Reversal Hierarchies for Small 2DFAs

Christos A. Kapoutsis[1,⋆] and Giovanni Pighizzini[2]

[1] LIAFA, Université Paris VII, France
[2] DI, Università degli Studi di Milano, Italia

Abstract. A two-way deterministic finite automaton *with* $r(n)$ *reversals* performs $\leq r(n)$ input head reversals on every n-long input. Let $\mathsf{2D}[r(n)]$ be all families of problems solvable by such automata of size polynomial in the index of the family. Then the *reversal hierarchy* $\mathsf{2D}[0] \subseteq \mathsf{2D}[1] \subseteq \mathsf{2D}[2] \subseteq \cdots$ is strict, but $\mathsf{2D}[O(1)] = \mathsf{2D}[o(n)]$. Moreover, the *inner-reversal hierarchy* $\mathsf{2D}(0) \subseteq \mathsf{2D}(1) \subseteq \mathsf{2D}(2) \subseteq \cdots$, where now the bound is only for reversals strictly between the input end-markers, is also strict.

1 Introduction

A long-standing open question of the Theory of Computation is whether every two-way nondeterministic finite automaton (2NFA) is equivalent to a deterministic one (2DFA) with only polynomially more states; or, in other terms, whether $\mathsf{2D} = \mathsf{2N}$, where $\mathsf{2D}$ and $\mathsf{2N}$ are the classes of (families of) problems which are solvable by 'small' (i.e., polynomial-size) 2DFAs and 2NFAs, respectively [8].

In 2002, J. Hromkovič suggested approaching this question by its variants for 2DFAs with restricted number of input-head reversals [3]. Specifically, let a '2DFA *with* $r(n)$ *reversals*' be one which performs $\leq r(n)$ reversals on every n-long input. Next, for any class $\mathcal{R}$ of natural functions, let $\mathsf{2D}[\mathcal{R}]$ be the restriction of $\mathsf{2D}$ to problems that are solvable by small 2DFAs with $r(n)$ reversals, for some $r \in \mathcal{R}$. Then, the following obvious inclusions hold, where $\mathsf{0}, \mathsf{1}, \ldots$ are singletons for the individual constant functions, and const is all these functions together:

$$\begin{aligned}\mathsf{2D}[\mathsf{0}] \underset{\text{(a)}}{\subseteq} \mathsf{2D}[\mathsf{1}] \underset{\text{(b)}}{\subseteq} \mathsf{2D}[\mathsf{2}] \subseteq \cdots \subseteq \mathsf{2D}[\mathsf{r}] \subseteq \cdots \\ \subseteq \mathsf{2D}[\mathsf{const}] \subseteq \mathsf{2D}[O(1)] \subseteq \mathsf{2D}[o(n)] \overset{\text{(c)}}{\subseteq} \mathsf{2D}[O(n)] \overset{\text{(d)}}{\subseteq} \mathsf{2D}\,.\end{aligned} \tag{1}$$

Hromkovič suggested resolving all these inclusions, as well as every relationship between a class and its counterpart for 2NFAs [3, Research Problems 2–4].

Some answers are known: (a), (b), (c) are strict, by [7, Prop. 1], [1, Th. 2.2], and [4, Th. 3]; (d) is equality, as small 2DFAs have small halting equivalents [9,2], which reverse $O(n)$ times [4, Fact 3]; and every class up to $\mathsf{2D}[o(n)]$ is strictly inside its counterpart for 2NFAs, as the witness to [4, Th. 1] admits small 2NFAs even with 0 reversals. Here, we resolve most of the remaining inclusions in (1).

We start in Sect. 3, with a crossing-sequence argument which proves that a 2DFA cannot reverse $o(n)$ times unless it already reverses $O(1)$ times (and thus the lower bound of [4, Th. 1] is really a bound for 2DFAs with $O(1)$ reversals).

⋆ Supported by a Marie Curie Intra-European Fellowship (PIEF-GA-2009-253368) within the European Union Seventh Framework Programme (FP7/2007-2013).

Theorem 1. *Every* 2DFA *with* $o(n)$ *reversals is a* 2DFA *with* $O(1)$ *reversals.*

We continue in Sect. 4, with a uniform argument for all $r \geq 1$, which proves that small 2DFAs with r reversals are strictly more powerful than small 2DFAs with $< r$ reversals. Crucially, our argument makes black-box use of [4, Th. 2].

Theorem 2. *Let* $r \geq 1$. *For each* $h \geq 1$, *some problem requires* $2^{\Omega(h/r)}$ *states on every* 2DFA *with* $< r$ *reversals, but only* $O(r+h)$ *states on a* 2DFA *with* r *reversals.*

Hence, with Theorems 1 and 2 counted in, the chain of (1) is updated as follows:

$$\begin{aligned}\mathsf{2D}[0] \underset{[7]}{\overset{*}{\subsetneq}} \mathsf{2D}[1] \underset{[1]}{\overset{*}{\subsetneq}} \mathsf{2D}[2] \overset{*}{\subsetneq} \cdots \overset{*}{\subsetneq} \mathsf{2D}[\mathsf{r}] \overset{*}{\subsetneq} \cdots \\ \underset{*}{\subsetneq} \mathsf{2D}[\mathsf{const}] \underset{?}{=} \mathsf{2D}[O(1)] \underset{*}{=} \mathsf{2D}[o(n)] \overset{[4]}{\subsetneq} \mathsf{2D}[O(n)] \overset{[9,2]}{=} \mathsf{2D}\,,\end{aligned} \tag{2}$$

where '$*$' marks our contributions, and '?' marks the only remaining unresolved inclusion —we conjecture that the seemingly obvious equality is indeed true.

Finally, in Sect. 5 we show that Theorem 2 remains valid even when we bound only the *inner reversals*, which occur strictly between the two input end-markers (as opposed to *outer reversals*, which occur on the end-markers). Crucially, our proof builds on a stronger variant of the argument behind [4, Th. 2].

Theorem 3. *Let* $r \geq 1$. *For each* $h \geq 1$, *some problem requires* $\Omega(2^{h/2})$ *states on every* 2DFA *with* $< r$ *inner reversals, but only* $O(h)$ *states on a* 2DFA *with* r *inner reversals* (*and* 0 *outer reversals, if* r *is even*; *or* 1 *outer reversal, if* r *is odd*).

Thus, an additional *inner-reversal hierarchy* $\mathsf{2D}(0) \subsetneq \mathsf{2D}(1) \subsetneq \cdots \subsetneq \mathsf{2D}(\mathsf{const})$ is established, where now $\mathsf{2D}(\mathcal{R})$ restricts $\mathsf{2D}$ to problems solvable by small 2DFAs with $r(n)$ *inner* reversals, for some $r \in \mathcal{R}$. (Clearly, $\mathsf{2D}[\mathcal{R}] \subseteq \mathsf{2D}(\mathcal{R})$ for all $\mathcal{R}$; moreover, from const upwards this inclusion is easily seen to be an equality.)

2 Preparation

If $h \geq 0$, then $[h] := \{0, \ldots, h-1\}$. If S is a set, then $|S|$, $\overline{S}$, $\mathbb{P}(S)$, $S_\perp$ are its size, complement, powerset, and augmentation $S \cup \{\perp\}$. If f, g are partial functions, then the composition $(f \circ g)(a)$ is defined iff both $f(a)$ and $g(f(a))$ are, and then equals $g(f(a))$; the k-fold composition of f with itself is denoted by f^k.

