Information Processing Letters 79 (2001) 105–109

Information Processing Letters

www.elsevier.com/locate/ipl

Efficient minimization of deterministic weak ω-automata

Christof Löding

Lehrstuhl Informatik VII, RWTH Aachen, 52056 Aachen, Germany

Received 26 April 2000; received in revised form 16 October 2000
Communicated by F.B. Schneider

Abstract

We analyze the minimization problem for deterministic weak automata, a subclass of deterministic Büchi automata, which recognize the regular languages that are recognizable by deterministic Büchi and deterministic co-Büchi automata. We reduce the problem to the minimization of finite automata on finite words and obtain an algorithm running in time $O(n \cdot \log n)$, where n is the number of states of the automaton. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: ω-automaton minimization; Computational complexity; Algorithms

1. Introduction

In contrast to deterministic finite automata on finite words (DFA), for ω-automata in general there is no unique minimal automaton and so far there is no characterization of minimal state automata in terms of congruences over words as for DFA. In [1] Staiger identified a "good" subclass of ω-languages that can be recognized by such minimal state automata derived from a congruence over finite words. These are the regular ω-languages from the topological class $G_\delta \cap F_\sigma$. In [2] Gutleben investigated the algorithmic aspect of minimizing Muller-automata for this fragment and obtained an algorithm running in time $O(n^2 \cdot \log n)$ (for an n state Muller-automaton).

In this paper we consider the model of deterministic weak automata (DWA), which form a subclass of deterministic Büchi (respectively Muller) automata and exactly capture the regular languages from $G_\delta \cap F_\sigma$ [3–5]. We obtain an $O(n \cdot \log n)$ time minimization algorithm by applying a minimization algorithm for DFA. The key of this method is the construction of a suitable set of final states of a DWA such that it is "good" in the following sense: The standard minimization algorithm applied as for DFA yields a minimal DWA for the ω-language under consideration. In a DWA there may be states that cannot occur infinitely often during a run of the automaton. Thus, the type of these states (final or non-final) does not play any role for acceptance. We shall show (in Section 3) that it is possible to fix these types such that a "good" DWA is obtained (allowing minimization as for DFA); moreover we prove that this declaration of final and non-final states can be done in linear time.

2. Preliminaries

For a finite set Σ we denote by Σ^* the set of finite words over Σ and by Σ^ω the set of infinite words over Σ. The empty word is denoted by ε. A deterministic finite automaton (DFA) is of the form $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$, where Q is a finite set of states,

E-mail address: loeding@i7.informatic.rwth-aachen.de (C. Löding).

Σ is a finite input alphabet, q_0 is the initial state, $\delta : Q \times \Sigma \to Q$ is the transition function, and $F \subseteq Q$ is the set of final states. We extend δ to a function $\delta : Q \times \Sigma^* \to Q$ by $\delta(q, \varepsilon) = q$ and $\delta(q, aw) = \delta(\delta(q, a), w)$ for $q \in Q$, $a \in \Sigma$, and $w \in \Sigma^*$ as usual. We write $\mathcal{A}_q$ for the automaton that corresponds to $\mathcal{A}$ except for the initial state that is changed to q. The DFA $\mathcal{A}$ defines a language

$$L_*(\mathcal{A}) = \{w \in \Sigma^* \mid \delta(q_0, w) \in F\}.$$

We can also view a DFA as a deterministic Büchi automaton (DBA) [6]. A run of the DBA $\mathcal{A}$ on $\alpha \in \Sigma^\omega$ is a sequence $\rho \in Q^\omega$ with $\rho(0) = q_0$ and $\rho(i+1) = \delta(\rho(i), \alpha(i))$ for all $i \in \mathbb{N}$, where $\rho(i)$ denotes the ith position in ρ. The *infinity set* of ρ is

$$In(\rho) = \{q \in Q \mid \rho(i) = q \text{ for infinitely many } i \in \mathbb{N}\}.$$

The *occurrence set* of ρ is

$$Oc(\rho) = \{q \in Q \mid \text{there is an } i \in \mathbb{N} \text{ with } \rho(i) = q\}.$$

A run is accepting iff $In(\rho) \cap F \neq \emptyset$ and α is accepted iff its run is accepting. The ω-language accepted by $\mathcal{A}$ is

$$L_\omega(\mathcal{A}) = \{\alpha \in \Sigma^\omega \mid \mathcal{A} \text{ accepts } \alpha\}.$$

For a language $U \subseteq \Sigma^*$ of finite words we define $\overrightarrow{U} = \{\alpha \in \Sigma^\omega \mid \alpha \text{ has infinitely many prefixes in } U\}$. The following fact is well known (see, e.g., [7]).

