Kuratowski Algebras Generated by Factor-, Subword-, and Suffix-Free Languages

Jozef Jirásek Jr.[1,2], Matúš Palmovský[1], and Juraj Šebej[2(✉)]

[1] Mathematical Institute, Slovak Academy of Sciences, Grešákova 6, 040 01 Košice, Slovakia
jirasekjozef@gmail.com, palmovsky@saske.sk

[2] Faculty of Science, Institute of Computer Science, P.J. Šafárik University, Jesenná 5, 040 01 Košice, Slovakia
juraj.sebej@gmail.com

Abstract. We study Kuratowski algebras generated by suffix-, factor-, and subword-free languages under the operations of star and complementation. We examine 12 possible algebras, and for each of them, we provide an answer to the question whether or not it can be generated by a suffix-, factor-, or subword-free language. In each case when an algebra can be generated by such a language, we show that this language may be taken to be regular, and we compute upper bounds on the state complexities of all the generated languages. Finally, we find generators that maximize these complexities.

1 Introduction

The famous Kuratowski's 14-theorem states that, in a topological space, repeatedly applying the operations of closure and complement to any given set can produce at most 14 distinct sets [6,12]. Kuratowski's theorem in the settings of formal languages has been studied by Brzozowski et al. [2]. It has been shown that repeatedly applying Kleene closure and complementation to a given language produces again up to 14 distinct languages. Moreover, all formal languages have been classified according to the structure of the algebras they generate under Kleene closure and complementation. It has been proved that there are precisely 12 such algebras, and even more, each of them can be generated by a binary regular language.

Recently, Kuratowski algebras generated by certain restricted classes of languages have been investigated. Brzozowski et al. [4] proved that prefix-, suffix-, factor-, and subword-closed languages can generate at most 8 languages under the above mentioned operations. They also gave an example of a regular language

M. Palmovský—Research supported by VEGA grant 2/0084/15 and grant APVV-15-0091. This work was conducted as a part of PhD study of Matúš Palmovský at the Faculty of Mathematics, Physics and Informatics of the Commenius University in Bratislava.

J. Šebej—Research supported by VEGA grant 1/0142/15 and grant APVV-15-0091.

© IFIP International Federation for Information Processing 2017
Published by Springer International Publishing AG 2017. All Rights Reserved
G. Pighizzini and C. Câmpeanu (Eds.): DCFS 2017, LNCS 10316, pp. 189–201, 2017.
DOI: 10.1007/978-3-319-60252-3_15

in each of these four classes which generates 8 languages, and also maximizes their state complexities.

In [10], Kuratowski algebras generated by prefix-free languages have been investigated in detail. For each of the 12 possible algebras, the following questions have been answered:

1. Can this algebra be generated by a prefix-free language?
2. Can this algebra be generated by a regular prefix-free language?
3. Can this algebra be generated by a regular prefix-free language of an arbitrary state complexity?
4. What are the maximal state complexities of languages generated in this algebra by a prefix-free regular language?
5. Is there a prefix-free regular generator which maximizes all of these complexities at the same time?

In this paper, we answer the same questions for suffix-, factor-, and subword-free languages. For each of these three classes and each of the 12 algebras, if the algebra can be generated by a language in this class, we give an example of a regular generator. We discuss state complexities of all the generated languages.

If an algebra can be generated by a prefix-free language, then it can also be generated by a suffix-free language, and vice versa. However, we show that there are algebras which are generated by a prefix- (or suffix-) free language, but cannot be generated by any factor-free language. One interesting conclusion is that while in the prefix-free case, if an algebra can be generated by a prefix-free language, the answer to question 5 is always yes, for suffix-free languages this is not always the case.

2 Preliminaries

We assume that the reader is familiar with basic notions in formal languages and automata theory. For details, the reader may refer to [9,13,14].

If Σ is a finite alphabet, then Σ^* is the set of strings over Σ, including the empty string ε. The length of a string w is denoted by $|w|$. A language is any subset of Σ^*. The complement of a language L is the language $L^c = \Sigma^* \setminus L$. The concatenation of languages K and L is $KL = \{uv \mid u \in K \text{ and } v \in L\}$. The Kleene closure, or star, of L is defined as $L^* = \cup_{i \geq 0} L^i$, while the positive closure of L is $L^+ = \cup_{i \geq 1} L^i$, where $L^0 = \{\varepsilon\}$ and $L^{i+1} = L^i L$. To simplify the exposition, we use an exponent notation, so for example, L^{c*} and L^{*c*} stand for $(L^c)^*$ and $((L^*)^c)^*$, respectively.

A *nondeterministic finite automaton* (NFA) is a quintuple $A = (Q, \Sigma, \cdot, s, F)$ defined in a usual way. A state q_d of an NFA A is called a *dead state* if no string is accepted by A from q_d, that is, if $q \cdot w \cap F = \emptyset$ for each string w. We say that (p, a, q) is a transition in NFA A if $q \in p \cdot a$. We also say that the state p has an *out-transition* on a, and the state q has an *in-transition* on a. An NFA is *non-exiting* if its final states have no out-transitions, and it is *non-returning* if its initial state does not have any in-transitions.

An NFA A is a *deterministic finite automaton* DFA if for each state q and each input symbol a, the set $q \cdot a$ has exactly one element. The *state complexity* of a regular language L, sc(L), is the smallest number of states in any DFA for L. It is well known that a DFA is minimal with respect to the number of states if all its states are reachable and pairwise distinguishable.

