Lower Bounds on the Size of Sweeping Automata*

Juraj Hromkovič[1] and Georg Schnitger[2]

[1] Department of Computer Science, ETH Zürich, ETH Zentrum, CH-8022 Zürich, Switzerland
[2] Institute of Computer Science, Goethe Universität, Robert Mayer-Strasse 11-15, D-6054 Frankfurt a. M., Germany

Abstract. A deterministic sweeping automaton is a two-way deterministic automaton (2-DFA) which makes turns only at the left or right end of the input. We give a survey on recent lower bounds for the conciseness of sweeping automata. We also show that any sweeping automaton for the language $(L\$)^*$ has to have at least $\sqrt{m/2}$ states whenever the nondeterministic message complexity of $\overline{L}$ is at least m. Thus we obtain the first general method to establish lower bounds on the conciseness of sweeping automata.

1 Introduction

In 1978 Sakoda and Sipser [18] asked whether there is a polynomial p such that every n-state two-way nondeterministic automaton (2-NFA) has an equivalent $p(n)$-state 2-DFA. The corresponding question for one-way nondeterministic automata (1-NFA) and 1-DFA was answered early. The subset construction allows to convert any n-state 1-NFA into an equivalent 1-DFA with at most 2^n states and there are n-state 1-NFA which require equivalent 1-DFA of size at least 2^n [15, 16]. However despite significant research effort the Sakoda-Sipser problem has remained wide open.

After a survey on the Sakoda-Sipser problem in Section 2 we provide a communication perspective of Sipser's fundamental result [19] on sweeping automata in Section 3. In particular we show that any language $(L\$)^*$ requires deterministic sweeping automata with many states, provided the complement of L has only nondeterministic communication protocols with many messages.

2 The Sakoda-Sipser Problem

We begin the survey with unary automata and then move on to the case of general alphabets where we consider deterministic, nondeterministic and randomized 2-DFA, resp. sweeping automata.

Unary automata The difference in conciseness between determinism and nondeterminism is far better understood for unary automata. Chrobak [2, 3] shows that $O(e^{\sqrt{n \ln n}})$ states suffice and $e^{\Omega(\sqrt{n \ln n})}$ states may be necessary to simulate a unary n-state 1-NFA or 2-DFA by a 1-DFA. Unary 2-NFA and 1-DFA are compared in [14] and $e^{\Theta(\sqrt{n \ln n})}$ is shown to be the optimal cost of simulating n-state 2-NFA by 1-DFA. Also, surprisingly, a unary n-state 1-NFA can be simulated by a 2-DFA with at most $O(n^2)$ states [2]. Thus for unary automata the gain in conciseness when using nondeterminism instead of determinism is considerably smaller than to be expected for unrestricted automata and even almost disappears when simulating one-way by two-way access.

* Work supported by SNF-grant 200020-120073 and DFG-grant SCHN 503/4-1.

The Sakoda-Sipser problem has remained open for unary automata as well, however significant progress was made in closing the gap in conciseness between 2-DFA and 2-NFA [4]. First it is observed that the sweeping mode is not a severe restriction for unary computations. In particular a "quasi-sweeping" 2-NFA (qs-NFA) is introduced which reverses the direction of its head, resp. performs a nondeterministic choice only at the left or right end of its input. Then it is shown that any 2-NFA has an equivalent qs-NFA of roughly same size, however the obtained qs-NFA is only "almost-equivalent". Nevertheless the almost-equivalent normal form can be utilized to show that each n-state unary 2-NFA can be simulated by an $O(n^{\lceil \log_2(n+1) \rceil + 3})$-state qs-DFA [4] and hence the conciseness gap between 2-NFA and 2-DFA is reduced to a quasi-polynomial.

1-NFA and Sweeping automata: Sipser's Separation Sipser [19] provided a negative, partial answer to the Sakoda-Sipser problem for general alphabets by investigating sweeping automata. He showed that there is a family $(B_n \mid n \in \mathbb{N})$ of languages with 1-NFA of size n, but that any deterministic sweeping automaton for B_n requires at least 2^n states. Observe that the gap between 1-NFA and deterministic sweeping automata is optimal, since any n-state 1-NFA has an equivalent 1-DFA with 2^n states, if we apply the subset construction.

