CHAPTER 4

Automata on Infinite Objects

Wolfgang THOMAS

Institut für Informatik und Praktische Mathematik,
Universität Kiel, Olshausenstrasse 40, D-2300 Kiel, FRG

Contents

HANDBOOK OF THEORETICAL COMPUTER SCIENCE
Edited by J. van Leeuwen

Introduction

The subject of finite automata on infinite sequences and infinite trees was established in the sixties by Büchi [9], McNaughton [62], and Rabin [93]. Their work introduced intricate automaton constructions, opened connections between automata theory and other fields (for example, logic and set-theoretic topology), and resulted in a theory which is fundamental for those areas in computer science where nonterminating computations are studied.

The early papers were motivated by decision problems in mathematical logic. Büchi showed that finite automata provide a normal form for certain monadic second-order theories. Rabin'sTree Theorem (which states that the monadic second-order theory of the infinite binary tree is decidable) turned out to be a powerful result to which a large number of other decision problems could be reduced.

From this core the theory has developed into many directions. Today, a rapidly growing literature is concerned with applications in modal logics of programs and in the specification and verification of concurrent programs. Many of the logical specification formalisms for programs (systems of program logic or temporal logic) are embeddable in the monadic second-order theories studied by Büchi and Rabin; indeed these theories can be considered as "universal process logics" for linear, respectively branching computations, as far as programs with finite state spaces are concerned.

Besides these applications (see also [30, 52]), the following lines of research should be mentioned:

- the investigation of other (usually more general) models of computation than finite automata: grammars, pushdown automata, Turing machines, Petri nets, etc.,
- the classification theory of sequence (or tree) properties, e.g. by different acceptance modes of automata or by topological conditions,
- algebraic aspects, in particular the semigroup-theoretical analysis of automata over ω-sequences,
- the connection with fixed point calculi,
- the study of more general structures than ω-sequences and ω-trees in connection with automata (e.g. sequences over the integers or certain graphs), and the extension of Rabin's decidability result to stronger theories.

The aim of this chapter is to provide the reader with a more detailed overview of these and related developments of the field, based on a self-contained exposition of the central results.

Some topics are only treated in brief remarks or had to be skipped, among them combinatorics on infinite words, connections with semantics of program schemes, and the discussion of related calculi for concurrency (such as CCS).

Also the references listed at the end of the chapter do not cover the subject. However, we expect that few papers will be missed when the reader traces the articles which are cited within the given references.

PART I. AUTOMATA ON INFINITE WORDS

Notation

Throughout this chapter A denotes a finite alphabet and A^*, respectively A^ω, denote the set of finite words, resp. the set of ω-sequences (or: *ω-words*) over A. An ω-word over A is written in the form $\alpha = \alpha(0)\alpha(1)\ldots$ with $\alpha(i) \in A$. Let $A^\infty = A^* \cup A^\omega$. Finite words are indicated by $u, v, w, \ldots$, the empty word by ε, and sets of finite words by $U, V, W, \ldots$ Letters $\alpha, \beta, \ldots$ are used for ω-words and $L, L', \ldots$ for sets of ω-words (i.e., *ω-languages*). Notations for segments of ω-words are

$$\alpha(m,n) := \alpha(m)\ldots\alpha(n-1) \quad (\text{for } m \leqslant n), \quad \text{and} \quad \alpha(m,\omega) := \alpha(m)\alpha(m+1)\ldots.$$

The logical connectives are written $\neg, \vee, \wedge, \rightarrow, \exists, \forall$. As a shorthand for the quantifiers "there exist infinitely many n" and "there are only finitely many n" we use "$\exists^\omega n$", respectively "$\exists^{<\omega} n$".

The following operations on sets of finite words are basic: for $W \subseteq A^*$ let

$$\operatorname{pref} W := \{u \in A^* \mid \exists v\, uv \in W\},$$

$$W^\omega := \{\alpha \in A^\omega \mid \alpha = w_0 w_1 \ldots \text{ with } w_i \in W \text{ for } i \geqslant 0\},$$

$$\overrightarrow{W} := \{\alpha \in A^\omega \mid \exists^\omega n\, \alpha(0,n) \in W\}.$$

Other notations for $\overrightarrow{W}$ found in the literature are $\lim W$ and W^δ. Finally, for an ω-sequence $\sigma = \sigma(0)\sigma(1)\ldots$ from S^ω, the "infinity set" of σ is

$$\operatorname{In}(\sigma) := \{s \in S \mid \exists^\omega n\, \sigma(n) = s\}.$$

1. Büchi automata

Büchi automata are nondeterministic finite automata equipped with an acceptance condition that is appropriate for ω-words: an ω-word is accepted if the automaton can read it from left to right while assuming a sequence of states in which some final state ocurs infinitely often (*Büchi acceptance*).

