Small Sweeping 2NFAs Are Not Closed under Complement

Christos A. Kapoutsis

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
cak@mit.edu

Abstract. A two-way nondeterministic finite automaton is *sweeping* (SNFA) if its input head can change direction only on the end-markers. For every n, we exhibit a language that can be recognized by an n-state SNFA but requires $2^{\Omega(n)}$ states on every SNFA recognizing its complement.

1 Introduction

Understanding the power of nondeterminism is one of the most important goals of the theory of computation. In the past four decades, huge efforts have been invested into problems like P vs. NP and L vs. NL, with limited success. To some, this is creating the suspicion that essentially the same elusive idea lies at the core of all problems of this kind, little affected by the particulars of the underlying computational model or resource.

In this context, a possibly advantageous approach is to focus on weak models of computation. Provided that they are also powerful enough to be relevant, such models allow us to meaningfully study the power of nondeterministic algorithms in a much simpler setting, closer to the set-theoretic objects produced by their computations and in some distance from our often misleading algorithmic intuitions about how these computations may behave.

One such model is the two-way finite automaton. The question whether nondeterminism strictly increases its power, in the sense that it allows exponential economy in the number of states, was raised by Seiferas [1] in the early 70's. Now known as the 2D vs. 2N question, it was reduced by Sakoda and Sipser [2] to the study of certain complete problems and remains essentially as wide open as its famous counterparts above. The conjecture is that indeed 2D≠2N, and its more precise variants are quite surprising—see [3] for a brief history and discussion.

Given that small two-way deterministic finite automata (2DFAs) are closed under complement [4, 5], one way to confirm the conjecture is by proving that this closure fails in the nondeterministic case (2NFAs). In this track, Geffert, Mereghetti and Pighizzini [5] have recently studied the special case of small *unary* 2NFAs, but concluded that these *are* in fact closed under complement.

Following the same track, we study a different special case. We focus on *sweeping* 2NFAs (SNFAs), which are 2NFAs that can change the direction of their input head only on the end-markers. We prove that small SNFAs are *not* closed under complement—reaffirming, in a sense, the promise of the general direction.

The sweeping restriction was originally introduced by Sipser [6], in the first major step towards the conjecture, where he showed that no small SDFA can solve *liveness*—a problem that even small one-way nondeterministic finite automata (1NFAs) can solve. Indeed, our proof has the structure of that argument: we show that *no small* SNFA *can solve the complement of liveness.* Note that this was already known for 1NFAs (by a relatively simple argument of [2]) and SDFAs (by a combination of the arguments of [6] and [4]), so our theorem can be seen as a generalization of those facts to sweeping bidirectionality and to nondeterminism, respectively. In fact, this generalization was already asked for in [6].

2 Preliminaries and Outline

We write $[n]$ for the set $\{1, 2, \ldots, n\}$. If Σ is an alphabet, Σ^* is the set of all finite strings over Σ. If z is a string, then $|z|$, z_t, and z^t are its length, t-th symbol, and t-fold concatenation with itself. A property $P \subseteq \Sigma^*$ is *infinitely right-extensible* if every string in P has a right extension in P: $(\forall y \in P)(\exists z)(|z| \neq 0 \;\&\; yz \in P)$; *infinitely left-extensible* properties are defined symmetrically.

2.1 Sets, Functions, and Relations

If U is a set, then $\overline{U}$, $|U|$, $\mathcal{P}(U)$, and U^2 denote its complement, size, powerset, and set of pairs. The following simple lemma plays a central role in our proof.

Lemma 1. *Let $(u_i)_{i\in I}$ and $(v_i)_{i\in I}$ be two sequences of subsets of a set U, where I is a set of indices totally ordered by $<$. If for all $i', i \in I$ we have*

$$i' < i \implies u_{i'} \cap v_i = \emptyset \qquad \text{and} \qquad i' = i \implies u_{i'} \cap v_i \neq \emptyset,$$

then $|I| \leq |U|$.

Proof. For each $i \in I$, let a_i be any element of the non-empty intersection $u_i \cap v_i$. If the list $(a_i)_{i\in I}$ contains a repetition, say $a_{i'} = a_i =: a$ for two indices $i' < i$, then $a = a_{i'} \in u_{i'}$ and $a = a_i \in v_i$; hence $a \in u_{i'} \cap v_i$, a contradiction. Therefore the list $(a_i)_{i\in I}$ contains $|I|$ distinct elements of U. Hence, $|I| \leq |U|$.

Let $V \subseteq \mathcal{P}(U)$ be a set of points in the lattice of subsets of U. For $u \in V$, the part of V below u is $V_u := \{u' \in V \mid u' \subseteq u\}$; the *height* $h_V(u)$ of u in V is the length of the longest chain $\emptyset \neq u_1 \subsetneq \cdots \subsetneq u_k$ in V_u. For $f : V \to V$, we say f is *monotone* if it respects inclusion: $u' \subseteq u \implies f(u') \subseteq f(u)$; we say f is an *automorphism* if its restriction to V_u is a bijection from V_u to $V_{f(u)}$, for all u. Clearly, every automorphism respects heights: $h_V(u) = h_V\big(f(u)\big)$, for all u. By f^t we mean the t-fold composition of f with itself; if $t = 0$, this is the identity.

Lemma 2. *Suppose $f : V \to V$, where $V \subseteq \mathcal{P}(U)$ is a* finite *set of points from the lattice of a set U. If f is* injective *and* monotone*, then it is an automorphism.*

Proof. Pick any $u \in V$, set $v := f(u)$, and let f_u be the restriction of f to V_u. We will show f_u is a bijection from V_u to V_v. Since f is monotone, f_u has all its values in V_v: $u' \in V_u \implies u' \subseteq u \implies f(u') \subseteq f(u) \implies f_u(u') \in V_v$. Since f is injective, so is f_u. So, f_u is an injection from V_u to V_v. To show that it is a bijection, it is sufficient to show that V_v does not have more elements than V_u.

Since f is injective and V is finite, f is a permutation of V. Hence, for some $t \geq 1$, f^t is the identity. Let $f' := f^{t-1}$. Since f is injective and monotone, f' is also injective and monotone. Moreover, $u = f^t(u) = f^{t-1}\big(f(u)\big) = f'(v)$. Now the same argument as in the previous paragraph shows that the restriction f'_v of f' to V_v is an injection from V_v to V_u. Consequently, $|V_v| \leq |V_u|$.

