The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-66252-3_7

Most Complex Non-Returning Regular Languages*

Janusz A. Brzozowski[1] and Sylvie Davies[2]

[1] David R. Cheriton School of Computer Science, University of Waterloo
Waterloo, ON, Canada N2L 3G1
brzozo@uwaterloo.ca
[2] Department of Pure Mathematics, University of Waterloo
Waterloo, ON, Canada N2L 3G1
sldavies@uwaterloo.ca

Abstract. A regular language L is non-returning if in the minimal deterministic finite automaton accepting it there are no transitions into the initial state. Eom, Han and Jirásková derived upper bounds on the state complexity of boolean operations and Kleene star, and proved that these bounds are tight using two different binary witnesses. They derived upper bounds for concatenation and reversal using three different ternary witnesses. These five witnesses use a total of six different transformations. We show that for each $n \geqslant 4$ there exists a ternary witness of state complexity n that meets the bound for reversal and that at least three letters are needed to meet this bound. Moreover, the restrictions of this witness to binary alphabets meet the bounds for product, star, and boolean operations. We also derive tight upper bounds on the state complexity of binary operations that take arguments with different alphabets. We prove that the maximal syntactic semigroup of a non-returning language has $(n-1)^n$ elements and requires at least $\binom{n}{2}$ generators. We find the maximal state complexities of atoms of non-returning languages. Finally, we show that there exists a most complex non-returning language that meets the bounds for all these complexity measures.

Keywords: atom, boolean operation, concatenation, different alphabets, most complex, non-returning, reversal, regular, star, state complexity, syntactic semigroup, transition semigroup, unrestricted complexity

1 Introduction

Formal definitions are postponed until Section 2; we assume the reader is familiar with basic properties of regular languages and finite automata as described in [11, 13], for example.

A deterministic finite automaton (DFA) is *non-returning* if there are no transitions into its initial state. A regular language is non-returning if its minimal DFA has that property. The *state complexity* of a regular language L, denoted by

* This work was supported by the Natural Sciences and Engineering Research Council of Canada grant No. OGP0000871.

$\kappa(L)$, is the number of states in the minimal DFA accepting L. The state complexity of an *operation* on regular languages is the maximal state complexity of the result of the operation, expressed as a function of the state complexities of the operands.

The state complexities of common operations (union, intersection, difference, symmetric difference, Kleene star, reverse and product/concatenation) were studied by Eom, Han and Jirásková [7]. They pointed out that several interesting subclasses of regular languages have the non-returning property; these subclasses include the class of suffix-free languages (suffix codes) and its subclasses (for example, bifix-free languages), and finite languages.

A regular language $L_n(a, b, c)$ of state complexity n is defined for all $n \geqslant 3$ in Figure 1. It was shown in [2] that the sequence $(L_3(a, b, c), \ldots, L_n(a, b, c), \ldots)$ of these languages meets the upper bounds (for regular languages) on the complexities of all the basic operations on regular languages as follows: If $L(b, a)$ is $L(a, b)$ with the roles of a and b interchanged, then $L_m(a, b) \circ L_n(b, a)$ meets the bound mn for all binary boolean operations $\circ$ that depend on both arguments; if $m \neq n$, $L_m(a, b) \circ L_n(a, b)$ meets the bound mn; $(L_n(a, b))^*$ meets the bound $2^{n-1} + 2^{n-2}$ for star; $(L_n(a, b, c))^R$ meets the bound 2^n for reversal; and $L_m(a, b, c)L_n(a, b, c)$ meets the bound $(m - 1)2^n + 2^{n-1}$ for product.

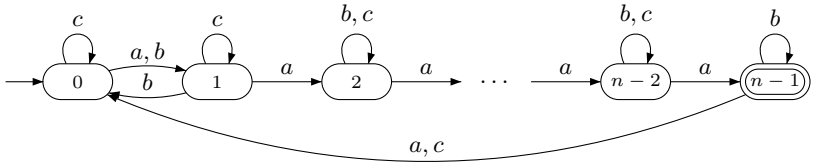

Fig. 1. Most complex regular language $L_n(a, b, c)$.

It was proposed in [2] that the size of the *syntactic semigroup* of a regular language is another worthwhile measure of the complexity of the language. The syntactic semigroup is isomorphic to the *transition semigroup* of the minimal DFA of L, that is, the semigroup of transformations of the state set of the DFA induced by non-empty words.

Another complexity measure suggested in [2] is the number and state complexities of the atoms of the language, where an atom is a certain kind of intersection of complemented and uncomplemented quotients of L.

It was shown in [2] that the languages $L_n(a, b, c)$ not only meet the bounds on the state complexities of operations, but also have the largest syntactic semigroups (of size n^n), and the largest number of atoms (2^n), all of which have the maximal possible state complexities. In this sense these are *most complex* regular languages.

In this paper we show that there also exist most complex non-returning languages. For each $n \geqslant 4$, we define a language of state complexity n. We prove that the syntactic semigroup of this language has $(n - 1)^n$ elements (the maxi-

mal possible for non-returning languages), that it is generated by $\binom{n}{2}$ elements, and that the number of generators cannot be reduced. We also show that this language has 2^n atoms, all of which have maximal state complexity. We demonstrate that the upper bound on the state complexity of reversal is met by a single ternary language, and that no binary language meets this bound. Moreover, restrictions of this language to binary alphabets meet the bounds for star, product and boolean operations. This is in contrast to [7] where several types of witnesses are used to meet the various bounds. We correct an error in [7, Table 1], where it is stated that the upper bound on the complexity of product cannot be reached with binary witnesses. Additionally, we consider both *restricted* and *unrestricted* state complexity [3] of binary operations on non-returning languages. When computing restricted state complexity, one assumes the operation takes in two languages over the same alphabet; for unrestricted state complexity we allow the inputs to be languages over different alphabets.

