State Complexity of Reversals of Deterministic Finite Automata with Output

Sylvie Davies(✉)

Department of Pure Mathematics, University of Waterloo, Waterloo, Canada
sldavies@uwaterloo.ca

Abstract. We investigate the worst-case state complexity of reversals of deterministic finite automata with output (DFAOs). In these automata, each state is assigned some output value, rather than simply being labelled final or non-final. This directly generalizes the well-studied problem of determining the worst-case state complexity of reversals of ordinary deterministic finite automata. If a DFAO has n states and k possible output values, there is a known upper bound of k^n for the state complexity of reversal. We show this bound can be reached with a ternary input alphabet. We conjecture it cannot be reached with a binary input alphabet except when $k = 2$, and give a lower bound for the case $3 \leq k < n$. We prove that the state complexity of reversal depends solely on the transition monoid of the DFAO and the mapping that assigns output values to states.

1 Introduction

The problem of determining the worst-case state complexity of the reversal operation on regular languages has been well-studied. Work on this problem dates back to the 1960 s; see Jirásková and Šebej [5] for a historical overview. It is known that if L is recognized by an n-state deterministic finite automaton (DFA), then the (deterministic) state complexity of the reverse L^R is at most 2^n, and this bound can be reached over a binary alphabet.

We study a generalization of this problem to *deterministic finite automata with output* (DFAOs). In a DFAO, each state is assigned an output from a finite *output alphabet* Δ. Rather than recognizing languages, DFAOs compute functions $f\colon \Sigma^* \to \Delta$, where Σ is the *input alphabet*. The value $f(w)$ is defined to be the output of the state reached by starting in the initial state and following the path spelling w. DFAOs directly generalize DFAs; the $|\Delta| = 2$ case is equivalent to assigning a value of "final" or "non-final" to each state. DFAOs are different from Moore machines [8], which build up an output word as each state is visited.

DFAOs are used in the study of *automatic sequences* [1]. If we treat the words $w \in \Sigma^*$ as representations of natural numbers in some base, we can view the function $f\colon \Sigma^* \to \Delta$ as a function $f\colon \mathbb{N} \to \Delta$, that is, an infinite sequence of

C. Câmpeanu (Ed.): CIAA 2018, LNCS 10977, pp. 133–145, 2018.
https://doi.org/10.1007/978-3-319-94812-6_12

elements of Δ. Sequences for which the corresponding function can be computed by a DFAO are called *automatic*.

The *reverse* of the function $f\colon \Sigma^* \to \Delta$ is the function $f^R\colon \Sigma^* \to \Delta$ defined by $f^R(w) = f(w^R)$. The reversal operation on DFAOs can thus be viewed as changing the direction in which the DFAO reads input: from left-to-right to right-to-left, or vice versa. We are concerned with the maximal blow-up in size (number of states) when the input reading direction of a DFAO is reversed. That is, given a function f computed by an n-state DFAO, what is the worst-case state complexity of f^R? The standard construction for reversal of DFAOs [1, Theorem 4.3.3] gives an upper bound of $|\Delta|^n$, where Δ is the output alphabet. However, it does not seem to be known whether this bound is reachable.

We prove that when the input alphabet has size three or greater, the upper bound $|\Delta|^n$ is indeed reachable. When the input alphabet is binary, the problem becomes much more complicated. We conjecture that if $|\Delta| \geq 3$, the upper bound $|\Delta|^n$ is not reachable over a binary alphabet, despite the fact that it is known to be reachable for $|\Delta| = 2$ (the ordinary DFA case). While we could not prove that the upper bound is unreachable in all cases, we have proved it is unreachable when $|\Delta| = n$ (that is, the cardinality of Δ equals the number of states n) and $|\Delta| \geq 3$, and verified computationally that it is unreachable for $(|\Delta|, n) \in \{(3,4),(3,5),(3,6),(4,5)\}$. We prove a lower bound for the case of a binary input alphabet and $3 \leq |\Delta| < n$.

We also demonstrate that the state complexity of DFAO reversal is completely determined by the transition monoid of the DFAO and the map which assigns outputs to states. In particular, if function f is computed by a minimal n-state DFAO with state set Q, transition monoid M, and output map $\tau\colon Q \to \Delta$, then the state complexity of f^R is exactly $|\tau M|$, where $\tau M = \{\tau \circ m : m \in M\}$ and $\circ$ denotes function composition. Since DFAs are special cases of DFAOs, this gives a new characterization of the state complexity of DFA reversal in terms of the transition monoid and the characteristic function of the final state set.

2 Preliminaries

We assume familiarity with basic concepts and results on regular languages and finite automata. There are many references on this subject, such as [4].

A *deterministic finite automaton with output* (DFAO) is a 6-tuple $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$, where:

- Q is a finite set of *states* and $q_0 \in Q$ is the *initial state*.
- Σ is the *input alphabet* and Δ is the *output alphabet*; both are finite.
- $\cdot\colon Q \times \Sigma \to Q$ is the *transition function*.
- $\tau\colon Q \to \Delta$ is the *output map*.

We use infix notation for the transition function: the image of the pair (q, a) under the transition function is denoted $q \cdot a$. We extend the transition function to words in Σ^* as follows: for $q \in Q$, we define $q \cdot \varepsilon = q$, and for $w = ax$, $a \in \Sigma$, $x \in \Sigma^*$ we inductively define $q \cdot ax = (q \cdot a) \cdot x$. If $p \cdot a = q$ for $p, q \in Q$ and

$a \in \Sigma$, we say there is a *transition* from p to q on a. If $p \cdot w = q$ for $w \in \Sigma^*$, we say there is a *path* from p to q spelling w.