Every NFA $A = (Q, \Sigma, \cdot, s, F)$ can be converted to an equivalent DFA $A' = (2^Q, \Sigma, \circ, \{s\}, F')$, where $F' = \{S \in 2^Q \mid S \cap F \neq \emptyset\}$ and $S \circ a = S \cdot a$ for each S in 2^Q and each a in Σ. We call the DFA A the *subset automaton* of the NFA A. The subset automaton may not be minimal since some of its states can be unreachable or equivalent to other states. To prove distinguishability of states of the subset automaton, the following notions from [3] are useful.

A state q of the NFA A is said to be *uniquely distinguishable* if there is a string w which is accepted by A from and only from the state q. Next, we say that a transition (p, a, q) is a *unique in-transition* if there is no state r different from p such that (r, a, q) is a transition in A. Finally, we say that a state q is *uniquely reachable* from a state p if there is a sequence of unique in-transitions (p_{i-1}, a_i, q_i) for $i = 1, 2, \ldots, k$ such that $p_0 = p$ and $p_k = q$.

If a uniquely distinguishable state q of an NFA A be uniquely reachable from a state p, then the state p is uniquely distinguishable as well. Next, if two subsets of a subset automaton of an NFA A differ in a uniquely distinguishable state of A, then the two subsets are distinguishable. It follows that if a uniquely distinguishable state of an NFA A is uniquely reachable from any other state of A, then the subset automaton of A does not have equivalents states.

If $u, v, w, x \in \Sigma^*$ and $w = uxv$, then u is a *prefix* of w, x is a *factor* of w, and v is a *suffix* of w. If $w = u_0 v_1 u_1 \cdots v_n u_n$, where $u_i, v_i \in \Sigma^*$, then $v_1 v_2 \cdots v_n$ is a *subword* of w. A prefix v (suffix, factor, subword) of w is *proper* if $v \neq w$.

A language L is *prefix-free* if $w \in L$ implies that no proper prefix of w is in L. Suffix-, factor-, and subword-free languages are defined analogously. A language L is *weakly-prefix-closed* if $w \in L$ implies that each non-empty prefix of w is in L. It is known that a minimal DFA for a prefix-free (suffix-free) language is non-exiting (non-returning) [7,8].

A language is *(positive-)closed* if it is closed under positive closure, that is, if $L = L^+$. It is *open*, if its complement is closed, and it is *clopen* if it is both closed and open. The terms *Kleene-closed* and *Kleene-open* are defined analogously. The *(positive) interior* of a language L is $L^{\oplus} = L^{c+c}$. The *Kleene interior* is $L^{\circledast} = L^{c*c}$. Notice that L is open iff $L = L^{\oplus}$. Next, for every language L, L^+ is closed and $L^{\oplus}$ is open.

Let $B(L)$ be the family of all languages generated from L by positive closure and positive interior; see [2, Subsect. 4.1]. Let $D(L)$ be the family of all languages generated from L by complementation and Kleene closure. Let $E(L)$ be the family of all languages generated from L by Kleene closure and Kleene interior. It is shown in [2, Lemma 20] that

$$D(L) = E(L) \cup \{M \mid M^c \in E(L)\}. \tag{1}$$

Moreover, if L is neither open nor closed, then by [2, Lemma 22],

$$E(L) = \{L\} \cup \{M \cup \{\varepsilon\} \mid M \in B(L) \text{ and } M \text{ is closed}\} \\ \cup \{M \setminus \{\varepsilon\} \mid M \in B(L) \text{ and } M \text{ is open}\}.$$

For each language L, the family $D(L)$ has at most 14 distinct languages, and Table 2 in [2, p. 312] describes 12 possible algebras, each of which is generated by a regular language. Notice that there is an oversight in cases (2a) and (2b): In case (2a) we should have $\varepsilon \notin L$, $|E(L)| = 3$, $|D(L)| = 6$, and it is generated by $\{a\}$. In case (2b) we should have $\varepsilon \in L$, $|E(L)| = 4$, $|D(L)| = 8$, and it is generated by $\{\varepsilon, a\}$ [1]. Here we show a modified table in which we do not display $|D(L)|$, and instead of $|E(L)|$, we display the set $E(L)$. We assume that L is a prefix-free (suffix-, factor-, subword-free) language and use the facts that $M^+ \cup \{\varepsilon\} = M^*$ and $M \setminus \{\varepsilon\} = M$ if $\varepsilon \notin M$. Notice that by (1), we only need to know the state complexities of the languages in $E(L)$.

Table 1. Classification of languages by the structure of $(E(L), ^*, ^{\circledast})$; cf. [2, p. 312].