Omitted proofs can be found at `http://arxiv.org/abs/1701.03944`.

2 Preliminaries

A *deterministic finite automaton (DFA)* is a quintuple $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite non-empty set of *states*, Σ is a finite non-empty *alphabet*, $\delta\colon Q \times \Sigma \to Q$ is the *transition function*, $q_0 \in Q$ is the *initial* state, and $F \subseteq Q$ is the set of *final* states. We extend δ to a function $\delta\colon Q \times \Sigma^* \to Q$ as usual. A DFA $\mathcal{D}$ *accepts* a word $w \in \Sigma^*$ if $\delta(q_0, w) \in F$. The language accepted by $\mathcal{D}$ is denoted by $L(\mathcal{D})$. If q is a state of $\mathcal{D}$, then the language $L_q(\mathcal{D})$ of q is the language accepted by the DFA $(Q, \Sigma, \delta, q, F)$. A state is *empty* if its language is empty. Two states p and q of $\mathcal{D}$ are *equivalent* if $L_p(\mathcal{D}) = L_q(\mathcal{D})$. A state q is *reachable* if there exists $w \in \Sigma^*$ such that $\delta(q_0, w) = q$. A DFA is *minimal* if all of its states are reachable and no two states are equivalent.

We use $Q_n = \{0, \ldots, n-1\}$ as our basic set with n elements. A *transformation* of Q_n is a mapping $t\colon Q_n \to Q_n$. The *image* of $q \in Q_n$ under t is denoted by qt, and this notation is extended to subsets of Q_n: if $P \subseteq Q_n$, then $Pt = \{qt : q \in P\}$. The *rank* of a transformation t is the cardinality of Q_nt. If s and t are transformations of Q_n, their composition is denoted $(qs)t$ when applied to $q \in Q_n$. Let $\mathcal{T}_{Q_n}$ be the set of all n^n transformations of Q_n; then $\mathcal{T}_{Q_n}$ is a monoid under composition.

For $k \geqslant 2$, a transformation t of a set $P = \{q_0, q_1, \ldots, q_{k-1}\} \subseteq Q_n$ is a *k-cycle* if $q_0t = q_1, q_1t = q_2, \ldots, q_{k-2}t = q_{k-1}, q_{k-1}t = q_0$. This k-cycle is denoted by $(q_0, q_1, \ldots, q_{k-1})$, and leaves the states in $Q_n \setminus P$ unchanged. A 2-cycle (q_0, q_1) is called a *transposition*. A transformation that sends state p to q and acts as the identity on the remaining states is denoted by $(p \to q)$. If a transformation of Q_n has rank $n-1$, then there is exactly one pair of distinct elements $i, j \in Q_n$ such that $it = jt$. We say a transformation t of Q_n is of *type* $\{i, j\}$ if t has rank $n-1$ and $it = jt$ for $i < j$.

The *syntactic congruence* of a language $L \subseteq \Sigma^*$ is defined on Σ^+ as follows: For $x, y \in \Sigma^+$, $x \approx_L y$ if and only if $wxz \in L \Leftrightarrow wyz \in L$ for all $w, z \in \Sigma^*$. The

quotient set $\Sigma^+/\approx_L$ of equivalence classes of $\approx_L$ is a semigroup, the *syntactic semigroup* T_L of L.

Let $\mathcal{D} = (Q_n, \Sigma, \delta, 0, F)$ be a DFA. For each word $w \in \Sigma^*$, the transition function induces a transformation δ_w of Q_n by w: for all $q \in Q_n$, $q\delta_w = \delta(q, w)$. The set $T_{\mathcal{D}}$ of all such transformations by non-empty words is the *transition semigroup* of $\mathcal{D}$ under composition [12]. Often we use the word w to denote the transformation t it induces; thus we write qw instead of $q\delta_w$. We also write $w\colon t$ to mean that w induces the transformation t.

If $\mathcal{D}$ is a minimal DFA of L, then $T_{\mathcal{D}}$ is isomorphic to the syntactic semigroup T_L of L [12], and we represent elements of T_L by transformations in $T_{\mathcal{D}}$. The size of this semigroup has been used as a measure of complexity [2, 6, 8, 10].

The (left) *quotient* of $L \subseteq \Sigma^*$ by a word $w \in \Sigma^*$ is the language $w^{-1}L = \{x : wx \in L\}$. It is well known that the number of quotients of a regular language is finite and equal to the state complexity of the language.

Atoms are defined by a left congruence, where two words x and y are congruent whenever $ux \in L$ if and only if $uy \in L$ for all $u \in \Sigma^*$. Thus x and y are congruent whenever $x \in u^{-1}L$ if and only if $y \in u^{-1}L$ for all $u \in \Sigma^*$. An equivalence class of this relation is an *atom* of L [5]. Atoms can be expressed as non-empty intersections of complemented and uncomplemented quotients of L (see Section 5). The number of atoms and their state complexities were suggested as measures of complexity of regular languages [2] because all quotients of a language and all quotients of its atoms are unions of atoms [4, 5, 9].

