Minimal Reversible Deterministic Finite Automata

Markus Holzer, Sebastian Jakobi, and Martin Kutrib(✉)

Institut für Informatik, Universität Giessen,
Arndtstr. 2, 35392 Giessen, Germany
{holzer,sebastian.jakobi,kutrib}@informatik.uni-giessen.de

Abstract. We study reversible deterministic finite automata (REV-DFAs), that are partial deterministic finite automata whose transition function induces an injective mapping on the state set for every letter of the input alphabet. We give a structural characterization of regular languages that can be accepted by REV-DFAs. This characterization is based on the absence of a forbidden pattern in the (minimal) deterministic state graph. Again with a forbidden pattern approach, we also show that the minimality of REV-DFAs among all equivalent REV-DFAs can be decided. Both forbidden pattern characterizations give rise to NL-complete decision algorithms. In fact, our techniques allow us to construct the minimal REV-DFA for a given minimal DFA. These considerations lead to asymptotic upper and lower bounds on the conversion from DFAs to REV-DFAs. Thus, almost all problems that concern uniqueness and the size of minimal REV-DFAs are solved.

1 Introduction

Reversibility is a fundamental principle in physics. Since abstract computational models with discrete internal states may serve as prototypes of computing devices which can be physically constructed, it is interesting to know whether these abstract models are able to obey physical laws. The observation that loss of information results in heat dissipation [16] strongly suggests to study computations without loss of information. Many different formal models have been studied from this point of view. The reversibility of a computation means in essence that every configuration has a unique successor configuration and a unique predecessor configuration. For example, reversible Turing machines have been introduced in [4], where it turned out that every Turing machine can be simulated by a reversible one—for improved simulation constructions see [3,20]. Since Rice's theorem shows that any non-trivial property on languages accepted by (reversible) Turing machines is undecidable, it is reasonable from a practical perspective to study reversibility in devices of lower computational capacity. On the opposite end of the automata hierarchy, reversibility has been studied for finite automata [1,6,8,11,17,21], pushdown automata [13], queue automata [15], and even multi-head finite automata [2,14,18,19].

I. Potapov (Ed.): DLT 2015, LNCS 9168, pp. 276–287, 2015.
DOI: 10.1007/978-3-319-21500-6_22

Originally, reversible deterministic finite automata have been introduced and studied in the context of algorithmic learning theory in [1]; see also [11]. Later this concept was generalized in [21] and [17]. Almost all of these definitions agree on the fact that the transition function induces a partial injective mapping for every letter. Nevertheless, there are subtle differences. For instance, in [1] a partial deterministic finite automaton (DFA) M is defined to be reversible if M and the dual of M, that is the automaton that is obtained from M by reversing all transitions and interchanging initial and final states, are both deterministic. In particular, this definition implies that for reversible DFAs in the sense of [1] only one final state is allowed; hence these devices were called *bideterministic* in [21]. Since there are regular languages that are not accepted by any DFA with a sole accepting state, by definition, there are non-reversible regular languages in this setting. Then the definition of reversibility has been extended in [21]. Now multiple accepting as well as multiple initial states are allowed. So, reversible DFAs in the sense of [21] may have limited nondeterminism plugged in from the outside world at the outset of the computation. But still, these devices turn out to be less powerful than general (possibly irreversible) finite automata. An example is the regular language a^*b^* which is shown [21] to be not acceptable by any reversible DFA. In the same paper it is proved that for a given DFA the existence of an equivalent reversible finite automaton can be decided in polynomial time. A further generalization of reversibility to quasi-reversibility, which even allows nondeterministic transitions was introduced in [17]—see also [6]. Different aspects of reversibility for classical automata are discussed in [12]. In view of these results natural questions concern the uniqueness and the size of a minimal reversible DFA in terms of the size of the equivalent minimal DFA. For the latter question, in [8], a lower bound of $\Omega(1.001^n)$ states has been obtained which, in turn, raises the question for the construction of a minimal reversible DFA from a given (minimal) DFA. The construction problem has partially been solved in [6,17], where so-called quasi-reversible automata are constructed. However, these quasi-reversible DFAs may themselves be exponentially more succinct than the minimal reversible DFAs. In fact, the witness automata in [8] are already quasi-reversible.