Case	Necessary and sufficient conditions	$E(L)$	Regular generator
(1a)	L is clopen; $\varepsilon \in L$	$L, L \setminus \{\varepsilon\}$	a^*
(1b)	L is clopen; $\varepsilon \notin L$	$L, L \cup \{\varepsilon\}$	a^+
(2a)	L is open but not clopen; $\varepsilon \notin L$	L, L^*, L^+	a
(2b)	L is open but not clopen; $\varepsilon \in L$	$L, L \setminus \{\varepsilon\}, L^*, L^+$	$a \cup \varepsilon$
(3a)	L is closed but not clopen; $\varepsilon \notin L$	$L, L^\oplus, L^\oplus \cup \{\varepsilon\}$	aaa^*
(3b)	L is closed but not clopen; $\varepsilon \in L$	$L, L \cup \{\varepsilon\}$, $L^\oplus, L^\oplus \cup \{\varepsilon\}$	$aaa^* \cup \varepsilon$
(4)	L is neither open nor closed; L^+ is clopen and $L^{\oplus+} = L^+$	$L, L^*, L^+, L^\oplus$	$a \cup aaa$
(5)	L is neither open nor closed; $L^\oplus$ is clopen and $L^{+\oplus} = L^\oplus$	$L, L^*, L^\oplus \cup \{\varepsilon\}, L^\oplus$	aa
(6)	L is neither open nor closed; L^+ is open but $L^\oplus$ is not closed; $L^{\oplus+} \neq L^+$	L, L^*, L^+, $L^\oplus, L^{\oplus*}, L^{\oplus+}$	$G :=$ $a \cup abaa$
(7)	L is neither open nor closed; $L^\oplus$ is closed but L^+ is not open; $L^{+\oplus} \neq L^\oplus$	$L, L^*, L^\oplus \cup \{\varepsilon\}, L^\oplus$, $L^{+\oplus} \cup \{\varepsilon\}, L^{+\oplus}$	$(a \cup b)^+ \setminus$ G
(8)	L is neither open nor closed; $L^\oplus$ is not closed and L^+ is not open; $L^{+\oplus} = L^{\oplus+}$	$L, L^*, L^\oplus$, $L^{+\oplus} \cup \{\varepsilon\}, L^{+\oplus}$	$a \cup bb$
(9)	L is neither open nor closed; $L^\oplus$ is not closed and L^+ is not open; $L^{+\oplus} \neq L^{\oplus+}$	$L, L^*, L^\oplus$, $L^{+\oplus} \cup \{\varepsilon\}, L^{+\oplus}$, $L^{\oplus*}, L^{\oplus+}$	$a \cup ab \cup bb$

3 Factor-Free and Subword-Free Languages

In this section we investigate Kuratowski algebras generated by factor- and subword-free languages. In [10] we have already shown that algebras in cases (2b), (3a), (3b), (4), and (7) cannot be generated by a prefix-free language. Therefore these cases cannot be generated by a factor- or subword-free language either. We examine all the remaining cases and either show that the algebra cannot be generated by any factor-free (and therefore also any subword-free) language, or we give an example of a subword-free (and therefore also factor-free) regular generator. Moreover, the given generators maximize the state complexities of all the generated languages. We begin by stating several helpful observations.

Proposition 1. *Let* $n \geq 3$. *If* L *is a factor free language over* Σ *with* $\operatorname{sc}(L) = n$, *then* $\operatorname{sc}(L^*) \leq n - 1$ *if* $|\Sigma| \geq 2$ *and* $\operatorname{sc}(L^*) = n - 2$ *if* $|\Sigma| = 1$.

Proof. Let $A = (\{s, 1, 2, \ldots, n-3, q_f, q_d\}, \Sigma, \cdot, s, \{q_f\})$ be a minimal non-returning and non-exiting DFA for L with the dead state q_d. Construct a DFA for L^* from A by making the state q_f initial and the state s non-initial, and by replacing each transition (q_f, a, q_d) with $(q_f, a, s \cdot a)$. In the resulting DFA, the state s is unreachable, so $\operatorname{sc}(L^*) \leq n-1$. In the unary case, we must have $L = \{a^{n-2}\}$, so $\operatorname{sc}(L^*) = n - 2$. $\square$

Since the language $L^{\oplus}$ contains those strings of L that cannot be expressed as a concatenation of strings of L^c, we get the next proposition.

Proposition 2.

(a) Let $K \subseteq L \subseteq \Sigma^*$ *and* K *be weakly-prefix-closed. Then* $K \subseteq L^{\oplus}$.
(b) Let $L \subseteq \Sigma^*$ *and* $\Gamma = L \cap \Sigma$. *Then* $\Gamma \subseteq L^{\oplus}$.
(c) Let $L \subseteq \Sigma^*$ *and* $\Gamma = L \cap \Sigma$. *If* L *is a factor-free language different from* $\{\varepsilon\}$, *then* $L^{\oplus} = \Gamma$ *and* $L^{+\oplus} = L^{\oplus+} = \Gamma^+$.

Proof. (a) Since K is weakly-prefix-closed, every non-empty prefix of every string in K is in K as well. Therefore, no string in K can be expressed as a concatenation of strings in L^c. Hence $K \subseteq L^{\oplus}$. Claim (b) follows directly from (a).

(c) We have $\Gamma \subseteq L^{\oplus}$ by (b). The empty string and strings in $\Sigma \setminus \Gamma$ are not in L, therefore they are not in $L^{\oplus}$. Let $w \in L$ and $|w| \geq 2$. Since L is factor-free, no symbol occurring in w is in L. It follows that w can be partitioned into one-symbol strings that are in L^c. Hence $w \notin L^{\oplus}$, so $L^{\oplus} = \Gamma$ and $L^{\oplus+} = \Gamma^+$. Since $\Gamma^+ \subseteq L^+$ and Γ^+ is weakly-prefix-closed, we have $\Gamma^+ \subseteq L^{+\oplus}$ by (a). Let w be a string in L^+ which contains a symbol in $\Sigma \setminus \Gamma$. Then w must contain at least two such symbols. Therefore we can split w into substrings, each of which contains exactly one symbol in $\Sigma \setminus \Gamma$. These strings cannot be in L^+, therefore $w \in L^{+\oplus}$. Hence $L^{+\oplus} = \Gamma^+$, so $L^{+\oplus} = L^{\oplus+}$. $\square$

Now we examine the individual cases of possible Kuratowski algebras generated by factor- and subword-free languages. Our aim is to get the results that are summarized in Table 2. In each case we first recall sufficient and necessary conditions from Table 1, and then we discuss the case in detail.