Let Σ be an alphabet. If $z \in \Sigma^*$ is a string, we write $|z|$, z_j, z^j, and z^{R} for its length, j-th symbol ($1 \leq j \leq |z|$), j-fold concatenation with itself ($j \geq 0$), and reverse; its j-th boundary ($1 \leq j \leq |z|+1$) is the left boundary of z_j, or the right one if $j = |z|+1$. If $Z \subseteq \Sigma^*$, then $Z^{\mathsf{R}} := \{z^{\mathsf{R}} \mid z \in Z\}$.

A (*promise*) *problem* over Σ is a pair $\mathfrak{L} = (L, \tilde{L})$ of disjoint subsets of Σ^*. Every w in the *promise* $L \cup \tilde{L}$ is an *instance* of $\mathfrak{L}$: *positive* if $w \in L$, or *negative* if $w \in \tilde{L}$. To solve $\mathfrak{L}$ is to accept every $w \in L$ but no $w \in \tilde{L}$.

2.1 Two-Way Automata

A *two-way deterministic finite automaton* (2DFA) is any $M = (Q, \Sigma, \delta, q_{\mathrm{s}}, q_{\mathrm{a}}, q_{\mathrm{r}})$, where Q is a set of *states*, Σ is an *alphabet*, $q_{\mathrm{s}}, q_{\mathrm{a}}, q_{\mathrm{r}} \in Q$ are the *start*, *accept*, and *reject* states, and $\delta : Q \times (\Sigma \cup \{\vdash,\dashv\}) \to Q \times \{\text{L},\text{R}\}$ is the (total) *transition function*, using two end-markers $\vdash,\dashv \notin \Sigma$ and the two directions L,R. An input $w \in \Sigma^*$ is presented to M between the end-markers, as $\vdash w \dashv$. The computation starts at q_s and on $\vdash$. At each step, the next state and head motion are derived from δ and the current state and symbol. End-markers may be violated only if the next state is q_a or q_r: $\delta(\cdot,\vdash)$ is always (q_a,L), (q_r,L), or $(\cdot,\text{R})$; and $\delta(\cdot,\dashv)$ is always (q_a,R), (q_r,R), or $(\cdot,\text{L})$. So, the computation loops, or falls off $\vdash w \dashv$ into q_r, or falls off $\vdash w \dashv$ into q_a. In this last case, we say *M accepts w*.

Formally, *the computation of M from state p and the j-th symbol of string z*, denoted $\text{COMP}_{M,p,j}(z)$, is the longest sequence $c = ((q_t, j_t))_{0 \le t < m}$ such that $0 < m \le \infty$, $(q_0, j_0) = (p, j)$, and every next (q_t, j_t) follows from the previous one via δ and z in the usual way (Fig. 1a). We call (q_t, j_t) the *t-th point* of c. If $m = \infty$ then c *loops*; otherwise it *halts*, and *hits left* (if $j_{m-1} = 0$) or *hits right* (if $j_{m-1} = |z|+1$) *into* q_{m-1}. The computation $\text{LCOMP}_{M,p}(z) := \text{COMP}_{M,p,1}(z)$ is *the* L*-computation of M from p on z*; depending on whether it loops, hits left, or hits right, we call it a L-*loop*, L-*turn*, or LR-*traversal*. Symmetrically, *the* R*-computation of M from p on z*, $\text{RCOMP}_{M,p}(z) := \text{COMP}_{M,p,|z|}(z)$, is a R-*loop*, R-*turn*, or RL-*traversal*. The (*full*) *computation of M on $w \in \Sigma^*$* is $\text{COMP}_M(w) := \text{LCOMP}_{M,q_s}(\vdash w \dashv)$. So, M accepts w iff $\text{COMP}_M(w)$ falls off $\vdash w \dashv$ into q_a.

The *j-th crossing sequence* of a computation c on a string z is the sequence $q_1, q_2, \ldots$ where q_i is the state immediately after c crosses the j-th boundary of z for the i-th time. Easily, if c halts, then every crossing sequence contains $\le 2|Q|$ states, and thus $\le (|Q|+1)^{2|Q|}$ of these sequences are distinct.

A *reversal* of c is a point $(\,.\,, j_t)$ whose predecessor and successor exist and lie on the same side relative to it: $t \ne 0, m-1$, and $j_{t-1}, j_{t+1} < j_t$ or $j_t < j_{t-1}, j_{t+1}$ (Fig. 1a). If c is full, a reversal is either an *outer reversal*, if it lies on $\vdash$ or $\dashv$, or an *inner reversal*, otherwise. We write $r(c)$ for the total number of reversals in c. Clearly $0 \le r(c) \le \infty$, with $r(c) = \infty$ iff c loops. We write $r_M(n)$ for the maximum $r(c)$ over all full computations c of M on n-long inputs. Easily, if finite, $r_M(n)$ is at most linear: $r_M(n) = \infty$ or $r_M(n) \le |Q| \cdot (n+2)$, for all n.

We say M is a 2DFA *with $r(n)$ reversals* if $r_M(n) \le r(n)$ for all n; or a 2DFA *with $r(n)$ inner reversals* if every full computation on an n-long input performs $\le r(n)$ inner reversals. If M is a 2DFA with 0 inner reversals, we call it *sweeping*

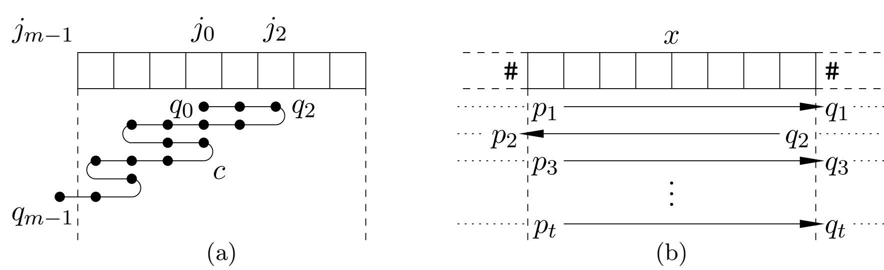

Fig. 1. (a) A left-hitting computation c with $m = 15$ and $r(c) = 5$ reversals, at points 2, 6, 8, 11, and 12. (b) A certificate for x (for the case of odd t).

(SDFA). If it is a 2DFA with 0 reversals and may also *hang* (i.e., δ is a partial function), we call it *one-way* (1DFA); then, the state $q_{\rm i}$ for which $\delta(q_{\rm s},\vdash) = (q_{\rm i},\text{R})$ is called *initial*, every state q with $\delta(q,\dashv) = (q_{\rm a},\text{R})$ is called *final*, and M accepts w iff $\text{LCOMP}_{M,q_{\rm i}}(w)$ hits right into a final state.