Let $R \subseteq U^2$ be a binary relation. We write $R(\cdot)$ for the mapping of each $u \subseteq U$ to the set $R(u) := \{b \in U \mid (\exists a \in u)(aRb)\}$ of all elements related to elements of u; we usually write $R(a)$ for $R(\{a\})$. Clearly, $R(\cdot)$ is monotone. If $R' \subseteq U^2$ is also a binary relation, we write $R' \circ R$ for the composition: $a(R' \circ R)b \iff (\exists c \in U)(aR'c \ \& \ cRb)$. Clearly, $(R' \circ R)(u) = R\big(R'(u)\big)$, for all u.

A total order $<$ on $\mathcal{P}(U)^2$ is *nice* if each pair "escapes" from every strictly smaller pair in at least one component: $(u', v') < (u, v) \implies u' \not\supseteq u \ \vee \ v' \not\supseteq v$. It is not hard to verify that nice orders on $\mathcal{P}(U)^2$ exist, for every finite U.

2.2 Sweeping Automata and Liveness

A *sweeping deterministic finite automaton* (SDFA, [6]) is a triple $M = (q_s, \delta, q_f)$, where δ is the *transition function*, partially mapping $Q \times (\Sigma \cup \{\Box\})$ to Q, for some set Q of *states*, some *alphabet* Σ, and some *end-marker* $\Box \notin \Sigma$, while q_s and q_f are the *start* and *final* states. An input $z \in \Sigma^*$ is presented to M between two copies of $\Box$. The computation starts at q_s, on the symbol to the right of the left copy of $\Box$, heading rightward. The next state is always derived from δ and the current state and symbol. The next position is always the adjacent one in the direction of motion; except when the current symbol is $\Box$ and the next state is not q_f, in which case the next position is the adjacent one in the opposite direction. Note that the computation can either loop, or hang, or fall off the string $\Box z \Box$ into q_f. In this last case we say that *M accepts z*.

More generally, for any $z \in \Sigma^*$ and $p \in Q$, the *left computation of M from p on z* is the unique sequence

$$\text{LCOMP}_{M,p}(z) := (q_t)_{1 \leq t \leq m}$$

where $q_1 = p$; every next state is $q_{t+1} = \delta(q_t, z_t)$, provided that $t \leq |z|$ and the value of δ is defined; and m is the first t for which this last provision fails. If $m = |z| + 1$, the computation *exits into q_m*; otherwise, $1 \leq m \leq |z|$ and the computation *hangs at q_m*. The *right computation of M from p on z*, $\text{RCOMP}_{M,p}(z) := \big(q_t\big)_{1 \leq t \leq m}$, is defined symmetrically, with $q_{t+1} = \delta(q_t, z_{|z|+1-t})$.

If M is allowed more than one next move at each step, we say that it is *nondeterministic* (SNFA). Formally, this means that δ *totally* maps $Q \times (\Sigma \cup \{\Box\})$ to the *powerset* of Q and implies that, on any $z \in \Sigma^*$, M exhibits a *set* of computations. If at least one of them falls off $\Box z \Box$ into q_f, then M accepts z.

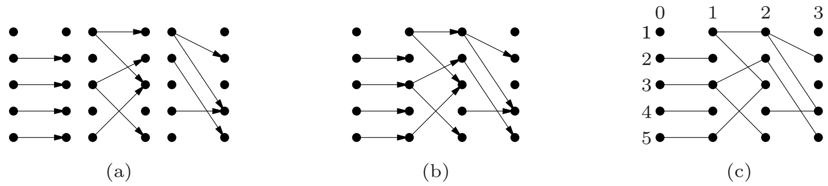


Fig. 1. (a) Three symbols in Σ_5; e.g., the third symbol is $\{(1,2),(1,4),(2,5),(4,4)\}$. (b) The string defined by them. (c) The string simplified and indexed; here $\xi = \{(3,5)\}$.

Similarly, $\text{LCOMP}_{M,p}(z)$ is now a *set* of computations. To encode how states connect via left computations, we define the binary relation $\text{LVIEW}_M(z) \subseteq Q^2$

$$(p,q) \in \text{LVIEW}_M(z) \iff \big(\exists c \in \text{LCOMP}_{M,p}(z)\big)(c \text{ exits into } q),$$

and call it the *left behavior of* M *on* z. Then, for $u \subseteq Q$, the set $\text{LVIEW}_M(z)(u)$ of states reachable via left computations from within u is the *left view of* u *on* z. The *right behavior* $\text{RVIEW}_M(z)$ *of* M *on* z and the *right view* $\text{RVIEW}_M(z)(u)$ *of* u *on* z are defined similarly. Note that, if $|z| = 1$, the automaton has the same behavior in both directions: $\text{LVIEW}_M(z) = \text{RVIEW}_M(z) = \{(p,q) \mid \delta(p,z) \ni q\}$. Also, if extending z does not cause a view to include any new states, then this remains true on all identical further extensions, as described in the next lemma.

Lemma 3. *The following implications are true, for all* $t \geq 1$:

- $\text{LVIEW}_M(z)(u) \supseteq \text{LVIEW}_M(z\tilde{z})(u) \implies \text{LVIEW}_M(z)(u) \supseteq \text{LVIEW}_M(z\tilde{z}^t)(u)$,
- $\text{RVIEW}_M(z)(u) \supseteq \text{RVIEW}_M(\tilde{z}z)(u) \implies \text{RVIEW}_M(z)(u) \supseteq \text{RVIEW}_M(\tilde{z}^t z)(u)$.

Liveness. For $n \geq 1$, we consider the alphabet $\Sigma_n := \mathcal{P}([n]^2)$ of all directed 2-column graphs with n nodes per column and only rightward arrows (Fig. 1a). An m-long string over Σ_n is naturally viewed as a directed $(m+1)$-column graph (Fig. 1b), in which for simplicity we often omit the direction of the arrows (Fig. 1c). We say that the string has *connectivity* $\xi \subseteq [n]^2$ if ξ correctly describes all connections between the outer columns: $(a,b) \in \xi$ iff there exists an m-long path from the a-th node of the 0-th column to the b-th node of the m-th column. We write $B_{n,\xi}$ for the set of all strings of connectivity ξ. The strings of $B_{n,\emptyset}$ are called *dead*; all other strings are called *live*. We define $B_n := \overline{B_{n,\emptyset}}$ as the collection of all live strings. So, B_n is the property of *liveness* —as defined in [2].