The *function computed by a DFAO* is the function $f\colon \Sigma^* \to \Delta$ defined by $f(w) = \tau(q_0 \cdot w)$. That is, we determine $f(w)$ by starting in the initial state q_0, following the path corresponding to w to reach some state q, then applying the output map τ to get the output value associated with q. A function that can be computed by a DFAO is called a *finite-state function*.

A state $q \in Q$ is *reachable* if there is a path to it from the initial state q_0, i.e., there exists $w \in \Sigma^*$ such that $q_0 \cdot w = q$. The DFAO $\mathcal{D}$ is called *trim* if all states are reachable. Two states $p, q \in Q$ are *distinguishable* if there exists $w \in \Sigma^*$ such that $\tau(p \cdot w) \neq \tau(q \cdot w)$. A DFAO is *minimal* if it has the least possible number of states among all DFAOs computing the same function. The following result is well-known for DFAs, and it can be shown to hold for DFAOs using essentially the same proof.

Proposition 1. *A DFAO is minimal if and only if all states are reachable and every pair of distinct states is distinguishable.*

For further reference on the DFAO model, see [1].

Let Q be a finite set; we usually assume without loss of generality that $Q = \{1, 2, \ldots, n\}$. A *transformation* of Q is a function $t\colon Q \to Q$. The *image* of a transformation $t\colon Q \to Q$ is the set $t(Q) = \{t(q) : q \in Q\}$. The *rank* of a transformation is the size of its image. Transformations of Q (or more generally, functions $f\colon Q \to X$ for some set X) can be specified explicitly using *matrix notation*:

$$t = \begin{pmatrix} 1 & 2 & 3 & \cdots & n \\ t(1) & t(2) & t(3) & \cdots & t(n) \end{pmatrix}.$$

Transformations (or functions $f\colon Q \to X$) can be written concisely using *list notation*; for example, the list $[1, 4, 3, 5, 2, 2, 3]$ denotes $\begin{pmatrix} 1\,2\,3\,4\,5\,6\,7 \\ 1\,4\,3\,5\,2\,2\,3 \end{pmatrix}$.

A bijective transformation is called a *permutation*. Permutations can be written concisely using *disjoint cycle notation*; for example, $(1, 2, 4, 5)(6, 7)$ denotes the permutation $[2, 4, 3, 5, 1, 7, 6]$.

Transformations can be *composed* using the $\circ$ operator; the image of q under $s \circ t$ is $s(t(q))$. A set of transformations of Q that is closed under composition is called a *transformation monoid* on Q. The size of Q is called the *degree* of the transformation monoid. The *full transformation monoid* on Q is the set of all transformations of Q. The *symmetric group* on Q is the set of all permutations of Q. A transformation monoid M is *generated* by a set of transformations T if every transformation in M can be written as a composition of transformations from T. We say a monoid is *k-generated* if it is generated by a set of size k.

Each DFAO $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$ has a transformation monoid associated with it, called the *transition monoid* of the DFAO. It is defined as follows. For each $w \in \Sigma^*$, define the function $\overline{w}\colon Q \to Q$ by $\overline{w}(q) = q \cdot w$. The function $\overline{w}$ is called the *action* of w in $\mathcal{D}$. Composition of actions obeys the following rule:

$$\overline{x} \circ \overline{y} = \overline{yx}, \quad \text{since } \overline{x}(\overline{y}(q)) = q \cdot y \cdot x = q \cdot yx = \overline{yx}.$$

Since the set $\{\overline{w} : w \in \Sigma^*\}$ of all word actions in $\mathcal{D}$ is closed under composition, this set forms a transformation monoid on Q. This is the *transition monoid* of $\mathcal{D}$. The transition monoid is generated by the set $\{\overline{a} : a \in \Sigma\}$ of letter actions.

When working with multiple DFAOs, say $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$ and $\mathcal{D}' = (Q', \Sigma', \cdot', q_0', \Delta', \tau')$, the notation $\overline{w}$ is ambiguous: it is unclear whether this is the action of w in $\mathcal{D}$ or in $\mathcal{D}'$. We adopt the following convention: the notation $\overline{w}$ refers to the action of w in a DFA whose transition function is named "$\cdot$". Thus in this case, $\overline{w}$ would refer to the action of w in $\mathcal{D}$, rather than $\mathcal{D}'$. This convention will be sufficient to keep things unambiguous in this paper.

If $w = a_1 a_2 \cdots a_{n-1} a_n$ is a word over Σ^* with $a_1, \ldots, a_n \in \Sigma$, the *reverse* of w is $w^R = a_n a_{n-1} \cdots a_2 a_1$. Note that $\overline{a_1} \circ \overline{a_2} \circ \cdots \circ \overline{a_{n-1}} \circ \overline{a_n} = \overline{a_n a_{n-1} \cdots a_2 a_1} = \overline{w^R}$. On the other hand, $\overline{a_n} \circ \overline{a_{n-1}} \circ \cdots \circ \overline{a_2} \circ \overline{a_1} = \overline{a_1 a_2 \cdots a_{n-1} a_n} = \overline{w}$. The *reverse* of a finite-state function $f\colon \Sigma^* \to \Delta$ is the function $f^R\colon \Sigma^* \to \Delta$ defined by $f^R(w) = f(w^R)$. Following [1, Theorem 4.3.3], we give a DFAO construction for f^R in terms of a DFAO for f.