2.2 Parallel Automata

A (*two-sided*) *parallel automaton* ($\text{P}_2$1DFA) [10] is any triple $M = (\mathcal{A}, \mathcal{B}, F)$ where $\mathcal{A}, \mathcal{B}$ are disjoint families of 1DFAs over an alphabet Σ, and F is a subset of the cartesian product of all sets $Q^D_\perp$, for $D \in \mathcal{A} \cup \mathcal{B}$ and Q^D the state set of D. To run M on input $w \in \Sigma^*$ means to run each D on w (without end-markers) from its initial state and record the *result* (the state in which D falls off w, or $\perp$ if it hangs), but with a twist: each $D \in \mathcal{A}$ reads w from left to right (as usual), while each $D \in \mathcal{B}$ reads w from right to left (i.e., it reads w^{R}). We say M accepts w iff the produced tuple of $|\mathcal{A}|+|\mathcal{B}|$ results is in F.

If F consists of the tuples where every result is a final state in the respective 1DFA, then M is a *parallel intersection automaton* ($\cap_2$1DFA) [8]: it accepts iff *all* its components do. If F consists of the tuples where at least one of the results is a final state, then M is a *parallel union automaton* ($\cup_2$1DFA) [8]: it accepts iff *any* of its component does. In both cases, we write only $M = (\mathcal{A}, \mathcal{B})$. When $\mathcal{B} = \emptyset$ or $\mathcal{A} = \emptyset$, we say M is *left-sided* ($\cap_{\text{L}}$1DFA, $\cup_{\text{L}}$1DFA) or *right-sided* ($\cap_{\text{R}}$1DFA, $\cup_{\text{R}}$1DFA).

We now recall notions and facts leading to *generic strings* and *blocks* [10,4].

For $D \in \mathcal{A}$ and $y \in \Sigma^*$, the set of states that can be produced on the right boundary of y by L-computations of D is denoted by:

$$Q^D_{\text{LR}}(y) := \{q \mid (\exists p)[\,\text{LCOMP}_{D,p}(y) \text{ hits right into } q\,]\}\,.$$

For every right extension yz of y, we let $\alpha^D_{y,z} : Q^D_{\text{LR}}(y) \rightharpoonup Q^D$ be the partial function whose value on each $q \in Q^D_{\text{LR}}(y)$ is either the state which $\text{LCOMP}_{D,q}(z)$ hits right into, or undefined if $\text{LCOMP}_{D,q}(z)$ hangs. Similarly, for $D \in \mathcal{B}$, we let $Q^D_{\text{RL}}(y) := \{q \mid (\exists p)[\,\text{RCOMP}_{D,p}(y) \text{ hits left into } q\,]\}$, and $\beta^D_{z,y} : Q^D_{\text{RL}}(y) \rightharpoonup Q^D$ be such that $\beta^D_{z,y}(q) = r$ iff $\text{RCOMP}_{D,q}(z)$ hits left into r. The next straightforward fact is a summary of [6, Facts 3.7–9] (as well as a special case of [5, Facts 3–4]).

Fact 1. *If $D \in \mathcal{A}$ then $\alpha^D_{y,z}$ partially surjects $Q^D_{\text{LR}}(y)$ to $Q^D_{\text{LR}}(yz)$, thus $|Q^D_{\text{LR}}(y)| \geq |Q^D_{\text{LR}}(yz)|$; in addition, $Q^D_{\text{LR}}(yz) \subseteq Q^D_{\text{LR}}(z)$. If $D \in \mathcal{B}$ then $\beta^D_{z,y}$ partially surjects $Q^D_{\text{RL}}(y)$ to $Q^D_{\text{RL}}(zy)$, thus $|Q^D_{\text{RL}}(zy)| \leq |Q^D_{\text{RL}}(y)|$; in addition, $Q^D_{\text{RL}}(z) \supseteq Q^D_{\text{RL}}(zy)$.*

For $L \subseteq \Sigma^*$, we say y is *generic for M over L* if $y \in L$ and no right (resp., left) extension of y in L reduces the number of states produced on the right (left) boundary by the L-computations (R-computations) of any $D \in \mathcal{A}$ (any $D \in \mathcal{B}$):

$$y \in L \quad \text{and} \quad \begin{array}{l} (\forall yz \in L)(\forall D \in \mathcal{A})[\,|Q^D_{\text{LR}}(yz)| = |Q^D_{\text{LR}}(y)|\,] \\ (\forall zy \in L)(\forall D \in \mathcal{B})[\,|Q^D_{\text{RL}}(zy)| = |Q^D_{\text{RL}}(y)|\,]\,. \end{array}$$

If ϑ is a fixed generic string for M over L, every string of the form $\vartheta x \vartheta$ is called a *block*, with *infix* x. For $D \in \mathcal{A}$, we write $\alpha^D_{\vartheta,x\vartheta}$ simply as α^D_x, and note that it partially maps $Q^D_{\text{LR}}(\vartheta)$ to itself, since $Q^D_{\text{LR}}(\vartheta x\vartheta) \subseteq Q^D_{\text{LR}}(\vartheta)$ (by Fact 1). Similarly, for $D \in \mathcal{B}$ we write $\beta^D_{\vartheta x,\vartheta} : Q^D_{\text{RL}}(\vartheta) \rightharpoonup Q^D_{\text{RL}}(\vartheta)$ simply as β^D_x. The tuple

$$\big(\ (\alpha^D_x)_{D\in\mathcal{A}},\ (\beta^D_x)_{D\in\mathcal{B}}\ \big)$$

is the *inner behavior* of M on $\vartheta x\vartheta$, and satisfies the next lemma, by standard 'cut-and-paste' arguments (e.g., see [4, Lemma 3], [5, Fact 6], [10]), and the following fact, by the definition of generic string (e.g., see [5, Fact 5], [10]).

Lemma 1. **(a)** *If the inner behavior of M on $\vartheta x\vartheta$ consists of identities, then M decides identically on ϑ and $\vartheta x\vartheta$.* **(b)** *If the inner behaviors of M on $\vartheta x\vartheta$ and $\vartheta y\vartheta$ are identical, then M decides identically on $\vartheta x\vartheta$ and $\vartheta y\vartheta$.*

Fact 2. *On every $\vartheta x\vartheta \in L$, the inner behavior of M consists of permutations.*

The next fact (variant of [5, Facts 7–8]) says that the inner behavior on a block of the form $\vartheta x\vartheta y\vartheta$, where ϑ appears in the infix, composes the two inner behaviors for the overlapping blocks $\vartheta x\vartheta$ and $\vartheta y\vartheta$; this then generalizes to blocks of the form $\vartheta x^{(k)}\vartheta$, where the infix $x^{(k)} := x(\vartheta x)^{k-1}$ is k ϑ-separated copies of the same string. Finally, Fact 4 (variant of [4, Fact 7]) follows as an easy corollary.

Fact 3. *For all $D \in \mathcal{A}$, it is $\alpha^D_{x\vartheta y} = \alpha^D_x \circ \alpha^D_y$; hence $\alpha^D_{x^{(k)}} = (\alpha^D_x)^k$ for all $k \geq 1$. Similarly, for all $D \in \mathcal{B}$, it is $\beta^D_{x\vartheta y} = \beta^D_y \circ \beta^D_x$; hence $\beta^D_{x^{(k)}} = (\beta^D_x)^k$ for all $k \geq 1$.*

Fact 4. *If the inner behavior of M on $\vartheta x\vartheta$ consists of permutations, then for some $k \geq 1$ the inner behavior of M on $\vartheta x^{(k)}\vartheta$ consists of identities.*

2.3 Hardness Propagation

In the "*hardness propagation*" style of [6], all our witnesses are built by applying appropriate 'hardness increasing' operators to a single, well-understood, 'core' problem. Below, we first recall some of these operations along with some associated hardness propagation lemmata. We then also recall our one 'core' problem.