2.3 Outline

It is easy to see that B_n can be recognized by a SNFA (a 1NFA, actually) with only n states. Our goal is to prove that, in contrast, for the complementary language $\overline{B_n} = B_{n,\emptyset}$ a SNFA would need exponentially many states.

Theorem 1. *Every* SNFA *that recognizes* $B_{n,\emptyset}$ *has* $2^{\Omega(n)}$ *states.*

The rest of the article proves this fact. We fix n and a SNFA $M = (q_s, \delta, q_f)$ over a set Q of k states that recognizes $B_{n,\emptyset}$. We will prove that $k = 2^{\Omega(n)}$.

The proof is based on Lemma 1. We build two sequences $(X_\iota)_{\iota\in\mathcal{I}}$ and $(Y_\iota)_{\iota\in\mathcal{I}}$ that are related as in the lemma. The indices are all pairs of non-empty subsets of $[n]$, the universe is all sets of 1 or 2 steps of M:[1]

$$\mathcal{I} := \{(\alpha, \beta) \mid \emptyset \neq \alpha, \beta \subseteq [n]\} \qquad\qquad \mathcal{S} := \big\{\{s', s\} \mid s', s \in Q^2\big\},$$

and the total order $<$ is the restriction on $\mathcal{I}$ of some *nice* order on $\mathcal{P}([n])^2$. If we indeed construct these sequences, then the lemma says $|\mathcal{I}| \leq |\mathcal{S}|$, therefore

$$(2^n - 1)^2 \leq k^2 + \tbinom{k^2}{2},$$

hence $k = 2^{\Omega(n)}$. For the remainder, we fix $\mathcal{I}$ and $\mathcal{S}$ as here.

Note that from now on some subscripts in our notation are redundant. We thus drop them: e.g., $B_{n,\emptyset}$ and $\text{LVIEW}_M(z)(u)$ become $B_\emptyset$ and $\text{LVIEW}(z)(u)$.

Also, before moving on, let us prove a fact that will be useful later: In order to accept a dead string but reject a live one, M must produce on the dead string a single-state view that "escapes" the corresponding view on the live string.

Lemma 4. *Let z' be live and z dead. Then at least one of the following is true*:
- $\text{LVIEW}(z')(p) \not\supseteq \text{LVIEW}(z)(p)$ *for some* $p \in Q$.
- $\text{RVIEW}(z')(p) \not\supseteq \text{RVIEW}(z)(p)$ *for some* $p \in Q$.

Proof. Suppose $\text{LVIEW}(z')(p) \supseteq \text{LVIEW}(z)(p)$ and $\text{RVIEW}(z')(p) \supseteq \text{RVIEW}(z)(p)$, for all p. Pick any *accepting* computation c of M on z. Break c into its *traversals* $c_1, \ldots, c_m$, in the natural way: for $j < m$, each c_j starts at some state p_j next to a $\Box$ and ends at some state q_j on the other $\Box$; $p_1 = q_s$; $\delta(q_j, \Box) \ni p_{j+1}$; and $c_m = (q_f)$. Then, for each odd (resp., even) $j < m$, we know q_j is in $\text{LVIEW}(z)(p_j)$ (resp., in $\text{RVIEW}(z)(p_j)$) and thus also in $\text{LVIEW}(z')(p_j)$ (resp., $\text{RVIEW}(z')(p_j)$); hence, some computation c'_j of M on z' starts and ends identically to c_j. If we also set $c'_m := (q_f)$ and concatenate $c'_1, \ldots, c'_m$, we end up with a computation c' of M on z' which is also accepting. So, M accepts z', a contradiction.

3 Hard Inputs and the Two Sequences

3.1 Generic Strings

Consider any $y \in \Sigma^*$ and the set of views produced via left computations on it:

$$\text{LVIEWS}(y) := \{\text{LVIEW}(y)(u) \mid u \subseteq Q\},$$

i.e., the range of $\text{LVIEW}(y)(\cdot)$. How does this set change if we extend y into yz?

Let $\text{LMAP}(y, z)$ be the function that for every left view produced on y returns its left view on z —i.e., $\text{LMAP}(y, z)$ simpy restricts $\text{LVIEW}(z)(\cdot)$ to $\text{LVIEWS}(y)$. It

[1] A *step* of M is any $s \in Q^2$. Also, note that $\{s', s\}$ represents a singleton when $s' = s$.

is easy to verify that $\text{LVIEWS}(yz)$ contains all values of this function, and is covered by them. In other words, $\text{LMAP}(y, z)$ is a *surjection* from $\text{LVIEWS}(y)$ to $\text{LVIEWS}(yz)$. This immediately implies that $|\text{LVIEWS}(y)| \geq |\text{LVIEWS}(yz)|$.

The next fact encodes this conclusion, along with the obvious remark that $\text{LMAP}(y, z)$ is monotone. It also shows the symmetric facts, for left extensions and right views. The set $\text{RVIEWS}(y)$ consists of all views produced on y via right computations, and $\text{RMAP}(z, y)$ is the restriction of $\text{RVIEW}(z)(\cdot)$ on $\text{RVIEWS}(y)$.

Fact 1. *For all* y, z*:* $\text{LMAP}(y, z)$ *monotonically surjects* $\text{LVIEWS}(y)$ *to* $\text{LVIEWS}(yz)$*, so* $|\text{LVIEWS}(y)| \geq |\text{LVIEWS}(yz)|$*; symmetrically, in the other direction,* $\text{RMAP}(z, y)$ *monotonically surjects* $\text{RVIEWS}(y)$ *to* $\text{RVIEWS}(zy)$*, so* $|\text{RVIEWS}(y)| \geq |\text{RVIEWS}(zy)|$.