Proposition 2. *Let $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$ be a DFAO computing the function f. There exists a DFAO $\mathcal{D}^R$ computing f^R.*

Proof. Let $\mathcal{D}^R = (\Delta^Q, \Sigma, \odot, \tau, \Delta, \Omega)$, where:

- The state set is Δ^Q, the set of all functions from Q to Δ.
- The initial state is $\tau\colon Q \to \Delta$, the output map of $\mathcal{D}$.
- The transition function $\odot$ is defined as follows: $g \odot a = g \circ \overline{a}$, for $g \in \Delta^Q$ and $a \in \Sigma$.
- The output map $\Omega\colon \Delta^Q \to \Delta$ is defined by $\Omega(g) = g(q_0)$.

By definition, the function computed by $\mathcal{D}$ is $f(w) = \tau(q_0 \cdot w)$. The function computed by $\mathcal{D}^R$ is $\Omega(\tau \odot w) = (\tau \odot w)(q_0)$; we must show this equals $f^R(w) = f(w^R)$. If $w = a_1 a_2 \cdots a_n$, then we have

$$\tau \odot w = \tau \odot a_1 \odot a_2 \odot \cdots \odot a_n = \tau \circ \overline{a_1} \circ \overline{a_2} \circ \cdots \circ \overline{a_n} = \tau \circ \overline{w^R}.$$

It follows that

$$(\tau \circ \overline{w^R})(q_0) = \tau(\overline{w^R}(q_0)) = \tau(q_0 \cdot w^R) = f(w^R) = f^R(w)$$

as required. □

The *state complexity* of a finite-state function is the size of a minimal DFAO computing the function. If a function f is computed by an n-state minimal DFAO (i.e., the function has state complexity n), Proposition 2 shows that the state complexity of f^R is bounded above by $|\Delta|^n$, since the size of the state set Δ^Q of $\mathcal{D}^R$ is $|\Delta|^{|Q|} = |\Delta|^n$.

The following proposition makes it easier to compute the state complexity of f^R. The analogous result for DFAs is known (e.g., see [5, Proposition 3]).

Proposition 3. *If $\mathcal{D}$ is trim, then all states of $\mathcal{D}^R$ are pairwise distinguishable.*

Proof. Let g and h be distinct states of $\mathcal{D}^R$. There exists $q \in Q$ such that $g(q) \neq h(q)$. Since $\mathcal{D}$ is trim, q is reachable. Choose $w \in \Sigma^*$ such that $q_0 \cdot w^R = q$. Observe that $\Omega(g \odot w) = (g \circ \overline{w^R})(q_0) = g(q_0 \cdot w^R) = g(q)$, and similarly $\Omega(h \odot w) = h(q)$. Since $\Omega(g \odot w) \neq \Omega(h \odot w)$, g and h are distinguishable. □

If we take $\mathcal{D}^R$ and remove all unreachable states from it, we obtain a DFAO for f^R with all states reachable and every pair of distinct states distinguishable. By Proposition 1, this is a minimal DFAO for f^R. Hence given a function f computed by a trim DFAO $\mathcal{D}$, to determine the state complexity of f^R, we can simply count the number of reachable states in $\mathcal{D}^R$.

3 Main Results

We first prove that the state complexity of reversal of DFAOs is completely determined by the transition monoid and the output map.

Proposition 4. *Let $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$ be a trim DFAO computing function f. Let M be the transition monoid of $\mathcal{D}$. The state complexity of f^R is $|\tau M|$, where $\tau M = \{\tau \circ \overline{w} : w \in \Sigma^*\}$.*

Proof. The DFAO $\mathcal{D}^R = (\Delta^Q, \Sigma, \odot, \tau, \Delta, \Omega)$ computes f^R. By Proposition 3, all states of $\mathcal{D}^R$ are distinguishable, so the state complexity of f^R is the number of reachable states in $\mathcal{D}^R$.

Recall from the proof of Proposition 2 that $g \odot w = g \circ \overline{w^R}$ for $g\colon Q \to \Delta$ and $w \in \Sigma^*$. In particular, since τ is the initial state of $\mathcal{D}^R$, every reachable state of $\mathcal{D}^R$ has the form $\tau \odot w = \tau \circ \overline{w^R}$. Hence the set of reachable states of $\mathcal{D}^R$ is $\{\tau \circ \overline{w^R} : w \in \Sigma^*\}$. But this is the same set as $\tau M = \{\tau \circ \overline{w} : w \in \Sigma^*\}$. It follows that the number of reachable states in $\mathcal{D}^R$ is precisely $|\tau M|$. □

Recall that DFAs are essentially DFAOs with $|\Delta| = 2$, if we view the output map as telling us whether a state is final. Hence we have the following corollary:

Corollary 1. *Let $\mathcal{D} = (Q, \Sigma, \cdot, q_0, F)$ be a trim DFA recognizing language L. Let M be the transition monoid of $\mathcal{D}$. The state complexity of L^R is $|\chi_F M|$, where $\chi_F\colon Q \to \{0, 1\}$ is the characteristic function of F.*

Throughout the rest of this section, Q and Δ will be finite sets with $|Q| = n$ and $|\Delta| = k$, the monoid M will be a transformation monoid on Q, and $\tau\colon Q \to \Delta$ will be a surjective function. Note that the surjectivity of τ implies $|\Delta| \leq |Q|$. It is fine to make this assumption, since if $|\Delta| > |Q|$ there are more possible outputs than there are states, and so we can shrink Δ without loss of generality.