Let $\mathfrak{L} = (L, \tilde{L})$. The *reverse* and the *complement* of $\mathfrak{L}$ are the problems $\mathfrak{L}^{\mathsf{R}} := (L^{\mathsf{R}}, \tilde{L}^{\mathsf{R}})$ and $\neg\mathfrak{L} := (\tilde{L}, L)$. Easily, $\neg(\mathfrak{L}^{\mathsf{R}}) = (\neg\mathfrak{L})^{\mathsf{R}}$, and [4, Fact 12] holds:

Lemma 2. **(a)** *If no $\cup_{\text{L}}$1DFA with s-state components solves $\mathfrak{L}$, then no $\cup_{\text{R}}$1DFA with s-state components solves $\mathfrak{L}^{\mathsf{R}}$.* **(b)** *If no $\cap_{\text{L}}$1DFA with $(s{+}1)$-state components solves $\mathfrak{L}$, then no $\cup_{\text{L}}$1DFA with s-state components solves $\neg\mathfrak{L}$.*

The *conjunctive star* of $\mathfrak{L}$ is the problem of checking that a #-delimited list of instances of $\mathfrak{L}$ contains only positives; dually, the *disjunctive star* is the problem where at least one instance in the list must be positive [6, §3.1]:

$$\textstyle\bigwedge\mathfrak{L} := \big(\ \{\#x_1\#\cdots\#x_l\# \mid (\forall i)(x_i \in L)\},\ \ \{\#x_1\#\cdots\#x_l\# \mid (\exists i)(x_i \in \tilde{L})\}\ \big)$$
$$\textstyle\bigvee\mathfrak{L} := \big(\ \{\#x_1\#\cdots\#x_l\# \mid (\exists i)(x_i \in L)\},\ \ \{\#x_1\#\cdots\#x_l\# \mid (\forall i)(x_i \in \tilde{L})\}\ \big),$$

where $\#x_1\#\cdots\#x_l\#$ means $l \geq 0$, each $x_i \in L \cup \tilde{L}$, and # is a fresh symbol. Easily,

$$\textstyle\neg(\bigwedge\mathfrak{L}) = \bigvee\neg\mathfrak{L} \qquad \neg(\bigvee\mathfrak{L}) = \bigwedge\neg\mathfrak{L} \qquad (\bigwedge\mathfrak{L})^{\mathsf{R}} = \bigwedge\mathfrak{L}^{\mathsf{R}} \qquad (\bigvee\mathfrak{L})^{\mathsf{R}} = \bigvee\mathfrak{L}^{\mathsf{R}},$$

by the definitions. In addition, the following lemma holds [6, Lemma 3.3]:

Lemma 3. *If no s-state* 1DFA *solves* $\mathfrak{L}$*, then no* $\cap_{\text{L}}$1DFA *with s-state components solves* $\bigvee\mathfrak{L}$*.*

The *ordered star* $\mathfrak{L}_{\text{L}}{<}\mathfrak{L}_{\text{R}}$ of two problems $\mathfrak{L}_{\text{L}} = (L_{\text{L}}, \tilde{L}_{\text{L}})$ and $\mathfrak{L}_{\text{R}} = (L_{\text{R}}, \tilde{L}_{\text{R}})$ of disjoint promises is defined as follows [4, §7.2]: an instance is promised to be a list $x = \#x_1\#\cdots\#x_l\#$ of #-delimited instances of $\mathfrak{L}_{\text{L}}$ and $\mathfrak{L}_{\text{R}}$ where all positives of one of the problems appear before all positives of the other (note that this includes lists where at most one problem contributes positives); the task is to check that *either* both problems contribute positives and the one that places them first is $\mathfrak{L}_{\text{L}}$ *or* neither problem contributes any positives. So, in a positive x, there are x_i both from L_{L} and from L_{R}, and all those from L_{L} precede all those from L_{R}; or all x_i are in $\tilde{L}_{\text{L}} \cup \tilde{L}_{\text{R}}$. In a negative x, there are x_i both from L_{L} and from L_{R}, and all those from L_{R} precede all those from L_{L}; or exactly one of L_{L},L_{R} contributes some x_i. The next hardness propagation lemma is [4, Lemma 8]:

Lemma 4. *If no* $\cup_{\text{L}}$1DFA *with* $1{+}\binom{s}{2}$*-state components solves* $\mathfrak{L}_{\text{L}}$ *and no* $\cup_{\text{R}}$1DFA *with* $1{+}\binom{s}{2}$*-state components solves* $\mathfrak{L}_{\text{R}}$*, then no s-state* SDFA *solves* $\mathfrak{L}_{\text{L}}{<}\mathfrak{L}_{\text{R}}$*.*

The *membership problem* is defined over the alphabet $[h] \cup \mathbb{P}([h])$ as follows: "Given an $i \in [h]$ and an $\alpha \subseteq [h]$ (in this order), check that $i \in \alpha$." Formally:

$$\mathfrak{M} = \text{MEMBERSHIP}_h := \big(\ \{i\alpha \mid \alpha \subseteq [h] \ \&\ i \in \alpha\},\ \{i\alpha \mid \alpha \subseteq [h] \ \&\ i \in \overline{\alpha}\}\ \big)\,. \quad (3)$$

Easily, $\mathfrak{M}$ has an h-state 1DFA, but $\mathfrak{M}^{\text{R}}$ and $\neg\mathfrak{M}^{\text{R}}$ (where α precedes i) have no 1DFA with $<2^h{-}1$ states [6,4]. (In [4, Eq. (7)], $\mathfrak{M}^{\text{R}}$ is called SET NUM$_h$.)

3 From Few Reversals to Bounded Reversals

We now prove Theorem 1. We pick a 2DFA M with $r_M(n) \neq O(1)$, and show that $r_M(n) \neq o(n)$, too. Note that this is trivial if $r_M(n) = \infty$ for infinitely many n. So, the interesting case is when $r_M(n)$ is finite for all sufficiently large n.

Since $r_M(n) \neq O(1)$, every bound r admits infinitely many n with $r_M(n) \geq r$. Consider in particular $r := s{\cdot}(s{+}1)^{2s}$, for s the number of states in M. Then, for infinitely many n, some full computation c_n on some n-long input performs $\geq s{\cdot}(s{+}1)^{2s}$ reversals. Moreover, for all sufficiently large n, these c_n are halting, exactly because we are in the interesting case. Let c be one of these halting c_n.

Let $j_1 < \cdots < j_m$ be the indices of the cells where c performs its $\geq s{\cdot}(s{+}1)^{2s}$ reversals. Then $m \geq (s{+}1)^{2s}$, or else $m < (s{+}1)^{2s}$ cells would host $\geq s{\cdot}(s{+}1)^{2s}$ reversals, so some cell would host $> s$ reversals, so c would repeat a point on that cell and thus loop, a contradiction.

