Three families of maximally nondeterministic automata

Citation for published version (APA):

Document status and date:
Published: 01/01/1987

Publisher Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Three Families of Maximally Nondeterministic Automata

by

Tom Verhoeff

87/10

June, 1987
This is a series of notes of the Computing Science Section of the Department of Mathematics and Computing Science Eindhoven University of Technology. Since many of these notes are preliminary versions or may be published elsewhere, they have a limited distribution only and are not for review. Copies of these notes are available from the author or the editor.
Three Families of Maximally Nondeterministic Automata

TOM VERHOEFF

Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
mcvax.UUCP!leutrc3!wstomv

April 1987

ABSTRACT

For every nondeterministic finite state machine there exists an equivalent deterministic automaton. Such an equivalent automaton can be obtained by the subset construction. An automaton may be called maximally nondeterministic when the subset construction yields a minimal automaton. This note presents three nice families of maximally nondeterministic automata together with some variations. The languages concerned are all prefix-closed.

CONTENTS

0. Introduction
1. Deterministic and maximally nondeterministic automata
2. A simple family
3. A family based on the doubling construction
4. A family with only two symbols
5. Concluding remarks
References

Maximally Nondeterministic Automata
0. INTRODUCTION

Some circuit synthesis techniques based on Trace Theory (cf. [0], [1]) require the conversion of a nondeterministic automaton into an equivalent deterministic one. This conversion may cause an increase in the size of the automaton. It is interesting to know how bad the increase can be. This note presents a couple of worst case examples.

There are many ways to define automata (cf. [2], [3], [4]). In this note an automaton is a quadruple \(<Q, A, \delta, q_0>\), where

- \(Q\) is a finite set of states,
- \(A\) is a finite set of symbols,
- \(\delta\) is a mapping of \(Q \times A\) to the power set of \(Q\), and
- \(q_0\) is in \(Q\).

The set \(A\) is called the alphabet of the automaton, \(\delta\) its transition function, and \(q_0\) its initial state.

Let \(N\) be an automaton, \(N = <Q, A, \delta, q_0>\). We also write \(qN\) for \(Q\), \(aN\) for \(A\), \(\delta_N\) for \(\delta\), and \(iN\) for \(q_0\).

The state transition graph of \(N\), or state graph for short, is the following rooted labeled directed graph \(G\). \(G\) has vertex set \(Q\). There is an arrow labeled \(b\), \(b \in A\), from vertex \(q\) to vertex \(r\) if and only if \(r \in \delta(q, b)\). The root of \(G\) is \(q_0\). A labeled arrow is also called a transition of the automaton. The set of transitions is denoted by \(xN\); it can be defined as a subset of \(Q \times A \times Q\). We are only interested in its size as a measure of the automaton’s complexity. We have

\[
\# xN = \left(\sum q, b : q \in Q \land b \in A : \# \delta(q, b)\right).
\] (0.0)

Notice that the alphabet of an automaton may contain symbols that do not occur as labels in its state graph. An automaton is, therefore, not uniquely determined by its state graph. For the automata that we shall encounter each symbol of the alphabet occurs in at least one transition.

The trace of a path in the state graph is the sequence of symbols that occur as labels on its constituent arrows. The trace set of a state is the set of all traces of paths that begin in that state. The trace set of a state is thus a subset of \((aN)^*\). The trace set of an automaton is the trace set of its initial state. Two automata are called equivalent if and only if their trace sets are equal.

Although the above definitions in terms of the state graph are straightforward, it is also convenient to express them more directly. Therefore the transition function \(\delta\) is extended to sets of states and traces of symbols as follows. For \(X\) a subset of \(Q\) and \(t\) a trace in \(A^*\) we define \(\hat{\delta}(X, t)\) by

\[
\hat{\delta}(X, \epsilon) = X.
\]
\[ \delta(X, tb) = (\bigcup q : q \in \hat{\delta}(X, t) : \delta(q, b)) \quad \text{for } b \in A, \]

where \( \epsilon \) denotes the empty trace. The extended transition function \( \hat{\delta} \) has some important properties. Let \( X \) and \( Y \) be subsets of \( Q \), \( q \) a state in \( Q \), \( t \) and \( u \) traces over \( A \), and \( b \) a symbol in \( A \). Then we have

\[ \begin{align*}
\hat{\delta}(\emptyset, t) &= \emptyset, \\
\hat{\delta}(\{q\}, b) &= \delta(q, b), \\
\hat{\delta}(X, tu) &= \hat{\delta}(\hat{\delta}(X, t), u), \text{ and} \\
\hat{\delta}(X \cup Y, t) &= \hat{\delta}(X, t) \cup \hat{\delta}(Y, t).
\end{align*} \]

In state graph terminology, \( \hat{\delta}(\{q\}, t) \) is the set of states that can be reached from state \( q \) via a path with trace \( t \). Hence, there exists a path in the state graph with trace \( t \) starting in state \( q \) if and only if \( \hat{\delta}(\{q\}, t) \neq \emptyset \). Thus, the trace set of state \( q \) is the set

\[ \{ t \in (A^*)^* | \hat{\delta}(\{q\}, t) \neq \emptyset \}. \]

**Technical Remark.** The kind of automata that we have defined are also known as

0) nondeterministic finite state machines.
1) with exactly one initial state.
2) without epsilon transitions.
3) that are not necessarily completely specified (since \( \delta \) may give \( \emptyset \) as image), and
4) of which all states are final states.