Suppose $\circ$ is a unary operation on languages, and $f(n)$ is an upper bound on the state complexity of this operation. If the state complexity of $(L_n)^\circ$ is $f(n)$, then L_n is called a *witness* to the state complexity of $\circ$ for that n. In general, we need a sequence $(L_k, L_{k+1}, \ldots,)$ of such languages; this sequence is called a *stream*. Often a stream does not start at 1 because the bound may not hold for small values of n. For a binary operation we need two streams. The languages in a stream usually have the same form and differ only in the parameter n.

Sometimes the same stream can be used for both operands of a binary operation, but this is not always possible. For example, for boolean operations when $m = n$, the state complexity of $L_n \cup L_n = L_n$ is n, whereas the upper bound is $mn = n^2$. However, in many cases the second language is a "dialect" of the first, that is, it "differs only slightly" from the first. A *dialect* of $L_n(\Sigma)$ is a language obtained from $L_n(\Sigma)$ by deleting some letters of Σ in the words of $L_n(\Sigma)$ – by this we mean that words containing these letters are deleted – or replacing them by letters of another alphabet Σ'. Here we encounter only two types of dialects:

1. A dialect in which some letters were deleted; for example, $L_n(a, b)$ is a dialect of $L_n(a, b, c)$ with c deleted, and $L_n(a, -, c)$ is a dialect with b deleted.
2. A dialect in which the roles of two letters are exchanged; for example, $L_n(b, a)$ is such a dialect of $L_n(a, b)$.

These two types of dialects can be combined, for example, in $L_n(a, -, b)$ the letter c is deleted, and b plays the role that c played originally. The notion of dialects also extends to DFAs; for example, if $\mathcal{D}_n(a, b, c)$ recognizes $L_n(a, b, c)$ then $\mathcal{D}_n(a, -, b)$ recognizes the dialect $L_n(a, -, b)$.

3 Main Results

From now on by *complexity* we mean *state complexity*.

Let $\Gamma = \{a_{i,j} : 0 \leqslant i < j \leqslant n-1\}$, where $a_{i,j}$ is a letter that induces any transformation of type $\{i, j\}$ and does not map any state to 0. Let $\Gamma' = \Gamma \setminus \{a_{0,n-1}, a_{0,1}, a_{1,n-1}, a_{0,2}\}$. Let $\Sigma = \{a, b, c, d\} \cup \Gamma'$, where $a : (1, \ldots, n-1)(0 \to 1)$, $b : (1, 2)(0 \to 2)$, $c : (2, \ldots, n-1)(1 \to 2)(0 \to 1)$, and $d : (0 \to 2)$. Note that a, b, c and d are transformations of types $\{0, n-1\}$, $\{0, 1\}$, $\{1, n-1\}$ and $\{0, 2\}$, respectively. Note also that a, b and c restricted to $Q_n \setminus \{0\}$ generate all the transformations of $\{1, \ldots, n-1\}$. This follows from the well-known fact that the full transformation semigroup on a set X can be generated by the symmetric group on X together with a transformation of X with rank $|X| - 1$. For $X = \{1, \ldots, n-1\}$, we see that $\{(1, \ldots, n-1), (1, 2)\}$ (the restrictions of a and b) generate the symmetric group, and $(2, \ldots, n-1)(1 \to 2)$ (the restriction of c) is a transformation of rank $|X| - 1 = n - 2$.

We are now ready to define a most complex non-returning DFA and language.

Definition 1. *For $n \geqslant 4$, let $\mathcal{D}_n = \mathcal{D}_n(\Sigma) = (Q_n, \Sigma, \delta_n, 0, \{n-1\})$, where $\Sigma = \{a, b, c, d\} \cup \Gamma'$, and δ_n is defined in accordance with the transformations described above. See Figure 2 for $\mathcal{D}_n(\Sigma)$ restricted to $\{a, b, c, d\}$. Let $L_n = L_n(\Sigma)$ be the language accepted by $\mathcal{D}_n(\Sigma)$.*

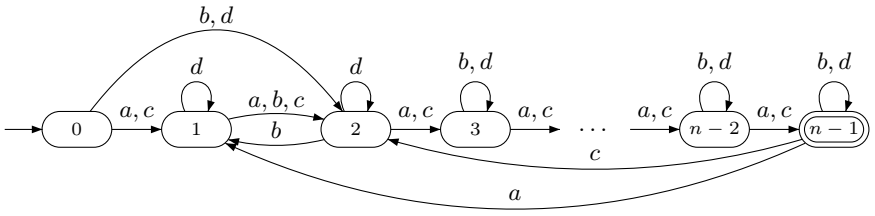

Fig. 2. Most complex non-returning language $L_n(\Sigma)$ of Definition 1. The letters in $\Gamma' = \Sigma \setminus \{a, b, c, d\}$ are omitted.