Now let $\overline{q}_0, \overline{q}_1, \ldots, \overline{q}_m$ be the crossing sequences of c on any $m{+}1$ boundaries that are separated by the m cells above. Since $m{+}1$ exceeds the number $(s{+}1)^{2s}$ of distinct crossing sequences in halting computations (cf. Sect. 2.1), two of the $\overline{q}_i$ must be identical. Let y be the infix between the corresponding two boundaries. Then the input is xyz, for some x,z.

We know y hosts ≥ 1 of the reversals of c, because it contains ≥ 1 of the cells indexed by the j_i. We also know, by a standard 'cut-and-paste' argument, that every full computation $c_t := \textsc{comp}_M(xy^tz)$ repeats on every copy of y every computation segment performed by c on y, including all reversals contained therein. So, every c_t performs ≥ 1 reversal on each copy of y, for a total of $\geq t$ reversals. Hence, for the infinitely many lengths $n_t := |xy^tz|$ some n_t-long input forces M to perform $\geq t = (n_t - |xz|)/|y|$ reversals. Hence, $r_M(n) \neq o(n)$.

So, Theorem 1 holds, making the inclusion $\mathsf{2D}[O(1)] \subseteq \mathsf{2D}[o(n)]$ an equality.

Concerning the inclusion $\mathsf{2D}[\mathsf{const}] \subseteq \mathsf{2D}[O(1)]$ one level down, it is tempting to suggest that it is also an equality, caused by the seemingly obvious reduction (analogous to that of Theorem 1) that every 2DFA with $O(1)$ reversals is a 2DFA with r reversals, for some r. But this suggestion is wrong (easily). The next tempting suggestion is that, although a 2DFA with $O(1)$ reversals is not *already* one with r reversals, it can be made into one, with some increase in size. Indeed:

Lemma 5. *Every s-state* 2DFA *with $O(1)$ reversals is equivalent to a $O(rs)$-state* 2DFA *with r reversals, for some r.*

Still, in this lemma, r may be super-polynomial in s (as in the 2DFA built in the proof of Theorem 1), resulting in a 2DFA too large to prove $\mathsf{2D}[\mathsf{const}] = \mathsf{2D}[O(1)]$.

4 Inside the Reversal Hierarchy

In this section we prove Theorem 2. We first introduce a new 'hardness increasing' operator and prove an associated 'hardness propagation' lemma.

The *r-th conjunctive power* of $\mathfrak{L} = (L, \tilde{L})$ is the problem of checking that a #-delimited list of exactly r instances of $\mathfrak{L}$ contains only positives:

$$\textstyle\bigwedge_r \mathfrak{L} := \Big(\ \{\#x_1\#\cdots\#x_r\# \mid (\forall i)(x_i \in L)\}, \ \ \{\#x_1\#\cdots\#x_r\# \mid (\exists i)(x_i \in \tilde{L})\} \ \Big)\,,$$

where $\#x_1\#\cdots\#x_r\#$ means that every $x_i \in L \cup \tilde{L}$ and # is a fresh symbol.

Lemma 6. *If no $4rs^{2r+1}$-state* SDFA *solves $\mathfrak{L}$, then no s-state* 2DFA *with $<r$ reversals solves $\bigwedge_r \mathfrak{L}$.*

Proof. Let $\mathfrak{L} = (L, \tilde{L})$. Let M be an s-state 2DFA with $<r$ reversals for $\bigwedge_r \mathfrak{L}$. We build a SDFA M' for $\mathfrak{L}$ with $4rs^{2r+1}$ states. We first introduce *certificates*, then show how they characterize the positives of $\mathfrak{L}$, then use them to design M'.

Pick any positive x of $\mathfrak{L}$. Then $w := \#(x\#)^r$ is a positive of $\bigwedge_r \mathfrak{L}$. Therefore, $c := \textsc{comp}_M(w)$ is accepting. Moreover, the reversals in c are fewer than the copies of x in w. So, on one or more of these copies, c performs 0 reversals. Fix any such copy (e.g., the leftmost one). On it, c consists of $t \leq r$ one-way traversals (one-way, since there are 0 reversals; and $\leq r$, because with $<r$ reversals in total c can traverse each infix $\leq r$ times). Let $\overline{p}_x := (p_1, \ldots, p_t)$ and $\overline{q}_x := (q_1, \ldots, q_t)$ be the crossing sequences of c on the outer boundaries of that copy of x (Fig. 1b). Finally, consider the set of all pairs of crossing sequences created in this way,

$$\mathcal{C} := \{(\overline{p}_x, \overline{q}_x) \mid x \in L\}\,,$$

as we iterate over all positives of $\mathfrak{L}$. We use this set in the next definition.

Definition. A pair $(\overline{p}, \overline{q})$ of t-long sequences of states of M is a *certificate* for an instance x of $\mathfrak{L}$ if it satisfies the following three clauses:

1. $(\overline{p}, \overline{q}) \in \mathcal{C}$.
2. For every odd $i = 1, \dots, t$: $\text{LCOMP}_{M,p_i}(x)$ is one-way and hits right into q_i.
3. For every even $i = 1, \dots, t$: $\text{RCOMP}_{M,q_i}(x)$ is one-way and hits left into p_i.

Claim. An instance of $\mathfrak{L}$ is positive iff it has a certificate.

Proof. [⇒] Let $x \in L$. Then clearly $(\overline{p}_x, \overline{q}_x)$ is a certificate for x. [⇐] Let $\tilde{x} \in \tilde{L}$. Suppose $\tilde{x}$ has a certificate $(\overline{p}, \overline{q})$. By Clause 1, there is $x \in L$ such that the accepting computation $c := \text{COMP}_M(\texttt{\#}(x\texttt{\#})^r)$ exhibits $\overline{p}$ and $\overline{q}$ on the outer boundaries of a copy of x on which it contains 0 reversals. By Clauses 2 and 3, M notices no difference if we replace that copy with a copy of $\tilde{x}$. So, the computation of M on the modified string is also accepting. But this modified string is a negative of $\bigwedge_r \mathfrak{L}$. Therefore, M does not solve $\bigwedge_r \mathfrak{L}$ —a contradiction. ⊡

By the Claim, one way to check an instance x of $\mathfrak{L}$ is to check whether any pair in $\mathcal{C}$ is a certificate for it; because $\mathcal{C}$ is 'small' and each pair is checkable by 'few' sweeps, this strategy can be implemented by a 'small' SDFA. Specifically, M' iterates over all $((p_1, \dots, p_t), (q_1, \dots, q_t)) \in \mathcal{C}$. For each of them and each odd (resp., even) $i = 1, \dots, t$, it simulates M on x from p_i (from q_i) on the leftmost (rightmost) symbol, to see whether it hits right (left) into q_i (into p_i) without ever reversing; on any attempt to reverse, M' stops simulating and just completes the sweep. If these checks succeed for all i, then M' accepts; otherwise, it continues to the next pair. If all pairs have been tried, then M' rejects.