This is the starting point of our investigation. For our definition of reversibility we stick to standard definitions. That is, partial DFAs with a unique initial state and potentially multiple accepting states. Then such an automaton is *reversible* if the transition function induces an injective mapping on the state set for every letter. These basic definitions are given in the next section together with an introductory example. For these reversible DFAs (REV-DFAs) we are able to solve the question on uniqueness and size of minimal representations almost completely. Section 3 is devoted to develop a method to decide the reversibility of a given regular language. While the notion of reversibility proposed in [21] is also decidable in polynomial time by an argument on the syntactic monoid of the language under consideration, here we obtain a structural characterization of regular languages that can be accepted by REV-DFAs in terms of their minimal DFAs. By this characterization an NL-complete decidability algorithm is shown, which is based on checking for the absence of forbidden patterns in the state graph. Then in Section 4 we turn to the minimality of REV-DFAs. First a structural characterization of minimal REV-DFAs is given. Again, this characterization allows to establish an NL-complete algorithm that decides whether a given DFA is already a minimal REV-DFA among all equivalent REV-DFAs. A further result is the construction of a minimal REV-DFA out of a given DFA that accepts a reversible language. Finally, this method is used to reconsider the example given in [8] and to improve the lower bound derived there to its maximum. Then we give a new family of binary witness languages that yield a better lower bound in order of Φ^n, where Φ is the golden ratio. This bound can be increased by larger alphabets, it has a limit of $\Omega(2^{n-1})$ as $|\Sigma|$ tends to infinity. Finally, our results allow to determine an upper bound of 2^{n-1} states for the conversion of DFAs to minimal REV-DFAs, even for arbitrary alphabet sizes.

2 Preliminaries

An *alphabet* Σ is a non-empty finite set, its elements are called *letters* or *symbols*. We write Σ^* for the *set of all words* over the finite alphabet Σ.

We recall some definitions on finite automata as contained, for example, in [7]. A *deterministic finite automaton* (DFA) is a system $M = \langle S, \Sigma, \delta, s_0, F\rangle$, where S is the finite set of *internal states*, Σ is the alphabet of *input symbols*, $s_0 \in S$ is the *initial state*, $F \subseteq S$ is the set of *accepting states*, and $\delta\colon S \times \Sigma \to S$ is the partial *transition function*. Note, that here the transition function is not required to be *total*. The *language accepted* by M is $L(M) = \{\, w \in \Sigma^* \mid \delta(s_0, w) \in F \,\}$, where the transition function is recursively extended to $\delta\colon S \times \Sigma^* \to S$. By $\delta^R\colon S \times \Sigma \to 2^S$, with $\delta^R(q, a) = \{\, p \in S \mid \delta(p, a) = q \,\}$, we denote the *reverse* transition function of δ. Similarly, also δ^R can be extended to words instead of symbols. Two devices M and M' are said to be *equivalent* if they accept the same language, that is, $L(M) = L(M')$. In this case we simply write $M \equiv M'$.

Let $M = \langle S, \Sigma, \delta, s_0, F\rangle$ be a DFA accepting the language L. The set of words $R_{M,q} = \{\, w \in \Sigma^* \mid \delta(q, w) \in F \,\}$ refers to the *right language* of the state q in M. In case $R_{M,p} = R_{M,q}$, for some states $p, q \in S$, we say that p and q are *equivalent* and write $p \equiv_M q$. The equivalence relation $\equiv_M$ partitions the state set S of M into equivalence classes, and we denote the equivalence class of $q \in S$ by $[q] = \{\, p \in S \mid p \equiv_M q \,\}$. Equivalence can also be defined between states of different automata: two states p and q of DFAs M and, respectively, M' are *equivalent*, denoted by $p \equiv q$, if $R_{M,p} = R_{M',q}$.

A state $p \in S$ is *accessible* in M if there is a word $w \in \Sigma^*$ such that $\delta(s_0, w) = p$, and it is *productive* if there is a word $w \in \Sigma^*$ such that $\delta(p, w) \in F$. If p is both accessible and productive then we say that p is *useful*. In this paper we only consider automata with all states useful. Let M and M' be two DFAs with $M \equiv M'$. Observe that if p is a useful state in M, then there exists a useful state p' in M', with $p \equiv p'$. A DFA is *minimal* (among all DFAs) if there does not exist an equivalent DFA with fewer states. It is well known that a DFA is minimal if and only if all its states are useful and no pair of states is equivalent.

Next we define reversible DFAs. Let $M = \langle S, \Sigma, \delta, s_0, F\rangle$ be a DFA. A state $r \in S$ is said to be *irreversible* if there are two distinct states p and q in S and a letter $a \in \Sigma$ such that $\delta(p, a) = r = \delta(q, a)$. Then a DFA is *reversible* if it does not contain any irreversible state. In this case the automaton is said to be a *reversible* DFA (REV-DFA). Equivalently the DFA M is reversible, if every letter $a \in \Sigma$ induces an *injective partial mapping* from S to itself *via* the mapping $\delta_a : S \to S$ with $p \mapsto \delta(p, a)$. In this case, the reverse transition function δ^R can then be seen as a (partial) injective function $\delta^R : S \times \Sigma \to S$. Notice that if p and q are two distinct states in a REV-DFA, then $\delta(p, w) \neq \delta(q, w)$, for all words $w \in \Sigma^*$. Finally, a REV-DFA is *minimal* (among all REV-DFAs) if there is no equivalent REV-DFA with a smaller number of states.