A consequence of points 1) and 4) is that the trace sets of our automata are nonempty and prefix-closed. The latter means that for each trace in the trace set all its prefixes are also in the trace set.

(End of Technical Remark)

1. **DETERMINISTIC AND MAXIMALLY NONDETERMINISTIC AUTOMATA**

Automaton \( N \) is deterministic if and only if

\[ (A \ q, b : q \in Q \land b \in A : \# \delta(q, b) \leq 1). \]

In terms of the state graph this means that all outgoing arrows of a vertex have distinct labels. Hence, for each vertex there is at most one path with a given trace that starts in this vertex. The well-known *subset construction*, when applied to \( N \), yields a deterministic automaton \( D \) that is equivalent to \( N \). We recall that

\[ qD = \{ X | \emptyset \neq X \subseteq qN \}, \]
\[ aD = aN. \]

*Maximally Nondeterministic Automata*
\[ \delta_D(X, b) = \begin{cases} \{ \delta_N(X, b) \} & \text{if } \delta_N(X, b) \neq \emptyset \\ \emptyset & \text{otherwise} \end{cases} \quad \text{for } X \in \mathcal{Q}_D \text{ and } b \in \Sigma_D. \]

and therefore \( \# \mathcal{Q}_D = 2^{\# \mathcal{Q}_N} - 1 \). \( D \) is also called the subset automaton of \( N \).

**Technical Remark.** Notice that we have excluded the empty set from \( \mathcal{Q}_D \). It could have been included, but then it should be a nonfinal state—otherwise \( D \) would not be equivalent to \( N \). Since no path starting in it would lead to a final state, it can be omitted as far as the trace set is concerned. That is what we have done.

(End of Technical Remark)

The following relation between the extended transition functions of \( N \) and \( D \) holds.

\[ \hat{\delta}_D(\{ X \}, t) = \begin{cases} \{ \hat{\delta}_N(X, t) \} & \text{if } \hat{\delta}_N(X, t) \neq \emptyset \\ \emptyset & \text{otherwise} \end{cases} \quad \text{for } X \in \mathcal{Q}_D \text{ and } t \in (\Sigma N)^* \text{.} \quad (1.0) \]

Automaton \( M \) is minimal if and only if \( M \) is deterministic and \( \# \mathcal{Q}_M \leq \# \mathcal{Q}_D \) for all deterministic automata \( D \) equivalent to \( M \). All minimal automata that are equivalent to automaton \( N \) trivially have the same number of states (by the way, they are even isomorphic). That number is denoted by \( m_N \). If \( N \) is deterministic then obviously \( m_N \leq \# \mathcal{Q}_N \). From the subset construction we, therefore, know that in general

\[ m_N \leq 2^{\# \mathcal{Q}_N} - 1. \quad (1.1) \]

Automaton \( N \) is called maximally nondeterministic if and only if \( m_N = 2^{\# \mathcal{Q}_N} - 1 \). Hence, \( N \) is maximally nondeterministic if and only if its subset automaton is minimal.

In the following sections we present three families of maximally nondeterministic automata. In each family there is for every integer \( n \), \( n \geq 1 \), one maximally nondeterministic automata with \( n \) states.

The first family is interesting because its definition and correctness proof are very simple. But it uses unnecessarily large alphabets and transition sets (exponential in the number of states). The second family is of interest because it is based on a general "doubling" construction. The sizes of its alphabets and transition sets are linear and quadratic in the number of states respectively. The third family is the compulsive optimizer's delight since it is very economic: all its automata use two symbols and the number of transitions is twice the number of states.

Before embarking on the definitions of these families we characterize minimal automata and maximally nondeterministic automata. Well-known (cf. Corollary 5.3 in [3]) is this

**Maximally Nondeterministic Automata**
Theorem. Deterministic automaton $D$ is minimal if and only if

0) all states of $D$ are accessible, i.e., in the state graph of $D$ there is a path from the root to every state.

and

1) all states of $D$ are distinguishable, i.e., every two distinct states have different trace sets.

(End of Theorem)

The accessibility condition for automaton $D$ can also be expressed as

$$ (A q : q \in qD : (E t : t \in (aD)^* : q \in \delta_D(\{iD\}, t))). $$

Application of relation (1.0) yields the following

Corollary. Automaton $N$ is maximally nondeterministic if and only if

$$ (A X : \emptyset \not= X \subseteq qN : (E t : t \in (aN)^* : X = \hat{\delta}_N(\{iN\}, t))). $$

and

$$ (A X, Y : \emptyset \not= X \subseteq qN \land \emptyset \not= Y \subseteq qN \land X \not= Y
 : \{t \in (aN)^* | \hat{\delta}_N(X, t) \not= \emptyset\} \not= \{t \in (aN)^* | \hat{\delta}_N(Y, t) \not= \emptyset\}). $$

(End of Corollary)

When proving that automaton $N$ is maximally nondeterministic we shall speak of the accessibility and distinguishability of nonempty subsets of $qN$ when dealing with (1.2) and (1.3) respectively. Actually these notions refer to the subset automaton of $N$.