Proposition 1. *Let $\mathcal{A}$ be a DFA. Then $L_\omega(\mathcal{A}) = \overrightarrow{L_*(\mathcal{A})}$.*

A state $q \in Q$ is called *recurrent* iff there is a $w \in \Sigma^* \setminus \{\varepsilon\}$ with $\delta(q, w) = q$. Otherwise q is called *transient*. A strongly connected component (SCC) $S \subseteq Q$ of $\mathcal{A}$ is a maximal subset of Q such that q is reachable from p for each $p, q \in S$. Transient states form a strongly connected component of size 1. An SCC is called transient if it consists of a transient state, otherwise it is called recurrent. Deterministic weak automata (DWA) form a subclass of DBA. A DBA is called *weak* iff every SCC of $\mathcal{A}$ only contains final states or only contains non-final states. According to this we call an SCC final or non-final.

3. Maximal colorings

Minimization for DFA is well investigated. We want to apply an algorithm for the minimization of DFA for minimizing DWA. For that aim we shall change the set of final states without changing the language of the DWA, however such that the standard minimization algorithm for DFA yields a minimal equivalent DWA. In this section we define colorings which will be used to compute an appropriate set of final states which allows this approach to minimization.

Definition 2. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ be a DWA and let $k \in \mathbb{N}$. A mapping $c : Q \to \mathbb{N}$ is called an *$\mathcal{A}$-coloring* iff $c(q)$ is even for every recurrent state $q \in F$, $c(q)$ is odd for every recurrent state $q \notin F$, and $c(p) \leqslant c(q)$ for every $p, q \in Q$ with $\delta(p, a) = q$ for some $a \in \Sigma$. The coloring c is called *k-maximal* iff

(1) for every $\mathcal{A}$-coloring $c' : Q \to \{0, \ldots, k\}$ and for every state $q \in Q$ one has $c'(q) \leqslant c(q)$, and
(2) $c(q) \leqslant k$ for all $q \in Q$.

A coloring is called *maximal* if it is k-maximal for some k.

If we choose k minimal, then the values of a maximal $\mathcal{A}$-coloring correspond to Wagner's super chains [4], i.e., his n^+ and n^- values.

The notion of an $\mathcal{A}$-coloring allows an alternative formulation of acceptance reminding of "parity automata" (see, e.g., [7]). For a run ρ of an automaton and a coloring c we denote by $c(\rho)$ the sequence $c(\rho(0))c(\rho(1))c(\rho(2))\cdots$.

Remark 3. A run $\rho \in Q^\omega$ is accepting iff $\max(Oc(c(\rho)))$ is even.

Definition 4. For an $\mathcal{A}$-coloring c let $F_c = \{q \in Q \mid c(q) \text{ is even}\}$. The DWA $\mathcal{A}$ is said to be in *normal form* iff $F = F_c$ for some k-maximal $\mathcal{A}$-coloring c with k even. (We could also make this definition for k odd. We just have to fix some parity to get a unique minimal DWA.)

For the next considerations we assume $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ to be a DWA and $c : Q \to \mathbb{N}$ to be a k-maximal $\mathcal{A}$-coloring for some $k \in \mathbb{N}$.

Remark 5. For each $p \in Q$ with $c(p) \leqslant k - 2$ there is a $q \in Q$ such that q is reachable from p and $c(q) = c(p) + 1$.

Proof. Suppose there is a $p \in Q$ not satisfying this property. Let $P = \{q \in Q \mid q$ is reachable from p and $c(q) = c(p)\}$, where we also consider p to be reachable from p. Define a new $\mathcal{A}$-coloring $c' : Q \to \{0, \dots, k\}$ by $c'(q) = c(q)$ if $q \notin P$ and $c'(q) = c(q) + 2$ if $q \in P$. Then c' is an $\mathcal{A}$-coloring contradicting the k-maximality of c. □

Remark 6. For each $q \in Q$ there is an $\alpha \in \Sigma^\omega$ with run ρ of $\mathcal{A}_q$ on α such that $\max(Oc(c(\rho))) = c(q)$.