Theorem 1. *Let M be the full transformation monoid on Q. Then $|\tau M| = k^n$ for all surjective functions $\tau\colon Q \to \Delta$.*

Proof. It suffices to show that every function $h\colon Q \to \Delta$ lies in τM, i.e., every such function h can be written as $\tau \circ g$ for some $g\colon Q \to Q$.

For $q \in Q$, we define $g(q)$ as follows. Since τ is surjective, there exists $p_q \in Q$ such that $\tau(p_q) = h(q)$. Define $g(q) = p_q$. Then $(\tau \circ g)(q) = \tau(g(q)) = \tau(p_q) = h(q)$ for all $q \in Q$, so $\tau \circ g = h$ as required. □

Corollary 2. *Let f be a finite-state function computed by a minimal DFAO $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$ with $|\Delta| \leq |Q|$ (i.e., $k \leq n$). The state complexity of f^R is at most $|\Delta|^{|Q|} = k^n$, and this bound can be reached when $|\Sigma| \geq 3$.*

Proof. The upper bound on f^R follows from the construction for $\mathcal{D}^R$. For the lower bound, we use the well-known fact that the full transformation monoid on Q can be generated by three elements: two generators of the symmetric group on Q, and a transformation of rank $|Q| - 1$. If $Q = \{1, \ldots, n\}$, an explicit example of three generators is $f_1 = (1, 2, \ldots, n)$, $f_2 = (1, 2)$ and $f_3 = (1 \to 2)$, where $(1 \to 2)$ is the function that maps 1 to 2 and fixes all other elements. Choose $\{a, b, c\} \subseteq \Sigma$ and let $\mathcal{D}$ be a DFAO such that $\overline{a} = f_1$, $\overline{b} = f_2$ and $\overline{c} = f_3$. Then the transition monoid M of $\mathcal{D}$ is the full transformation monoid. Furthermore, $\mathcal{D}$ is trim (all states can be reached via $\overline{a}$). Hence Proposition 4 applies. If we take the output map τ to be surjective, by Theorem 1 we see that the state complexity of f^R is $|\tau M| = k^n$, as required. □

We now turn to the case where the input alphabet of the DFAO is binary, i.e., $|\Sigma| = 2$. This case is significantly harder than the $|\Sigma| \geq 3$ case. We assume $|\Delta| \geq 3$, since if $|\Delta| = 2$, this case is equivalent to studying reversal of DFAs with binary alphabets, and for DFAs the upper bound of 2^n is reachable [5].

Since the state complexity of DFAO reversal is completely determined by the transition monoid and output map, there are connections between the $|\Sigma| = 2$ case and the problem of finding the largest 2-generated transformation monoids of a particular degree. This problem has been studied by Holzer and König [3] and by Krawetz, Lawrence and Shallit [7].

Following Holzer and König, we define two families of monoids. First and most important are the $U_{\ell,m}$ monoids [3, Definition 5]. The monoid $U_{\ell,m}$ is a transformation monoid on $Q = \{1, \ldots, \ell + m\}$ defined as follows. Let $\alpha\colon Q \to Q$ be the permutation $(1, \ldots, \ell)(\ell + 1, \ldots, \ell + m)$. A function $\gamma\colon Q \to Q$ belongs to $U_{\ell,m}$ if and only if it satisfies one of the following conditions:

1. There exists $i \geq 0$ such that $\gamma = \alpha^i$, that is, $\gamma = \alpha \circ \alpha \circ \cdots \circ \alpha$ (where there are i occurrences of α).
2. $\gamma(\{1, \ldots, \ell\}) \cap \gamma(\{\ell + 1, \ldots, \ell + m\}) \neq \emptyset$, and there exists an element $i \in \{\ell + 1, \ldots, \ell + m\}$ such that i is not in the image of γ.

If $1 < \ell < m$ and $\gcd(\ell, m) = 1$, then $U_{\ell,m}$ can be generated by two elements [3, Theorem 8]. Krawetz [6] gives an explicit generating set: one of the generators is α, and the other is $\beta\colon Q \to Q$, where

$$\beta = \begin{pmatrix} 1 & 2\,3\,4 \cdots \ell + m - 1 & \ell + m \\ \ell + 1 & 2\,3\,4 \cdots \ell + m - 1 & 1 \end{pmatrix}$$

if $k = 2$ or ℓ is even, and otherwise

$$\beta = \begin{pmatrix} 1 & 2\,3\,4 \cdots \ell+m-1 & \ell+m \\ \ell+1 & 3\,2\,4 \cdots \ell+m-1 & 1 \end{pmatrix}.$$

Let $n = \ell + m$. For $n \geq 7$ and n prime, Holzer and König proved that there exist ℓ and m with $1 < \ell < m$ and $\gcd(\ell, m) = 1$ such that $U_{\ell,m}$ is the largest 2-generated transformation monoid [3, Theorem 15]. They conjecture that this also holds when $n \geq 7$ and n is not prime.

When $n \leq 6$, the largest 2-generated transformation monoids belong to a different family: the V_n^d monoids [3, Definition 16]. Let α be the permutation $(1, 2, \ldots, n)$. A function $\gamma\colon Q \to Q$ belongs to V_n^d if and only if it satisfies one of the following conditions:

1. There exists $i \geq 0$ such that $\gamma = \alpha^i$.
2. There exist $i, j \in \{1, \ldots, n\}$ such that $\gamma(i) = \gamma(j)$ and $j \equiv i + d \pmod{n}$.