Now suppose y belongs to an infinitely right-extensible property $P \subseteq \Sigma^*$. What happens to the size of $\text{LVIEWS}(y)$ if we keep extending y into $yz, yzz', \ldots$ inside P? Although there are infinitely many extensions, the size of the set can decrease only finitely many times. So, at some point it must stop changing. When this happens, we have arrived at a very useful tool. We define it as follows.

Definition 1. *Let* $P \subseteq \Sigma^*$*. A string* y *is* L-generic over P *if* $y \in P$ *and*

$$(\forall yz \in P)\big[|\text{LVIEWS}(y)| = |\text{LVIEWS}(yz)|\big].$$

An R-generic *string over* P *is defined symmetrically, with left-extensions and* $\text{RVIEWS}(\cdot)$*. A string that is both* L*-generic and* R*-generic over* P *is called* generic.

Lemma 5. *Let* $P \subseteq \Sigma^*$*. If* P *is non-empty and infinitely right-extensible (resp., left-extensible), then there exist* L*-generic (resp.,* R*-generic) strings over* P*. If* y_{L} *is* L*-generic and* y_{R} *is* R*-generic, then every string* $y_{\text{L}} x y_{\text{R}} \in P$ *is generic.*

Proof. For the last claim, we just note that all right-extensions of an L-generic string inside P are also L-generic, and the same is true in the other direction.

Generic strings were introduced in [6] (for SDFAs and over B_n). Intuitively, they are among the *richest* strings with property P, in the sense that they exhibit a greatest subset of the "features" that M is "prepared to pay attention to". This makes them useful in building hard inputs, as described in the next lemma and in Sect. 3.2. For the lemma, we will also need the following simple fact.

Fact 2. *For all* y, z*:* $\text{LVIEWS}(yz) \subseteq \text{LVIEWS}(z)$ *and* $\text{RVIEWS}(zy) \subseteq \text{RVIEWS}(z)$.

Proof. By Fact 1, $\text{LVIEWS}(yz)$ is the range of $\text{LMAP}(y, z)$, which is a restriction of $\text{LVIEW}(z)(\cdot)$; so, the first containment follows. Similarly in the other direction.

Lemma 6. *Suppose* y *is generic over* $P \subseteq \Sigma^*$*, and* $x \in \Sigma^*$*. If* $yxy \in P$*, then*

- $\text{LMAP}(y, xy)$ *is an automorphism on* $\text{LVIEWS}(y)$*, and*
- $\text{RMAP}(yx, y)$ *is an automorphism on* $\text{RVIEWS}(y)$.

Proof. Suppose $yxy \in P$. Then $|\text{LVIEWS}(y)| = |\text{LVIEWS}(yxy)|$ (since y is generic) and $\text{LVIEWS}(yxy) \subseteq \text{LVIEWS}(y)$ (by Fact 2). Hence, $\text{LVIEWS}(y) = \text{LVIEWS}(yxy)$. By this and Fact 1, we conclude $\text{LMAP}(y, xy)$ surjects $\text{LVIEWS}(y)$ onto itself, which is possible only if it is injective. Since $\text{LMAP}(y, xy)$ is also monotone, Lemma 2 implies it is an automorphism. The fact about $\text{RMAP}(yx, y)$ is proved similarly.

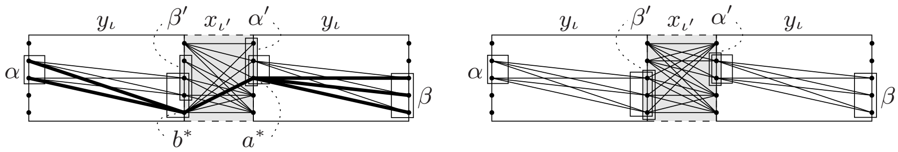
3.2 Constructing the Hard Inputs

Fix $\iota = (\alpha, \beta) \in \mathcal{I}$ and let $P_\iota := B_{\alpha\times\beta}$ be the property of connecting exactly every leftmost node in α to every rightmost node in β. Easily, P_ι is non-empty and infinitely extensible in both directions. So, an L-generic string y_{L} and an R-generic string y_{R} exist (Lemma 5). Then, for $\eta = [n]^2$ the complete symbol, we easily see that $y_{\text{L}}\eta y_{\text{R}} \in P_\iota$, too. Hence, this string is generic over P_ι (Lemma 5). We define $y_\iota := y_{\text{L}}\eta y_{\text{R}}$. We also define the symbol $x_\iota := \overline{\beta \times \alpha}$.

Lemma 7. *The two sequences $(y_\iota)_{\iota\in\mathcal{I}}$ and $(x_\iota)_{\iota\in\mathcal{I}}$ are such that, for all $\iota', \iota \in \mathcal{I}$:*

$$\iota' < \iota \implies y_\iota x_{\iota'} y_\iota \in P_\iota \qquad \text{and} \qquad \iota' = \iota \implies y_\iota x_{\iota'} y_\iota \in B_\emptyset.$$

Proof. Fix $\iota' = (\alpha', \beta')$ and $\iota = (\alpha, \beta)$ and let $z := y_\iota x_{\iota'} y_\iota$. Note that the connectivities of y_ι and $x_{\iota'}$ are respectively $\xi := \alpha \times \beta$ and $\xi' := \overline{\beta' \times \alpha'}$.



If $\iota' < \iota$ (on the left), then $\alpha' \not\supseteq \alpha$ or $\beta' \not\supseteq \beta$ (since $<$ is nice). Suppose $\beta' \not\supseteq \beta$ (if $\alpha' \not\supseteq \alpha$, use a similar argument) and fix any $b^* \in \beta \setminus \beta'$ and any $a^* \in \alpha$. For any $a, b \in [n]$, consider the a-th leftmost and b-th rightmost nodes of z. If $a \notin \alpha$ or $b \notin \beta$, then the two nodes do not connect in z, since neither can "see through" y_ι. If $a \in \alpha$ and $b \in \beta$, then $(a, b^*) \in \xi$ and $(b^*, a^*) \in \xi'$ and $(a^*, b) \in \xi$, so the two nodes connect via a path of the form $a \rightsquigarrow b^* \to a^* \rightsquigarrow b$. Overall, $z \in P_\iota$.