2. A SIMPLE FAMILY

This section presents our first family of maximally nondeterministic automata. It is a simple family in the sense that the proofs are straightforward.

For integer $n \geq 1$, we define automaton $N$ by

$$ qN = \{k \mid 0 \leq k < n \}, $$

$$ aN = \{b_X \mid \emptyset \not= X \subseteq qN\} \cup \{c_j \mid k \in qN\}. $$

$$ \delta_N(k, b_X) = \begin{cases} X & \text{if } k = 0 \\ \emptyset & \text{otherwise} \end{cases} \text{ for } b_X \in aN. $$

$$ \delta_N(k, c_j) = \begin{cases} \{k\} & \text{if } k = j \\ \emptyset & \text{otherwise} \end{cases} \text{ for } c_j \in aN, \text{ and} $$

$$ 1D = 0. $$

Maximally Nondeterministic Automata
Hence, we have

\[ \# qN = n, \]
\[ \# aN = 2^n + n - 1, \] and
\[ \# xN = n.2^{n-1} + n. \]

The automata in this family are maximally nondeterministic.

**Proof.** We show that the accessibility and distinguishability conditions are satisfied.

First we deal with accessibility. Let \( X \) be a nonempty subset of \( qN \) (a state of the subset automaton). Hence, \( b_X \) is a symbol of \( N \). We show that \( X \) is accessible (in the subset automaton) via the trace consisting of the single symbol \( b_X \):

\[ \hat{\delta}_N(\{0\}, b_X) \]
\[ = \{ \text{property (0.2) of} \ \hat{\delta} \} \]
\[ \delta_N(0, b_X) \]
\[ = \{ \text{definition of} \ \delta_N \} \]
\[ X \]

Next comes distinguishability. Let \( X \) and \( Y \) be distinct nonempty subsets of \( qN \). Without loss of generality assume \( j \in X \setminus Y \). We show that the trace consisting of the single symbol \( c_j \) is in the trace set of \( X \) but not in that of \( Y \) (w.r.t. the subset automaton). Hence, \( X \) and \( Y \) are distinguishable states (in the subset automaton). We derive

\[ \hat{\delta}_N(X, c_j) \]
\[ = \{ \text{definition of} \ \hat{\delta}, \text{or properties (0.2) and (0.4)} \} \]
\[ (U k : k \in X : \delta_N(k, c_j)) \]
\[ = \{ \text{definition of} \ \delta_N, \text{using} \ j \in X \} \]
\[ \{ j \} \]
\[ \neq \emptyset \]

and similarly

\[ \hat{\delta}_N(Y, c_j) \]
\[ = \{ \text{as above, using} \ j \notin Y \} \]
\[ \emptyset \]

(End of Proof)

Since these automata have so many symbols and transitions, we do not give examples of their state graphs. One variation that slightly reduces the number of symbols and transitions, is obtained by omitting symbols \( b_{\{k\}} \) for \( 0 \leq k < n \) and \( c_0 \), and all transitions

*Maximally Nondeterministic Automata*
involving these symbols. The set \( \{ k \} \), \( 0 < k < n \), is now accessible via the trace \( b_{qN} c_k \).

For the purpose of distinguishability, \( b_{qN} \) can take over the role of \( c_0 \).

For \( n = 2 \) the state graph of this variation is given below, together with the state graph of its subset automaton. The initial state is underlined. In the state graph of the subset automaton the states are denoted by the concatenation of their elements.

![State Graph Example](image)

### 3. A FAMILY BASED ON THE DOUBLING CONSTRUCTION

In this section we present a construction that can be applied to any automaton \( N \) to yield an automaton \( N' \) with

\[
\begin{align*}
\#qN' &= \#qN + 1. \quad (3.0) \\
\#aN' &= \#aN + 2. \quad (3.1) \\
\#xN' &= \#xN + \#qN + \#aN + 2. \quad (3.2) \\
mN' &= 2mN + 1. \quad (3.3)
\end{align*}
\]

For an obvious reason we call it the doubling construction. From (3.3) and

\[
2(2^n - 1) + 1 = 2^{n+1} - 1
\]

it follows that \( N' \) is maximally nondeterministic if and only if \( N \) is maximally nondeterministic.

Using the doubling construction we can obtain a family of maximally nondeterministic automata in the following way. Define \( N_1 \) as \( \langle \emptyset, \emptyset, \emptyset, 0 \rangle \) and take for \( N_{n+1} \), with \( n \geq 1 \), the result of the doubling construction applied to \( N_n \). \( N_1 \) is deterministic and minimal, since it has only one state. Furthermore, because \( 2^1 - 1 = 1 \), automaton \( N_1 \) is maximally nondeterministic. Hence, by mathematical induction on \( n \) all automata \( N_n \), for \( n \geq 1 \), are maximally nondeterministic. Their parameters are

\[
\begin{align*}
\#qN_n &= n, \\
\#aN_n &= 2n - 2, \text{ and} \\
\#xN_n &= 3n(n - 1)/2.
\end{align*}
\]
Doubling Construction. Let $N$ be an automaton. Then $N'$ is defined by