Example 1. Consider the finite language $L = \{aa, ab, ba\}$. The minimal DFA and a REV-DFA for this language are shown in Figure 1. Obviously, the minimal DFA is *not* reversible, since it contains the irreversible state 3. Moreover, it is also easy to see that the REV-DFA shown is minimal. Here minimality is meant with respect to all equivalent REV-DFAs. Note that redirecting the b-transition connecting state 1 and 3 in the REV-DFA to become a transition from state 1 to 4 results in a minimal REV-DFA as well. □

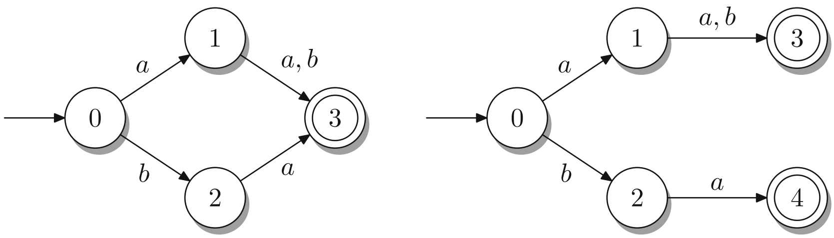

Fig. 1. The minimal DFA (left) and a minimal REV-DFA (right) for the finite language $L = \{aa, ab, ba\}$. Thus, L is a reversible language.

Finally we need some notations on computational complexity theory. We classify problems on REV-DFAs with respect to their computational complexity. Consider the complexity class NL which refers to the set of problems accepted by nondeterministic logspace bounded Turing machines.

To describe some of our algorithms we make use of nondeterministic space bounded oracle Turing machines, where the oracle tape is written deterministically. This oracle mechanism is known as RST-relativization in the literature [22]. If L is a set, we denote by $\mathsf{NL}^{\langle L\rangle}$ the class of languages accepted by nondeterministic logspace bounded RST oracle Turing machines with L oracle, and if C is a family of language, then $\mathsf{NL}^{\langle \mathsf{C}\rangle} = \bigcup_{L\in\mathsf{C}} \mathsf{NL}^{\langle L\rangle}$. Note that whenever C is a subset of NL, then $\mathsf{NL}^{\langle \mathsf{C}\rangle} \subseteq \mathsf{NL}$. This is due to the well-known fact that NL is closed under complementation [10,23], that is, NL = coNL, where coNL is the set of complements of languages from NL.

Further, hardness and completeness are always meant with respect to deterministic logspace bounded reducibility, unless otherwise stated.

3 Deciding the Reversibility of a Regular Language

We consider the problem to decide whether a given regular language is reversible, that is, it is accepted by a REV-DFA. Observe, that the minimal DFA for a language need not be reversible, although the language is accepted by a REV-DFA. This is seen by Example 1. Checking reversibility for the notion of [1] is trivial, because it boils down to verify the reversibility of the minimal DFA for the language, which must have a unique final state. Hence, the language from Example 1 is *not* reversible in the sense of [1]. On the other hand, the notion of reversibility proposed in [21] is also decidable, but by a more involved argument on the syntactic monoid of the language under consideration. We prove the following structural characterization of regular languages that can be accepted by REV-DFAs in terms of their minimal DFAs. The conditions of the characterization are illustrated in Figure 2.

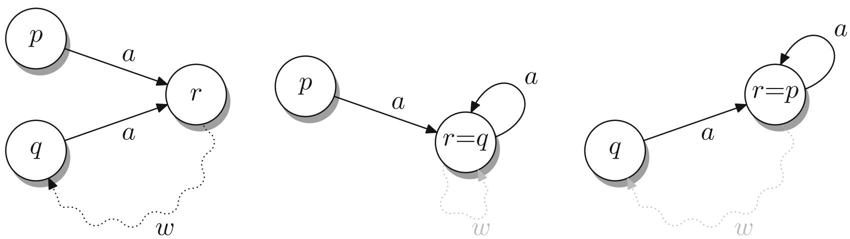

Fig. 2. The "forbidden pattern" of Theorem 2: the language accepted by a minimal DFA M can be accepted by a REV-DFA if and only if M does not contain the structure depicted on the left. Here the states p and q must be distinct, but state r could be equal to state p or state q. The situations where $r = q$ or $r = p$ are shown in the middle and on the right, respectively—here the word w and its corresponding path are grayed out because they are not relevant: in the middle, the word w that leads from r to q is not not relevant since it can be identified with the a-loop on state $r = q$. Also on the right hand side, word w is not important because we can simply interchange the roles of the states q and $r = p$.