If $\iota' = \iota$ (on the right), then $\xi' = \overline{\beta \times \alpha}$. Suppose $z \notin B_\emptyset$. Then some path in z connects the leftmost to the rightmost column. Suppose it is of the form $a \rightsquigarrow b^* \to a^* \rightsquigarrow b$. Then $b^* \in \beta$ and $(b^*, a^*) \in \xi'$ and $a^* \in \alpha$, a contradiction.

3.3 Constructing the Two Sequences

Suppose $\iota' < \iota$. Since the extension $y_\iota x_{\iota'} y_\iota$ of y_ι preserves P_ι (Lemma 7), each of $\text{LMAP}(y_\iota, x_{\iota'} y_\iota)$ and $\text{RMAP}(y_\iota x_{\iota'}, y_\iota)$ is an automorphism (Lemma 6). Put another way, the interaction between the steps of M on $x_{\iota'}$ and its two behaviors on y_ι is such that these two mappings are automorphisms. Put formally, both

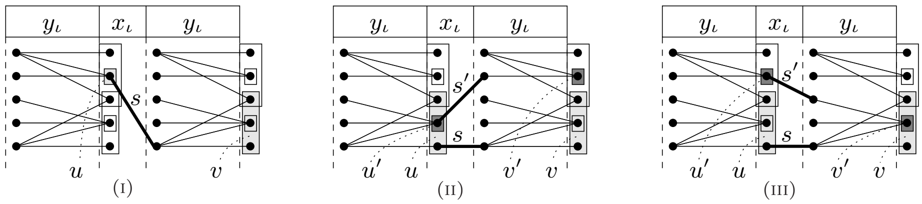
- the restriction of $\big(S_{\iota'} \circ \text{LVIEW}(y_\iota)\big)(\cdot)$ on $\text{LVIEWS}(y_\iota)$ and
- the restriction of $\big(S_{\iota'} \circ \text{RVIEW}(y_\iota)\big)(\cdot)$ on $\text{RVIEWS}(y_\iota)$

are automorphisms, for $S_{\iota'} := \{(p, q) \mid \delta(p, x_{\iota'}) \ni q\} = \text{LVIEW}(x_{\iota'}) = \text{RVIEW}(x_{\iota'})$.

What if $\iota' = \iota$? What is the status of $\text{LMAP}(y_\iota, x_\iota y_\iota)$ and $\text{RMAP}(y_\iota x_\iota, y_\iota)$? We can show that, since $y_\iota x_\iota y_\iota$ is dead (Lemma 7), we cannot have both functions be automorphisms.[2] However, something stronger is true: *we can even convince*

[2] If they were, they would be bijections (because each of $\text{LVIEWS}(y_\iota)$ and $\text{RVIEWS}(y_\iota)$ has a maximum). Hence, M would not be able to distinguish between the live y_ι and the dead $y_\iota(x_\iota y_\iota)^t$, for t any exponent that turns both bijections into identities. (Note that this is true even for the n-state SNFA that solves liveness. Therefore, this observation alone can give rise to no interesting lower bound for k.)

ourselves that one of the functions is not an automorphism by pointing at only 1 or 2 of the steps of M on x_ι. The next figure shows three examples of this. In each, we sketch the left behavior of M on y_ι and all single-state views, and consider all heights to be with respect to LVIEWS(y_ι).



Example I shows only 1 of the steps of M on x_ι, say $s = (p, q)$ —many more may be included in S_ι. Is LMAP$(y_\iota, x_\iota y_\iota)$ an automorphism? Normally, we would need to know the entire S_ι to answer this question. Yet, in this case s is enough to answer no. To see why, note that the view v of q on y_ι has height 2, while one of the views that contain p is u, of height 1. Irrespective of the rest of S_ι, LMAP$(y_\iota, x_\iota y_\iota)$ will map u to a view that contains v and thus has height 2 or more. So, it does not respect heights, which implies it is not an automorphism.

Example II shows 2 of the steps in S_ι, say $s' = (p', q')$ and $s = (p, q)$. Is LMAP$(y_\iota, x_\iota y_\iota)$ an automorphism? Observe that neither step alone can force a negative answer: the view v' of q' on y_ι has height 1, as does the lowest view u' containing p'; similarly for s, u, v, and height 2. Hence, individually each of s' and s may very well participate in sets of steps that induce automorphisms. Yet, they cannot belong to the same such set. To see why, suppose they do. Since $u' \subseteq u$, the image of u would be $v' \cup v$ or a superset. Since $v' \nsubseteq v$, the height of that image would be greater than the height of v, and thus greater than the height of u, violating the respect to heights.

Example III also shows 2 of the steps in S_ι, say $s' = (p', q')$ and $s = (p, q)$, neither of which can disqualify LMAP$(y_\iota, x_\iota y_\iota)$ from being an automorphism. Yet, together they can. To see why, suppose both steps participate in the same automorphism. Then the image of u' must be exactly v': otherwise, it would be some strict superset of v', of height 2 or more, disrespecting the height of u'. On the other hand, u must map to a set that contains v, and thus also v'. Hence, v' must be the exact image of some $u^* \subseteq u$. But then both u^* and u' map to v', when $u^* \neq u'$ (since $u' \nsubseteq u$), a contradiction to the map being injective.

In short, each step in S_ι severely restricts the form of LMAP$(y_\iota, x_\iota y_\iota)$ and RMAP$(y_\iota x_\iota, y_\iota)$. And, either individually or in pairs, some steps can be so restrictive that they cannot be part of any set of steps that induces an automorphism in both directions. To describe this formally, we introduce the next definition.

Definition 2. *A set of steps $S \subseteq Q^2$ is* compatible *with y_ι if there exists a set $\hat{S}$ such that $S \subseteq \hat{S} \subseteq Q^2$ and the following are both automorphisms:*

- *the restriction of $\big(\hat{S} \circ \text{LVIEW}(y_\iota)\big)(\cdot)$ on* LVIEWS(y_ι)*, and*
- *the restriction of $\big(\hat{S} \circ \text{RVIEW}(y_\iota)\big)(\cdot)$ on* RVIEWS(y_ι)*.*

E.g., $\{s\}$ in Example I and $\{s', s\}$ in Examples II,III are incompatible with y_ι.