Sipser's languages B_n utilize an alphabet of size 2^{n^2}, but a separation with identical parameters can also be obtained for the binary alphabet [13].

A result of Berman and Lingas [1] underlines the importance of the Sakoda-Sipser question for the investigation of logarithmic space. Assume that $\mathbf{L} = \mathbf{NL}$, i.e., that deterministic logarithmic space equals nondeterministic logarithmic space. Then for all $n \in \mathbb{N}$ and all s-state 2-NFA A there is a $\mathrm{poly}(s, n)$-state 2-DFA A' which accepts a subset of $L(A)$ including all strings of $L(A)$ of length n. As a consequence, if the Sakoda-Sipser is answered negatively and if only strings of length $\mathrm{poly}(s, n)$ are required for the separation, then $\mathbf{L} \neq \mathbf{NL}$ follows. Sipser observed that the Berman-Lingas result remains true even if A' is required to be a deterministic sweeping automaton. However his separation of 1-NFA from sweeping automata utilizes strings of exponential length.

2-DFA with a bounded degree of non-obliviousness An oblivious 2-DFA visits all tape cells for inputs of fixed length in the same predetermined order and hence sweeping automata can be viewed as a special case of oblivious two-way automata, where the number of reversals is bounded by the number of states. In [8] Sipser's result is extended to general oblivious two-way automata. An even stronger result can be given, if we measure the degree of non-obliviousness for a given 2-DFA A. In particular, if A visits tape cells for inputs of length n in at most $\phi_A(n)$ different orders, then we say that $\phi_A(n)$ is the degree of non-oblivousness of A. [8] shows that if ϕ_A is sublinear and if A recognizes the "Sipser language" B_n, then A has at least $2^{\Omega(n)}$ states.

Nondeterministic Sweeping Automata In [10] Kapoutsis generalizes Sipser's approach from deterministic to nondeterministic sweeping automata. In particular he shows that any nondeterministic sweeping automaton for the complement $\overline{B_n}$ of the Sipser language has to have at least $2^{\Omega(n)}$ states. As a consequence, "small sweeping automata are not closed under complement". Moreover, it is shown that 2-DFA may be exponentially more succinct than nondeterministic sweeping automata, generalizing a result of Micali [17]. As a consequence, Sipser's separation method will not be immediately generalizable to separate 2-NFA from 2-DFA, since the sweeping mode is a too severe restriction.

Las Vegas Two-Way and Sweeping Automata A Las Vegas two-way automaton (2-LVFA) is a probabilistic two-way automaton with three possible outputs, an input is either accepted, rejected or a non-committal answer, a question mark say, is given. Any committing answer has to be correct and the probability of a commitment has to be at least $1/2$ for any input. A self-verifying automaton is a nondeterministic automaton with the same three types of output. If a self-verifying automaton commits, its answer has to be correct and for any answer at least one computation is committing.

In [7] the following results are obtained. First, minimal self-verifying nondeterministic automata and minimal Las Vegas automata are asymptotically of same size.[3] Here it is utilized that the expected running time of a Las Vegas automaton can be exponential in the input length by designing a stochastic experiment with correspondingly small success probability. However, 2-NFA are provably more concise than 2-LVFA, specifically there are languages L_n with 2-NFA of size $O(n)$, but any 2-LVFA requires at least $\Omega(n^2)$ states. A similar (almost) quadratic gap appears, if we compare deterministic and Las Vegas computations: there are languages L_n with 2-LVFA of size n, but any 2-DFA has to have at least $\Omega(n^2/\log_2 n)$ states.

The separation between deterministic and Las Vegas computations can be considerably sharpened, if we consider sweeping automata. In [11] languages L_n are constructed with $O(n^2)$-state sweeping Las Vegas automata, whereas any deterministic sweeping automaton requires at least $2^{\Omega(n)}$ states.

3 Sweeping Automata: A Communication Perspective

In this section we describe Sipser's [19] argument within the context of communication complexity.

3.1 Communication Complexity

Arguments from communication complexity have been successful in giving lower bounds on time and space for general models of computation [5, 12] as well as giving sharp bounds on the conciseness of 1-NFA [9]. In all these results the respective computation model $\mathcal{M}$ is simulated by a communication protocol such that the resource under consideration is linked with the number of exchanged messages. The argument is completed by showing that any communication protocol has to exchange "many" messages and hence $\mathcal{M}$ has to consume "many" resources.

