A Note on the Reduction of Two-Way Automata to One-Way Automata*

Moshe Y. Vardi†
IBM Almaden Research Center

Abstract

We describe a new elementary reduction of two-way automata to one-way automata. The reduction is based on the subset construction rather than on crossing sequence analysis.

1 Introduction

Rabin and Scott [RS59] introduced two-way finite automata, which are allowed to move in both directions along their input tape, and proved that they are equivalent to one-way automata in their ability to define languages, i.e, they define precisely the regular languages. Their proof is rather complicated and is based on *crossing sequence analysis*. A simpler proof was given by Shepherdson [Sh59]; that proof is also indirectly based on crossing sequence analysis.

We describe here a new elementary proof, which is based on Rabin and Scott's *subset construction* rather than crossing sequence analysis. Our approach is somewhat less direct than the approaches in [RS59,Sh59]. Given a two-way automaton A, rather than construct a one-way automaton that defines the same language as A, we construct a one-way automaton that defines the complementary language. As we will show, it is the indirectness of the approach that enables us to use the subset construction.

Interestingly, the new construction was motivated by a practical application related to optimization of database logic program [CGKV88]. In that application we are given a nondeterministic two-way automata and we have to construct a (possibly nondeterministic) automaton that defines the complementary language. The constructions in [RS59,Sh59] (properly generalized) yield nondeterministic automata when applied to nondeterministic two-way automata, while at the same time they involve an exponential

* *Information Processing Letters* 30(1989), pp. 261–264.
† Address: IBM Almaden Research K53-802, 650 Harry Rd., San Jose, CA 95120-6099.

blow-up in the number of automaton states. Since complementing a nondeterministic automaton also involves an exponential blow-up in the number of automaton states [RS59], a straightforward approach to the optimization problem would involve a *doubly* exponential blow-up. The new construction manages to accomplish the task with a single exponential blow-up.

2 Basic Definitions

A *two-way automaton* $A = (\Sigma, S, S_0, \rho, F)$ consists of an alphabet Σ, a finite set of state S, a set of initial states $S_0 \subseteq S$, a transition function $\rho : S \times \Sigma \rightarrow 2^{S\times\{-1,0,1\}}$, and a set of accepting states $F \subseteq S$. Intuitively, a transition indicates not only the new state of the automaton, but also whether the head should move left, right, or stay in place. If for all $s \in S$ and $a \in \Sigma$ we have that $|\rho(s, a)| \leq 1$ and also $|S_0| = 1$, then A is said to be *deterministic*. If for all $s \in S$ and $a \in \Sigma$ we have that $\rho(s, a) \subseteq S \times \{1\}$, then the head always moves to the right, so the automaton is called a *one-way* automaton. It that case it is convenient to view the transition function as a mapping $\rho : S \times \Sigma \rightarrow 2^S$.

A *configuration* of A is a member of $S \times I\!N$, i.e., a pair consisting of a state and a "position". A *run* is a sequence of configurations, i.e., an element in $(S \times I\!N)^\star$. The run $(s_0, j_0), \ldots, (s_m, j_m)$ is a run of A on a word $w = a_0, \ldots, a_n$ in $\Sigma^\star$ if $s_0 \in S_0$, $j_0 = 0$, $j_m \leq n + 1$, and for all i, $0 \leq i < m$, we have that $0 \leq j_i \leq n$, and there is some $(t, k) \in \rho(s_i, a_{j_i})$ such that $s_{i+1} = t$ and $j_{i+1} = j_i + k$. This run is *accepting* if $j_m = n+1$ and $s_m \in F$. *A accepts w* if it has an accepting run on w. The set of words accepted by A is denoted $L(A)$.

3 The New Construction

At the heart of our construction is a characterization of inacceptance by two-way automata.