We are now ready to define the sequences promised in Sect. 2.3. For each $\iota \in \mathcal{I}$, we let X_ι consist of all sets of 1 or 2 steps of M on x_ι, and Y_ι consist of all sets of 1 or 2 steps of M that are incompatible with y_ι:

$$X_\iota := \big\{S \in \mathcal{S} \mid S \subseteq S_\iota\big\}, \qquad Y_\iota := \big\{S \in \mathcal{S} \mid S \text{ is incompatible with } y_\iota\big\}.$$

We need, of course, to show that the sequences relate as in Lemma 1.

The case $\iota' < \iota$ is easy. Each $S \in X_{\iota'}$ can be extended to the set of all steps of M on $x_{\iota'}$ (i.e., $\hat{S} := S_{\iota'}$), which does induce automorphisms, so $X_{\iota'} \cap Y_\iota = \emptyset$.

The case $\iota' = \iota$ is harder. We analyze it in the next section.

4 The Main Argument

Suppose $\iota' = \iota$. Our goal is to exhibit a singleton or two-set $S \subseteq S_\iota$ that is incompatible with y_ι. First, some preparation.

The witness. Consider the strings $y_\iota(x_\iota y_\iota)^t = (y_\iota x_\iota)^t y_\iota$, for all $t \geq 1$. Since $y_\iota x_\iota y_\iota$ is dead, so are all of them. Since y_ι is live, Lemma 4 says for all $t \geq 1$:

- $\text{LVIEW}(y_\iota)(p) \not\supseteq \text{LVIEW}\big(y_\iota(x_\iota y_\iota)^t\big)(p)$ for some $p \in Q$, or
- $\text{RVIEW}(y_\iota)(p) \not\supseteq \text{RVIEW}\big((y_\iota x_\iota)^t y_\iota\big)(p)$ for some $p \in Q$.

Namely, in order to accept the extensions $y_\iota(x_\iota y_\iota)^t = (y_\iota x_\iota)^t y_\iota$ but reject the original y_ι, M must exhibit on each of them a single-state view that "escapes" its counterpart on the original. In a sense, among all $2k$ single-state views on each extension, the escaping one is a "witness" for the fact that the extension is accepted, and Lemma 4 says that *every extension has a witness.* Of course, this allows for the possibility that different extensions may have different witnesses. However, we can actually find the same witness for all extensions:

Fact 3. *At least one of the following is true*:

- $\text{LVIEW}(y_\iota)(p) \not\supseteq \text{LVIEW}\big(y_\iota(x_\iota y_\iota)^t\big)(p)$ *for some* $p \in Q$ *and all* $t \geq 1$.
- $\text{RVIEW}(y_\iota)(p) \not\supseteq \text{RVIEW}\big((y_\iota x_\iota)^t y_\iota\big)(p)$ *for some* $p \in Q$ *and all* $t \geq 1$.

Proof. Suppose neither is true. Then each of the $2k$ single-state views has an extension on which it fails to escape from its counterpart on y_ι. Namely, every p has some $t_{p,\text{L}} \geq 1$ such that $\text{LVIEW}(y_\iota)(p) \supseteq \text{LVIEW}\big(y_\iota(x_\iota y_\iota)^{t_{p,\text{L}}}\big)(p)$ and some $t_{p,\text{R}} \geq 1$ such that $\text{RVIEW}(y_\iota)(p) \supseteq \text{RVIEW}\big((y_\iota x_\iota)^{t_{p,\text{R}}} y_\iota\big)(p)$. Consider the exponent

$$t^* := \big(\textstyle\prod_{p \in Q} t_{p,\text{L}}\big) \cdot \big(\textstyle\prod_{p \in Q} t_{p,\text{R}}\big)$$

and the extension $z := y_\iota(x_\iota y_\iota)^{t^*} = (y_\iota x_\iota)^{t^*} y_\iota$. Then each p has some $t \geq 1$ such that $z = y_\iota((x_\iota y_\iota)^{t_{p,\text{L}}})^t$, and thus Lemma 3 implies $\text{LVIEW}(y_\iota)(p) \supseteq \text{LVIEW}(z)(p)$; similarly, $\text{RVIEW}(y_\iota)(p) \supseteq \text{RVIEW}(z)(p)$. Overall, all single-state views on z fall within their counterparts on y_ι, contradicting Lemma 4.

We fix p to be a witness as in Fact 3. We assume p is of the first type, involving left views (otherwise, a symmetric argument applies). Moreover, among all witnesses of this type, we select p so as to minimize the height of $\text{LVIEW}(y_\iota)(p)$ in $\text{LVIEWS}(y_\iota)$. We let $V := \text{LVIEWS}(y_\iota)$, $h := h_V$, and $v_0 := \text{LVIEW}(y_\iota)(p)$.

By the selection of p, no $\tilde{p}$ with $\textsc{lview}(y_\iota)(\tilde{p}) \subsetneq v_0$ can be a witness of the first type. Hence, for every such $\tilde{p}$ there is some $\tilde{t} \geq 1$ such that $\textsc{lview}(y_\iota)(\tilde{p}) \supseteq \textsc{lview}\big(y_\iota(x_\iota y_\iota)^{\tilde{t}}\big)(\tilde{p})$. We fix t^* to be the product of all such $\tilde{t}$. Then:

Fact 4. *For all such $\tilde{p}$ and all $\lambda \geq 1$: $\textsc{lview}(y_\iota)(\tilde{p}) \supseteq \textsc{lview}(y_\iota(x_\iota y_\iota)^{\lambda t^*})(\tilde{p})$.*

Proof. Fix such a $\tilde{p}$ and the $\tilde{t}$ for which $\textsc{lview}(y_\iota)(\tilde{p}) \supseteq \textsc{lview}\big(y_\iota(x_\iota y_\iota)^{\tilde{t}}\big)(\tilde{p})$. Fix any $\lambda \geq 1$. Then λt^* is a multiple of $\tilde{t}$ and Lemma 3 applies.

Escape computations. For all $t \geq 1$, collect into a set $\mathcal{C}_t$ all computations $c \in \textsc{lcomp}_p\big(y_\iota(x_\iota y_\iota)^t\big)$ that exit into some $q \notin v_0$. These are the *escape computations* for p on the t-th extension. We also define $\mathcal{C} := \cup_{t\geq 1}\mathcal{C}_t$.