Proof. For recurrent states the claim obviously holds. So let $q \in Q$ be transient. If the claim does not hold, then there is no recurrent state reachable from q that has the same color as q. But then we can define a new coloring by just adding 1 to the color of all states that are reachable from q and have the same color as q. This coloring is obviously an $\mathcal{A}$-coloring and contradicts the k-maximality of c. □

Lemma 7. *Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ be a DWA and $c : Q \to \mathbb{N}$ be a k-maximal $\mathcal{A}$-coloring for some $k \in \mathbb{N}$. For all $p, q \in Q$ if $L_\omega(\mathcal{A}_p) = L_\omega(\mathcal{A}_q)$, then $c(p) = c(q)$.*

Proof. Assume by contradiction that there are $p, q \in Q$ with $L_\omega(\mathcal{A}_p) = L_\omega(\mathcal{A}_q)$ and $c(p) \neq c(q)$. Choose such p, q such that $c(p) + c(q)$ is maximal and $c(p) < c(q)$. We show that $c(q) = c(p) + 1$. If $c(p) > k - 2$, then $c(q) = k$ and therefore $c(p) = k - 1$. If $c(p) \leqslant k - 2$, then there is a $w \in \Sigma^*$ such that $c(r) = c(p) + 1$ for $\delta(p, w) = r$ (by Remark 5). Let $s = \delta(q, w)$. Because of $L_\omega(\mathcal{A}_p) = L_\omega(\mathcal{A}_q)$ we have $L_\omega(\mathcal{A}_r) = L_\omega(\mathcal{A}_s)$ and because of the maximality condition on p and q we get $c(r) = c(s)$ and since $c(s) \geqslant c(q)$ and $c(q) \geqslant c(r)$ we also get $c(q) = c(r)$. Hence, also in this case $c(q) = c(p) + 1$.

Now let $\alpha \in \Sigma^\omega$ be such that $\max(Oc(c(\rho))) = c(p)$ for the run ρ of $\mathcal{A}_p$, according to Remark 6. If in the run ρ' of $\mathcal{A}_q$ on α a color greater than $c(q)$ occurs, let w be a prefix of α with $c(\delta(q, w)) > c(q)$. Since w is a prefix of α we have $c(\delta(p, w)) = c(p)$. Let $r = \delta(p, w)$ and $s = \delta(q, w)$. Then $L_\omega(\mathcal{A}_r) = L_\omega(\mathcal{A}_s)$, $c(r) \neq c(s)$, and $c(s) + c(r) > c(p) + c(q)$ which is a contradiction to the choice of p and q. Therefore $\max(Oc(c(\rho))) = c(p)$ and $\max(Oc(c(\rho'))) = c(q) = c(p) + 1$. Thus, $\alpha \in L_\omega(\mathcal{A}_p)$ iff $\alpha \notin L_\omega(\mathcal{A}_q)$ (by Remark 3) contradicting the assumption. □

Finally we show how to transform a DWA $\mathcal{A}$ into normal form. For the time complexity of this computation we assume that $|\Sigma|$ is a constant.

Theorem 8. *For a given DWA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ with n states there exists a set $F' \subseteq Q'$ such that $\mathcal{A}' = (Q, \Sigma, q_0, \delta, F')$ is in normal form and equivalent to $\mathcal{A}$. This set F' can be computed in time $\mathrm{O}(n)$.*