Theorem 2. *Let* $M = \langle S, \Sigma, \delta, s_0, F\rangle$ *be a* minimal *deterministic finite automaton. The language* $L(M)$ *can be accepted by a reversible deterministic finite automaton if and only if there do not exist useful states* $p, q \in S$*, a letter* $a \in \Sigma$*, and a word* $w \in \Sigma^*$ *such that* $p \neq q$*,* $\delta(p, a) = \delta(q, a)$*, and* $\delta(q, aw) = q$*.*

By this characterization it is now easy to see that, e.g., both languages a^*ba^* and b^*ab^* are reversible, but their union is *not* reversible—obviously this union is a reversible language in the sense of [21]. Next we prove Theorem 2 by the upcoming two lemmata,

Lemma 3. *Let $M = \langle S, \Sigma, \delta, s_0, F\rangle$ be a deterministic finite automaton. If there exist useful states $p, q \in S$, a letter $a \in \Sigma$ and a word $w \in \Sigma^*$ such that $p \not\equiv q$, $\delta(p, a) \equiv \delta(q, a)$, and $\delta(q, aw) \equiv q$, then the language $L(M)$ cannot be accepted by a reversible deterministic finite automaton.*

Proof. Assume $M' = \langle S', \Sigma, \delta', s'_0, F'\rangle$ is a REV-DFA with $L(M') = L(M)$, then of course $s'_0 \equiv s_0$. Since the states p and q are useful, there must also be states $p', q' \in S'$ with $p' \equiv p$ and $q' \equiv q$. Thus, the relations $p' \not\equiv q'$, $\delta'(p', a) \equiv \delta'(q', a)$, and $\delta'(q', aw) \equiv q'$ must also hold in the REV-DFA M'. Let us now consider the sequence of states $\delta'(p', (aw)^i)$, for $i \geq 0$. From the equivalences $\delta'(p', a) \equiv \delta'(q', a)$ and $\delta'(q', aw) \equiv q'$, we conclude $\delta'(p', (aw)^i) \equiv q'$, for all $i \geq 1$. Thus, except for the first state p', all states of the above sequence are equivalent to q'. Notice that state p' cannot be equivalent to the other states of the sequence since $p' \not\equiv q'$. Since the number of states of M' must be finite, there must be a loop in the considered state sequence. This means that there must be integers $k \geq 0$ and $\ell \geq 1$ such that $\delta(p', (aw)^k) = \delta(p', (aw)^{k+\ell})$, and such that all states in the sequence $\delta(p', (aw)^0)$, $\delta(p', (aw)^1)$, $\ldots$, $\delta(p', (aw)^{k+\ell-1})$ are pairwise distinct. In fact we know that $k \geq 1$ because $\delta'(p', (aw)^{k+\ell}) \equiv q'$ cannot even be equivalent to state $\delta'(p', (aw)^0) = p'$. But now we have found two distinct states $\delta'(p', (aw)^{k-1})$ and $\delta'(p', (aw)^{k+\ell-1})$ that both map to the same state $\delta'(p', (aw)^k)$ on reading the input aw. This is a contradiction to M' being reversible, hence $L(M)$ cannot be accepted by any REV-DFA. ☐

When considering only minimal DFAs, the equivalences between states in Lemma 3 become equalities, so we obtain one implication of Theorem 2. Now let us prove that also the reverse implication is true. The idea how to make a given DFA reversible is very intuitive: as long as there is an irreversible state, copy this state and all states reachable from it, and distribute the incoming transitions to the new copies. The absence of the "forbidden pattern" ensures that this procedure eventually comes to an end.

For the proof of our next result we use the following notion. The state set S of a DFA $M = \langle S, \Sigma, \delta, s_0, F\rangle$ can be partitioned into *strongly connected components*: such a component is an inclusion maximal subset $C \subseteq S$ such that for all pairs of states $(p, q) \in C \times C$ there is a word $w \in \Sigma^*$ leading from p to q. Notice that also a single state q may constitute a strongly connected component, even if there is no looping transition on q.