Theorem 1 (Most Complex Non-Returning Languages). *For each $n \geqslant 4$, the DFA of Definition 1 is minimal and non-returning. The stream $(L_n(\Sigma) : n \geqslant 4)$ with some dialect streams is most complex in the class of regular non-returning languages in the following sense:*

1. *The syntactic semigroup of $L_n(\Sigma)$ has cardinality $(n-1)^n$, and at least $\binom{n}{2}$ letters are required to reach this bound.*
2. *Each quotient of $L_n(a)$ has complexity $n-1$, except L itself, which has complexity n.*
3. *The reverse of $L_n(a, b, c)$ has complexity 2^n, and at least three letters are needed to meet this bound. Moreover, $L_n(a, b, c)$ has 2^n atoms.*

4. *For each atom A_S of $L_n(\Sigma)$, the complexity $\kappa(A_S)$ satisfies:*

$$\kappa(A_S) = \begin{cases} 2^{n-1}, & \text{if } S \in \{\emptyset, Q_n\}; \\ 2 + \sum_{x=1}^{|S|} \sum_{y=1}^{|S|} \binom{n-1}{x}\binom{n-1-x}{y}, & \text{if } \emptyset \subsetneq S \subsetneq Q_n. \end{cases}$$

   Moreover, at least $\binom{n}{2}$ letters are required to meet these bounds.
5. *The star of $L_n(a,b)$ has complexity 2^{n-1}.*
6. *Let $m, n \geqslant 4$.*
   (a) *Restricted product: $\kappa(L_m(a,b)L_n(a,-,b)) = (m-1)2^{n-1}+1$.*
   (b) *Unrestricted product: $\kappa(L_m(a,b)L_n(a,-,b,d)) = m2^{n-1}+1$.*
7. *Let $m, n \geqslant 4$.*
   (a) *Restricted boolean operations: $\kappa(L_m(a,b) \circ L_n(b,a)) = mn-(m+n-2)$.*
   (b) *Additionally, when $m \neq n$, we can use the same witness for both arguments: $\kappa(L_m(a,b) \circ L_n(a,b)) = mn-(m+n-2)$.*
   (c) *Unrestricted boolean operations: The complexity of $L_m(a,b,c) \circ L_n(b,a,d)$ is $mn+1$ if $\circ \in \{\cup, \oplus\}$, that of $L_m(a,b,c) \setminus L_n(b,a)$ is $mn-n+2$, and that of $L_m(a,b) \cap L_n(b,a)$ is $mn-(m+n-2)$.*

All of these bounds are maximal for non-returning languages.

Proof. From the definition of the letters of Σ it is obvious that the DFA $\mathcal{D}_n$ is non-returning, and that any pair (p,q) of states can be distinguished by the shortest word in a^* accepted by p but not by q.

1. This follows from Propositions 1 and 2 below. In particular, note that the syntactic semigroup of $L_n(\Sigma)$ contains the symmetric group on $Q_n \setminus \{0\}$, so the conditions of Proposition 2 are met.
2. Observe that for $i > 0$, the quotient $(a^i)^{-1}L_n(a)$ has complexity $n-1$; for $i = 0$, the quotient $(a^0)^{-1}L_n(a) = L_n(a)$ has complexity n.
3. By Proposition 3 the number of atoms of $L_n(a,b,c)$ is 2^n. By [5] the complexity of the reverse is the same as the number of atoms. By Proposition 4 at least three letters are required to meet this bound on the number of atoms and the complexity of reverse.
4. See Propositions 5, 6, and 7.
5. See Proposition 8.
6. See Propositions 9 and 10.
7. See Propositions 11 and 12.

We prove Propositions 1–4 and 11 below. □

4 Syntactic Semigroup

For all basic operations on non-returning languages, the complexity bounds can be met with either binary or ternary witnesses [7]. However, to meet the bound for the size of the syntactic semigroup, our most complex stream is forced to use an alphabet that grows quadratically in size.

For $n \geqslant 2$, let N_n denote the semigroup of transformations of Q_n such that $it \neq 0$ for all $i \in Q_n$. We call N_n the *full non-returning semigroup* on Q_n. We give a necessary condition and a sufficient condition for a set G to generate N_n.

Proposition 1. *If G is a generating set for N_n, then G contains a transformation of type $\{i,j\}$ for each $\{i,j\} \subseteq Q_n$. Thus a minimal generating set has exactly one element of type $\{i,j\}$ for each of the $\binom{n}{2}$ sets $\{i,j\} \subseteq Q_n$.*

Proof. Suppose t is a transformation of type $\{i,j\}$, and let t' be an arbitrary transformation. If tt' has rank $n-1$, then tt' has type $\{i,j\}$. Indeed, since $it = jt$, it follows that $itt' = jtt'$. Thus composing a transformation of type $\{i,j\}$ with an arbitrary transformation either preserves the type, or lowers the rank.

Suppose G generates N_n. Observe that N_n does not contain transformations of rank n (since these map some element to 0). Since composition with a transformation of type $\{i,j\}$ either preserves type or lowers rank, the semigroup generated by G contains only transformations that either have the same type as some element of G, or have rank less than $n-1$ and so are typeless. But N_n contains a transformation of type $\{i,j\}$ for each $\{i,j\} \subseteq Q_n$. So if G generates N_n, then G must contain an element of type $\{i,j\}$ for each $\{i,j\} \subseteq Q_n$. □

Proposition 2. *Let G be a subset of N_n that contains a transformation of type $\{i,j\}$ for each set $\{i,j\} \subseteq Q_n$, $i < j$. Let G' be obtained by restricting every transformation in G to $Q_n \setminus \{0\}$. If G' generates the symmetric group on $Q_n \setminus \{0\}$, then G generates N_n.*

Proof. First, we show that G' in fact generates the full transformation semigroup on $Q_n \setminus \{0\}$. Recall that the full transformation semigroup on X is generated by the symmetric group on X together with a transformation of X of rank $|X|-1$. By assumption, G' contains generators of the symmetric group on $Q_n \setminus \{0\}$. Transformations in G of type $\{i,j\}$ with $0 < i < j$ have rank $n-1$, and furthermore their restrictions to $Q_n \setminus \{0\}$ are of rank $n-2$.