Proof. The main task is to compute a k-maximal $\mathcal{A}$-coloring. We fix k to be some even number greater than n to be sure that k is large enough. The problem can be reduced to finding a coloring of the SCC graph of $\mathcal{A}$ since states in the same SCC get the same color. Let $Q_1, \dots, Q_m$ be the SCCs of $\mathcal{A}$. The SCC graph $G = (V, E)$ has the vertex set $\{1, \dots, m\}$ and the edge relation is defined by $(i, j) \in E$ iff $i \neq j$ and there are states $q_i \in Q_i$ and $q_j \in Q_j$ with $\delta(q_i, a) = q_j$ for some $a \in \Sigma$. We identify $i \in \{1, \dots, m\}$ with its SCC Q_i. This SCC graph can be computed in linear time by standard SCC algorithms [8]. Furthermore we can assume that the transient SCCs are marked. The size of G is bounded by the size of $\mathcal{A}$, i.e., $|V| + |E| \leqslant n + |\Sigma| n$. To find a k-maximal $\mathcal{A}$-coloring c we shall find a mapping $d : V \to \{0, \dots, k\}$ with maximal values such that $d(i)$ is even if Q_i is recurrent and final, $d(i)$ is odd if Q_i is recurrent and non-final, and $d(i) \leqslant d(j)$ if $(i, j) \in E$. The d-value of an SCC then corresponds to the c-value of its states. With an algorithm for topological sorting [8] we can determine in linear time a permutation $v_1, \dots, v_m$ of $1, \dots, m$ such that if $(v_i, v_j) \in E$, then $i < j$. The algorithm from Fig. 1 computes the mapping d, where $succ(i)$ denotes the E-successors of $i \in V$. For the running time of the **for**-loop the **if**-statements clearly are not critical. The minimum in line 8 is computed m times, but the total number of steps for this is in $\mathrm{O}(|E|)$ which is bounded by $|\Sigma| \cdot n$. Thus, the algorithm runs in time $\mathrm{O}(n)$. The property of the list $(v_1, \dots, v_m)$ to be topologically sorted guarantees that the d-value of all successors of v_i is already computed when v_i is treated. If we assume that the value for the successors of v_i is maximal, then clearly the value for v_i is maximal. This assumption is justified because the value for the SCCs without successors is maximal. From d we can compute the corresponding k-maximal $\mathcal{A}$-coloring c. Given such a coloring c we obviously can compute the set F_c in time $\mathrm{O}(n)$. By the definition

```
$G = \text{SCCgraph}(\mathcal{A})$
$(v_1, \ldots, v_m) = \text{topsort}(G)$
**for** $i = m$ to 1
  **if** $succ(v_i) = \emptyset$ **then**
    **if** $v_i$ is final **then** $d(v_i) = k$
    **else** $d(v_i) = k - 1$
  **else**
    $l = \min\{d(j) \mid j \in succ(v_i)\}$
    **if** $v_i$ is transient **then** $d(v_i) = l$
    **else**
      **if** $l$ is even and $v_i$ is final **then** $d(v_i) = l$
      **else**
        **if** $l$ is odd and $v_i$ is not final **then** $d(v_i) = l$
        **else** $d(v_i) = l - 1$
```

Fig. 1. An algorithm for a maximal coloring on the SCC graph.

of $\mathcal{A}$-colorings the automaton $(Q, \Sigma, q_0, \delta, F')$ with $F' = F_c$ is equivalent to $\mathcal{A}$. $\Box$

4. Minimization

We use the terminology of congruences. For a language $L \subseteq \Sigma^\omega$ define the right congruence $\sim_L \subseteq \Sigma^* \times \Sigma^*$ by $u \sim_L v$ iff for all $\alpha \in \Sigma^\omega : u\alpha \in L \Leftrightarrow v\alpha \in L$. For $u \in \Sigma^*$ we denote the $\sim_L$-class of u by $[u]_L$. The index of $\sim_L$, i.e., the number of $\sim_L$-classes, is denoted by $Ind(\sim_L)$. We call a DWA minimal iff there is no equivalent DWA with less states. Given a language $L \subseteq \Sigma^\omega$ that can be recognized by a DWA we show that every DWA recognizing L has hat least $Ind(\sim_L)$ states.

Lemma 9. *Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ be a DWA and let $L = L_\omega(\mathcal{A})$. For each $q \in Q$ that is reachable from q_0 there is a $u_q \in \Sigma^*$ such that $u \in [u_q]_L$ for all $u \in \Sigma^*$ with $\delta(q_0, u) = q$. In particular we get $|Q| \geqslant Ind(\sim_L)$ and if $L_\omega(\mathcal{A}_p) \neq L_\omega(\mathcal{A}_q)$ for all $p, q \in Q$, then $\mathcal{A}$ is a minimal DWA.*

Proof. Let $q \in Q$ and let $u, v \in \Sigma^*$ with $\delta(q_0, u) = q = \delta(q_0, v)$. Since q is reachable from q_0 such u and v exist. It suffices to verify that $u \sim_L v$. Let $\alpha \in \Sigma^\omega$. $\mathcal{A}$ accepts $u\alpha$ iff it accepts $v\alpha$ because the runs can only differ on the u and v prefixes and acceptance only depends on the infinity set of the runs. $\Box$

As shown in [1] the above lemma even holds for DBA in general. Also in [1] it is shown that the lower bound from Lemma 9 in fact also is the upper bound for minimal DWA. Similar to the case of DFA a minimal state automaton for a DWA recognizable language L with the $\sim_L$-classes as states is defined. In the following we show how to compute from a given DWA an equivalent minimal DWA. For a DWA $\mathcal{A}$ let $\mathcal{A}^{\min}$ be the minimal DFA equivalent to the DFA $\mathcal{A}$. From Proposition 1 it is clear that $L_\omega(\mathcal{A}) = L_\omega(\mathcal{A}^{\min})$. But in general $\mathcal{A}^{\min}$ is not a minimal DWA. We show that if $\mathcal{A}$ is in normal form, then $\mathcal{A}^{\min}$ is a minimal DWA.