Lemma 4. *Let $M = \langle S, \Sigma, \delta, s_0, F\rangle$ be a* minimal *deterministic finite automaton. If there do not exist useful states $p, q \in S$, a letter $a \in \Sigma$ and a word $w \in \Sigma^*$ such that $p \neq q$, $\delta(p, a) = \delta(q, a)$, and $\delta(q, aw) = q$, then the language $L(M)$ can be accepted by a reversible deterministic finite automaton.*

Proof. We show how to convert the DFA M into an equivalent REV-DFA. First we build a topological order $\preceq$ of the strongly connected components of M, such that if $C_1 \preceq C_2$, for two such components C_1 and C_2, then no state in C_1 can be reached from a state in C_2. Consider a minimal (with respect to $\preceq$) strongly connected component C_k that contains an irreversible state—if no such component exists then the automaton is reversible. To determine the number of necessary copies of C_k, compute

$$\alpha = \max\{\, \left|\delta^R(r,a)\right| \mid r \in C_k, a \in \Sigma \,\}. \tag{1}$$

Now we replace the component C_k by α copies of C_k and redistribute all incoming transitions among these copies, such that no state in the copies of C_k is the target of two or more transitions on the same letter. Notice that all transitions that witness the irreversibility of states in C_k come from outside of C_k, because if there were states $p, q, r \in S$ and a letter $a \in \Sigma$ with $\delta(p,a) = \delta(q,a) = r$ and $q, r \in C_k$ then M would have the "forbidden pattern" since $\delta(q, aw) = q$ for some $w \in \Sigma^*$. Therefore, the copies of C_k do not contain irreversible states.

Since also the transitions from states in the component C_k to states outside of C_k are copied, of course previously reversible states directly "behind" the copies of C_k could now become irreversible. However, in this way we only introduce irreversible states in components that are of higher rank in the topological order $\preceq$. Moreover, the obtained automaton is still equivalent to the original one. Therefore the described procedure can be applied iteratively, each time enlarging the minimal $\preceq$-rank of components that contain irreversible states, which eventually leads to a reversible DFA for $L(M)$. □

For an example explaining the previous construction in further detail we refer to the upcoming Example 9—there all strongly connected components are singleton sets, but it is easy to see how the construction works for larger size components as well. Now we have proven Theorem 2. In fact, we will later see that the automaton constructed in the proof of Lemma 4 even is a *minimal* REV-DFA.

It can be shown that the regular language reversibility problem—given a DFA M, decide whether $L(M)$ is accepted by any REV-DFA—is NL-complete. The idea of the proof is to decide in NL whether a given DFA $M = \langle S, \Sigma, \delta, s_0, F\rangle$ accepts a *non-reversible* language with the help of Theorem 2, by witnessing the forbidden pattern depicted in Figure 2. Since NL is closed under complementation the containment of the reversibility problem within NL follows.

Theorem 5. *The regular language reversibility problem is* NL*-complete.* □

4 Minimal Reversible Deterministic Finite Automata

We recall that it is well know that the minimal DFA accepting a given regular language is unique up to isomorphism. So there is the natural question asking for the relations between minimality and reversibility. It turned out that in this connection the different notions of reversibility do matter. For instance, Example 1 already shows that minimal REV-DFAs are not unique (even not up to isomorphism) in general. In [21] it is mentioned that a language L is accepted by a bideterministic finite automaton if and only if the minimal finite automaton of L is reversible and has a unique final state. This answers the question about the notion of reversibility in [1]. However, for the other notions of reversibility considered, the *minimal reversible* finite automaton for some language can be exponentially larger than the minimal automaton. In [8] finite witness languages are given that require $6n + 1$ states for a minimal DFA, but $\Omega(1.001^n)$ states for a minimal *reversible* DFA. Before we turn to determine the exact number of states for this example as well as an improved lower bound, first we derive a structural characterization of minimal REV-DFAs.

Theorem 6. *Let $M = \langle S, \Sigma, \delta, s_0, F\rangle$ be a reversible deterministic finite automaton with all states useful. Then M is minimal if and only if for every equivalence class $[q_1] = \{q_1, q_2, \ldots, q_n\}$ in S, with $n > 1$, there exists a word $w \in \Sigma^+$ such that $\delta^R(q_i, w)$ is defined for $1 \leq i \leq n$, and $\delta^R(q_k, w) \not\equiv \delta^R(q_\ell, w)$, for some k and ℓ with $1 \leq k, \ell \leq n$.* □

With the characterization of minimal REV-DFAs as stated in the previous theorem we are ready to prove that deciding minimality for these devices is NL-complete, and thus computationally not too complicated.

Theorem 7. *Deciding whether a given deterministic finite automaton M is already a minimal reversible deterministic finite automaton is* NL*-complete.*

Proof. Due to limited space, we only prove containment in NL, which seems the more interesting here than NL-hardness. Let the DFA $M = \langle S, \Sigma, \delta, s_0, F\rangle$ be given. We will prove minimality with respect to all REV-DFAs using Theorem 6. Our algorithm uses the following oracle subroutines:

(i) Is state p from the DFA M useful?
(ii) Is $p \equiv_M q$ in the DFA M?
(iii) Is $|[p]| = k$ in the DFA M?—besides M and p the problem instance contains also k in binary.