Let us see how an escape computation looks like. Pick any $c \in \mathcal{C}$ (Fig. 2a), say on the t-th extension, exiting into q. Let $s_1, \ldots, s_t$ be the steps of c on x_ι, where $s_j = (p_j, q_j) \in S_\iota$. These are the *critical steps* along c. Let $v_j := \textsc{lview}(y_\iota)(q_j)$ be the view of the right end-point of s_j. Along with v_0, these views form the list $v_0, v_1, \ldots, v_t$ of the *major views* along c. Clearly, each of them contains the left end-point of the following critical step: $v_{j-1} \ni p_j$ (similarly, $v_t \ni q$). So, for each s_j there exist views $u \in V$ that contain its left end-point and are contained in the preceding major view: $v_{j-1} \supseteq u \ni p_j$ (similarly, $v_t \supseteq u \ni q$). Among them, let u_{j-1} be one of minimum height in V (select u_t similarly). Then the list $u_0, \ldots, u_{t-1}, u_t$ are the *minor views* along c.

We will find an incompatible S among the critical steps of such computations.

Case 1: Some $c \in \mathcal{C}$ contains some critical step s such that the singleton $\{s\}$ is incompatible with y_ι. Then we can select $S := \{s\}$, and we are done.

Case 2: For all $c \in \mathcal{C}$ and all critical steps s in c, the singleton $\{s\}$ is compatible with y_ι. In this case, we will find an incompatible two-set.

Steepness. First of all, every $c \in \mathcal{C}$ (say with t, s_j, v_j, u_j as above) has every major view at least as high as the next minor one ($h(v_j) \geq h(u_j)$, since $v_j \supseteq u_j$) and every minor view at least as high as the next major one ($h(u_j) \geq h(v_{j+1})$, otherwise $\{s_{j+1}\}$ would be incompatible, as in Example I). Hence, every $c \in \mathcal{C}$ has views of monotonically decreasing height ($h(v_0) \geq h(u_0) \geq h(v_1) \geq \cdots \geq h(u_t)$). To capture the "rate" of this decrease, we record the list of minor view heights $H_c := \big(h(u_j)\big)_{0\leq j\leq t}$, and order each $\mathcal{C}_t$ lexicographically: $c' \leq c$ iff $H_{c'} \leq_{\text{lex}} H_c$. With respect to this total order, "smaller" computation means "steeper".

Long and steepest computation. We fix t to be a multiple of t^* which is at least $|V|$, and select c to be steepest in $\mathcal{C}_t$. We let q, s_j, v_j, u_j be as usual.

Since $t \geq |V|$, the list $u_0, \ldots, u_t$ contains repetitions. Let $j' < j$ be the indices for the earliest one. Then $u_{j'} = u_j$, so $h(u_{j'}) = h(u_j)$, and thus all views in between have the same height: $h(u_{j'}) = h(v_{j'+1}) = \cdots = h(v_j) = h(u_j)$. As a result, each major view equals the next minor one: $v_{j'+1} = u_{j'+1}, \ldots, v_j = u_j$.

Case 2A: $j' = 0$. Then $h(u_0) = h(v_1) = \cdots = h(v_j) = h(u_j)$, and therefore $v_1 = u_1, \ldots, v_j = u_j$. In fact, we also have $h(v_0) = h(u_0)$, and therefore $v_0 = u_0$.

To see why, suppose $h(v_0) \neq h(u_0)$. Then $v_0 \supsetneq u_0$. Since $u_0 \in V$, some state $\tilde{p}$ has $\textsc{lview}(y_\iota)(\tilde{p}) = u_0$ (Fig. 2a), and thus Fact 4 applies to it (since $u_0 \subsetneq v_0$). In particular, $\textsc{lview}(y_\iota)(\tilde{p}) \supseteq \textsc{lview}\big(y_\iota(x_\iota y_\iota)^t\big)(\tilde{p})$ (since t is a multiple of t^*).

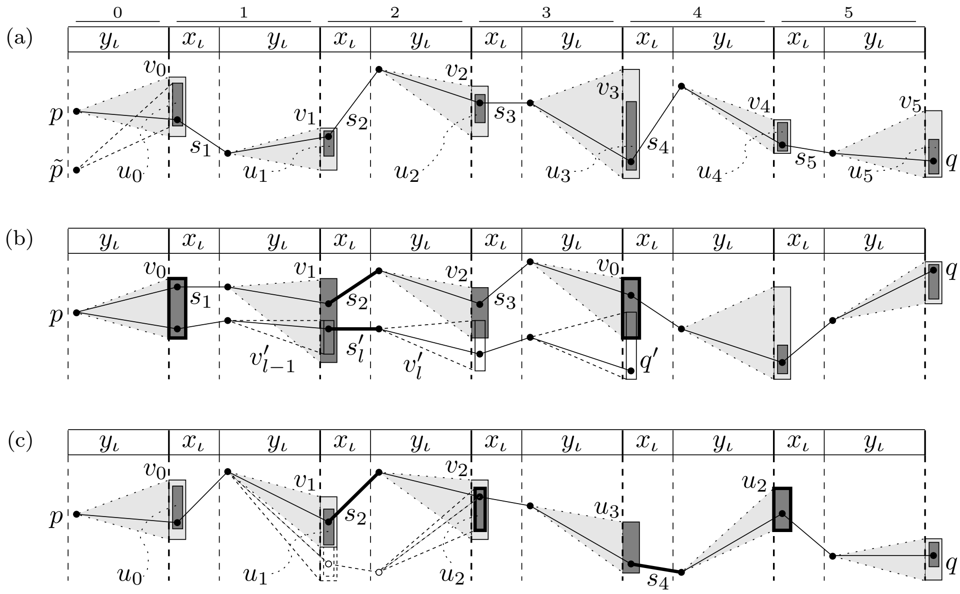


Fig. 2. (a) An escape computation $c \in \mathcal{C}_5$, exiting into q. (b) An example of Case 2A, for $j = 3$ and $l = 2$; in dashes, the new computation $c' \in \mathcal{C}_j$. (c) An example of Case 2B, for $j' = 2$ and $j = 4$; in dashes, the hypothetical case $u_{j'-1} \supseteq u_{j-1}$ and c'.