Lemma 3.1: *Let $A = (\Sigma, S, S_0, \rho, F)$ be a two-way automaton, and $w = a_0, \ldots, a_n$ be a word in $\Sigma^\star$. A does not accept w if and only if there exists a sequence $T_0, \ldots, T_{n+1}$ of subsets of S such that the following conditions hold:*

1. *$S_0 \subseteq T_0$,*
2. *$T_{n+1} \cap F = \emptyset$, and*
3. *for $0 \leq i \leq n$, if $s \in T_i$, $(s', k) \in \rho(s, a)$, and $i + k > 0$, then $s' \in T_{i+k}$.*

Proof: Suppose first that A does not accept w. Define T_i, $0 \leq i \leq n+1$, to be the set of all states $s \in S$ such that (s, i) is a configuration in a run of A on w. We now verify that the sequence $T_0, \ldots, T_{n+1}$ satisfies the conditions of the lemma. If $s \in S_0$, then $(s, 0)$ is a run of A on w, so we have $S_0 \in T_0$. Also, since A does not accept w, if $s \in F$, then

$(s, n+1)$ does not occur in any run of A on w. Therefore, $T_{n+1} \cap F = \emptyset$. Finally, if $0 \leq i \leq n$ and $s \in T_i$, then there is a run $(s_0, j_0), \ldots, (s_l, j_l)$ of A on w, where $s_l = s$, and $j_l = i$. If now $(s', k) \in \rho(s, a)$, and $i + k > 0$, then $(s_0, j_0), \ldots, (s_l, j_l), (s', i+k)$ is also a run of A on w. It follows that $s' \in T_{i+k}$.

Suppose now that A accepts w and let $(s_0, j_0), \ldots, (s_m, j_m)$ be an accepting run of A on w. In particular, $j_m = n+1$ and $s_m \in F$. Assume that $T_0, \ldots, T_{n+1}$ is a sequence that satisfies the conditions of the lemma. We show by induction on i that $s_i \in T_{j_i}$ for $0 \leq i \leq m$. Clearly, $s_0 \in T_0$, since $s_0 \in S_0$ and $S_0 \subseteq T_0$. Assume we have shown that $s_i \in T_{j_i}$ for $0 \leq i < m$. Then $0 \leq j_i \leq n$, and there is some $(t, k) \in \rho(s_i, a_{j_i})$ such that $s_{i+1} = t$ and $j_{i+1} = j_i + k$. Consequently, $j_i + k \geq 0$ and $s_{i+1} \in T_{j_{i+1}}$. It follows that $T_{n+1} \cap F \neq \emptyset$ — a contradiction. ∎

It may be tempting to think that it is easy to get a similar condition to acceptance of w by A. It seems that all we have to do is to change the second clause in Lemma 3.1 to $T_{n+1} \cap F \neq \emptyset$. Unfortunately, this is not the case; to characterize acceptance we also have to demand that the T_i's be *minimal*. While the conditions in the lemma are *local*, and therefore checkable by a finite-state automaton, minimality is a *global* condition.

To build an automaton that accepts the words not accepted by a two-way automaton A, we construct an automaton that checks nondeterministically whether the conditions of Lemma 3.1 are satisfied.

Theorem 3.2: *Let A be a two-way automata with n states. Then there is a one-way automaton B with $O(\exp n)$ states such that $L(B) = \Sigma^\star - L(A)$.*

Proof: B is the automaton $(\Sigma, Q, Q_0, \delta, G)$. The state set Q is $2^S \cup (2^S)^2$, i.e., sets of states and pairs of sets of states The starting state set Q_0 is $\{T : S_0 \subseteq T \subseteq S\}$, i.e, the collection of state sets that contain S_0. The accepting state set G is $\{T : T \cap F = \emptyset\} \cup \{(T, U) : U \cap F = \emptyset\}$, i.e., the collection of sets that do not intersect F and pair of sets where the second component do not intersect F.