For $2 \leq n \leq 6$, Holzer and König determined explicit generating sets for the largest 2-generated transformation monoids on $Q = \{1, \ldots, n\}$, which are all V_n^d monoids for some d. One of the generators is always $\alpha_n = (1, 2, \ldots, n)$. For $2 \leq n \leq 6$, the other generator β_n is:

$$\beta_2 = \begin{pmatrix} 1\,2 \\ 1\,1 \end{pmatrix}, \quad \beta_3 = \begin{pmatrix} 1\,2\,3 \\ 1\,1\,3 \end{pmatrix}, \quad \beta_4 = \begin{pmatrix} 1\,2\,3\,4 \\ 1\,1\,4\,3 \end{pmatrix},$$

$$\beta_5 = \begin{pmatrix} 1\,2\,3\,4\,5 \\ 1\,1\,4\,5\,3 \end{pmatrix}, \quad \beta_6 = \begin{pmatrix} 1\,2\,3\,4\,5\,6 \\ 1\,4\,1\,5\,6\,2 \end{pmatrix}.$$

Holzer and König also give a more general construction for 2-element generating sets of V_n^d monoids [3, Theorem 18].

With these definitions done, we return to the problem of computing worst-case state complexity of reversal for binary input alphabets. First we consider the special case $|Q| = |\Delta|$. Here it turns out that the state complexity problem almost completely reduces to the 2-generated monoid problem:

Theorem 2. *Let f be a finite-state function computed by a minimal DFAO $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$ with $|\Sigma| = 2$ and $|Q| = |\Delta| = n$. Let $m_2(n)$ denote the size of the largest 2-generated transformation monoid on $Q = \{1, 2, \ldots, n\}$ that occurs as the transition monoid of some trim DFA. The state complexity of f^R is at most $m_2(n)$, and this bound is reachable.*

Proof. Let $\Sigma = \{a, b\}$. By assumption, we can construct a trim DFAO $\mathcal{D}$ so that $\overline{a}$ and $\overline{b}$ generate a monoid of size $m_2(n)$. and let $\tau\colon Q \to \Delta$ be a bijection. By Proposition 4, the state complexity of f^R is $|\tau M|$. But τ is a bijection, so $|\tau M| = |M| = m_2(n)$. □

It may be the case that for some values of n, the largest transformation monoid on $\{1, 2, \ldots, n\}$ generated by two elements does *not* occur as the transition monoid of a trim DFA. Thus we do not quite get a complete reduction to the 2-generated monoid problem. Note that the $U_{\ell,m}$ and V_n^d monoids do occur as transition monoids of trim DFAs.

It is well known that if $|Q| \geq 3$, the full transformation monoid on a finite set Q cannot be generated by two elements. Hence $m_2(n)$ never reaches the upper bound of $|\Delta|^{|Q|} = n^n$ except when $|Q| = n = 2$.

Table 1 shows the known values for $m_2(n)$ for $2 \leq n \leq 7$, taken from [3, Table 1]. The value is not known for $n > 7$ except when n is prime, in which case $m_2(n)$ is the size of the largest 2-generated $U_{\ell,m}$ monoid. The values of n^n are also shown for comparison.

Table 1. Values of $m_2(n)$ for $2 \leq n \leq 7$.

n	2	3	4	5	6	7
$m_2(n)$	4	24	176	2110	32262	610871
n^n	4	27	256	3125	46656	823543

We now turn to the case where $|\Delta| < |Q|$. Our main result in this case is a formula for the size of $|\tau U_{\ell,m}|$, which in turn leads to a lower bound on the worst-case state complexity of f^R. The notation $\left\{ {\ell \atop i} \right\}$ below means the number of partitions of $\{1, \ldots, \ell\}$ into i parts (a Stirling number of the second kind).

Theorem 3. *Let $|\Delta| = k$ and let $|Q| = \ell + m = n$, with $2 \leq k < n$ and $1 \leq \ell \leq m$. Define*

$$F(k, \ell, m) = \sum_{i=1}^{\ell} \binom{k}{i} i! \left\{ {\ell \atop i} \right\} (k - i)^m.$$

$$G(k, \ell, m) = \begin{cases} \operatorname{lcm}(\ell, m), & \text{if } k \geq 4; \\ m, & \text{if } k = 3; \\ 1, & \text{if } k = 2. \end{cases}$$

There exists a function $\tau \colon Q \to \Delta$ such that

$$|\tau U_{\ell,m}| = k^n - F(k, \ell, m) + G(k, \ell, m).$$

To prove this theorem, we will need the following technical lemma. For space considerations, we omit the proof of the lemma; the proof can be found in the arXiv version of this paper [2].

Lemma 1. *Let $\Delta = \{1, \ldots, k\}$ and let $Q = \{1, \ldots, n\}$, with $2 \leq k < n$. Fix ℓ and m such that $\ell + m = n$ and $1 \leq \ell \leq m$. Let $\alpha \colon Q \to Q$ be the permutation $\alpha = (1, \ldots, \ell)(\ell + 1, \ldots, \ell + m)$. There exists a function $\tau \colon Q \to \Delta$ with the following properties:*

- $\tau\colon Q \to \Delta$ *is surjective.*
- $\tau(\{1,\ldots,\ell\}) \cap \tau(\{\ell+1,\ldots,\ell+m\}) = \emptyset$.
- *There exist distinct* $p, p' \in \{\ell+1,\ldots,\ell+m\}$ *such that* $\tau(p) = \tau(p')$.
- *The size of the set* $\{\tau \circ \alpha^i : i \geq 0\}$ *is precisely given by the function* $G(k,\ell,m)$.