Thus G' contains generators of the symmetric group on $Q_n \setminus \{0\}$, as well as a transformation of rank $|Q_n \setminus \{0\}| - 1 = n-2$; it follows that G' generates the full transformation semigroup on $Q_n \setminus \{0\}$.

Now, we prove that G generates every transformation in N_n. Let t be an element of N_n; we want to show that t is in the semigroup generated by G. Since N_n does not contain any transformations of rank n, the transformation t has rank less than n, and thus there exist distinct $i, j \in Q_n$ such that $it = jt$. Select a transformation s of type $\{i,j\}$ in G. Then for distinct $q, q' \in Q_n$, we have $qs = q's$ if and only if $\{q,q'\} = \{i,j\}$.

Hence there is a well-defined transformation r' of $Q_n \setminus \{0\}$ given by $(qs)r' = qt$ for all $q \in Q$; it is well-defined since if we have $qs = q's$, then $\{q,q'\} = \{i,j\}$ and is and js get mapped to a common element $it = jt$. The transformation r' lies in the full transformation semigroup on $Q_n \setminus \{0\}$, and so it is in the semigroup generated by G'. Hence there is some transformation r of Q_n in the semigroup generated by G such that r is equal to r' when restricted to $Q_n \setminus \{0\}$.

Since $qs \in Q_n \setminus \{0\}$ for all $q \in Q_n$, it follows that $(qs)r = (qs)r' = qt$ for all $q \in Q_n$, and thus sr and t are equal as transformations. Since s is in G and r is in the semigroup generated by G, it follows $sr = t$ is in the semigroup generated by G. Thus the semigroup generated by G contains all elements of N_n; but G is a subset of N_n, so G generates N_n. □

5 Number and Complexities of Atoms

Denote the complement of a language L by $\overline{L} = \Sigma^* \setminus L$. Let $Q_n = \{0, \ldots, n-1\}$ and let L_n be a non-empty regular language with quotients $K = \{K_0, \ldots, K_{n-1}\}$. Each subset S of Q_n defines an *atomic intersection* $A_S = \bigcap_{i \in S} K_i \cap \bigcap_{i \in \overline{S}} \overline{K_i}$, where $\overline{S} = Q_n \setminus S$. An *atom* of L is a non-empty atomic intersection; this definition is equivalent to that given in Section 2 in terms of a left congruence. Note that if $S \neq T$, then $A_S \cap A_T = \emptyset$; that is, atoms corresponding to distinct subsets of Q_n are disjoint. A language of complexity n can have at most 2^n atoms, since there are 2^n subsets of Q. We show that this bound can be met by non-returning languages. Additionally, we derive upper bounds on the complexities of atoms of non-returning languages, and show that our most complex stream meets these bounds.

We now describe a construction due to Iván [9]. Let L be a regular language with DFA $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$. For each $S \subseteq Q$, we define a DFA $\mathcal{D}_S = (Q_S, \Sigma, \Delta, (S, \overline{S}), F_S)$ as follows.

- $Q_S = \{(X, Y) : X, Y \subseteq Q, X \cap Y = \emptyset\} \cup \{\bot\}$. State $\bot$ is the *sink state*.
- $\Delta((X, Y), a) = (Xa, Ya)$ if $Xa \cap Ya = \emptyset$, and otherwise $\Delta((X, Y), a) = \bot$; also $\Delta(\bot, a) = \bot$.
- $F_S = \{(X, Y) : X \subseteq F, Y \subseteq \overline{F}\}$.

The DFA $\mathcal{D}_S$ recognizes the atomic intersection A_S of L; if it recognizes a non-empty language, then A_S is an atom. We can determine the complexity of A_S by counting reachable and distinguishable states in $\mathcal{D}_S$.

Proposition 3. *The language $L_n = L_n(a, b, c)$ has 2^n atoms.*

Proof. We want to show that A_S is an atom of L_n for all $S \subseteq Q_n$. It suffices to show for each S that the DFA $\mathcal{D}_S$ recognizes at least one word. Then since atoms corresponding to different subsets of Q_n are disjoint, this proves there are 2^n distinct atoms.

First, we show that from the initial state $(S, \overline{S})$, we can reach some state of the form (X, Y) where $0 \notin X$ and $0 \notin Y$. Consider the set $\{0, 1, n-1\}$. Notice that for each subset $\{i, j\}$ of $\{0, 1, n-1\}$, we have a transformation of type $\{i, j\}$: a has type $\{0, n-1\}$, b has type $\{0, 1\}$, and c has type $\{1, n-1\}$. Additionally, by the pigeonhole principle, either S contains two distinct elements from $\{0, 1, n-1\}$, or $\overline{S}$ contains two distinct elements from $\{0, 1, n-1\}$.

Suppose without loss of generality it is S which contains two distinct elements from $\{0, 1, n-1\}$. Let $\{i, j\} \subseteq S$ for some $\{i, j\} \subseteq \{0, 1, n-1\}$ with $i \neq j$. Let $\sigma \in \Sigma$ be the letter inducing the transformation of type $\{i, j\}$. Then we claim $(S, \overline{S})\sigma \neq \bot$. Indeed, suppose that $q \in S\sigma \cap \overline{S}\sigma$. Then since σ is a transformation of type $\{i, j\}$, we must have $i\sigma = j\sigma = q$, and no other element is mapped to q. But $\{i, j\} \subseteq S$, so we cannot have $q \in \overline{S}\sigma$.