It remains to define the transition function δ. We have $(T, U) \in \delta(T, a)$ if the following holds:

- If $s \in T$ and $(t, 0) \in \rho(s, a)$, then $t \in T$, and
- if $s \in T$ and $(t, 1) \in \rho(s, a)$, then $t \in U$.

We have $(U, V) \in \delta((T, U), a)$ if the following holds:

- If $s \in U$ and $(t, -1) \in \rho(s, a)$, then $t \in T$,
- If $s \in U$ and $(t, 0) \in \rho(s, a)$, then $t \in U$, and
- if $s \in U$ and $(t, 1) \in \rho(s, a)$, then $t \in V$.

It is easy to verify that B accepts a word $w = a_1, \ldots, a_n$ if and only if there exists a sequence $T_0, \ldots, T_{n+1}$ that satisfies the conditions of Lemma 3.1. Thus, by the lemma, B accepts w if and only if A does not accept w. ∎

Since regular language are closed under complement, Theorem 3.2 implies that two-way automata defines regular languages. Of course, if you are given a two-way automaton and you want a one-way automaton that defines the same language, then it is more efficient to use the construction in [RS59,Sh59], since our construction yields a nondeterministic automaton for the complementary language.

4 A Deterministic Construction

The construction described in the previous section yields a nondeterministic one-way automaton. In this section we show that is possible to obtain a deterministic one-way automaton without a doubly exponential blow-up, even when the two-way automaton is nondeterministic. This construction is based on Shepherdson's construction, which keep in a finitary way all the information about "backwards" runs [Sh59].

Let $A = (\Sigma, S, S_0, \rho, F)$ be a two-way automaton. An *A-label* is a subset of S^2, i.e, a set pairs of states. Intuitively, a pair $\langle s, t\rangle$ denotes the fact that there is a backward run starting at a state s and ending at a state t. Let $w = a_0, \ldots, a_n$ be a word in $\Sigma^\star$. An *A-labeling* $\mathbf{m} \in (2^{S^2})^\star$ for w consists of a sequence $m_0, \ldots, m_n$ of labels that satisfy the following condition: there exists a sequence $l_0, \ldots, l_n$ of labels such that

- $\langle s, t\rangle \in l_i$ iff either $(t, 0) \in \rho(s, a_i)$ or $i > 1$ and there are states $s', t' \in S$ such that $\langle s', t'\rangle \in m_{i-1}$, $(s', -1) \in \rho(s, a_i)$, and $(t, 1) \in \rho(t', a_{i-1})$.
- $\langle s, t\rangle \in m_i$ iff there is a sequence $s_0, ..., s_k$, $k > 0$, such that $s_0 = s$, $s_k = t$ and $\langle s_j, s_{j+1}\rangle \in l_i$ for $0 \geq j > k$.

Intuitively, m keeps the information about backward runs. It is easy to see that every word has a unique A-labeling.

We now consider words over the alphabet $\Sigma_A = \Sigma \times 2^{S^2}$. Let u be the word $\langle a_0, m_0\rangle, \ldots, \langle a_n, m_n\rangle$ in $\Sigma_A^\star$. We say that the word is *A-legal* if $\mathbf{m}$ is an A-labeling of w, where $w = a_0, \ldots, a_n$, and $\mathbf{m} = m_0, \ldots, m_n$. We abuse notation and denote u by the pair $\langle w, m\rangle$.

The usefulness of A-labeling is that it supplies enough information to check in a "one-way sweep" whether a word is accepted by the two-way automaton A.

Lemma 4.1: *Let A be a two-way automaton with n states. There are one-way deterministic automata A_1 and A_2 with $O(\exp n)$ states such that a given legal word $\langle w, m\rangle$ is accepted by A_1 if and only if w is accepted by A and a given legal word $\langle w, m\rangle$ is accepted by A_2 if and only if w is not accepted by A.*

Proof: Essentially the automata A_1 and A_2 use the information supplied in the labelling to avoid going backwards.