It is not hard to see that these problems and their complements can also be solved on a nondeterministic Turing machine in logspace.

Now our algorithm proceeds as follows: first the Turing machine verifies that the input is a REV-DFA, by inspecting all states and checking that for every letter a there is at most one a-transition leaving and entering the state. If this is not the case the Turing machine halts and rejects. Next, it is checked whether all states are useful. Here RST oracle queries are used. If this is not the case, the computation halts and rejects. Otherwise, we start verifying the conditions given in Theorem 6. To this end we cycle through all states $q \in S$—note that we already know that all these states are useful. Then we determine the size of $|[q]|$. This is done by cycling through all k with $1 \leq k \leq |S|$ and asking our oracle subroutine whether $|[q]| = k$ holds in M. If $k = 1$ nothing has to be done and the algorithm proceeds with the next q. Otherwise, let $k > 1$, and the algorithm has to verify the property stated in Theorem 6. Therefore we nondeterministically guess a word $w = av$ in reversed order on the fly letter by letter. In case $v = b_1 b_2 \cdots b_m$ with $b_i \in \Sigma$, for $1 \leq i \leq m$, then the machine guesses b_m, $b_{m-1}, \ldots, b_1$ and a in this order. Then for the letter b_m we deterministically compute $q' = \delta^R(q, b_m)$ and verify (i) that $|[q']| = k$ and (ii) that $\delta^R(p, b_m)$ is defined for every state p in $[q]$. Notice that in this case, every state from the equivalence class $[q']$ enters the equivalence class $[q]$ on input b_m. Again, both questions can be answered with the help of oracles on a RST oracle Turing machine. Then we continue the backward computation of M with state q' and the letter b_{m-1} proceeding as just described above. This step by step backward computation continues until we reach state q'' with the next to last guessed letter b_1. Finally, reading letter a backward must result in a situation that the condition of Theorem 6 is fulfilled. This means that $\delta^R(q'', a)$ is defined and (i) results in an equivalence class that is strictly smaller than k and (ii) moreover, $\delta^R(p, a)$ is defined for every state p in $[q'']$. As above these questions are answered with the help of the oracles described above. The Turing machine halts and rejects if any of these oracle questions is not answered appropriately. Then the equivalence class $[q]$ satisfies the condition of Theorem 6 *via* the witness $w = av$, and the Turing machine proceeds with the next q in order.

If we have found witnesses for all equivalence classes $[q]$, for all states q in S, then Turing machine halts and accepts. Otherwise, it halts and rejects. It is not hard to see that the described algorithm can be implemented on a nondeterministic logspace bounded RST oracle Turing machine. Thus, we can decide minimality of REV-DFAs in $\mathsf{NL}^{\langle \mathsf{NL} \rangle} = \mathsf{NL}$. □

A closer look on the construction of a REV-DFA from a given minimal DFA in the proof of Lemma 4 reveals that the constructed automaton satisfies the condition given in Theorem 6, and thus, is a minimal REV-DFA.

Lemma 8. *Let $M = \langle S, \Sigma, \delta, s_0, F \rangle$ be a minimal deterministic finite automaton and $M' = \langle S', \Sigma, \delta', s_0', F' \rangle$ the reversible deterministic finite automaton constructed from M as in the proof of Lemma 4. Then M' is a minimal reversible deterministic finite automaton.* □

Now we are prepared to derive lower bounds on the number of states for minimal reversible DFAs. The currently best known lower bound $\Omega(1.001^n)$ originates in [8]. It relies on the $2n$-fold concatenation L^{2n} of the finite language $L = \{aa, ab, ba\}$—see Figure 3. Using our technique for constructing a minimal REV-DFA, one can derive the exact number of states of a minimal REV-DFA for the language L^{2n}, which is $2^{2n+2} - 3$. Since the minimal DFA for L^{2n} has $6n+1$ states, the blow-up in the number of states is in the order of $2^{n/3} = (\sqrt[3]{2})^n$, which is approximately 1.259^n. In our next example we present a better lower bound which is related to the Fibonacci numbers, and thus is approximately 1.618^n, the golden ratio Φ to the power of n.