If Q are the states of M, then M' uses states $Q' := \mathcal{C} \times \{1, \dots, r\} \times Q_\perp$. State $(\overline{p}, \overline{q}, i, p)$ means we are at state p in simulating M in the i-th check for the candidate certificate $(\overline{p}, \overline{q})$; if $p = \perp$, then the i-th check has already failed due to an attempt to reverse, and we are just completing the sweep. Easily, $|\mathcal{C}| \le \sum_{t=0}^{r} (s^t \cdot s^t) \le 2s^{2r}$, therefore $|Q'| = |\mathcal{C}| \cdot r \cdot (|Q|+1) \le 2s^{2r} \cdot r \cdot (s+1) \le 4rs^{2r+1}$. □

We are now ready to introduce our witness. For $r \ge 1$ and $\mathfrak{M}$ as in (3), it is

$$\mathfrak{R}_r := \bigwedge_r \left[\left(\bigwedge \mathfrak{M}^{\text{R}} \right) < \left(\bigwedge \mathfrak{M} \right) \right]. \tag{4}$$

So, an instance of $\mathfrak{R}_r$ is a list of the form $\texttt{\$}y_1\texttt{\$} \cdots \texttt{\$}y_r\texttt{\$}$; each y_j is a list of the form $\texttt{*}x_1\texttt{*} \cdots \texttt{*}x_l\texttt{*}$, for arbitrary l; and each x_j is a list of the form $\texttt{\#}\alpha_1 i_1\texttt{\#} \cdots \texttt{\#}\alpha_l i_l\texttt{\#}$ or $\texttt{\#}i_1\alpha_1\texttt{\#} \cdots \texttt{\#}i_l\alpha_l\texttt{\#}$, again for arbitrary l. The task is to check that, in every y_j: *either* every x_j has some i_j not in the adjacent α_j (i.e., all x_j are negatives of $\bigwedge\mathfrak{M}^{\text{R}}$ and $\bigwedge\mathfrak{M}$); *or* x_j of both forms exist with all their i_j in the adjacent α_j, and those of the set-number form precede those of the number-set form (i.e., both $\bigwedge\mathfrak{M}^{\text{R}}$ and $\bigwedge\mathfrak{M}$ contribute positives, and those of $\bigwedge\mathfrak{M}^{\text{R}}$ precede those of $\bigwedge\mathfrak{M}$).

For the lower bound, we know that every SDFA for $\bigwedge\mathfrak{M}^{\text{R}} < \bigwedge\mathfrak{M}$ has $2^{\Omega(h)}$ states (by the lower-bound argument of [4, §7.3], which uses Lemma 4). Therefore, by Lemma 6, every 2DFA with $< r$ reversals for $\mathfrak{R}_r$ has $2^{\Omega(h/r)}$ states.

For the upper bound, we start as in [4, §7.3]. We let M_0 be the h-state 1DFA for $\mathfrak{M}$. We then build a $O(h)$-state 1DFA M_1 for $\bigwedge\mathfrak{M}$, which just repeatedly simulates M_0 on the successive instances of $\mathfrak{M}$ and accepts iff all are accepted.

Next, we build a $O(h)$-state 2DFA M_2 with 1 reversal for $\bigwedge\mathfrak{M}^{\text{R}}{<}\bigwedge\mathfrak{M}$. On input $*x_1*\cdots*x_l*$, M_2 scans forward simulating M_1 on every instance of $\bigwedge\mathfrak{M}$ until it detects a positive or reaches $\dashv$. In either case, it reverses and scans backwards simulating M_1 on (the reverse of) every instance of $\bigwedge\mathfrak{M}^{\text{R}}$ until it detects a positive or reaches $\vdash$. Then M_2 knows what to do: (1) if neither scan detected a positive, then all x_j are negative, so M_2 must accept; (2) if the forward scan detected no positive but the backward scan did, then only $\bigwedge\mathfrak{M}^{\text{R}}$ contributes positives, so M_2 must reject; (3) if the forward scan detected a positive but the backward scan did not, then M_2 must reject either because only $\bigwedge\mathfrak{M}$ contributes positives or because both problems do but the order is wrong; (4) if both scans detected a positive, then both problems contribute and the order is correct, so M_2 must accept. So, M_2 finishes the backward scan (if needed) and decides on $\vdash$.

Finally, we build a 2DFA R_r with r reversals for $\mathfrak{R}_r$. On input $\$y_1\$\cdots\$y_r\$$, a successive pair $\$y_j\$y_{j+1}\$$ is checked by a 2-reversal LR-traversal, as follows: scan forward past y_j; simulate M_2 on y_{j+1} by a 1-reversal L-turn which ends on the middle $\$$; from there, simulate M_2 on (the reverse of) y_j by a 1-reversal R-turn which also ends on the middle $\$$; from there, scan forward past y_{j+1}. Easily, this check needs $O(h)$ states. Now, if r is even, then R_r simply repeats this check on every pair of successive y_j until it reaches $\dashv$. If r is odd, then R_r first scans forward past $y_1,\dots,y_{r-1}$, to simulate M_2 on y_r by a 1-reversal L-turn that ends on the penultimate $\$$; from there, it starts checking pairs of successive y_j by repeating the above check (backwards and in reverse) until $\vdash$. Easily, the number of states in R_r is $O(r{+}h)$ —for even r, it is only $O(h)$.

5 Inside the Inner-Reversal Hierarchy

We now prove Theorem 3. Most crucially, we improve the lower bound of Sect. 4 to make it (i) independent of r, and (ii) valid even when only *inner* reversals are restricted. For this, we enhance our chain of hardness propagation, by proving variants of Lemmata 4 and 6 where SDFAs are replaced by $\text{P}_2$1DFAs.

Lemma 4*. *If no* $\cup_{\text{L}}$1DFA *with* $1{+}\binom{s}{2}$*-state components solves* $\mathfrak{L}_{\text{L}}$ *and no* $\cup_{\text{R}}$1DFA *with* $1{+}\binom{s}{2}$*-state components solves* $\mathfrak{L}_{\text{R}}$*, then no* $\text{P}_2$1DFA *with* s*-state components solves* $\mathfrak{L}_{\text{L}}{<}\mathfrak{L}_{\text{R}}$.

Proof. The structure of the argument is exactly as in the proof of [4, Lemma 8]; we just adapt some of its steps for $\text{P}_2$1DFAs. So, let $\mathfrak{L}_{\text{L}}=(L_{\text{L}},\tilde{L}_{\text{L}})$, $\mathfrak{L}_{\text{R}}=(L_{\text{R}},\tilde{L}_{\text{R}})$. Suppose some $\text{P}_2$1DFA $M=(\mathcal{A},\mathcal{B},F)$ solves $\mathfrak{L}_{\text{L}}{<}\mathfrak{L}_{\text{R}}$ with s-state components.

We first consider the strings of #-delimited instances of $\mathfrak{L}_{\text{L}}$ and $\mathfrak{L}_{\text{R}}$ where neither problem contributes positives, and those where exactly one does:

$$L := \{\#x_1\#\cdots\#x_l\# \mid (\forall i)(x_i\in\tilde{L}_{\text{L}}\cup\tilde{L}_{\text{R}})\}\text{, and}$$
$$\tilde{L} := \{\#x_1\#\cdots\#x_l\# \mid (\exists i)(x_i\in L_{\text{L}}\cup L_{\text{R}})\ \ \&\ \ \neg(\exists i)(\exists j)(x_i\in L_{\text{L}}\ \&\ x_j\in L_{\text{R}})\}\,,$$

where $\#x_1\#\cdots\#x_l\#$ means $l\geq 0$ and every $x_i\in L_{\text{L}}\cup\tilde{L}_{\text{L}}\cup L_{\text{R}}\cup\tilde{L}_{\text{R}}$. Note that all strings in $L\cup\tilde{L}$ are instances of $\mathfrak{L}_{\text{L}}{<}\mathfrak{L}_{\text{R}}$: positive if in L, negative if in $\tilde{L}$. So,

M solves $(L, \tilde{L})$. From now on, fix ϑ to be a generic string for M over L. (The existence of such a string follows from standard observations [6, §3.3.2].)