$$q'N = qN \cup \{q_0\} \quad \text{where} \quad q_0 \notin qN \text{ is a fresh state.}$$

$$a'N = aN \cup \{x, y\} \quad \text{where} \quad x \notin aN \text{ and } y \notin aN \text{ are fresh symbols.}$$

$$\delta_N'(p,b) = \delta_N(p,b) \quad \text{for} \quad p \in qN \text{ and } b \in aN,$$

$$\delta_N'(p,x) = \emptyset \quad \text{for} \quad p \in qN,$$

$$\delta_N'(p,y) = \{p\} \quad \text{for} \quad p \in qN,$$

$$\delta_N'(q_0,b) = \{q_0\} \quad \text{for} \quad b \in aN,$$

$$\delta_N'(q_0,x) = \{\text{in}, q_0\},$$

$$\delta_N'(q_0,y) = \emptyset, \text{ and}$$

$$\text{in}' = q_0.$$

The state graph of $N'$ can be obtained from that of $N$ by adding (i) a transition labeled $y$ from each state to itself, (ii) the new initial state $q_0$, (iii) a transition labeled $x$ from $q_0$ to itself and to in, and (iv) for each symbol of aN a transition labeled with that symbol from q0 to itself.

(End of Doubling Construction)

Examples. We give the state graphs of the automata $N_n$ as discussed above, for $1 \leq n \leq 3$, and also of their subset automata. For the new initial state of $N_{n+1}$ we have chosen $n$: the fresh symbols of $N_{n+1}$ are $a_n$ and $b_n$, playing the roles of $x$ and $y$ respectively. An arrow labeled with a list of symbols abbreviates multiple "parallel" arrows, each labeled with a symbol from that list. Notice that an isomorphic version of $N_2$ appeared in the preceding section as a variation on the simple family.

\[\begin{align*}
N_1 : & 0 \\
N_2 : & a_1 \quad b_1 \\
       & 1 \quad a_1 \quad b_1
\end{align*}\]
That the doubling construction fulfills claims (3.0) through (3.2) is obvious from its definition. We shall now prove that it also satisfies (3.3) in the case that \( N \) is maximally nondeterministic. The general case is left as an exercise to the reader.

**Proof.** Let \( N \) be a maximally nondeterministic automaton and let \( N' \) be the automaton obtained by applying the doubling construction to \( N \). We use \( q_0, x, \) and \( y \) to denote the corresponding objects called for in the doubling construction.

On account of (3.4) it suffices to prove that \( N' \) is maximally nondeterministic. We intend to do this by showing the accessibility and distinguishability conditions for the subset automaton of \( N' \). But first we notice the following properties of the extended transition function of \( N' \). They derive from the properties of \( \delta \) mentioned in Section 0 and the definition of \( \hat{\delta}_N \). For \( X \) a (possibly empty) subset of \( qN \) and \( t \) a trace over \( aN \) we have

\[
\hat{\delta}_N(X, t) = \hat{\delta}_N(X, t),
\]

\[
\hat{\delta}_N(X, x) = \emptyset,
\]

\[
\hat{\delta}_N(X, y) = X.
\]
\[ \hat{\delta}_N(\{q_0\}, t) = \{q_0\}. \]  
\[ \hat{\delta}_N(\{q_0\}, x) = \{1N \cdot q_0\}. \] and
\[ \hat{\delta}_N(\{q_0\}, y) = \emptyset. \]

Now we prove that the accessibility condition is satisfied (cf. (1.2)). Let \( X \) be a nonempty subset of \( qN' \). We distinguish two cases: \( q_0 \in X \) and \( q_0 \not\in X \).

**Case** \( q_0 \in X \). If \( X = \{q_0\} \) then the empty trace trivially suffices. Suppose \( X \setminus \{q_0\} \) is nonempty, and therefore a nonempty subset of \( qN \). On account of (1.2) applied to \( N \), which is assumed maximally nondeterministic, let \( t \) be a trace over \( aN \) such that
\[ \hat{\delta}_N(\{1N \cdot t\}) = X \setminus \{q_0\}. \]
We show that the trace \( xt \) accesses \( X \):
\[ \hat{\delta}_N(\{q_0\}, xt) \]
\[ = \{ \text{property (0.3) of } \hat{\delta} \} \]
\[ \hat{\delta}_N(\hat{\delta}_N(\{q_0\}, x), t) \]
\[ = \{ (3.9) \} \]
\[ \hat{\delta}_N(\{iN \cdot q_0\}, t) \]
\[ = \{ \text{set calculus and property (0.4) of } \hat{\delta} \} \]
\[ \hat{\delta}_N(\{iN \cdot t\} \cup \hat{\delta}_N(\{q_0\}, t) \]
\[ = \{ (3.5) \text{ and (3.8) using that } t \in (aN)^* \} \]
\[ \hat{\delta}_N(\{q_0\}, t) \cup \{q_0\} \]
\[ = \{ \text{definition of } t \} \]
\[ X \setminus \{q_0\} \cup \{q_0\} \]
\[ = \{ \text{set calculus using that } q_0 \in X \} \]