We begin by reviewing the conventional two-party communication setting [20]. Two players, Alice and Bob want to compute a Boolean function $f : \{0,1\}^{2n} \to \{0,1\}$. At the beginning of the computation Alice receives the prefix x of the first n input bits and Bob receives the final n bits. Alice and Bob cooperate in order to determine $f(x,y)$; in particular they communicate by exchanging binary messages and try to determine $f(x,y)$ by exchanging as few message sequences as possible. We say that a protocol is one-way if only Alice sends messages and two-way otherwise. Various types of communication protocols such as deterministic, nondeterministic or randomized protocols can be considered.

When applying communication complexity to conciseness problems, the setup of uniform communication [6] is preferable however: here an input is not partitioned into strings x and y of (approximate) same length but into an arbitrary prefix x and the remaining suffix y. To give an example let us assume that we are interested in determining the minimal number of states of a 1-DFA for a given language L. We

[3] In the case of one-way access, the situation is completely different, since minimal 1-DFA have at most quadratically more states than Las Vegas one-way automata [6].

may simulate any 1-DFA A for L as follows. Alice first simulates A on her input x and then sends the final state q to Bob. Bob continues to simulate A on his input y by beginning in state q and is then able to determine whether the combined input xy belongs to L. We say that the total number of different messages, sent over all inputs, is the message complexity of this protocol. Thus we have simulated any 1-DFA A with $|Q|$ states by a deterministic communication protocol with message complexity $|Q|$.

What is the worst possible gap between the minimal number of messages and the minimal number of states? It turns out that the minimal number of messages required to recognize L by a deterministic one-way protocol coincides with the minimal number of states required by a 1-DFA for L [6]. This result only holds for uniform communication and is false for non-uniform communication. For instance the language $L_n = \{0^{2^n}xx | x \in \{0,1\}^n\}$ can be recognized with message complexity two, if Alice receives the first half of the input: Alice indicates by sending a single bit whether she only sees zeroes. However the size of the minimal DFA is at least 2^n and message complexity does not predict the minimal number of states correctly.

Of course this simulation of finite automata by one-way protocols works for any mode of computation and not just deterministic computations, however for nondeterministic and probabilistic computations the minimal number of states may be exponentially larger than the minimal number of messages [9].

Let $\mathrm{d}(L)$ and $\mathrm{n}(L)$ be the deterministic and nondeterministic message complexity of L, i.e., $\mathrm{d}(L)$ equals the minimal number of message sequences of a two-way deterministic protocol for L sent for all possible inputs; $\mathrm{n}(L)$ is the corresponding measure for nondeterministic communication. Finally let $\mathrm{d}_1(L)$ be the minimal number of messages of a one-way deterministic protocol for L. There may be exponential gaps between $\mathrm{d}_1(L)$ and $\mathrm{d}(L)$ as well as between $\mathrm{d}(L)$ and $\mathrm{n}(L)$ [5, 12].

3.2 Sweeping Automata and Nondeterministic Communication

The application of communication complexity to 1-DFA is straightforward, but the situation changes dramatically when stepping from a 1-DFA to a 2-DFA. In particular assume that A is a 2-DFA with q states: if we simulate A on an input xy, then A may generate up to $q!$ message sequences, since now crossing sequences may have length q. This phenomenon is the main reason that communication arguments fail when trying to give non-trivial lower bounds on the minimal number of states of a 2-DFA. Surprisingly, if we follow Sipser's argumentation, communication methods are applicable to sweeping automata.

The concept of a generic string [19] is crucial for the following arguments. We investigate the marked Kleene closure $(L\$)^*$ for a new separating symbol \$ and essentially call a string $g \in (L\$)^*$ generic for L and a sweeping automaton A iff g "maximally compresses" the set of reachable states.