Table 2. Binary subword-free generators of Kuratowski algebras maximizing complexities of generated languages. Cases (2b), (3a), (3b), (4), (6), (7), and (9) cannot be generated by any factor- or subword-free language.

Case	$E(L)$	Upper bounds on state complexities	Subword-free generator
(1a)	$L, L \setminus \{\varepsilon\}$	$2, 1$	$\{\varepsilon\}$
(1b)	$L, L \cup \{\varepsilon\}$	$1, 2$	$\emptyset$
(2a)	L, L^*, L^+	$3, 2, 3$	$\{a\}$ over $\{a, b\}$
(5)	$L, L^*, L^{\oplus} \cup \{\varepsilon\}$	$n, n-1, 2, 1$	$\{a^{n-2}\}$ over $\{a, b\}$
(8)	$L, L^*, L^{\oplus}, L^{\oplus *}, L^{\oplus +}$	$n, n-1, 3, 2, 3$	$\{a, b^{n-2}\}$

(1a) L is clopen; $\varepsilon \in L$ **(1b)** L is clopen; $\varepsilon \notin L$
If a factor-free language contains a non-empty string, then it is not closed. It follows that the only two clopen factor-free languages are $\emptyset$ and $\{\varepsilon\}$, which gives the results in the first two rows of Table 2.

(2a) L is open but not clopen; $\varepsilon \notin L$
Let L be a factor-free language over an alphabet Σ such that $\varepsilon \notin L$, $L = L^{\oplus}$ and $L \neq L^+$. By Proposition 2, we have $L^{\oplus} = L \cap \Sigma$. Hence, we must have $\emptyset \neq L \subseteq \Sigma$, so $\mathrm{sc}(L) = 3$. Moreover, every such language satisfies the conditions in case (2a). If $L = \Sigma$, then $L^* = \Sigma^*$ and $L^+ = \Sigma^+$, so $\mathrm{sc}(L^*) = 1$ and $\mathrm{sc}(L^+) = 2$. Otherwise, $\mathrm{sc}(L^*) = 2$ and $\mathrm{sc}(L^+) = 3$. The language $\{a\}$ over $\{a, b\}$ mets the upper bounds $(3, 2, 3)$ and $\{a\}$ as a unary language meets the upper bounds $(3, 1, 2)$. Row (2a) in Table 2 displays the binary case.

(5) L is neither open nor closed; $L^{\oplus}$ is clopen and $L^{+\oplus} = L^{\oplus}$
Let L be a factor-free language over an alphabet Σ satisfying the conditions in case (5). Then $L \neq \{\varepsilon\}$, and by Proposition 2, we have $L^{\oplus} = L \cap \Sigma$. Since $L^{\oplus}$ is closed, we must have $L^{\oplus} = \emptyset$, so $\mathrm{sc}(L^{\oplus}) = 1$ and $\mathrm{sc}(L^{\oplus} \cup \{\varepsilon\}) = 2$. Next, by Proposition 1, $\mathrm{sc}(L^*) \leq n - 1$ if $|\Sigma| \geq 2$, and $\mathrm{sc}(L^*) \leq n - 2$ if $|\Sigma| = 1$. The binary generator $\{a^{n-2}\}$ meets the upper bounds $(n, n-1, 2, 1)$. In the unary case, the upper bounds $(n, n-2, 2, 1)$ are met by the unary subword-free generator $\{a^{n-2}\}$. Row (5) in Table 2 displays the binary case.

(6) L is neither open nor closed; L^+ is open but $L^{\oplus}$ is not closed; $L^{\oplus +} \neq L^+$
Let L be a factor-free language satisfying (6). In particular, we have $L \neq \emptyset, L \neq \{\varepsilon\}$, and L^+ is open. Notice that $ua \in L$ implies $a \in L$ because otherwise we would have $u \in L^{+c}$ and $a \in L^{+c}$, so L^+ would not be open. It follows that L contains no string of length at least two. Hence $L \subseteq \Sigma$. However then $L = L^{\oplus}$, a contradiction with the assumption that L is not open. Therefore the Kuratowski algebra in case (6) cannot be generated by any factor-free language.

(8) L is neither open nor closed; $L^{\oplus}$ is not closed; L^+ is not open; $L^{+\oplus} = L^{\oplus +}$
Let L be a factor free language satisfying (8). By Proposition 1, we have $\mathrm{sc}(L^*) \leq n - 1$. Let $\Gamma = L \cap \Sigma$. Then $L^{\oplus} = \Gamma$ by Proposition 2. Since $L^{\oplus}$ is not closed,

we must have $\Gamma \neq \emptyset$. Therefore $\mathrm{sc}(L^{\oplus}) = \mathrm{sc}(\Gamma) = 3$, $\mathrm{sc}(L^{\oplus *}) = \mathrm{sc}(\Gamma^*) \leq 2$, and $\mathrm{sc}(L^{\oplus +}) = \mathrm{sc}(\Gamma^+) \leq 3$. Next, let $L = \{a, b^{n-2}\}$. Then $L^{\oplus} = \{a\}$, so L is not open. Since we have $aa \in L^+ \setminus L$ and $aa \in L^{\oplus +} \setminus L^{\oplus}$, the languages L and $L^{\oplus}$ are not closed. Since $b^{n-2} \in L^+ \setminus L^{+\oplus}$, the language L^+ is not open. By Proposition 2, $L^{\oplus +} = L^{+\oplus}$. Hence $\{a, b^{n-2}\}$ is a binary subword-free generator of case (8), and notice that it maximizes all the corresponding complexities. In the unary case, we must have $L = \{a^{n-2}\}$. Then $L^{\oplus} = \emptyset$ or $L^{\oplus} = L$, so L does not satisfy (8). Row (8) in Table 2 displays the binary case.