**Case** \( q_0 \not\in X \). On account of the previous case \( X \cup \{q_0\} \) is accessible. Let \( t \) be a trace over \( aN' \) such that
\[ \hat{\delta}_N(\{q_0\}, t) = X \cup \{q_0\}. \]
We show that \( X \) is accessed via the trace \( ty \):
\[ \hat{\delta}_N(\{q_0\}, ty) \]
\[ = \{ \text{property (0.3) of } \hat{\delta} \} \]
\[ \hat{\delta}_N(\hat{\delta}_N(\{q_0\}, t), y) \]
\[ = \{ \text{definition of } t \} \]
\[ \hat{\delta}_N(X \cup \{q_0\}, y) \]
\[ = \{ \text{property (0.4) of } \hat{\delta} \} \]
\[ \hat{\delta}_N(X, y) \cup \hat{\delta}_N(\{q_0\}, y) \]
\[ = \{ (3.7), (3.10), \text{ and set calculus } \}
\[ X \]

*Maximally Nondeterministic Automata*
And now we prove the distinguishability condition (cf. (1.3)). Let $X$ and $Y$ be distinct nonempty subsets of $qN$. We consider three cases: $q_0 \notin X \cup Y$, $q_0 \in X \cup Y$, and $q_0 \in X \cap Y$.

**Case** $q_0 \notin X \cup Y$. Then $X$ and $Y$ are distinct nonempty subsets of $qN$. On account of (1.3) and the assumption that $N$ is maximally nondeterministic there is a trace over $aN$ that distinguishes (the trace sets of) $X$ and $Y$ in (the subset automaton of) $N$. From (3.5) we see that the same trace reveals the distinguishability of $X$ and $Y$ in $N'$.

**Case** $q_0 \in X \cup Y$. Without loss of generality assume $q_0 \in Y$, and thus $X \subseteq qN$. Using property (0.4) of $\hat{\delta}$, the definition of $\hat{\delta}_N$, and (3.6) for $Y \setminus \{q_0\}$, one obtains $\hat{\delta}_N(Y, y) = \{iN, q_0\}$. Together with (3.6) for $X$, this shows that the trace consisting of the single symbol $x$ distinguishes $X$ and $Y$.

**Case** $q_0 \in X \cap Y$. If either $X$ or $Y$ equals $\{q_0\}$, then the trace $y$ distinguishes $X$ and $Y$ on account of (3.7) and (3.10). Otherwise, $X \setminus \{q_0\}$ and $Y \setminus \{q_0\}$ are nonempty and distinct. Hence, the first case applies. Say trace $t$, $t \in (aN)^*$, distinguishes them. Then trace $yt$ distinguishes $X$ and $Y$, since $\hat{\delta}_N(X, y) = X \setminus \{q_0\}$ and similarly for $y$.

(End of Proof)

In general the doubling construction requires two new symbols for each additional state. If, however, it is repeatedly applied to $N_1$—as was done above—then one new symbol suffices. In this special case one can take the "new" $x$ equal to the analogous symbol of the previous automaton. In that way one obtains a family of maximally nondeterministic automata for which the number of symbols equals the number of states (except for the single-state automaton which has an empty alphabet). Below we give a condensed proof of this assertion.

**Examples.** We give the state graphs with 3 and 4 states obtained by carrying out the above variation.

```
\begin{tikzpicture}
  \node (1) at (0,0) {$a_1, b_1$};
  \node (2) at (2,0) {$a_1, b_2$};
  \node (3) at (4,0) {$b_1, b_2$};
  \node (4) at (0,-2) {$a_1$};
  \node (5) at (2,-2) {$a_1$};
  \node (6) at (4,-2) {$a_1$};
  \draw[->] (1) to (2);
  \draw[->] (2) to (3);
  \draw[->] (3) to (1);
  \draw[->] (4) to (5);
  \draw[->] (5) to (6);
  \draw[->] (6) to (4);
\end{tikzpicture}
```

*Maximally Nondeterministic Automata*
The alternative family of automata can also be described nonrecursively. For that description it is convenient to rename the symbol $a_1$ as $b_0$. The transition function $\delta$ of the $n$-state automaton is then given by

$$
\delta(k, b_j) = \begin{cases} 
\{k, k-1\} & \text{if } k \neq j = 0 \\
\{k\} & \text{if } k \neq j > 0 \\
\emptyset & \text{if } k = j
\end{cases}
$$

where $0 \leq k < n$ and $0 \leq j < n$. This allows us to deduce the following properties of the extended transition function. For integer $m$, $0 \leq m < n$, we have

$$
\hat{\delta}((n-1),(b_0)^m) = \{k \mid n - m - 1 \leq k < n\}.
$$

(3.11)

and for $X$ a subset of states and $j$ a state, $j > 0$.

$$
\hat{\delta}(X, b_j) = X \setminus \{b_j\}.
$$

(3.12)

Since $n - 1$ is the initial state of the $n$-state automaton, the accessibility condition (1.2) is now easily verified. Let $X$ be a nonempty set of states. Apply (3.11) with $m = n - 2$ if $0 \notin X$ and with $m = n - 1$ if $0 \in X$. Follow this by a suitable sequence of applications of (3.12) to remove unwanted states and thereby obtain $X$.