Definition 1. *Let L be a language and let \$ be a new symbol. Also, let A be a deterministic sweeping automaton for $(L\$)^*$ with Q as its set of states.*
(a) For any string s set

$$Q_{lr}(s) = \{q \mid \textit{there is } q' \in Q \textit{ such that } A \textit{ reaches } q \textit{ in a } \textbf{\textit{single}} \textit{ left-to-right sweep, when reading } s \textit{ and starting the sweep in state } q' \}.$$

$Q_{rl}(s)$ is defined analogously, but now we consider right-to-left sweeps of A.
(b) A string $g \in (L\$)^*$ is called lr-generic (resp. rl-generic) for A iff $|Q_{lr}(g)| \leq |Q_{lr}(gt)|$ (resp. $|Q_{rl}(g)| \leq |Q_{rl}(tg)|$) holds for any string $t \in (L\$)^*$. We call g generic for A iff g is lr-generic and rl-generic for A.

We show that generic strings exist for arbitrary languages L. Observe first that

$$Q_{lr}(uv) \subseteq Q_{lr}(v) \text{ and } Q_{rl}(uv) \subseteq Q_{rl}(u) \tag{1}$$

hold for arbitrary strings $u, v \in (L\$)^*$. For instance, assume that state q belongs to $Q_{lr}(uv)$. Then q is reached in a left-to-right sweep of A for input uv. If $q' \in Q$ is the obtained state after reading u, then q is reached when starting in q' and reading v. But then q belongs to $Q_{lr}(v)$.

Now, if an arbitrary string $u \in (L\$)^*$ is not lr-generic, then there is a string $v \in (L\$)^*$ with $|Q_{lr}(uv)| < |Q_{lr}(u)|$. We apply (1) and obtain $|Q_{rl}(uv)| \leq |Q_{rl}(u)|$. Replace u by uv and repeat the argument. If u is not rl-generic with "counterexample" $v \in (L\$)^*$, then we apply an analogous argument, replace u by vu and repeat the argument as well.

Lemma 1. *[19] Let L be an arbitrary regular language and let A be a deterministic sweeping automaton for $(L\$)^*$ with state set Q. Then there is a generic string g for A. Moreover, for any string s,*

$$s \in L \Rightarrow Q_{lr}(gs\$g) = Q_{lr}(g) \text{ and } Q_{rl}(gs\$g) = Q_{rl}(g)$$
$$s \notin L \Rightarrow Q_{lr}(gs\$g) \subset Q_{lr}(g) \text{ or } Q_{rl}(gs\$g) \subset Q_{rl}(g).$$

Proof. We have already established that there is a generic string g for A. We apply (1) and obtain

$$Q_{lr}(gs\$g) \subseteq Q_{lr}(g) \text{ and } Q_{rl}(gs\$g) \subseteq Q_{rl}(g)$$

for any string s. Now let s belong to L. If $Q_{lr}(gs\$g)$ is a proper subset of $Q_{lr}(g)$ or if $Q_{rl}(gs\$g)$ is a proper subset of $Q_{rl}(g)$, then contrary to our assumption g is not generic.

Finally assume that s does not belong to L, but that $Q_{lr}(gs\$g) = Q_{lr}(g)$ as well as $Q_{rl}(gs\$g) = Q_{rl}(g)$ hold. As a consequence, we also get $Q_{lr}(g(s\$g)^i) = Q_{lr}(g)$ as well as $Q_{rl}((gs\$)^i g) = Q_{rl}(g)$ for all i. Hence $(s\$g)^i$ and $(gs\$)^i$ induce a permutation on $Q_{lr}(g)$ and $Q_{rl}(g)$ respectively. There has to be a power j such that both $(s\$g)^j$ and $(gs\$)^j$ induce the respective identity permutation. We have obtained a contradiction, since $g \in (L\$)^*$ and $g(s\$g)^j = (gs\$)^j g \notin (L\$)^*$ are indistinguishable for A. □

Theorem 1. *Assume that $\mathrm{n}(\overline{L}) = m$. Then any deterministic sweeping automaton for $(L\$)^*$ has to have at least $\sqrt{m/2}$ states.*

Proof. Assume that $(L\$)^*$ has a deterministic sweeping automaton A with state set Q. Let $q = |Q|$ and assume that $\mathrm{n}(\overline{L}) = m$ holds. We utilize A to come up with a nondeterministic protocol for $\overline{L}$ with message complexity $2q^2$. But then $2q^2 \geq m$ has to hold and the claim follows.