(9) L is neither open nor closed; $L^{\oplus}$ is not closed; L^+ is not open; $L^{+\oplus} \neq L^{\oplus +}$ If L is a factor-free language satisfying (9), then $L \neq \{\varepsilon\}$. However, then $L^{+\oplus} = L^{\oplus +}$ by Proposition 2, so L cannot generate case (9).

4 Suffix-Free Languages

Now we turn our attention to suffix-free languages. Since reversal commutes with complementation and star, whenever an algebra is generated by a prefix-free language, it is also generated by a suffix-free language. However, while the complexities of L^* and L^{*c*} in the prefix-free case are at most n and $2^{n-3} + 2$, respectively, for a suffix-free language, the complexity of L^* may be up to $2^{n-2} + 1$, and the complexity of L^{*c*} is not known. The exact complexity of this combined operation is not known even in the general case of regular languages [11].

Surprisingly, we need the language L^{*c*} only in case (9), and this is the only case which is left open in this paper. In every other case, we are able to compute the maximal complexities of all the generated languages. Next, again surprisingly, the complexities of $L^{\oplus}, L^{\oplus +}$, and $L^{\oplus *}$ are at most n, and a DFA for $L^{\oplus}$ can be obtained from a DFA for L just by omitting the non-final states. Finally, it is interesting that in most cases, all the complexities cannot be maximized by a single generator.

We start with a very useful Cmorik's lemma which helps us easily prove the suffix-freeness of our generators. Then we state and prove some observations concerning suffix-free languages; let us recall that a minimal DFA for a suffix-free language is non-returning.

Lemma 3 [5, Lemma 1]. Let A be a non-returning DFA that has a unique final state. If each state of A, except for the dead state, has at most one in-transition on every input symbol, then $L(A)$ is suffix-free.

Lemma 4. *Let $\varepsilon \notin L$ and $L \cap \Sigma = \emptyset$. Then $L^{\oplus} = \emptyset$ and $L^{+\oplus} = \emptyset$.*

Proof. If $L = \emptyset$, then $L^{\oplus} = \emptyset$. Otherwise let w be a non-empty string in L. Then w can be partitioned into one-symbol strings that are in L^c. Thus $w \notin L^{\oplus}$, and we have $L^{\oplus} = \emptyset$. If $L \cap \Sigma = \emptyset$, then also $L^+ \cap \Sigma = \emptyset$, and by the same argument $L^{+\oplus} = \emptyset$. □

Lemma 5. *Let $n \geq 3$ and L be a suffix-free language accepted by a minimal non-returning DFA $A = (\{s, 1, 2, \ldots, n-2, q_d\}, \Sigma, \cdot, s, F)$. Then*

(a) $\mathrm{sc}(L^{c+}) \le n$;
(b) $\mathrm{sc}(L^{\oplus}) \le |F| + 2$;
(c1) *L is open if and only if* $F = \{1, 2, \ldots, n-2\}$;
(c2) *if L is open, then* $\mathrm{sc}(L^{+}) \le \mathrm{sc}(L)$;
(d) $\mathrm{sc}(L^{\oplus+}) \le n$;
(e) *L^{+} is open if and only if L^{+} is weakly-prefix-closed;*
(f) $\mathrm{sc}(L^{+}) \le 2^{n-2} + 1$ *and* $\mathrm{sc}(L^{*}) \le 2^{n-2} + 1$.

Proof. (a) Let $w \in \Sigma^*$. If $w \in L^c$ then $w \in L^{c+}$. If $w \in L$ and some non-empty prefix u of w is in L^c, that is, $w = uv$ with $u \neq \varepsilon$ and $u \in L^c$, then $v \in L^c$ since L is suffix-free. Hence $w \in L^{c+}$. It follows that an n-state DFA for L^{c+} can be constructed from A as follows:

- interchange final and non-final states of A;
- in each final state p of the resulting DFA, except for the initial state, replace each out-transition (p, a, q) with the loop (p, a, p).

(b) Since $L^{\oplus} = L^{c+c}$, we get an n-state DFA for $L^{\oplus}$ by complementing the DFA obtained in case (a). It follows that all non-final states of A, except for the initial state s, are dead in the DFA for $L^{\oplus}$, so $\mathrm{sc}(L^{\oplus}) \le |F| + 2$.

(c1) The language L is open if and only if $L = L^{\oplus}$. By the construction in case (b), this holds if and only if $F = \{1, 2, \ldots, n-2\}$.

(c2) To get an NFA for L^{+}, we add the transitions $(q, a, s \cdot a)$ for each final state q and each input symbol a. If $s \cdot a = q_d$, then we can remove the transition (q, a, q_d) since it is not used in any accepting computation. Otherwise, we have $s \cdot a \in F$, and we must have $q \cdot a = q_d$ because otherwise L would not be suffix-free. Hence we can remove the transition (q, a, q_d) for the same reason as above. The resulting automaton is deterministic and has n states.