Proof (Theorem 3). We start with a brief outline of the proof strategy. Without loss of generality, assume $\Delta = \{1,\ldots,k\}$ and $Q = \{1,\ldots,n = \ell+m\}$. Define $F_{\ell,m} = \{f\colon Q \to \Delta : f(\{1,\ldots,\ell\}) \cap f(\{\ell+1,\ldots,\ell+m\}) = \emptyset\}$.

- First, we show that $\Delta^Q = \tau U_{\ell,m} \cup F_{\ell,m}$ for certain τ.
- After proving this, the inclusion-exclusion principle gives the formula

$$k^n = |\Delta^Q| = |\tau U_{\ell,m}| + |F_{\ell,m}| - |\tau U_{\ell,m} \cap F_{\ell,m}|.$$

- We show that $|F_{\ell,m}| = F(k,\ell,m)$.
- We show that $|\tau U_{\ell,m} \cap F_{\ell,m}| = G(k,\ell,m)$.
- Rearranging the inclusion-exclusion formula above gives the result.

Let us show that for an appropriate choice of $\tau\colon Q \to \Delta$, we have $\Delta^Q = \tau U_{\ell,m} \cup F_{\ell,m}$. That is, every function from Q to Δ lies in one of $\tau U_{\ell,m}$ or $F_{\ell,m}$.

We select τ with the following properties:

- $\tau\colon Q \to \Delta$ is surjective.
- $\tau(\{1,\ldots,\ell\}) \cap \tau(\{\ell+1,\ldots,\ell+m\}) = \emptyset$, that is, $\tau \in F_{\ell,m}$.
- There exist distinct $p, p' \in \{\ell+1,\ldots,\ell+m\}$ such that $\tau(p) = \tau(p')$.
- The size of the set $\{\tau \circ \alpha^i : i \geq 0\}$ is precisely $G(k,\ell,m)$.

Such a function τ exists by Lemma 1. Note that we need $k < n$ and $\ell \leq m$ to apply Lemma 1; this is the only place we use these hypotheses.

Now, let $g\colon Q \to \Delta$ be arbitrary. We will show that if g is *not* in $F_{\ell,m}$, then it must be in $\tau U_{\ell,m}$, thus proving that $\Delta^Q = \tau U_{\ell,m} \cup F_{\ell,m}$. To show that $g \in \tau U_{\ell,m}$, we define a function $f\colon Q \to Q$ such that $f \in U_{\ell,m}$ and $\tau \circ f = g$.

Since $g \notin F_{\ell,m}$, there exist distinct elements $r \in \{1,\ldots,\ell\}$ and $r' \in \{\ell+1,\ldots,\ell+m\}$ such that $g(r) = g(r')$. Since τ is surjective, there exists s such that $\tau(s) = g(r)$. Furthermore, we can choose s so that $s \neq p'$. Indeed, if p' is one of the possible choices for s, then by the fact that $\tau(p) = \tau(p')$, we can choose $s = p$ instead. Now, we define $f\colon Q \to Q$ for each $q \in Q$ as follows:

- If $q \in \{r, r'\}$, define $f(q) = s$.
- If $g(q) = \tau(p)$ and $q \notin \{r, r'\}$, define $f(q) = p$.
- Otherwise, choose an element q' such that $\tau(q') = g(q)$ (by surjectivity) and define $f(q) = q'$.

We verify in each case that $\tau \circ f = g$:

- If $q = r$, then $f(r) = s$, so $\tau(f(r)) = \tau(s) = g(r)$.
- If $q = r'$, then $f(q) = s$, and since $g(r) = g(r')$ we have $\tau(f(r')) = \tau(s) = g(r) = g(r')$.

- If $q \notin \{r, r'\}$ and $g(q) = \tau(p)$, then $f(q) = p$, so $\tau(f(q)) = \tau(p) = g(q)$.
- Otherwise, we have $f(q) = q'$ such that $\tau(f(q)) = \tau(q') = g(q)$.

Now, we show that $f \in U_{\ell,m}$. First, note that there exist elements $r \in \{1, \ldots, \ell\}$ and $r' \in \{\ell+1, \ldots, \ell+m\}$ such that $f(r) = f(r')$. Next, observe that the element $p' \in \{\ell+1, \ldots, \ell+m\}$ is not in the image of f. To see this, note that if we have $f(q) = p'$, then we have $\tau(f(q)) = \tau(p') = \tau(p)$. But $\tau(f(q)) = g(q)$, so this implies $g(q) = \tau(p)$. In the case where $g(q) = \tau(p)$, we defined $f(q) = p \neq p'$, so this is a contradiction. It follows that f meets the conditions to belong to $U_{\ell,m}$.

This proves that if $g\colon Q \to \Delta$ is not in $F_{\ell,m}$, then $g \in \tau U_{\ell,m}$ and thus $\Delta^Q = \tau U_{\ell,m} \cup F_{\ell,m}$. Next, we show that $|F_{\ell,m}| = F(k, \ell, m)$.