Lemma 10. *Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ be a DWA in normal form. If $L_*(\mathcal{A}_p) \neq L_*(\mathcal{A}_q)$ for some $p, q \in Q$, then $L_\omega(\mathcal{A}_p) \neq L_\omega(\mathcal{A}_q)$.*

Proof. Since $\mathcal{A}$ is in normal form, there is a maximal $\mathcal{A}$-coloring c with $F = F_c$. Let $p, q \in Q$ with $L_*(\mathcal{A}_p) \neq L_*(\mathcal{A}_q)$. This means there is a word $w \in \Sigma^*$ with $\delta(p, w) \in F$ iff $\delta(q, w) \notin F$. Let $r = \delta(p, w)$ and let $s = \delta(q, w)$. Since $F = F_c$ this means that $c(r) \neq c(s)$ and therefore $L_\omega(\mathcal{A}_r) \neq L_\omega(\mathcal{A}_s)$ by Lemma 7. Let $\alpha \in \Sigma^\omega$ with $\alpha \in L_\omega(\mathcal{A}_r)$ iff $\alpha \notin L_\omega(\mathcal{A}_s)$. Then $w\alpha \in L_\omega(\mathcal{A}_p)$ iff $w\alpha \notin L_\omega(\mathcal{A}_q)$. $\Box$

We also want to show that we obtain a minimal DWA that does not depend on the given DWA but only on the ω-language defined by the given DWA. For this we need the following lemma.

Lemma 11. *Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ and $\mathcal{A}' = (Q', \Sigma, q_0', \delta', F')$ be two DWA in normal form with $L_\omega(\mathcal{A}) = L_\omega(\mathcal{A}')$. Then $L_*(\mathcal{A}) = L_*(\mathcal{A}')$.*

Proof. Let c be an l-maximal $\mathcal{A}$-coloring and let c' be a l'-maximal $\mathcal{A}'$-coloring such that $F = F_c$ and $F' = F_{c'}$. Since $\mathcal{A}$ and $\mathcal{A}'$ are in normal form, l and l' are even. Thus, we can also assume that c and c' are k-maximal for some even k large enough by adding appropriate constants to the c and c' values.

For $u \in \Sigma^*$ let $q_u = \delta(q_0, u)$ and $q_u' = \delta'(q_0', u)$. Assume there is a $u \in \Sigma^*$ with $c(q_u) \neq c'(q_u')$. We consider the case $c(q_u) < c(q_u')$. The other case is symmetrical. Define the mapping $d : Q \to \{0, \ldots, k\}$ by

$$d(q_v) = \max\{c(q_v), c'(q_v')\} \quad \text{for each } v \in \Sigma^*.$$

By Lemmas 9 and 7 we know that this definition of the d-value of a state does not depend on the choice of v. This means if $q_w = q_v$, then $c'(q'_w) = c'(q'_v)$. Obviously $d(q_u) > c(q_u)$. Thus, if we can show that d is an $\mathcal{A}$-coloring, then we have a contradiction because c is a k-maximal $\mathcal{A}$-coloring. By the definition of d it is clear that for $q \in Q$ and $a \in \Sigma$ we have $d(q) \leqslant d(\delta(q,a))$. So let $q \in F$ be a recurrent state and let $v, w \in \Sigma^*$ be such that $\delta(q_0, v) = q$ (i.e., $q = q_v$) and $\delta(q, w) = q$. If $d(q)$ is odd, then $d(q) \neq c(q)$ and therefore $d(q) = c'(q'_v)$. Since $\mathcal{A}$ on $\alpha := vw^\omega$ infinitely often visits q, α contains infinitely many prefixes that are $\sim_L$-equivalent to v (by Lemma 9). Therefore $\mathcal{A}'$ on α visits infinitely often a state q' with $L_\omega(\mathcal{A}'_{q'}) = L_\omega(\mathcal{A}'_{q'_v})$. By Lemma 7 we know that $c'(q') = c'(q'_v)$ and thus the maximal color visited by $\mathcal{A}'$ on α is $c'(q'_v) = d(q)$. But then $\mathcal{A}'$ rejects α and $\mathcal{A}$ accepts α, a contradiction. Thus, the d-value of recurrent states from F is even. In the same way one can show that the d value of recurrent states from $Q \setminus F$ is odd. Therefore d is an $\mathcal{A}$-coloring and we have a contradiction. This means for all $u \in \Sigma^*$ we have $c(q_u) = c'(q'_u)$ and hence $q_u \in F$ iff $q'_u \in F'$. $\square$