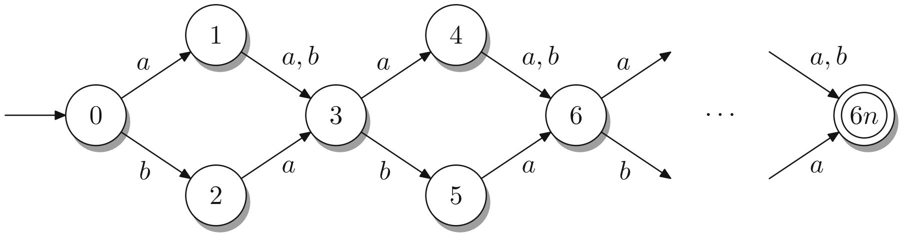

Fig. 3. The minimal DFA accepting the language L^{2n}, for $n \geq 1$

Example 9. Let $n \geq 3$ and consider the DFA $M_n = \langle S, \Sigma, \delta, s_0, F \rangle$ with state set $S_n = \{1, 2, \ldots, n\}$, initial state $s_0 = 1$, final state $F_n = \{n\}$, and transition function δ_n given through:

$$\delta_n(s,a) = \begin{cases} s+1 & \text{if } s \leq n-1 \text{ and } s \text{ is odd,} \\ s+2 & \text{if } s \leq n-2 \text{ and } s \text{ is even,} \end{cases}$$

$$\delta_n(s,b) = \begin{cases} s+2 & \text{if } s \leq n-2 \text{ and } s \text{ is odd,} \\ s+1 & \text{if } s \leq n-1 \text{ and } s \text{ is even.} \end{cases}$$

Figure 4 shows an example of the automaton M_n for $n = 6$. Notice that no transitions are defined in state n, and only one transition is defined in state $n - 1$. Clearly, the DFA M_n is minimal, but not reversible. However, since the language $L(M_n)$ is finite, one readily sees that it can be accepted by a REV-DFA.

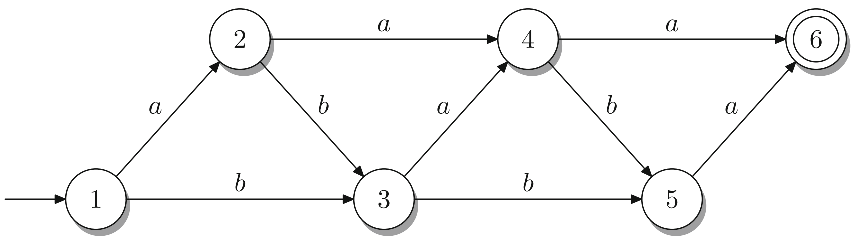

Fig. 4. The minimal DFA M_n, for $n = 6$, where the minimal REV-DFA needs $\sum_{i=1}^{n} F_i = F_{n+2} - 1$ states

Let us apply the construction from the proof of Lemma 4 to construct an equivalent REV-DFA, which, by Lemma 8 is a minimal REV-DFA. The topological order $\preceq$ of the strongly connected components of M_n clearly is the natural order $1 \preceq 2 \preceq \cdots \preceq n$. States 1 and 2 do not need to be copied, but we need two copies of state 3 because of its two predecessor states 1 and 2 by letter b. Then we need three copies of state 4 because of its three predecessors by letter a, namely state 2 and two copies of state 3. It is clear how this continues: every state s of M_n with $s \geq 3$ has two predecessors $s-1$ and $s-2$ either on letter a (if s is even) or letter b (if s is odd). Therefore the number of copies of state s is the sum of the number of copies of $s-1$ and those of $s-2$. Since we start with one copy of state 1 and one copy of state 2, the number of copies of a state $s \in S_n$ in the minimal REV-DFA for $L(M_n)$ is exactly F_s, the s-th Fibonacci number. Therefore the number of states of the minimal REV-DFA is $\sum_{i=1}^{n} F_n$. This is equal to $F_{n+2} - 1$. From the closed form

$$F_n = \frac{1}{\sqrt{5}} \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n$$

and the fact that $\left(\frac{1-\sqrt{5}}{2}\right)^n$ tends to zero, for large n, we see that state blow-up when transforming M into an equivalent REV-DFA is in the order of $\left(\frac{1+\sqrt{5}}{2}\right)^n$, that is, approximately 1.618^n. □

Thus, we have shown the following theorem.

Theorem 10. *For every n with $n \geq 3$ there is an n-state DFA M_n over a binary input alphabet accepting a reversible language, such that any equivalent REV-DFA needs at least $\Omega(\Phi^n)$ states with $\Phi = (1+\sqrt{5})/2$, the golden ratio.* □

It is worth mentioning that the lower bound of Example 9 is for a binary alphabet. It can be increased at the cost of more symbols. For a k-ary alphabet one can derive the lower bound from the k-ary Fibonacci function $F_n = F_{n-1} + F_{n-2} + \cdots + F_{n-k}$. For $k = 3$ the lower bound is of order 1.839^n and for $k = 4$ it is of order 1.927^n. For growing alphabet sizes the bound asymptotically tends to 2^{n-1}, that is, $\Omega(2^{n-1})$.

Finally, our techniques allow us to determine an upper bound of 2^{n-1} states for the conversion from DFAs to equivalent REV-DFAs, even for arbitrary alphabet sizes.