To show that the distinguishability condition (1.3) holds, consider two distinct nonempty sets of states $X$ and $Y$. Without loss of generality assume $X \setminus Y$ is nonempty. Define trace $t$ as the sequence of symbols $b_j$ where the indexes form a permutation of $Y$, such that $b_0$ is at the end if $0 \in Y$. Trace $t$ distinguishes $X$ and $Y$ on account of (3.12) and $\delta([0], b_0) = \emptyset$.

The automata of the alternative family can be reduced further by omitting those transitions labeled with symbol $b_0$ ( = $a_1$) that go from state $k$ to itself, where $k$ is not the initial state. This reduction does not affect properties (3.11) and (3.12) and the fact that $\delta([0], b_0) = \emptyset$. The resulting automata have only one state that violates the condition for being deterministic, viz. the initial state.
4. A FAMILY WITH ONLY TWO SYMBOLS

A moment's thought will convince the reader that a maximally nondeterministic automaton with more than one state requires an alphabet with at least two symbols. The family presented in this section shows that two symbols suffice for any number of states.

For integer \( n, n \geq 2 \), automaton \( N \) is defined by

\[
\begin{align*}
q_N &= \{ k \mid 0 \leq k < n \}, \\
a_N &= \{ a, b \}, \\
\delta_N(0, a) &= \{ 0, 1 \}, \\
\delta_N(0, b) &= \emptyset, \\
\delta_N(k, a) &= \{(k + 1) \mod n \} \quad \text{for} \quad 0 < k < n, \\
\delta_N(k, b) &= \{ k \} \quad \text{for} \quad 0 < k < n, \text{ and} \\
i_N &= 0.
\end{align*}
\]

We do not bother to make an explicit choice for a single-state automaton of this family. It shall not concern us here. The automata are not indexed with \( n \), since \( n \) will be fixed throughout this section. We shall also omit the subscript \( N \) on \( \delta_N \) and \( i_N \).

The family has the following parameters.

\[
\begin{align*}
\#q_N &= n, \\
\#a_N &= 2, \text{ and} \\
\#x_N &= 2n.
\end{align*}
\]

**Examples.** We give the state graphs for the automata with 2, 3, and 6 states.
Notice that the initial state is the only state that violates the condition for being deterministic: it has two outgoing arrows with label \( a \). Also notice that the initial state is the only state that does not have outgoing arrows with both labels \( a \) and \( b \).

We use a special notation for sets of states—viz. bit vectors—to prove that these automata are maximally nondeterministic.

The characteristic function over \( Q \) of the set \( X \), \( X \subseteq Q \), is the mapping \( f \) from \( Q \) into \( \{0, 1\} \) defined by

\[
f(q) = \begin{cases} 
0 & \text{if } q \notin X \\
1 & \text{if } q \in X
\end{cases} \quad \text{for } q \in Q.
\]

A bit vector is just a way of writing the characteristic function of a set. We are interested in subsets of \( Q^N \). The bit vector of \( X \), \( X \subseteq Q^N \), is the string obtained by concatenating the images of \( X \)'s characteristic function over \( Q \), while traversing the arguments (elements of \( Q^N \)) in increasing order. These bit vectors all have length \( n \). We shall apply standard abbreviations regarding strings to bit vectors as well. For example, the empty set has bit vector \( 0^n \) and the initial state of \( N \)'s subset automaton, i.e. the set \( \{0\} \), has bit vector \( 10^{n-1} \).

In what follows \( t \) denotes a trace over \( aN \), \( u \), \( v \), and \( w \) denote bit vectors, and \( x \), \( y \), and \( z \) denote bits (bit vectors of length 1). The lengths of \( u \), \( v \), and \( w \) are implicitly assumed to be such that the bit vector in which they appear has the proper length.

The automata of the two-symbol family are maximally nondeterministic.

**Proof.** Let \( n \) be an integer, \( n \geq 2 \). We consider the automaton \( N \) with \( n \) states of this family. From its definition and the properties of \( \hat{S} \) as mentioned in Section 0, one can derive the following properties of \( N \)'s extended transition function.

*Maximally Nondeterministic Automata*
\( S(u_a) = 1u. \)  
(4.0)
\( S(0u_y, a) = 0u. \) and  
(4.1)
\( S(xv, b) = 0v. \)  
(4.2)

When taking \( y = 1 \), properties (4.0) and (4.1) can be combined into
\( S(v_1, a) = 1v. \)  
(4.3)

From properties (4.0), (4.1), and (4.2) also follow
\( S(vy, ab) = 0v, \) and  
(4.4)
\( S(xuy, ba) = y0u. \)  
(4.5)

First, we prove by mathematical induction on the length of \( u \) (denoted by \( k \)) that the nonempty set of states with bit vector \( u1^{n-k} \) is accessible. Taking \( k = n \) then shows that the accessibility condition (1.2) is satisfied.

**Base** \( u = \epsilon \). Repeated application of (4.0) yields
\( \hat{S}(10^{n-1}, a^{n-1}) = 1^n, \)
which shows that \( 1^n \) is accessible.