(d) We have $L^{\oplus} \subseteq L$, so $L^{\oplus}$ is a suffix-free language. Since $L^{\oplus}$ is open, we get $\mathrm{sc}(L^{\oplus+}) \le n$ by $(c2)$ and (b).

(e) Assume that L^{+} is open. Let $w \in L^{+}, w = uv$ and $u \neq \varepsilon$. If $w \in L$, then $v \notin L$ and also $v \notin L^{+}$, since is L is suffix-free. Thus $v \in L^{+c}$, and therefore $u \notin L^{+c}$ since L^{+} is open. Hence $u \in L^{+}$. If $w = w_1 w_2 \cdots w_k$ with $k \ge 2$ and $w_i \in L$, and u is a non-empty prefix of w, then $u = w_1 w_2 \cdots w_{i-1} x$ where x is a non-empty prefix of w_i. As shown above, we have $x \in L^{+}$. Therefore $u \in L^{+}$. It follows that L^{+} is weakly-prefix-closed.

Conversely, assume that L^{+} is weakly-prefix-closed. Suppose for a contradiction that there is a string w in L^{+} such that $w \notin L^{+\oplus}$. Then $w = w_1 w_2 \cdots w_k$ with $k \ge 2$ and $w_i \in L^{+c}$ and $w_i \neq \varepsilon$. Since L^{+} is weakly-prefix-closed, we must have $w_1 \in L^{+}$, a contradiction.

(f) To get an NFA for L^{+} from the DFA A, we first remove the dead state d, and then we add the transition $(q, a, s \cdot a)$ for each final state q and each input symbol a such that $s \cdot a \neq d$. The resulting NFA is non-returning, so its subset automaton is non-returning and it has at most $2^{n-2} + 1$ reachable states. To get a DFA for L^{*}, we only make the initial state of the subset automaton final. □

Now we inspect the individual cases of possible Kuratowski algebras generated, this time, by suffix-free languages. Our aim is to get the results shown in Table 3. Cases (1a) and (1b) are analogous to the previous section.

Table 3. Suffix-free generators of Kuratowski algebras maximizing complexities of corresponding generated languages. Cases (2b), (3a), (3b), (4), and (7) cannot be generated by any suffix-free language

Case	$E(L)$	Upper bounds on state complexities	Suffix-free generator
(1a)	$L, L \setminus \{\varepsilon\}$	$2, 1$	ε
(1b)	$L, L \cup \{\varepsilon\}$	$1, 2$	$\emptyset$
(2a)	L, L^*, L^+	n, n, n	Fig. 1
(5)	$L, L^*, L^\oplus \cup \{\varepsilon\}, L^\oplus$	$n, 2^{n-2}+1, 2, 1$	Fig. 2
(6)	$L, L^*, L^+, L^\oplus, L^{\oplus *}, L^{\oplus +}$	$n, 2^{n-3}+2, 2^{n-3}+2, n-1, n-1, n-1$	Fig. 3 (top) Fig. 3 (bottom)
(8)	$L, L^*, \ L^\oplus, L^{+\oplus} \cup \{\varepsilon\}, L^{+\oplus}$	$n, 2^{n-2}+1, n-1, n-1, n-1$	Fig. 4 (top) Fig. 4 (bottom)
(9)	$L, L^*, L^{+\oplus} \cup \{\varepsilon\}, L^{+\oplus}, L^\oplus, L^{\oplus *}, L^{\oplus +}$	$n, 2^{n-2}+1, 2^{3n \log n}, 2^{3n \log n}, n-1, n-1, n-1$	Fig. 5 (top) ? Fig. 5 (bottom)

(2a) L is open, L is not closed, $\varepsilon \notin L$
Since L is open, we have $\mathrm{sc}(L^+) \le n$ by Lemma 5 (c2). To get an n-state DFA for L^*, we only make the initial state s final in the DFA for L^+ obtained in Lemma 5 (c2). Let L be the ternary suffix-free language accepted by the DFA shown in Fig. 1. By Lemma 5 (c1), L is open. Since $aa \in L^+ \setminus L$, L is not closed. Thus L satisfies the conditions (2a). We have $\mathrm{sc}(L) = \mathrm{sc}(L^*) = \mathrm{sc}(L^+) = n$ since the final states in $\{1, 2, \ldots, n-2\}$ can be distinguished by strings in b^*, and in the case of L^*, the final states s and $n-2$ are distinguished by c. This gives the results in row (2a) of Table 3.

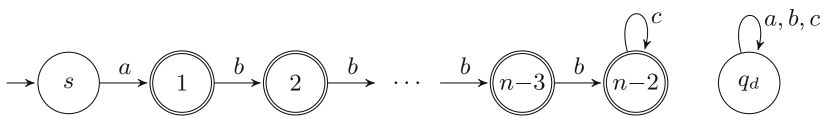

Fig. 1. A suffix-free generator of the Kuratowski algebra in case (2a); the transitions not shown are going to the dead state q_d.

(5) L is neither open nor closed; $L^\oplus$ is clopen and $L^{+\oplus} = L^\oplus$
Since L is neither open nor closed, we have $L \neq \emptyset$ and $L \neq \{\varepsilon\}$. Thus $\varepsilon \notin L$. Next $L^\oplus \subseteq L$, so $L^\oplus$ is suffix-free. Moreover $L^\oplus$ is assumed to be clopen, therefore $L^\oplus = \emptyset$ or $L^\oplus = \{\varepsilon\}$. Since $\varepsilon \notin L$, we must have $L^\oplus = \emptyset$. Hence $\mathrm{sc}(L^\oplus \cup \{\varepsilon\}) = 2$ and $\mathrm{sc}(L^\oplus \setminus \{\varepsilon\}) = 1$. Next we have $\mathrm{sc}(L^*) \le 2^{n-2}+1$ by Lemma 5 (f). Let

L be the language accepted by the DFA A shown in Fig. 2. By Lemma 3, L is suffix-free. We can show that L is the desired generator.