Theorem 12. *For a given DWA $\mathcal{A}$ with n states one can compute an equivalent DWA with a minimal number of states in time* $O(n \cdot \log n)$. *Furthermore this minimal DWA only depends* (*up to isomorphism*) *on* $L_\omega(\mathcal{A})$.

Proof. (For the facts about minimization of DFA used in this proof see, e.g., [9].) By Theorem 8 one can transform $\mathcal{A}$ into normal form in time $O(n)$. So we assume that $\mathcal{A}$ is in normal form. Then we compute $\mathcal{A}^{\min}$ in time $O(n \cdot \log n)$ [10,11]. First of all $\mathcal{A}^{\min}$ is a DWA, since there is a homomorphism from $\mathcal{A}$ to $\mathcal{A}^{\min}$ preserving final and non-final states. Therefore a cycle in $\mathcal{A}^{\min}$ containing final and non-final states would also give such a cycle in $\mathcal{A}$. It is well known that $L_*(\mathcal{A}^{\min}_p) \neq L_*(\mathcal{A}^{\min}_q)$ for all distinct states p, q of $\mathcal{A}^{\min}$. Thus, by Lemma 10, $L_\omega(\mathcal{A}^{\min}_p) \neq L_\omega(\mathcal{A}^{\min}_q)$ for all distinct states p, q of $\mathcal{A}^{\min}$ and therefore $\mathcal{A}^{\min}$ is a minimal DWA by Lemma 9. The automaton $\mathcal{A}^{\min}$ only depends on $L_*(\mathcal{A})$. From Lemma 11 follows that $L_*(\mathcal{A})$ only depends on $L_\omega(\mathcal{A})$ because $\mathcal{A}$ is in normal form. $\square$

This theorem also allows to reduce the equivalence problem of DWA to the equivalence problem for DFA because two DWA are equivalent iff the minimal DWA obtained by Theorem 12 are equivalent as DFA. So we also obtain an efficient equivalence check for DWA.

Acknowledgement

I thank Ludwig Staiger, Wolfgang Thomas, and Jens Vöge.

References

[1] L. Staiger, Finite-state ω-languages, J. Comput. System Sci. 27 (1983) 434–448.

[2] N. Gutleben, On the minimization of Muller-automata, Tech. Rep. Reihe Informatik I-9, BTU Cottbus, Fak. f. Mathematik, Naturwissenschaften und Informatik, December 1996.

[3] L. Staiger, K. Wagner, Automatentheoretische und automatenfreie Charakterisierungen topologischer Klassen regulärer Folgenmengen, Elektron. Informationsverarbeitung und Kybernetik EIK 10 (1974) 379–392.

[4] K. Wagner, On ω-regular sets, Inform. and Control 43 (1979) 123–177.

[5] O. Maler, L. Staiger, On syntactic congruences for ω-languages, Theoret. Comput. Sci. 183 (1) (1997) 93–112.

[6] J. Büchi, On a decision method in restricted second order arithmetic, in: Proc. International Congress on Logic, Method and Philos. Sci. 1960, 1962, pp. 1–11.

[7] W. Thomas, Languages, automata, and logic, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Language Theory, Vol. III, Springer, Berlin, 1997, pp. 385–455.

[8] R. Sedgewick, Algorithms, Addison-Wesley, Reading, MA, 1988.

[9] J. Hopcroft, J. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, MA, 1979.

[10] J. Hopcroft, An $n \log n$ algorithm for minimizing states in a finite automaton, in: Theory of Machines and Computations, Academic Press, New York, 1971, pp. 189–196.

[11] D. Gries, Describing an algorithm by Hopcroft, Acta Inform. 2 (1973) 97–109.