Theorem 11. *Let M be a minimal deterministic finite automaton with n states, that accepts a reversible language. Then a minimal reversible deterministic finite automaton for $L(M)$ has at most 2^{n-1} states.* □

References

1. Angluin, D.: Inference of reversible languages. J. ACM **29**, 741–765 (1982)
2. Axelsen, H.B.: Reversible Multi-head Finite Automata Characterize Reversible Logarithmic Space. In: Dediu, A.-H., Martín-Vide, C. (eds.) LATA 2012. LNCS, vol. 7183, pp. 95–105. Springer, Heidelberg (2012)
3. Axelsen, H.B., Glück, R.: A simple and efficient universal reversible turing machine. In: Dediu, A.-H., Inenaga, S., Martín-Vide, C. (eds.) LATA 2011. LNCS, vol. 6638, pp. 117–128. Springer, Heidelberg (2011)

4. Bennett, C.H.: Logical reversibility of computation. IBM J. Res. Dev. **17**, 525–532 (1973)
5. Cho, S., Huynh, D.T.: The parallel complexity of finite-state automata problems. Inform. Comput. **97**, 1–22 (1992)
6. García, P., Vázquez de Parga, M., López, D.: On the efficient construction of quasi-reversible automata for reversible languages. Inform. Process. Lett. **107**, 13–17 (2008)
7. Harrison, M.A.: Introduction to Formal Language Theory. Addison-Wesley (1978)
8. Héam, P.C.: A lower bound for reversible automata. RAIRO Inform. Théor. **34**, 331–341 (2000)
9. Holzer, M., Jakobi, S.: Minimal and Hyper-Minimal Biautomata. In: Shur, A.M., Volkov, M.V. (eds.) DLT 2014. LNCS, vol. 8633, pp. 291–302. Springer, Heidelberg (2014)
10. Immerman, N.: Nondeterministic space is closed under complement. SIAM J. Comput. **17**, 935–938 (1988)
11. Kobayashi, S., Yokomori, T.: Learning approximately regular languages with reversible languages. Theoret. Comput. Sci. **174**, 251–257 (1997)
12. Kutrib, M.: Aspects of Reversibility for Classical Automata. In: Calude, C.S., Freivalds, R., Kazuo, I. (eds.) Gruska Festschrift. LNCS, vol. 8808, pp. 83–98. Springer, Heidelberg (2014)
13. Kutrib, M., Malcher, A.: Reversible Pushdown Automata. In: Dediu, A.-H., Fernau, H., Martín-Vide, C. (eds.) LATA 2010. LNCS, vol. 6031, pp. 368–379. Springer, Heidelberg (2010)
14. Kutrib, M., Malcher, A.: One-Way Reversible Multi-head Finite Automata. In: Glück, R., Yokoyama, T. (eds.) RC 2012. LNCS, vol. 7581, pp. 14–28. Springer, Heidelberg (2013)
15. Kutrib, M., Malcher, A., Wendlandt, M.: Reversible queue automata. In: Non-Classical Models of Automata and Applications (NCMA 2014). books@ocg.at, vol. 304, pp. 163–178. Austrian Computer Society, Vienna (2014)
16. Landauer, R.: Irreversibility and heat generation in the computing process. IBM J. Res. Dev. **5**, 183–191 (1961)
17. Lombardy, S.: On the Construction of Reversible Automata for Reversible Languages. In: Widmayer, P., Triguero, F., Morales, R., Hennessy, M., Eidenbenz, S., Conejo, R. (eds.) ICALP 2002. LNCS, vol. 2380, pp. 170–182. Springer, Heidelberg (2002)
18. Morita, K.: Two-way reversible multi-head finite automata. Fund. Inform. **110**, 241–254 (2011)
19. Morita, K.: A Deterministic Two-Way Multi-head Finite Automaton Can Be Converted into a Reversible One with the Same Number of Heads. In: Glück, R., Yokoyama, T. (eds.) RC 2012. LNCS, vol. 7581, pp. 29–43. Springer, Heidelberg (2013)
20. Morita, K., Shirasaki, A., Gono, Y.: A 1-tape 2-symbol reversible Turing machine. Trans. IEICE **E72**, 223–228 (1989)
21. Pin, J.E.: On reversible automata. In: Latin 1992: Theoretical Informatics. LNCS, vol. 583, pp. 401–416. Springer (1992)
22. Ruzzo, W.L., Simon, J., Tompa, M.: Space-bounded hierarchies and probabilistic computations. J. Comput. System Sci. **28**, 216–230 (1984)
23. Szelepcsényi, R.: The method of forced enumeration for nondeterministic automata. Acta Inform. **26**, 279–284 (1988)
24. Wagner, K., Wechsung, G.: Computational Complexity. Reidel (1986)