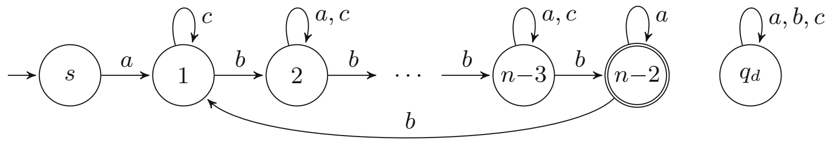

Fig. 2. A suffix-free generator of the Kuratowski algebra in case (5); the transitions not shown are going to the dead state q_d.

(6) L is not open, L is not closed, L^+ is open, $L^{\oplus}$ is not closed, $L^{\oplus +} \neq L^+$
Let L be accepted by a minimal DFA $A = (\{s, 1, \ldots, n-2, q_d\}, \Sigma, \cdot, s, F)$. First, we prove that $\text{sc}(L^*) \leq 2^{n-3} + 2$ in this case. If L satisfies the conditions in case (6), then L^+ is open. By Lemma 5 (e), L^+ is weakly-prefix-closed. Construct an NFA N for L^+ from A by adding the transitions $(q, a, s \cdot a)$ for each final state q and each input symbol a.

In the subset automaton of the NFA N, each reachable non-final subset, except for the initial subset, must be dead since L^+ is weakly-prefix-closed. We can show that no reachable subset contains two final states of A. Hence the subset automaton has at most $|F| \cdot 2^{n-|F|-2}$ reachable pairwise distinguishable states. This is at most $2^{n-3} + 2$, and to meet this bound, $|F|$ must be 1 or 2. To get a DFA for L^*, we make the initial state final in the subset automaton of the NFA N.

Now consider $L^{\oplus}$. By Lemma 5 (b), we have $\text{sc}(L^{\oplus}) \leq |F| + 2$. Thus $\text{sc}(L^{\oplus})$ is maximal if $F = \{1, 2, \ldots, n-2\}$. However, then L would be open by Lemma 5 (c1). Therefore we have $\text{sc}(L^{\oplus}) \leq n - 1$. Notice that if $n \geq 6$, then there is no language that maximizes both the complexities of L^+ and $L^{\oplus}$.

We can show that the suffix-free generator accepted by the DFA A shown in Fig. 3 (top) maximizes the complexities of L^+ and L^*, and the suffix-free generator accepted by the DFA B shown in Fig. 3 (bottom) maximizes the complexities of the remaining languages in $E(L)$.

(8) L is neither open nor closed; $L^{\oplus}$ is not closed; L^+ is not open; $L^{+\oplus} = L^{\oplus +}$
Let L be a suffix-free generator in case (8). We can show that the complexities of the generated languages are as in the corresponding row of Table 3. Similarly as in case (6) we can show that the upper bounds on the complexity of L^* and $L^{\oplus}$ cannot be met by a single generator. The suffix-free generator accepted by the DFA A shown in Fig. 4 (top) maximizes the complexity of L^*, and the suffix-free generator accepted by the DFA B shown in Fig. 4 (bottom) maximizes the complexities of the remaining languages in $E(L)$.

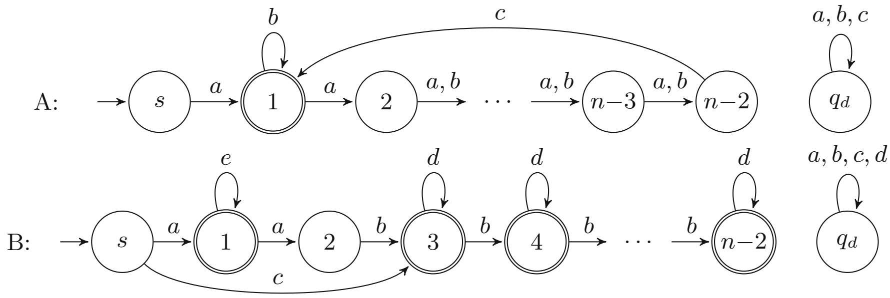

Fig. 3. Suffix-free generators of the Kuratowski algebra in case (6); the transitions not shown are going to the dead state q_d.

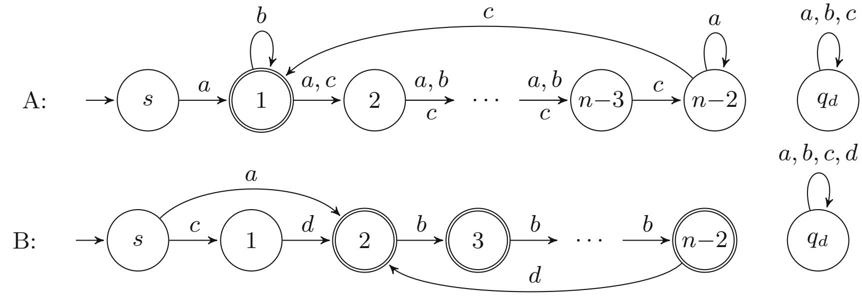

Fig. 4. Suffix-free generators of the Kuratowski algebra in case (8); the transitions not shown are going to the dead state q_d.