Let $g \in (L\$)^*$ be a generic string for A. Assume that Alice receives the prefix x and Bob the suffix y of an input $s = xy$. Alice starts by guessing a "direction" $d \in \{lr, rl\}$. We first assume $d = lr$.

Alice guesses two states $p_1, p_2 \in Q_{lr}(g)$ and communicates lr as well as states q_1, q_2, where q_i is the final state after reading x when starting in state p_i. Bob continues the simulation of a left-to-right sweep of A by determining the final state q_i' after reading $y\$g$ starting with state q_i. Bob accepts if $q_1' = q_2'$ and rejects otherwise. If Bob accepts, then $Q_{lr}(gxy\$g) \subset Q_{lr}(g)$ and hence $xy \notin L$ follows as a consequence of Lemma 1.

If on the other hand $d = rl$, then Alice only communicates rl. Bob guesses states $p_1, p_2 \in Q_{rl}(\$g)$, simulates right-to-left sweeps on his input y for p_1, p_2 and communicates the obtained states q_1, q_2. Alice continues the simulation for her input gx and accepts iff $q_1' = q_2'$ for the final states q_1', q_2'.

If the protocol accepts, then $Q_{lr}(gxy\$g)\ Q_{lr}(g)$, $Q_{rl}(gxy\$g) = Q_{rl}(g)$ and $xy \in L$ follows. Thus the protocol is correct and accepts $\overline{L}$ by sending at most $2 \cdot q^2$ nondeterministic messages. □

Thus any marked Kleene closure $(L\$)^*$ requires sweeping automata with many states, provided the nondeterministic message complexity of the complement of L is large. In many cases Theorem 1 can be applied easily. We give a first example for the language $L = L_n = \{xy \mid x, y \in \{0,1\}^n, x \neq y\}$. The nondeterministic message complexity of $\overline{L_n}$ is 2^n: any accepting message sequence which is sent for xx and yy is also sent for xy and hence any accepting message sequence is sent for exactly one string in L_n. (Thus the strings xx form a fooling set.) As a consequence, sweeping automata for $(L_n\$)^*$ need at least $2^{n/2}$ states.

Observe that L_n and hence $(L_n\$)^*$ has 1-NFA of size quadratic in n. But we can also construct 2-DFA for $(L_n\$)^*$ of quadratic size, since L_n can be recognized by a 2-DFA of quadratic size.

Corollary 1. *Let*

$$L_n = \{xy \mid x, y \in \{0,1\}^n, x \neq y\}.$$

Then any deterministic sweeping automaton for $(L_n\$)^*$ has at least $2^{(n-1)/2}$ states. However L_n can be recognized by a 1-NFA or a 2-DFA with $O(n^2)$ states.

Observe that, in general, the sizes of minimal automata recognizing L resp. $(L\$)^*$ are almost identical for 1-DFA, 1-NFA and 2-DFA, however $(L\$)^*$ may be much harder if one enforces the sweeping mode. The first result showing the severe limitation of the sweeping mode was given in [17].

A second example is Sipser's language B_n. To define B_n let Σ_n be the alphabet consisting of all bipartite graphs with n "left" and n "right" nodes. For a given word $w = b_1 \cdots b_m$ identify the left nodes of b_{i+1} with the right nodes of b_i. Then B_n consists of all words w with a path from a left node of b_1 to a right node of b_m.

Let \$ correspond to the complete bipartite graph and let B'_n be the subset of B_n consisting of all words over $\Sigma_n \backslash \{\$\}$. Then $(B'_n\$)^* = B_n\$$. Moreover, w belongs to B_n if and only if $w\$$ belongs to $B_n\$$ and hence it suffices to show that $(B'_n\$)^*$ requires large sweeping automata. We therefore have to show that the nondeterministic message complexity of $\overline{B'_n}$ is large.