Definition. A pair $\{p, q\}$ of distinct states in M is a *forward certificate* for an instance x of $\mathfrak{L}_{\mathrm{L}}$ or $\mathfrak{L}_{\mathrm{R}}$ if there exists $D \in \mathcal{A}$ such that

$$p, q \in Q^D_{\mathrm{LR}}(\vartheta) \quad \text{and} \quad \begin{array}{l}\text{if both } \mathrm{LCOMP}_{D,p}(x\vartheta) \text{ and } \mathrm{LCOMP}_{D,q}(x\vartheta) \text{ hit right,}\\ \text{then they do so into the same state.}\end{array} \tag{5}$$

A *backward certificate* is defined symmetrically, with $\mathcal{A}$, Q^D_{LR}, $\mathrm{LCOMP}_{D,\cdot}(x\vartheta)$, and "hit right" replaced respectively by $\mathcal{B}$, Q^D_{RL}, $\mathrm{RCOMP}_{D,\cdot}(\vartheta x)$, and "hit left".

Claim 1. An instance of $\mathfrak{L}_{\mathrm{L}}$ or $\mathfrak{L}_{\mathrm{R}}$ is positive iff it has a certificate.

Proof. As in [4, Lemma 8]. [⇒] By Fact 4 and Lemma 1a. [⇐] By Fact 2. ⊡

Note that, for positive instances, Claim 1 does not specify whether the existing certificates are of the forward or of the backward kind. It turns out that a stronger criterion is possible for at least one of $\mathfrak{L}_{\mathrm{L}}$ or $\mathfrak{L}_{\mathrm{R}}$.

Claim 2. At least one is true: (i) every positive instance of $\mathfrak{L}_{\mathrm{L}}$ has a forward certificate, or (ii) every positive instance of $\mathfrak{L}_{\mathrm{R}}$ has a backward certificate.

Proof. Suppose not. Then there is $x \in L_{\mathrm{L}}$ with no forward certificate and $y \in L_{\mathrm{R}}$ with no backward certificate. As in the proof of Claim 1, this means that every α^D_x for $D \in \mathcal{A}$ permutes $Q^D_{\mathrm{LR}}(\vartheta)$ and every β^D_y for $D \in \mathcal{B}$ permutes $Q^D_{\mathrm{RL}}(\vartheta)$. Pick $k \geq 1$ so that each of these $|\mathcal{A}|+|\mathcal{B}|$ permutations becomes an identity after k iterations:

$$(\forall D \in \mathcal{A})[\ (\alpha^D_x)^k = \mathrm{id}\ ] \qquad \text{and} \qquad (\forall D \in \mathcal{B})[\ (\beta^D_y)^k = \mathrm{id}\ ]\,,$$

where 'id' is the identity function on the appropriate domain. Then, by Fact 3,

$$(\forall D \in \mathcal{A})[\ \alpha^D_{x^{(k)}} = \mathrm{id}\ ] \qquad \text{and} \qquad (\forall D \in \mathcal{B})[\ \beta^D_{y^{(k)}} = \mathrm{id}\ ]\,. \tag{6}$$

Intuitively, this means that no $D \in \mathcal{A}$ can distinguish $\vartheta x^{(k)}\vartheta$ from ϑ, and no $D \in \mathcal{B}$ can distinguish $\vartheta y^{(k)}\vartheta$ from ϑ. Hence, M cannot distinguish between

$$\vartheta x^{(k)}\vartheta y^{(k)}\vartheta \qquad \text{and} \qquad \vartheta y^{(k)}\vartheta x^{(k)}\vartheta\,, \tag{7}$$

because they both 'look' like $\vartheta y^{(k)}\vartheta$ to every $D \in \mathcal{A}$, and like $\vartheta x^{(k)}\vartheta$ to every $D \in \mathcal{B}$. If this intuition is correct, then M treats identically a positive (on the left) and a negative (on the right) instance of $\mathfrak{L}_{\mathrm{L}}{<}\mathfrak{L}_{\mathrm{R}}$—a contradiction.

Indeed, the inner behavior of every $D \in \mathcal{A}$ on the two instances of (7) is:

$$\begin{aligned}\alpha^D_{x^{(k)}\vartheta y^{(k)}} &= \alpha^D_{x^{(k)}} \circ \alpha^D_{y^{(k)}} = \mathrm{id} \circ \alpha^D_{y^{(k)}} = \alpha^D_{y^{(k)}}\\ \alpha^D_{y^{(k)}\vartheta x^{(k)}} &= \alpha^D_{y^{(k)}} \circ \alpha^D_{x^{(k)}} = \alpha^D_{y^{(k)}} \circ \mathrm{id} = \alpha^D_{y^{(k)}}\,,\end{aligned}$$

where in each line all functions are partial from $Q^D_{\mathrm{LR}}(\vartheta)$ to itself, the first step uses Fact 3, and the second step uses (6). Hence, $\alpha^D_{x^{(k)}\vartheta y^{(k)}} = \alpha^D_{y^{(k)}} = \alpha^D_{y^{(k)}\vartheta x^{(k)}}$. By this and a symmetric argument for every $D \in \mathcal{B}$, we eventually conclude that

$$\left.\begin{array}{l}\big(\ (\alpha^D_{x^{(k)}\vartheta y^{(k)}})_{D\in\mathcal{A}},\ (\beta^D_{x^{(k)}\vartheta y^{(k)}})_{D\in\mathcal{B}}\ \big)\\ \big(\ (\alpha^D_{y^{(k)}\vartheta x^{(k)}})_{D\in\mathcal{A}},\ (\beta^D_{y^{(k)}\vartheta x^{(k)}})_{D\in\mathcal{B}}\ \big)\end{array}\right\} = \big(\ (\alpha^D_{y^{(k)}})_{D\in\mathcal{A}},\ (\beta^D_{x^{(k)}})_{D\in\mathcal{B}}\ \big)\,.$$

Hence, M treats the blocks of (7) the same (Lemma 1b), as expected. ⊡

Now, if Claim 2i is true, then along with Claim 1 it implies a criterion for $\mathfrak{L}_{\text{L}}$: *an instance of* $\mathfrak{L}_{\text{L}}$ *is positive iff it has a forward certificate*. We thus get:

Claim 3. Some $\cup_{\text{L}}$1DFA with $1+\binom{s}{2}$-state components solves $\mathfrak{L}_{\text{L}}$.