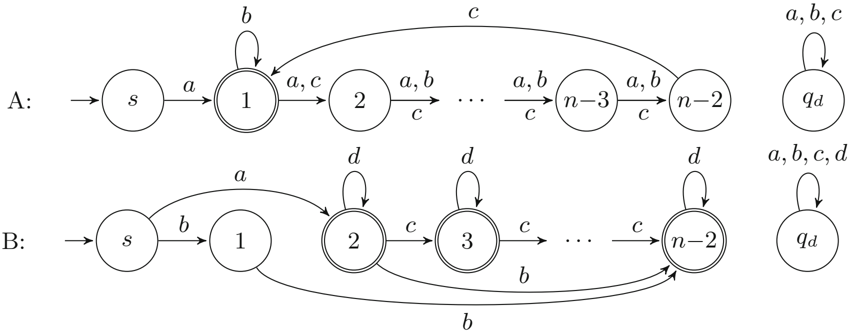

Fig. 5. Suffix-free generators of the Kuratowski algebra in case (9); the transitions not shown are going to the dead state q_d.

(9) L is neither open nor closed; $L^{\oplus}$ is not closed; L^{+} is not open; $L^{+\oplus} \neq L^{\oplus+}$ Since the complexity of L^{*c*} is not known for suffix-free languages, this part of case (9) remains open. The suffix-free generator accepted by the DFA A shown in Fig. 5 (top) maximizes the complexity of L^{*}, and the suffix-free generator accepted by the DFA B shown in Fig. 5 (bottom) maximizes the complexities of $L^{\oplus}, L^{\oplus *}, L^{\oplus +}$.

5 Conclusions

We investigated Kuratowski algebras generated by factor-, subword-, and suffix-free languages under the operations of star and complement. For each of these three classes and each of the 12 possible algebras we either showed that this algebra cannot be generated by a language in this class, or we gave a regular generator. For each of the possible algebras, we gave upper bounds on the state complexities of the generated languages. For factor- and subword- free languages, all the upper bounds can be met simultaneously by a single generator.

This also holds for cases (1a), (1b), (2a), and (5) for suffix-free languages. In cases (6) and (8), not all upper bounds can be met simultaneously. We gave examples of generators maximizing each of the upper bounds separately. In case (9), we were unable to find an automaton maximizing the complexity of $L^{+\oplus} \cup \{\varepsilon\}$ and $L^{+\oplus}$, here we only gave upper bounds.

References

1. Brzozowski, J.A.: Kuratowski algebras generated from L by applying the operators of Kleene closure and complement. Personal communication (2016)
2. Brzozowski, J.A., Grant, E., Shallit, J.: Closures in formal languages and Kuratowski's theorem. Internat. J. Found. Comput. Sci. **22**(2), 301–321 (2011)
3. Brzozowski, J., Jirásková, G., Liu, B., Rajasekaran, A., Szykuła, M.: On the state complexity of the shuffle of regular languages. In: Câmpeanu, C., Manea, F., Shallit, J. (eds.) DCFS 2016. LNCS, vol. 9777, pp. 73–86. Springer, Cham (2016). doi:10.1007/978-3-319-41114-9_6
4. Brzozowski, J.A., Jirásková, G., Zou, C.: Quotient complexity of closed languages. Theory Comput. Syst. **54**(2), 277–292 (2014)
5. Cmorik, R., Jirásková, G.: Basic operations on binary suffix-free languages. In: Kotásek, Z., Bouda, J., Černá, I., Sekanina, L., Vojnar, T., Antoš, D. (eds.) MEMICS 2011. LNCS, vol. 7119, pp. 94–102. Springer, Heidelberg (2012). doi:10.1007/978-3-642-25929-6_9
6. Fife, J.H.: The Kuratowski closure-complement problem. Math. Mag. **64**, 180–182 (1991)
7. Han, Y., Salomaa, K.: State complexity of basic operations on suffix-free regular languages. Theor. Comput. Sci. **410**(27–29), 2537–2548 (2009)
8. Han, Y., Salomaa, K., Wood, D.: Operational state complexity of prefix-free regular languages. In: Ésik, Z., Fülöp, Z. (eds.) Automata, Formal Languages, and Related Topics, pp. 99–115. Institute of Informatics, University of Szeged, Szeged (2009)
9. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory Languages and Computation. Addison-Wesley, Boston (1979)
10. Jirásek, J., Šebej, J.: Kuratowski algebras generated by prefix-free languages. In: Han, Y.-S., Salomaa, K. (eds.) CIAA 2016. LNCS, vol. 9705, pp. 150–162. Springer, Cham (2016). doi:10.1007/978-3-319-40946-7_13

11. Jirásková, G., Shallit, J.: The state complexity of star-complement-star. In: Yen, H.-C., Ibarra, O.H. (eds.) DLT 2012. LNCS, vol. 7410, pp. 380–391. Springer, Heidelberg (2012). doi:10.1007/978-3-642-31653-1_34
12. Kuratowski, K.: Sur l'operation $\bar{\mathrm{A}}$ de l'analysis situs. Fundam. Math. **3**, 182–199 (1922)
13. Sipser, M.: Introduction to the Theory of Computation. Cengage Learning, Boston (2012)
14. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A., et al. (eds.) Handbook of Formal Languages, Volume 1 Word, Language, Grammar, pp. 41–110. Springer, Heidelberg (1997)