Write $f \in F_{\ell,m}$ in list notation as $[a_1, a_2, \ldots, a_\ell, b_1, b_2, \ldots, b_m]$, where $f(i) = a_i$ and $f(\ell + i) = b_i$. For this function to lie in $F_{\ell,m}$, we must have the property that $\{a_1, a_2, \ldots, a_\ell\} \cap \{b_1, b_2, \ldots, b_m\} = \emptyset$. Note that since $F_{\ell,m}$ is a set of functions from Q to Δ, we have $\{a_1, \ldots, a_\ell\}, \{b_1, \ldots, b_m\} \subseteq \Delta$. We count the number of distinct "function lists" in $F_{\ell,m}$ as follows:

- Fix a set $S \subseteq \Delta$ and assume $\{a_1, \ldots, a_\ell\} = S$. Let $|S| = i$.
- In the first segment $[a_1, \ldots, a_\ell]$ of the list, each a_i can be an arbitrary element of S. However, since $\{a_1, \ldots, a_\ell\} = S$, each element of S must appear at least once in the list. Thus the first segment $[a_1, \ldots, a_\ell]$ of the list represents a *surjective* function from $\{1, \ldots, \ell\}$ onto S. Since $|S| = i$, the number of such surjective functions is $i!\left\{ {\ell \atop i} \right\}$. (It is known in general that the number of surjective functions from $\{1, \ldots, m\}$ to $\{1, \ldots, n\}$ is $n!\left\{ {m \atop n} \right\}$.)
- In the second segment $[b_1, \ldots, b_m]$ of the list, each b_i must be an element of $\Delta \setminus S$, since we want $\{a_1, \ldots, a_\ell\} \cap \{b_1, \ldots, b_m\} = \emptyset$. Since $|S| = i$ and $|\Delta| = k$, there are $k - i$ elements to pick from in $\Delta \setminus S$, and we need to choose m of them. Thus there are $(k-i)^m$ choices for the second segment of the list. In total, for a fixed set S of size i, there are $i!\left\{ {\ell \atop i} \right\}(k-i)^m$ distinct lists with $\{a_1, \ldots, a_k\} = S$.
- Now, we take the sum over all possible choices for the set S. Since $S = \{a_1, \ldots, a_\ell\}$ and S is non-empty, we have $1 \leq |S| \leq \ell$. For each set size i, there are $\binom{k}{i}$ ways to choose $S \subseteq \Delta$ with $|S| = i$. Thus the total number of functions in $F_{\ell,m}$ is

$$\sum_{i=1}^{\ell} \binom{k}{i} i! \left\{ {\ell \atop i} \right\} (k-i)^m = F(k, \ell, m).$$

Next, we show that $|\tau U_{\ell,m} \cap F_{\ell,m}| = G(k, \ell, m)$. We claim that

$$\tau U_{\ell,m} \cap F_{\ell,m} = \begin{cases} \emptyset, & \text{if } \tau \notin F_{\ell,m}; \\ \{\tau \circ \alpha^i : i \geq 0\}, & \text{if } \tau \in F_{\ell,m}. \end{cases}$$

Then the size equality with $G(k, \ell, m)$ follows from the properties of τ.

To see the claim, suppose that $\tau \circ g \in F_{\ell,m}$ for some $g \in U_{\ell,m}$. Since $g \in U_{\ell,m}$, either $g = \alpha^i$ for some i, or there exists $p \in \{1, \ldots, \ell\}$ and $q \in \{\ell+1, \ldots, \ell+m\}$

such that $g(p) = g(q)$. In the latter case, $\tau(g(p)) = \tau(g(q))$, which contradicts the assumption that $\tau \circ g$ is in $F_{\ell,m}$. Hence $g = \alpha^i$ for some $i \geq 0$, and so $\tau \circ g = \tau \circ \alpha^i$. Now, note that $\tau(\alpha^i(\{1, \ldots, \ell\})) = \tau(\{1, \ldots, \ell\})$, and $\tau(\alpha^i(\{\ell+1, \ldots, \ell+m\})) = \tau(\{\ell+1, \ldots, \ell+m\})$. Thus $\tau \circ \alpha^i$ is in $F_{\ell,m}$ if and only if τ is in $F_{\ell,m}$, and the claim follows.

Finally, we can conclude the proof. Recall that $|\Delta| = k$ and $|Q| = n$, and thus $|\Delta^Q| = |\Delta|^{|Q|} = k^n$. Thus by the inclusion-exclusion principle, we have

$$k^n = |\Delta^Q| = |\tau U_{\ell,m}| + |F_{\ell,m}| - |\tau U_{\ell,m} \cap F_{\ell,m}|.$$

Rearranging this, we get:

$$|\tau U_{\ell,m}| = k^n - |F_{\ell,m}| + |\tau U_{\ell,m} \cap F_{\ell,m}|.$$

We proved that $|F_{\ell,m}| = F(k, \ell, m)$ and $|\tau U_{\ell,m} \cap F_{\ell,m}| = G(k, \ell, m)$. It follows that $|\tau U_{\ell,m}| = k^n - F(k, \ell, m) + G(k, \ell, m)$, as required. □

This theorem gives the following lower bound on the worst-case state complexity of DFAO reversal when $|\Sigma| = 2$.