Proof. By the criterion, an instance x of $\mathfrak{L}_{\text{L}}$ is positive iff there is $D \in \mathcal{A}$ and distinct $p, q \in Q^{D}_{\text{LR}}(\vartheta)$ such that *either* one of $\text{LCOMP}_{D,p}(x\vartheta)$ or $\text{LCOMP}_{D,q}(x\vartheta)$ hangs *or* both hit right into the same state. A $\cup_{\text{L}}$1DFA can check this using a $1+\binom{s}{2}$-state component $D_{p,q}$ for every such combination of D and p, q. ⊡

If Claim 2ii holds, we work similarly with $\mathfrak{L}_{\text{R}}$ and backward certificates. □

Lemma 6*. *If no* P$_2$1DFA *with s-state components solves $\mathfrak{L}$, then no s-state* 2DFA *with $<r$ inner reversals solves $\bigwedge_r \mathfrak{L}$.*

Proof. Let $\mathfrak{L} = (L, \tilde{L})$. Let M be a 2DFA with $<r$ inner reversals for $\bigwedge_r \mathfrak{L}$, with set of states $Q = [s]$. We build a P$_2$1DFA M' with s-state components for $\mathfrak{L}$.

We use *certificates* as in Lemma 6. For each $x \in L$, $c := \text{COMP}_M(\#(x\#)^r)$ is accepting and avoids reversals on one or more copies of x (since every reversal on a copy of x is inner). So, the crossing sequences $\overline{p}_x,\overline{q}_x$ on the outer boundaries of the leftmost such copy are again 'linked' by t one-way traversals (Fig. 1b). This time, however, it is not guaranteed that $t \le r$, as some pairs of successive traversals may be separated by *outer* reversals, whose number is not bounded by r. We just know that $t \le 2s$ (or else c would repeat a state on an outer cell of x, and loop), so the set $\mathcal{C} := \{(\overline{p}_x, \overline{q}_x) \mid x \in L\}$ of candidate certificates may be exponentially large, forbidding an exhaustive search by a small SDFA.

However, a small-component P$_2$1DFA can delegate this exhaustive search to its set of accepting tuples. So, we let $M' := (\{A_p \mid p \in Q\}, \{B_p \mid p \in Q\}, F)$. Each 1DFA A_p simulates M from p for as long as it moves right; if M ever attempts to reverse, A_p hangs. Similarly, each B_p simulates M from p for as long as it moves left, and hangs at any attempt to reverse. Hence, on input x, M' simulates M from every state and in every fixed direction, covering every possible one-way traversal of x by M. In the end, it checks whether x has a certificate by comparing the results of these $2s$ computations against each $(\overline{p}, \overline{q}) \in \mathcal{C}$. Formally, for each $\overline{p} = (p_1, \dots, p_t)$ and $\overline{q} = (q_1, \dots, q_t)$ we let $F_{(\overline{p},\overline{q})}$ be the set of all $2s$-tuples that we can build from two copies of all states in $Q = \{0, 1, \dots, s-1\}$

$$(\ 0, 1, \dots, s-1, \ \ 0, 1, \dots, s-1 \ ),$$

by replacing (i) every odd-indexed p_i in the left copy with the respective q_i (to ask A_{p_i} to hit right into q_i); (ii) every even-indexed q_i in the right copy with the respective p_i (to ask B_{q_i} to hit left into p_i) and (iii) all other states in either copy with any result in $Q_\perp$ (to let all other 1DFAs free). This way, $F_{(\overline{p},\overline{q})}$ is all tuples which prove that $(\overline{p}, \overline{q})$ is a certificate. So, letting $F := \bigcup_{(\overline{p},\overline{q}) \in \mathcal{C}} F_{(\overline{p},\overline{q})}$, we ensure that M' accepts x iff x has a certificate, namely iff $x \in L$. □

We are now ready to proceed to the main argument that proves Theorem 3.

For the lower bound, we start as in [4, §7.3]. We know no (2^h-2)-state 1DFA solves $\neg\mathfrak{M}^{\mathsf{R}}$. So, Lemma 3 implies no $\cap_{\mathrm{L}}$1DFA with (2^h-2)-state components solves $\bigvee\neg\mathfrak{M}^{\mathsf{R}}$. Hence, Lemma 2b for $\bigvee\neg\mathfrak{M}^{\mathsf{R}} = \neg\bigwedge\mathfrak{M}^{\mathsf{R}}$ implies that

$$\text{no } \cup_{\mathrm{L}}\text{1DFA with } (2^h-3)\text{-state components solves } \bigwedge\mathfrak{M}^{\mathsf{R}}.$$

This, together with Lemma 2a for $\bigwedge\mathfrak{M}^{\mathsf{R}} = (\bigwedge\mathfrak{M})^{\mathsf{R}}$, implies that

$$\text{no } \cup_{\mathrm{R}}\text{1DFA with } (2^h-3)\text{-state components solves } \bigwedge\mathfrak{M}.$$

So, by Lemma 4*, in every $\mathrm{P}_2$1DFA for $\bigwedge\mathfrak{M}^{\mathsf{R}}{<}\bigwedge\mathfrak{M}$ some component has $\Omega(2^{h/2})$ states. By Lemma 6*, the same holds for all 2DFAs with $<r$ inner reversals for $\mathfrak{R}_r$.

For the upper bound, we note that our 2DFA R_r from Sect. 4 performs only inner reversals. Moreover, its size can stay independent of r, if we allow ≤ 1 outer reversal: for odd r, we modify R_r to work as if r were even; this causes 1 outer reversal during the check of y_r. So, the modified R_r solves $\mathfrak{R}_r$ with $O(h)$ states, r inner reversals, and 0 or 1 outer reversals (depending on the parity of r).

6 Conclusion

We studied 2DFAs with *few*, *bounded*, and *fixed* reversals ($o(n)$, $O(1)$, r, respectively). We showed that the first two are actually the same, whereas small 2DFAs of the last kind strictly increase their power with every additional reversal, even if we focus only on those performed strictly between the end-markers.

It would have been nice if we had also resolved $\mathsf{2D}[\mathsf{const}] \subseteq \mathsf{2D}[O(1)]$. It would also be interesting to repeat this analysis for 2NFAs [3, Research Problem 4].

References

1. M. Balcerzak and D. Niwiński. Two-way deterministic automata with two reversals are exponentially more succinct than with one reversal. *Information Processing Letters*, 110:396–398, 2010.
2. V. Geffert, C. Mereghetti, and G. Pighizzini. Complementing two-way finite automata. *Information and Computation*, 205(8):1173–1187, 2007.
3. J. Hromkovič. Descriptional complexity of finite automata: concepts and open problems. *Journal of Automata, Languages and Combinatorics*, 7(4):519–531, 2002.
4. C. Kapoutsis. Nondeterminism is essential in small two-way finite automata with few reversals. *Information and Computation*. To appear.
5. C. Kapoutsis. Deterministic moles cannot solve liveness. *Journal of Automata, Languages and Combinatorics*, 12(1-2):215–235, 2007.
6. C. Kapoutsis, R. Královič, and T. Mömke. Size complexity of rotating and sweeping automata. *Journal of Computer and System Sciences*, 78(2):537–558, 2012.
7. A. R. Meyer and M. J. Fischer. Economy of description by automata, grammars, and formal systems. In *Proceedings of FOCS*, pages 188–191, 1971.
8. W. J. Sakoda and M. Sipser. Nondeterminism and the size of two-way finite automata. In *Proceedings of STOC*, pages 275–286, 1978.
9. M. Sipser. Halting space-bounded computations. *Theoretical Computer Science*, 10:335–338, 1980.
10. M. Sipser. Lower bounds on the size of sweeping automata. *Journal of Computer and System Sciences*, 21(2):195–202, 1980.