Let $A = (\Sigma, S, S_0, \rho, F)$. A_1 is the one-way automaton $(\Sigma, Q, Q_0, \delta, G)$. The state set Q is 2^S. The starting state set Q_0 is $\{S_0\}$. The accepting state set is $G = \{T : T \cap F \neq \emptyset\}$.

It remains to define the transition function δ. We have $U \in \delta(T, \langle m, a\rangle)$ if $U = \{s : \exists s', s'' \in S$ such that $\langle s', s''\rangle \in m$ and $s \in \rho(s'', a)\}$.

It is easy to see that A_1 is deterministic. We leave it to the reader to verify that a given legal word $\langle w, m\rangle$ is accepted by A_1 if and only if w is accepted by A.

The definition of A_2 is almost identical to the definition of A_1; the only difference is that the accepting state set is $\{T : T \cap F = \emptyset\}$. ∎

To complete our construction we have to show that a legal labelling can be computed by a finite-state deterministic automaton.

Theorem 4.2: *Let A be a two-way automata with n states. Then there are one-way deterministic automata B_1 and B_2 with $O(\exp(n^2))$ states such that $L(B_1) = L(A)$ and $L(B_2) = \Sigma^\star - L(A)$.*

Proof: The automata B_1 and B_2 will have two part: one part generates the labelling, and the other part emulate the automata A_1 and A_2 of Lemma 4.1. We focus first on B_1.

Let $A = (\Sigma, S, S_0, \rho, F)$. Let $A_1 = (\Sigma, Q, Q_0, \delta, G)$ described in Lemma 4.1. B_1 is the one-way automaton $(\Sigma, P, P_0, \theta, H)$. The state set P is $Q \times (2^{S^2})^2$, that is, a state in P consists of a state of A_1 and a pair of A-labels. The starting state set Q_0 is $\{S_0\} \times \{\emptyset\} \times \{\emptyset\}$. The accepting state set is $G = \{\langle T, p, m\rangle : T \cap F \neq \emptyset\}$.

It remains to define the transition function θ. We have $\langle T', p', m'\rangle \in \delta(\langle T, p, m\rangle, a)$ if

1. $T' \in \delta(T, \langle a, m\rangle)$
2. $\langle s, t\rangle \in p'$ iff $(t, 1) \in \rho(s, a)$, and
3. there exists a label $l \subseteq S^2$ such that
   - $\langle s, t\rangle \in l$ iff either $(t, 0) \in \rho(s, a)$ or there are states $s', t' \in S$ such that $\langle s', t'\rangle \in m$, $(s', -1) \in \rho(s, a)$, and $(t', t) \in p$.
   - $\langle s, t\rangle \in m'$ iff there is a sequence $s_0, ..., s_k$, $k > 0$, such that $s_0 = s$, $s_k = t$ and $\langle s_j, s_{j+1}\rangle \in l$ for $0 \geq j > k$.

The construction for A_2 is almost identical; the only difference is that the accepting state set is $\{\langle T, p, m\rangle : T \cap F = \emptyset\}$. ∎

Note that we could have used Theorem 4.2 to accomplish the task mentioned in the introduction: given a two-way automaton, construct a one-way automaton that defines the complementary language. The blow-up in Theorem 3.2 ($O(\exp n)$) is, however, better than the blow-up in Theorem 4.2 ($(O(\exp n^2))$.

5 References

[**CGKV88**] Cosmadakis, S.S., Gaifman, H., Kanellakis, P.C., Vardi, M.Y.: Decidable Optimization Problems for Database Logic Programs. *Proc. 20th ACM Symp. on Theory of Computing*, May 1988.

[**RS59**] Rabin, M.O., Scott, D.: Finite automata and their decision problems. *IBM J. Research and Development*, 3(1959), pp. 114–125.

[**Sh59**] Shepherdson, J.C.: The reduction of two-way automata to one-way automata. *IBM J. Research and Development*, 3(1959), pp. 199–201.