On the other hand, u_0 contains the left end-point of s_1, so the part of c after s_1 shows that $q \in \text{LVIEW}\big(y_\iota(x_\iota y_\iota)^t\big)(\tilde{p})$, and thus $q \in \text{LVIEW}(y_\iota)(\tilde{p}) = u_0$. Since $u_0 \subseteq v_0$, this means that c is not an escape computation, a contradiction.

So, $h(v_0) = h(u_0) = \cdots = h(v_j) = h(u_j)$ and $v_0 = u_0, \ldots, v_j = u_j$ (Fig. 2b). By the selection of p, its view on the j-th extension escapes v_0. Pick any $c' \in \mathcal{C}_j$, with exit state $q' \notin v_0$, critical steps $s'_1, \ldots, s'_j$, and major views $v'_0, \ldots, v'_j$. Then $v'_0 = v_0$ (since both c' and c start at p) and $q' \in v'_j \setminus v_j$ (since $v_j = u_j = u_0 = v_0$ and $q' \notin v_0$). So, the respective major views start with inclusion $v'_0 \subseteq v_0$ but end with non-inclusion $v'_j \nsubseteq v_j$. So there is $1 \le l \le j$ so that $v'_{l-1} \subseteq v_{l-1}$ but $v'_l \nsubseteq v_l$.

We are now ready to prove that $\{s'_l, s_l\}$ is incompatible with y_ι. The argument is as in Example II. Suppose the two steps participate in a set inducing an automorphism f. Since $v'_{l-1} \subseteq v_{l-1}$, both s'_l and s_l have their left end-points in v_{l-1}. Hence, $f(v_{l-1}) \supseteq v'_l \cup v_l$. Since $v'_l \nsubseteq v_l$, the height of $f(v_{l-1})$ is greater than that of v_l. But $h(v_{l-1}) = h(v_l)$. Therefore $h\big(f(v_{l-1})\big) > h(v_{l-1})$, a contradiction.

Case 2B: $j' \neq 0$. Then we can talk of the minor views $u_{j'-1}$ and u_{j-1} that precede the first repetition. Of course, $u_{j'-1} \neq u_{j-1}$. In fact, $u_{j'-1} \nsupseteq u_{j-1}$.

To see why, suppose $u_{j'-1} \supseteq u_{j-1}$ (Fig. 2c). Then $u_{j'-1} \supsetneq u_{j-1}$ (since $u_{j'-1} \neq u_{j-1}$) and thus $h(u_{j'-1}) > h(u_{j-1})$. Moreover, s_j has its left end-point in $v_{j'-1}$ (since $v_{j'-1} \supseteq u_{j'-1} \supseteq u_{j-1}$) while its right end-point has view $u_{j'}$ (since $v_j = u_j = u_{j'}$). Hence, by replacing $s_{j'}$ with s_j, we get a new computation c' that is also in $\mathcal{C}_t$. In addition, $H_{c'}$ differs from H_c only in that $h(u_{j'-1})$ is replaced by $h(u_{j-1})$. But then c' is strictly steeper than c, a contradiction.

We are now ready to prove that $\{s_{j'}, s_j\}$ is incompatible with y_ι. The argument is as in Example III. Suppose the two steps participate in a set inducing an automorphism f. Because of s_j, $f(u_{j-1}) \supseteq u_j$; but $h(u_{j-1}) = h(u_j)$ and f respects heights, so in fact $f(u_{j-1}) = u_j$. Because of $s_{j'}$, $f(u_{j'-1}) \supseteq u_{j'} = u_j$; so there exists $u^* \subseteq u_{j'-1}$ such that $f(u^*) = u_j$. Overall, $u^* \neq u_{j-1}$ (since exactly one is in $u_{j'-1}$) and $f(u^*) = f(u_{j-1})$. Hence f is not injective, a contradiction.

This concludes the analysis of the case $\iota' = \iota$ and thus the proof of Theorem 1.

5 Conclusion

We proved that small SNFAs are not closed under complement. In order to stay close to the combinatorial core of the problem, we used a non-standard transition function (implicit direction of motion; unusual reject and accept) and a large alphabet (exponential in n). It is not hard to show that *the lower bound remains exponential even under more standard definitions and over the binary alphabet.* In addition, by selecting the hard inputs more carefully in Sect. 3.2, we can ensure that a small 2DFA can correctly decide liveness on all of them. This way, we also have a proof that *2DFAs can be exponentially more succinct than SNFAs*, which generalizes the analogous known relationship between 2DFAs and SDFAs [6–8]. More details about these claims will appear in the full version of this article.

An interesting next question concerns the exact value of our lower bound (for our definition and alphabet). The smallest known SNFA for $B_{n,\emptyset}$ is the obvious 2^n-state 1DFA. *Is this really the best SNFA algorithm?* If so, then nondeterminism and sweeping bidirectionality together are completely useless in this context.

Of course, the full 2D vs. 2N question remains as wide open and challenging as ever: *Is there a small 2DFA for liveness?*

References

1. Seiferas, J.I.: Manuscript communicated to Michael Sipser. (1973)
2. Sakoda, W.J., Sipser, M.: Nondeterminism and the size of two-way finite automata. In: Proceedings of the Symposium on the Theory of Computing. (1978) 275–286
3. Kapoutsis, C.: Deterministic moles cannot solve liveness. In: Proceedings of the Workshop on Descriptional Complexity of Formal Systems. (2005) 194–205
4. Sipser, M.: Halting space-bounded computations. Theoretical Computer Science **10** (1980) 335–338
5. Geffert, V., Mereghetti, C., Pighizzini, G.: Complementing two-way finite automata. In: Proceedings of the International Conference on Developments in Language Theory. (2005) 260–271
6. Sipser, M.: Lower bounds on the size of sweeping automata. Journal of Computer and System Sciences **21**(2) (1980) 195–202
7. Berman, P.: A note on sweeping automata. In: Proceedings of the International Colloquium on Automata, Languages, and Programming. (1980) 91–97
8. Micali, S.: Two-way deterministic finite automata are exponentially more succinct than sweeping automata. Information Processing Letters **12**(2) (1981) 103–105