hard to say whether $x \xrightarrow{*}_R x'$, even for a fixed R over a 2-letter alphabet Σ. We reduce (the negation of) this question to our problem.

Fix x and x' of length $n > 1$: a word $x_1 x_2 \cdots x_m$ of length $n \times m$ encodes a derivation if $x_1 = x$, $x_m = x'$, and $x_i \to_R x_{i+1}$ for all $i = 1, \ldots, m-1$. The language L of words that do *not* encode a derivation from x to x' is regular and recognized by a NFA with $O(n)$ states. Now, there is a derivation $x \xrightarrow{*}_R x'$ iff $L \neq \Sigma^*$. Since L contains all words of length not divisible by $n > 1$, it is upward-closed, or downward-closed, iff $L = \Sigma^*$, iff $\neg(x \xrightarrow{*}_R x')$. □

Proposition 5.2. *Deciding whether $L(A)$ is upward-closed or downward-closed is* NL*-complete when A is a DFA, even in the 2-letter alphabet case.*

Proof. Since L is downward-closed if, and only if, $\Sigma^* \smallsetminus L$ is upward-closed, and since one easily builds a DFA for the complement of $L(A)$, it is sufficient to prove the result for upward-closedness.

We rely on the following easy lemma: L is upward-closed iff for all $u, v \in \Sigma^*$, $u\,v \in L$ implies $u\,a\,v \in L$ for all $a \in \Sigma$. Therefore, $L(A)$ is not upward-closed —for $A = (\Sigma, Q, \delta, \{i\}, F)$— iff there are states $p, q \in Q$, a letter a, and words u, v such that $\delta(i, u) = p$, $\delta(p, a) = q$, $\delta(p, v) \in F$ and $\delta(q, v) \notin F$. If such words exist, in particular one can take u and v of length $< n = |Q|$ and respectively $< n^2$. Hence testing (the negation of) upward-closedness can be done in nondeterministic logarithmic space by guessing u, a, and v within the above length bounds, finding p and q by running u and then a from i, then running v from both p and q.

For hardness, one may reduce from vacuity of DFAs, a well-known NL-hard problem that is essentially equivalent to GAP, the Graph Accessibility Problem. Note that for any DFA A (in fact any NFA) the following holds:

$$L(A) = \varnothing \quad \text{iff} \quad L(A) \cap \Sigma^{<n} = \varnothing \quad \text{iff} \quad L(A) \cap \Sigma^{<n} \text{ is upward-closed,}$$

where n is the number of states of A. This provides the required reduction since, given a FSA A, one easily builds a FSA for $L(A) \cap \Sigma^{<n}$. □

6 Concluding remarks

For words ordered by the (scattered) subword relation, we considered the state complexity of computing closures and interiors, both upward and downward, of regular languages given by finite-state automata. These operations are essential when reasoning with subwords, e.g., in symbolic model checking for lossy channel systems, see [11, Section 6]. We completed the known results on closures by demonstrating an exponential lower bound on downward closures even in the case of a two-letter alphabet.

The state complexity of interiors is a new problem that we introduced in this paper and for which we only have partial results: we show that the doubly-exponential upper bound for interiors of NFAs is matched by a doubly-exponential lower bound in the case of downward interiors when the alphabet is unbounded. For upward interiors of NFAs, or for fixed alphabets, there remains an exponential gap between the existing upper and lower bounds.

These results contribute to a more general research agenda: what are the right data structures and algorithms for reasoning with subwords and superwords? The algorithmics of subwords has mainly been developed in string matching and combinatorics [4, 17] but other applications exist that require handling sets of strings rather than individual strings, e.g., model-checking and constraint solving. For these applications, there are many different ways of representing closed sets and automata-based representation are not always the preferred option, see, e.g., the SREs used for downward-closed languages in [2]. The existing trade-offs between all the available options are not yet well understood and certainly deserve scrutiny.

Acknowledgments. We thank S. Schmitz and the anonymous reviewers for their helpful comments.

References

1. Sheng Yu. State complexity: Recent results and open problems. *Fundamenta Informaticae*, 64(1–4):471–480, 2005.
2. P. A. Abdulla, A. Collomb-Annichini, A. Bouajjani, and B. Jonsson. Using forward reachability analysis for verification of lossy channel systems. *Formal Methods in System Design*, 25(1):39–65, 2004.
3. Ch. Haase, S. Schmitz, and Ph. Schnoebelen. The power of priority channel systems. In *Proc. CONCUR 2013*, volume 8052 of *Lecture Notes in Computer Science*, pages 319–333. Springer, 2013.
4. R. A. Baeza-Yates. Searching subsequences. *Theoretical Computer Science*, 78(2):363–376, 1991.
5. L. H. Haines. On free monoids partially ordered by embedding. *Journal of Combinatorial Theory*, 6(1):94–98, 1969.
6. H. Gruber, M. Holzer, and M. Kutrib. More on the size of Higman-Haines sets: Effective constructions. *Fundamenta Informaticae*, 91(1):105–121, 2009.
7. J.-C. Birget. Partial orders on words, minimal elements of regular languages and state complexity. *Theoretical Computer Science*, 119(2):267–291, 1993.
8. A. Okhotin. On the state complexity of scattered substrings and superstrings. *Fundamenta Informaticae*, 99(3):325–338, 2010.
9. J. A. Brzozowski, G. Jirásková, and Baiyu Li. Quotient complexity of ideal languages. *Theoretical Computer Science*, 470:36–52, 2013.
10. P.-C. Héam. On shuffle ideals. *RAIRO Theoretical Informatics and Applications*, 36(4):359–384, 2002.
11. N. Bertrand and Ph. Schnoebelen. Computable fixpoints in well-structured symbolic model checking. *Formal Methods in System Design*, 43(2):233–267, 2013.
12. J.-É. Pin and P. Weil. Polynomial closure and unambiguous product. *Theory of Computing Systems*, 30(4):383–422, 1997.
13. J. van Leeuwen. Effective constructions in well-partially-ordered free monoids. *Discrete Mathematics*, 21(3):237–252, 1978.
14. B. Courcelle. On constructing obstruction sets of words. *EATCS Bulletin*, 44:178–185, 1991.
15. H. Gruber and M. Holzer. Finding lower bounds for nondeterministic state complexity is hard. In *Proc. DLT 2006*, volume 4036 of *Lecture Notes in Computer Science*, pages 363–374. Springer, 2006.
16. M. Arfi. Polynomial operations on rational languages. In *Proc. STACS '87*, volume 247 of *Lecture Notes in Computer Science*, pages 198–206. Springer, 1987.
17. C. H. Elzinga, S. Rahmann, and Hui Wang. Algorithms for subsequence combinatorics. *Theoretical Computer Science*, 409(3):394–404, 2008.