**Step** \( u = xv \). On account of the induction hypothesis let \( t \) be a trace over \( aN \) with
\( \hat{S}(01^{n-1}, t) = v1^{n-k+1} \)
Since \( k \leq n \) we can apply (4.3) and (4.4) to \( v1^{n-k+1} \). Together with (0.3) and the definition of \( t \) this gives
\( \hat{S}(10^{n-1}, ta) = 1v1^{n-k}, \) and  
\( \hat{S}(10^{n-1}, tab) = 0v1^{n-k}. \)

Distinguishing the cases \( x = 1 \) and \( x = 0 \), this shows that \( xu1^{n-k} \) is accessed via traces \( ta \) and \( tab \) respectively.

Finally, we prove that the distinguishability condition (1.3) holds. Let \( u \) and \( v \) be the bit vectors of two distinct nonempty sets of states. Because the sets are nonempty, \( u \) can be written as \( 0^{n-1-k}1u_0 \), and \( v \) as \( 0^{n-1-l}1v_0 \), where \( k \) and \( l \) are the lengths of \( u_0 \) and \( v_0 \) respectively. We distinguish two cases: \( k \neq l \) and \( k = l \).

**Case** \( k \neq l \). Without loss of generality assume \( k > l \). We claim that the trace \( (ab)^{l+1} \) belongs to the trace set of \( u \) but not to that of \( v \). Write \( u_0 \) as \( u_1u_2 \), where the length of \( u_2 \) is \( l+1 \). We compute
\[
\hat{S}(u, (ab)^{l+1}) = \{ \text{rewriting } u \} \]
\[
\hat{\delta}(0^{n-1-k} 1u_1 u_2, (ab)^{k+1}) \\
= \{ \text{repeated application of string calculus. (0.3), and (4.4), using } k > l \} \\
0^n - k + l 1u_1 \\
= \{ \text{string calculus} \} \\
0^n
\]

and

\[
\hat{\delta}(v, (ab)^{l+1}) \\
= \{ \text{rewriting } v \} \\
\hat{\delta}(0^{n-1-l} 1v_0, (ab)^{l+1}) \\
= \{ \text{repeated application of string calculus. (0.3), and (4.4)} \} \\
0^n
\]

Case \( k = l \). Since \( u \) and \( v \) are distinct we can rewrite \( u_0 \) as \( u_1 y u_2 \) and \( v_0 \) as \( v_1 z v_2 \), such that \( y \neq z \) and \( v_2 \) has the same length as \( v_2 \). Let \( m \) be the length of \( u_2 \). Consider the trace \((ab)^m ba\). Writing \( xw \) for \( 0^{n-1-k+m} 1 \), we compute

\[
\hat{\delta}(u, (ab)^m ba) \\
= \{ \text{rewriting } u, \text{string calculus, and (0.3)} \} \\
\hat{\delta}(0^{n-1-k} 1u_1 y u_2, (ab)^m), ba) \\
= \{ m \text{ applications of (4.4), and definition of } xw \} \\
\hat{\delta}(xw u_1 y, ba) \\
= \{ (4.5) \} \\
y 0 w u_1
\]

Similarly we obtain \( \hat{\delta}(v, (ab)^m ba) = z 0 w v_1 \). Because \( y \neq z \) the previous case applies to \( y 0 w u_1 \) and \( z 0 w v_1 \) (they indeed represent distinct nonempty sets). If these states can be distinguished by trace \( t \), then it follows that trace \((ab)^n bat\) distinguishes \( u \) and \( v \).

(End of Proof)

The two-symbol automata discussed above are a variation on a family suggested by Berry Schoenmakers. For completeness' sake the state graphs of his automata with 3 and 6 states are depicted below. The proofs for these automata can be given along the same lines, but they are somewhat more complicated (especially accessibility).
5. CONCLUDING REMARKS

Jo Ebergen brought the question whether upper bound (1.1) can be attained under my attention. Everyone that I asked about it told me that maximally nondeterministic automata—attaining this upper bound—were indeed known to exist. They could not, however, give me any references. Exercise 2.2.27 of [2], for example, poses the problem, but no solution is provided. Finally, I set out to construct them myself. This resulted in the first two families. Berry Schoenmakers put me on the scent of the third family. Only afterwards did I hit upon Theorem 1.10 of [5], which exhibits a family of maximally nondeterministic automata with three symbols. References [3] and [4] are also good introductions to automata theory, but they don’t mention the problem. Nevertheless both have somehow influenced my presentation.

The class of nondeterministic automata considered in this note is restricted in the sense that the corresponding trace sets are nonempty and prefix-closed. This is the class that is most prominent in the circuit synthesis methods. It was imaginable that the upper bound (1.1) could be attained by more general nondeterministic automata but not within the restricted class. It turns out, however, that the upper bound is attained even in the restricted class.

Another restricted class that is of interest consists of the automata obtained from deterministic ones by concealment of certain symbols. Concealed symbols can be modeled by transitions without symbols (also called epsilon transitions; cf. [4], [0]). Upper bound (1.1) cannot be attained within this class, as is shown in [6]. The tightest upper bound for this class is

\[ mN \leq 3 \cdot 2^* q^{N-2} - 1 \]

for automata with at least two states. The upperbound is attained by the second variation discussed in Section 3 by concealing the symbol on the transition from the initial state to the “next” state.
I have given the definitions and correctness proofs in such detail because it is very easy to come up with sloppy suggestions that afterwards turn out to be ill-defined or plainly wrong. Thanks are due to a number of colleagues for pointing out shortcomings of a—much shorter—draft version.