Corollary 3. *Let $|Q| = n \geq 2$ and $|\Delta| = k \geq 2$. There exists a trim DFAO $\mathcal{D} = (Q, \Sigma, \cdot, q_0, \Delta, \tau)$ computing function f, with $|\Sigma| = 2$ and $k < n$, such that the state complexity of f^R is*

$$\max\{k^n - F(k, \ell, m) + G(k, \ell, m) : 1 < \ell < m, \ell + m = n, \gcd(\ell, m) = 1\}.$$

Proof. Pick ℓ and m such that $1 < \ell < m$, $\ell + m = n$ and $\gcd(\ell, m) = 1$. Then $U_{\ell,m}$ can be generated by two elements. Hence we can construct a DFAO $\mathcal{D}$ over a binary alphabet with state set $Q = \{1, \ldots, n\}$ and transition monoid $U_{\ell,m}$. This DFAO will be trim: all states in $\{1, \ldots, \ell\}$ are reachable by $\alpha = (1, \ldots, \ell)(\ell+1, \ldots, \ell+m)$, and $U_{\ell,m}$ contains elements which map 1 to $\ell+1$, so the rest of the states are reachable. By Theorem 3, there exists $\tau \colon Q \to \Delta$ such that

$$|\tau U_{\ell,m}| = k^n - F(k, \ell, m) + G(k, \ell, m).$$

Take τ as the output map of $\mathcal{D}$. Then by Proposition 4, the state complexity of f^R is $|\tau U_{\ell,m}|$. Taking the maximum over all values of ℓ and m that satisfy the desired properties gives the result. □

Table 2 gives the values of this lower bound for various values of $|\Delta| = k$ and $|Q| = n$ with $k < n$. For $n \in \{1, 2, 3, 4, 6\}$ there are no pairs (ℓ, m) such that $1 < \ell < m$, $\ell + m = n$ and $\gcd(\ell, m) = 1$, so those values of n are ignored.

Note that for $|\Delta| = 2$, this lower bound is off by one from the upper bound of 2^n. The known examples where 2^n is achieved do not use $U_{\ell,m}$ monoids. We conjecture that for $|\Delta| \geq 3$, the upper bound $|\Delta|^n = k^n$ is not reachable. Jason Bell has recently claimed a proof of this conjecture (private communication).

We close this section by mentioning the results of some computational experiments. The goal of these experiments was to find, for small values of $|Q| = n$

Table 2. Values for the lower bound of Corollary 3.

$k\backslash n$	5	6	7	8	9
2	31	-	127	255	511
3	216	-	2125	6452	19550
4	826	-	15472	63403	258360
5	-	-	71037	368020	1902365
6	-	-	243438	1539561	9657446

and $|\Delta| = k$, the maximal size of $|\tau M|$, where M is a monoid generated by two functions $\alpha\colon Q \to Q$ and $\beta\colon Q \to Q$, and $\tau\colon Q \to \Delta$ is a surjective function. The results of our experiments are shown in Table 3. The values in **bold** are true maximal values for $|\tau M|$ (and thus for the state complexity of binary DFAO reversal), which have been confirmed by brute force search. The other, non-bold values in the table are simply the largest we found through random search.

Table 3. Largest known values for $|\tau M|$, where M is a 2-generated transformation monoid on $\{1, \ldots, n\}$ and $\tau\colon \{1, \ldots, n\} \to \{1, \ldots, k\}$ is surjective. **Bold** values have been confirmed to be maximal by brute force search.

$k\backslash n$	3	4	5	6	7	8
3	**24**	**67**	**218**	**699**	2125	6452
4	-	**176**	**826**	3526	15472	63403

Note that for $n \geq 7$, the conjectured maximal values in Table 3 match the values in Table 2 for lower bound of Corollary 3. For this reason, we suspect the bound of Corollary 3 may in fact be optimal for $n \geq 7$. However, the evidence at this point is limited.

4 Conclusions

For DFAs, the worst-case state complexity of the reversal operation is 2^n for languages of state complexity n. When we generalize to DFAOs, the worst-case state complexity is bounded above by k^n, where k is the number of outputs of the DFAO. We proved that this upper bound can be attained by DFAOs over a ternary input alphabet. For binary input alphabets, we demonstrated there are connections with the problem of finding the largest 2-generated transformation monoid, and gave a lower bound on the worst-case state complexity for the $k < n$ case. Computational experiments suggest that the upper bound k^n is not reachable using binary input alphabets if $k \geq 3$.

Acknowledgements. I thank Jason Bell, Janusz Brzozowski, Jeffrey Shallit, and the anonymous referees for proofreading and helpful comments. This work was supported by the Natural Sciences and Engineering Research Council of Canada under grant No. OGP0000871.

References

1. Allouche, J.P., Shallit, J.: Automatic Sequences: Theory, Applications, Generalizations. Cambridge University Press, Cambridge (2003)
2. Davies, S.: State complexity of reversals of deterministic finite automata with output. CoRR abs/1705.07150 (2017). http://arxiv.org/abs/1705.07150
3. Holzer, M., König, B.: On deterministic finite automata and syntactic monoid size. Theoret. Comput. Sci. **327**(3), 319–347 (2004)
4. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory Languages and Computation, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston (1979)
5. Jirásková, G., Šebej, J.: Reversal of binary regular languages. Theoret. Comput. Sci. **449**, 85–92 (2012)
6. Krawetz, B.: Monoids and the state complexity of root(L). Master's thesis (2003). https://cs.uwaterloo.ca/~shallit/krawetz.pdf
7. Krawetz, B., Lawrence, J., Shallit, J.: State complexity and the monoid of transformations of a finite set. Int. J. Found. Comput. Sci. **16**(03), 547–563 (2005)
8. Moore, E.F.: Gedanken experiments on sequential machines. In: Automata Studies, pp. 129–153. Princeton University Press (1956)