In the communication problem for $\overline{B'_n}$ assume that Alice and Bob receive the inputs $w_A = b_1 \cdots b_{m'}$ and $w_B = b_{m'+1} \cdots b_m$ respectively. All Alice has to know is the subset $x \subseteq \{1, \ldots, n\}$ of the right nodes of $b_{m'}$ which are reachable from one of the left nodes of b_1 and all Bob has to know is the subset $y \subseteq \{1, \ldots, n\}$ of left nodes of $w_{m'+1}$ which reach a right node of w_m. Their combined input belongs to $\overline{B'_n}$ iff $x \cap y = \emptyset$ and the unreachability problem is equivalent to a disjointness problem. We again apply a fooling set argument to analyze the nondeterministic message complexity of the disjointness problem: consider all pairs $(x, \overline{x})$ of subsets of $\{1, \ldots, n\}$ with $|x| = n/2$ and observe that any accepting message sequence which is sent for $(x, \overline{x})$ and $(y, \overline{y})$ is also sent for $(x, \overline{y})$ and hence at least $\binom{n}{n/2}$ accepting message sequence have to be sent. Theorem 1 gives the lower bound of $\sqrt{\binom{n}{n/2}/2}$ states. This bound is roughly quadratically weaker than Sipser's bound 2^n and we are paying for larger applicability with slightly smaller bounds.

References

1. P. Berman and A. Lingas, On the complexity or regular languages in terms of finite automata, ICS PAS Report 304, Polish Academy of Sciences, 1977.
2. M. Chrobak, Finite automata and unary languages, *Theor. Comput. Sci.* 47, pp. 149-158, 1986.

3. M. Chrobak, Errata to "Finite automata and unary languages", *Theor. Comput. Sci.* 302 (1), pp. 497-498, 2003.
4. V. Geffert, C. Mereghetti and G. Pighizzini, Converting two-way nondeterministic unary automata into simpler automata, *Theor. Comput. Sci.* 295, pp. 189-203, 2003.
5. J. Hromkovič, Communication Complexity and Parallel Computating. Springer, 1997.
6. J. Hromkovič, G. Schnitger, On the power of Las Vegas for one-way communication complexity, OBDD's, and finite automata. *Inf. Comput.* 169, pp. 284-296, 2001.
7. J. Hromkovič and G. Schnitger, On the power of Las Vegas II: Two-way finite automata, *Theor. Comput. Sci.* 262, pp. 1-24, 2001.
8. J. Hromkovič and G. Schnitger, Nondeterminism versus determinism for two-way finite automata: generalizations of Sipser's separation, *Proc. of 30th Intern. Colloquium on Automata, Languages and Programming*, pp. 439-451, 2003.
9. J. Hromkovič and G. Schnitger, On the hardness of determining small NFA's and of proving lower bounds on their sizes, to appear in *Proc. of 12th Intern. Conference on Develpments in Language Theory*, pp. 34-55, 2008.
10. C.A. Kapoutsis, Small sweeping 2NFAs are not closed under complement, *Proc. of 33rd Intern. Colloquium on Automata, Languages and Programming*, pp. 144-156, 2006.
11. C.A. Kapoutsis, R. Kralovic and T. Mömke, An exponential gap between Las Vegas and deterministic sweeping finite automata, *4th Intern. Symp. on Stochastic Algorithms: Foundations and Applications*, pp. 130-141, 2007.
12. E. Kushilevitz, N. Nisan, Communication Complexity. Cambridge University Press, 1997.
13. H. Leung, Tight lower bounds on the size of sweeping automata, *J. Comput. Syst. Sci.* 63, 384-393, 2001.
14. C. Mereghetti and G. Pighizzini, Optimal simulations between unary automata, *SIAM J. Comput.* 30, pp. 1976-1992, 2001.
15. A. Meyer and M. Fischer, Economy of description by automata, grammars and formal systems, *in Proc. of the 12th SWAT Symposium*, pp. 188-191, 1971.
16. F. Moore, On the bounds of state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way automata, *IEEE Trans. Comput.* 20, pp. 1211-1214, 1971.
17. S. Micali, Two-way deterministic finite automata are exponentially more succinct than sweeping automata, *Inform. Process. Letters* 12, pp. 103-105, 1981.
18. W. Sakoda and M. Sipser, Nondeterminism and the size of two-way finite automata, *Proc. 10th ACM Symp. on Theory of Comput.*, pp. 275-286, 1978.
19. M. Sipser, Lower bounds on the size of sweeping automata, *J. Comput. Syst. Sci.* 21, 195-202, 1980.
20. A.C. Yao, Some Complexity Questions Related to Distributed Computing, *Proc. 11th ACM Symp. on Theory of Comput.*, pp. 209-213, 1979.