Hence $S\sigma \cap \overline{S}\sigma = \emptyset$. Furthermore, since σ is a non-returning transformation, we have $0 \notin S\sigma$ and $0 \notin \overline{S}\sigma$. Thus starting from the initial state $(S, \overline{S})$, we can apply σ to reach a state of the form (X, Y) with $0 \notin X$ and $0 \notin Y$.

Now, recall that the three transformations $\{a, b, c\}$, when restricted to $Q_n \setminus \{0\}$, generate all transformations of $Q_n \setminus \{0\}$. Since $X \subseteq Q_n \setminus \{0\}$, there exists a transformation of $Q_n \setminus \{0\}$ that maps every element of X to $n-1$ and every element of $(Q_n \setminus \{0\}) \setminus X$ to 1. Let $w \in \{a, b, c\}^*$ be a word that induces this transformation when restricted to $Q_n \setminus \{0\}$. Since $Y \subseteq Q_n \setminus \{0\}$ and Y is disjoint from X, it follows that w maps every element of Y to 1. Since $F_n = \{n-1\}$ is the final state set of $\mathcal{D}_n$, we see that $Xw \subseteq F_n$ and $Yw \subseteq \overline{F_n}$. Thus $(Xw, Yw) = (\{n-1\}, \{1\})$ is a final state of $\mathcal{D}_S$.

This shows that there exists a word $\sigma w \in \{a, b, c\}^*$ that maps the initial state $(S, \overline{S})$ of $\mathcal{D}_S$ to a final state. Thus A_S is an atom. □

Next, we prove that the bound on number of atoms cannot be met by a binary witness. From [5] we know that the number of atoms of a regular language is equal to the state complexity of the reverse of the language. Hence this also proves a conjecture from [7], that a ternary witness is necessary to meet the bound for reversal of non-returning languages.

Proposition 4. *Let L be a non-returning language of complexity n over $\Sigma = \{a, b\}$. Then the number of atoms of L is strictly less than 2^n.*

Proof. Let $\mathcal{D}$ be the minimal DFA of L, with state set Q_n. We introduce some special terminology for this proof, which generalizes the notion of transformations of type $\{i, j\}$. We say that a transformation t *unifies* i and j, or unifies the set $\{i, j\}$, if $it = jt$. For example, transformations of type $\{i, j\}$ unify $\{i, j\}$. But furthermore, every transformation of Q_n of rank $n-1$ or less unifies at least one pair of elements of Q_n. The transition semigroup of $\mathcal{D}$ cannot have transformations of rank n, since L is non-returning; thus all the transformations in the transition semigroup must unify some pair of states.

Suppose that in $\mathcal{D}$, the letter a induces a transformation that unifies $\{i, j\}$, and b induces a transformation that unifies $\{k, \ell\}$. Assume also that $i \neq j$ and $k \neq \ell$. We will show that at least one atomic intersection A_S of L is empty, and thus is not an atom.

Suppose $\{i, j\} = \{k, \ell\}$. Let $S = \{i\}$ and consider the atomic intersection A_S. The initial state of the DFA for A_S is $(\{i\}, \overline{S})$. Note that $j \in \overline{S}$, so $ja \in \overline{S}a$. But a unifies i and j, so $ja = ia \in \{i\}a$. Thus since $\{i\}a \cap \overline{S}a \neq \emptyset$, the letter a sends the initial state $(\{i\}, \overline{S})$ to the sink state. Since b also unifies i and j, the letter b also sends $(\{i\}, \overline{S})$ to the sink state. Thus A_S is non-empty if and only if $(\{i\}, \overline{S})$ is a final state. In fact, either A_S is non-empty or $A_S = \{\varepsilon\}$, since every non-empty word sends the initial state $(\{i\}, \overline{S})$ to the sink state. If we let $T = \{j\}$, the same argument shows that A_T is either empty or $A_T = \{\varepsilon\}$. But $A_S \cap A_T = \emptyset$, so one of A_S or A_T must be empty.

Now, suppose $\{i, j\} \cap \{k, \ell\} = \emptyset$. Let $S = \{i, k\}$ and consider the atomic intersection A_S. The initial state of the DFA for A_S is $(\{i, k\}, \overline{S})$ with $j, \ell \in \overline{S}$. Thus as before, the transformation a which unifies $\{i, j\}$ and the transformation b which unifies $\{k, \ell\}$ both send A_S to the sink state. So either A_S is empty or $A_S = \{\varepsilon\}$. For $T = \{j, \ell\}$, the same argument shows that either A_T is empty or $A_T = \{\varepsilon\}$. Hence as before, one of A_S or A_T is empty.