REFERENCES


Maximally Nondeterministic Automata
## COMPUTING SCIENCE NOTES

In this series appeared:

<table>
<thead>
<tr>
<th>No.</th>
<th>Author(s)</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>85/01</td>
<td>R.H. Mak</td>
<td>The formal specification and derivation of CMOS-circuits</td>
</tr>
<tr>
<td>85/02</td>
<td>W.M.C.J. van Overveld</td>
<td>On arithmetic operations with M-out-of-N-codes</td>
</tr>
<tr>
<td>85/03</td>
<td>W.J.M. Lemmens</td>
<td>Use of a computer for evaluation of flow films</td>
</tr>
<tr>
<td>85/04</td>
<td>T. Verhoeff, H.M.J.L. Schols</td>
<td>Delay insensitive directed trace structures satisfy the foam rubber wrapper postulate</td>
</tr>
<tr>
<td>86/01</td>
<td>R. Koymans</td>
<td>Specifying message passing and real-time systems</td>
</tr>
<tr>
<td>86/02</td>
<td>G.A. Bussing, K.M. van Hee, M. Voorhoeve</td>
<td>ELISA, A language for formal specifications of information systems</td>
</tr>
<tr>
<td>86/03</td>
<td>Rob Hoogerwoord</td>
<td>Some reflections on the implementation of trace structures</td>
</tr>
<tr>
<td>86/04</td>
<td>G.J. Houben, J. Paredaens, K.M. van Hee</td>
<td>The partition of an information system in several parallel systems</td>
</tr>
<tr>
<td>86/05</td>
<td>Jan L.G. Dietz, Kees M. van Hee</td>
<td>A framework for the conceptual modeling of discrete dynamic systems</td>
</tr>
<tr>
<td>86/06</td>
<td>Tom Verhoeff</td>
<td>Nondeterminism and divergence created by concealment in CSP</td>
</tr>
<tr>
<td>86/07</td>
<td>R. Gerth, L. Shira</td>
<td>On proving communication closedness of distributed layers</td>
</tr>
<tr>
<td>Year/Issue</td>
<td>Authors</td>
<td>Title</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>86/09</td>
<td>C. Huizing, R. Gerth, W.P. de Roever</td>
<td>Full abstraction of a real-time denotational semantics for an OCCAM-like language</td>
</tr>
<tr>
<td>86/10</td>
<td>J. Hooman</td>
<td>A compositional proof theory for real-time distributed message passing</td>
</tr>
<tr>
<td>86/11</td>
<td>W.P. de Roever</td>
<td>Questions to Robin Milner - A responder's commentary (IFIP86)</td>
</tr>
<tr>
<td>86/12</td>
<td>A. Boucher, R. Gerth</td>
<td>A timed failures model for extended communicating processes</td>
</tr>
<tr>
<td>86/14</td>
<td>R. Koymans</td>
<td>Specifying passing systems requires extending temporal logic</td>
</tr>
<tr>
<td>87/01</td>
<td>R. Gerth</td>
<td>On the existence of sound and complete axiomatizations of the monitor concept</td>
</tr>
<tr>
<td>87/02</td>
<td>Simon J. Klaver, Chris F.M. Verberne</td>
<td>Federatieve Databases</td>
</tr>
<tr>
<td>87/03</td>
<td>G.J. Houben, J.Paredaens</td>
<td>A formal approach to distributed information systems</td>
</tr>
<tr>
<td>87/04</td>
<td>T. Verhoeff</td>
<td>Delay-insensitive codes - An overview</td>
</tr>
<tr>
<td>Year</td>
<td>Author(s)</td>
<td>Title</td>
</tr>
<tr>
<td>------</td>
<td>-------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>87/05</td>
<td>R.Kuiper</td>
<td>Enforcing non-determinism via linear time temporal logic specification.</td>
</tr>
<tr>
<td>87/06</td>
<td>R.Koymans</td>
<td>Temporele logica specificatie van message passing en real-time systemen (in Dutch).</td>
</tr>
<tr>
<td>87/07</td>
<td>R.Koymans</td>
<td>Specifying message passing and real-time systems with real-time temporal logic.</td>
</tr>
<tr>
<td>87/08</td>
<td>H.M.J.L. Schols</td>
<td>The maximum number of states after projection.</td>
</tr>
<tr>
<td>87/09</td>
<td>J. Kalisvaart</td>
<td>Language extensions to study structures for raster graphics.</td>
</tr>
<tr>
<td></td>
<td>L.R.A. Kessener</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W.J.M. Lemmens</td>
<td></td>
</tr>
<tr>
<td></td>
<td>M.L.P. van Lierop</td>
<td></td>
</tr>
<tr>
<td></td>
<td>F.J. Peters</td>
<td></td>
</tr>
<tr>
<td></td>
<td>H.M.M. van de Wetering</td>
<td></td>
</tr>
<tr>
<td>87/10</td>
<td>T.Verhoeff</td>
<td>Three families of maximally nondeterministic automata.</td>
</tr>
<tr>
<td>87/11</td>
<td>P.Lemmens</td>
<td>Eldorado ins and outs. Specifications of a data base management toolkit according to the functional model.</td>
</tr>
</tbody>
</table>