Finally, suppose $\{i,j\} \cap \{k,\ell\}$ has exactly one element. Then either $k \in \{i,j\}$ or $\ell \in \{i,j\}$. Assume without loss of generality that $\ell \in \{i,j\}$ and $\ell = i$; otherwise rename the elements so this is the case. Then a unifies $\{i,j\}$, and b unifies $\{i,k\}$. Let $S = \{i\}$ and consider A_S. As before, the initial state of the DFA for A_S is sent to the sink state by both a and b. Thus either A_S is empty or $A_S = \{\varepsilon\}$. For $T = \{j,k\}$, the same argument shows that either A_T is empty or $A_T = \{\varepsilon\}$. Hence one of A_S or A_T is empty. □

Proposition 5. *Let L be a non-returning language of complexity n, and let Q_n be the state set of its minimal DFA. Let $S \subseteq Q_n$; then we have*

$$\kappa(A_S) \leqslant \begin{cases} 2^{n-1}, & \text{if } S \in \{\emptyset, Q_n\}; \\ 2 + \sum_{x=1}^{|S|} \sum_{y=1}^{|S|} \binom{n-1}{x}\binom{n-1-x}{y}, & \text{if } \emptyset \subsetneq S \subsetneq Q_n. \end{cases}$$

Proposition 6. *The atoms of the language $L_n = L_n(\Sigma)$ meet the complexity bounds of Proposition 5.*

Proposition 7. *Let L be a non-returning language over Σ of complexity n. If the atoms of L meet the bounds of Proposition 5, then Σ has size at least $\binom{n}{2}$.*

6 Other Operations

Proposition 8 (Star). *Let $\mathcal{D}_n(a,b)$ be the DFA of Definition 1 and let $L_n(a,b)$ be its language. Then the complexity of $(L_n(a,b))^*$ is 2^{n-1}.*

When dealing with binary operations, to avoid confusion between the sets of states $\{0,\ldots,m-1\}$ and $\{0,\ldots,n-1\}$ we use $\mathcal{D}'_m(\Sigma) = (Q'_m, \Sigma, \delta'_m, 0', \{(m-1)'\})$, and $\mathcal{D}_n(\Sigma) = (Q_n, \Sigma, \delta_n, 0, \{n-1\})$, where $Q'_m = \{0',\ldots,(m-1)'\}$. We write $L'_m(\Sigma)$ for the language of $\mathcal{D}'_m(\Sigma)$.

Proposition 9 (Restricted Product). *Let $\mathcal{D}_n(a,b,c)$ be the DFA of Definition 1 and let $L_n(a,b,c)$ be its language. Then for $m,n \geqslant 4$ the complexity of $L'_m(a,b)L_n(a,-,b)$ is $(m-1)2^{n-1}+1$.*

Proposition 10 (Unrestricted Product). *For $m,n \geqslant 4$, let L'_m (respectively, L_n) be a non-returning language of complexity m (respectively, n) over an alphabet Σ', (respectively, Σ). Then the complexity of product is at most $m2^{n+1}+1$, and this bound is met by $L'_m(a,b)$ and $L_n(a,-,b,d)$.*

A binary boolean operation is *proper* if it is not a constant function or a function of only one argument.

Proposition 11 (Restricted Boolean Operations). *Let $\mathcal{D}_n(a,b)$ be the DFA of Definition 1 and let $L_n(a,b)$ be its language. Then for $m,n \geqslant 4$ and for any proper binary boolean operation $\circ$ the complexity of $L'_m(a,b) \circ L_n(b,a)$ is $mn-(m+n-2)$. If $m \neq n$ then $\kappa(L'_m(a,b) \circ L_n(a,b)) = mn-(m+n-2)$.*

Proof. The upper bound was established in [7]. For the lower bound, Figure 3 restricted to the alphabet $\{a, b\}$ shows the two argument DFAs. As usual we construct their direct product. State $(0', 0)$ is initial and can never be reached again. If we apply a, we reach state $(1', 2)$, and the states reachable from this state form the direct product of DFA $\mathcal{E}'_{m-1}(a, b) = (\{1, \ldots, (m-1)'\}, \{a, b\}, \delta', 1', \{(m-1)'\})$ and DFA $\mathcal{E}_{n-1}(b, a) = (\{1, \ldots, n-1\}, \{a, b\}, \delta, 2, \{n-1\})$, where δ' and δ are $\delta_{m'}$ and δ_n restricted to $Q'_m \setminus \{0'\}$ and $Q_n \setminus \{0\}$. Since the transition semigroups of $\mathcal{E}'_m$ and $\mathcal{E}_n$ are the symmetric groups S_m and S_n, respectively, the result from [1, Theorem 1] applies, except in the cases where (m, n) is in $\{(4, 5), (5, 4), (5, 5)\}$, which have been verified by computation. Our first claim follows for the remaining cases by [1, Theorem 1]. If $m \neq n$, [1, Theorem 1] applies to $\mathcal{D}'_m(a, b)$ and $\mathcal{D}_n(a, b)$, and the second claim follows. In both cases the direct product of $\mathcal{E}'_m$ and $\mathcal{E}_n$ has $(m-1)(n-1)$ states; hence in the direct product of $\mathcal{D}'_m$ and $\mathcal{D}_n$ there are $(m-1)(n-1)+1 = mn-(m+n-2)$ states. By [1, Theorem 1] all these states are reachable and pairwise distinguishable for every proper binary boolean operation $\circ$.

Finally, note also that $(0', 0)$ is distinguishable from all other states. Since $(0', 0)a = (1', 2)$ and the preimage of $(1', 2)$ under a is $\{(0', 0), ((m-1)', 1)\}$, we see that $(0', 0)$ is distinguishable from $(p', q) \neq ((m-1)', 1)$ by first applying a, then applying a word that distinguishes $(0', 0)a = (1', 2)$ from $(p', q)a$. It is distinguishable from $((m-1)', 1)$ by first applying b, then applying a word that distinguishes $(0', 0)b = (2', 1)$ from $((m-1)', 1)b = ((m-1)', 2)$. □

Proposition 12. *For $m, n \geqslant 4$, let $L'_m(\Sigma')$ (respectively, $L_n(\Sigma)$) be a non-returning language of complexity m (respectively, n) over an alphabet Σ', (respectively, Σ). Then the complexity of union and symmetric difference is $mn+1$ and this bound is met by $L'_m(a, b, c)$ and $L_n(b, a, d)$; the complexity of difference is $mn-n+1$, and this bound is met by $L'_m(a, b, c)$ and $L_n(b, a)$; the complexity of intersection is $mn-(m+n-2)$ and this bound is met by $L'_m(a, b)$ and $L_n(b, a)$.*

7 Conclusions

We have shown that there exists a most complex non-returning language stream $(L_4(\Sigma), \ldots, L_n(\Sigma), \ldots)$. The cardinality of the syntactic semigroup of $L_n(\Sigma)$ is $(n-1)^n$ and its atoms have the highest state complexity possible for non-returning languages; both of these bounds can be reached only if Σ has at least $\binom{n}{2}$ letters. The bounds for the common restricted operations, however, can be met by streams over $\{a, b, c\}$ or $\{a, b\}$: $\kappa(L_m(a, b) \circ L_n(b, a) = mn-(m+n-2)$ for all proper boolean operations $\circ$; $\kappa(L_n(a, b))^* = 2^{n-1}$; $\kappa(L_n(a, b, c)^R) = 2^n$; and $\kappa(L'_m(a, b)L_n(a, -, b)) = (m-1)2^{n-1}+1$. The bounds for unrestricted boolean operations can be met by $L'_m(a, b, c)$ and $L_n(b, a, d)$, whereas those for the unrestricted product, by $L'_m(a, b)$ and $L_n(a, -, b, d)$.

Acknowledgments We are very grateful to Corwin Sinnamon and an anonymous reviewer for careful proofreading and constructive comments.

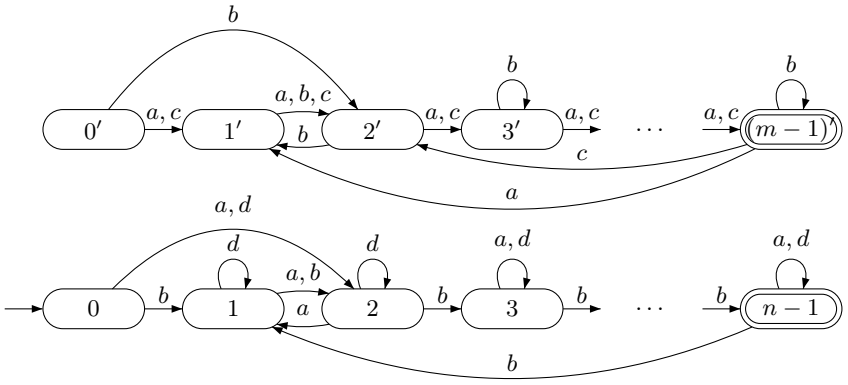

Fig. 3. DFAs $\mathcal{D}'_m(a,b,c)$ and $\mathcal{D}_n(b,a,d)$ for unrestricted boolean operations.

References

1. Bell, J., Brzozowski, J.A., Moreira, N., Reis, R.: Symmetric groups and quotient complexity of boolean operations. In: Esparza, J., et al. (eds.) ICALP 2014. LNCS, vol. 8573, pp. 1–12. Springer (2014)
2. Brzozowski, J.A.: In search of the most complex regular languages. Int. J. Found. Comput. Sc. 24(6), 691–708 (2013)
3. Brzozowski, J.A.: Unrestricted state complexity of binary operations on regular languages. In: Câmpeanu, C., et al. (eds.) DCFS 2016. LNCS, vol. 9777, pp. 60–72. Springer (2016), revised version in `http://arxiv.org/abs/1602.01387`
4. Brzozowski, J.A., Tamm, H.: Complexity of atoms of regular languages. Int. J. Found. Comput. Sc. 24(7), 1009–1027 (2013)
5. Brzozowski, J.A., Tamm, H.: Theory of átomata. Theoret. Comput. Sci. 539, 13–27 (2014)
6. Brzozowski, J.A., Ye, Y.: Syntactic complexity of ideal and closed languages. In: Mauri, G., Leporati, A. (eds.) DLT. LNCS, vol. 6795, pp. 117–128. Springer (2011)
7. Eom, H.S., Han, Y.S., Jirásková, G.: State complexity of basic operations on non-returning regular languages. Fund. Inform. 144, 161–182 (2016)
8. Holzer, M., König, B.: On deterministic finite automata and syntactic monoid size. Theoret. Comput. Sci. 327, 319–347 (2004)
9. Iván, S.: Complexity of atoms, combinatorially. Inform. Process. Lett. 116(5), 356–360 (2016)
10. Krawetz, B., Lawrence, J., Shallit, J.: State complexity and the monoid of transformations of a finite set. In: Domaratzki, M., et al. (eds.) CIAA. LNCS, vol. 3317, pp. 213–224. Springer (2005)
11. Perrin, D.: Finite automata. In: van Leewen, J. (ed.) Handbook of Theoretical Computer Science, vol. B, pp. 1–57. Elsevier (1990)
12. Pin, J.E.: Syntactic semigroups. In: Handbook of Formal Languages, vol. 1: Word, Language, Grammar, pp. 679–746. Springer, New York, NY, USA (1997)
13. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, pp. 41–110. Springer (1997)