DEFINITION. A *Büchi automaton* over the alphabet A is of the form $\mathcal{A} = (Q, q_0, \Delta, F)$ with finite state set Q, initial state $q_0 \in Q$, transition relation $\Delta \subseteq Q \times A \times Q$, and a set $F \subseteq Q$ of final states. A *run* of $\mathcal{A}$ on an ω-word $\alpha = \alpha(0)\alpha(1)\ldots$ from A^ω is a sequence $\sigma = \sigma(0)\sigma(1)\ldots$ such that $\sigma(0) = q_0$ and $(\sigma(i), \alpha(i), \sigma(i+1)) \in \Delta$ for $i \geqslant 0$; the run is called *successful* if $\operatorname{In}(\sigma) \cap F \neq \emptyset$, i.e. some state of F occurs infinitely often in it. $\mathcal{A}$ *accepts* α if there is a successful run of $\mathcal{A}$ on α. Let

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

be the ω-language *recognized* by $\mathcal{A}$. If $L = L(\mathcal{A})$ for some Büchi automaton $\mathcal{A}$, L is said to be *Büchi recognizable*.

EXAMPLE. Consider the alphabet $A = \{a, b, c\}$. Define $L_1 \subseteq A^\omega$ by

$\alpha \in L_1$ iff after any occurrence of letter a there is some occurrence of letter b in α.

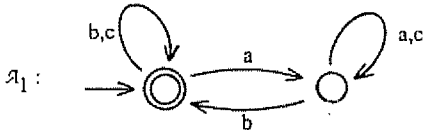


Fig. 1.

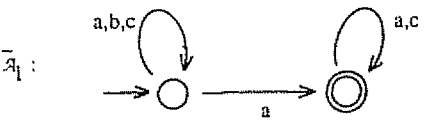


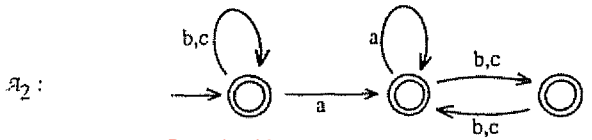
Fig. 2

Recognizes complement language of L1:
there is at least one a that is not followed by a b

A Büchi automaton recognizing L_1 is (in state graph representation) shown in Fig. 1. The complement $A^\omega - L_1$ is recognized by the Büchi automaton in Fig. 2. Finally, the ω-language $L_2 \subseteq A^\omega$ with

$\alpha \in L_2$ iff between any two occurrences of letter a in α there is an even number of letters b, c

is recognized by the automaton shown in Fig. 3.



Recognizes L2

Fig. 3

For a closer analysis of Büchi recognizable ω-languages we use the following notations, given some fixed Büchi automaton $\mathcal{A} = (Q, q_0, \Delta, F)$: if $w = a_0 \ldots a_{n-1}$ is a finite word over A, write $s \xrightarrow{w} s'$ if there is a state sequence $s_0, \ldots, s_n$ such that $s_0 = s$, $(s_i, a_i, s_{i+1}) \in \Delta$ for $i < n$, and $s_n = s'$. Let

bringing the automaton from state s to s' in finitely many steps

$$W_{ss'} = \{w \in A^* \mid s \xrightarrow{w} s'\}.$$

Each of the finitely many languages $W_{ss'}$ is regular. By definition of Büchi acceptance, the ω-language recognized by $\mathcal{A}$ is

$$L(\mathcal{A}) = \bigcup_{s \in F} W_{q_0 s} \cdot (W_{ss})^\omega. \tag{1.1}$$

1.1. THEOREM (Büchi [9]). *An ω-language $L \subseteq A^\omega$ is Büchi recognizable iff L is a finite union of sets $U \cdot V^\omega$ where U, $V \subseteq A^*$ are regular sets of finite words (and where, moreover, one may assume $V \cdot V \subseteq V$).*

In the proof, the direction from left to right is clear from equation (1.1); note that $W_{ss} \cdot W_{ss} \subseteq W_{ss}$. For the converse, we verify the following closure properties of Büchi recognizable sets.

1.2. LEMMA. (a) *If $V \subseteq A^*$ is regular, then V^ω is Büchi recognizable.*

(b) *If $U \subseteq A^*$ is regular and $L \subseteq A^\omega$ is Büchi recognizable, then $U \cdot L$ is Büchi recognizable.*

(c) *If* $L_1, L_2 \subseteq A^\omega$ *are Büchi recognizable, then* $L_1 \cup L_2$ *and* $L_1 \cap L_2$ *are Büchi recognizable.*

PROOF. (*a*) Since $V^\omega = (V - \{\varepsilon\})^\omega$, assume that V does not contain the empty word; further suppose that there is no transition into the initial state q_0 of the finite automaton which recognizes V. A Büchi automaton $\mathscr{A}'$ recognizing V^ω is obtained from the given automaton $\mathscr{A}$ by adding a transition (s, a, q_0) for any transition (s, a, s') with $s' \in F$, and by declaring q_0 as single final state of $\mathscr{A}'$.

The claims in (b) and (c) concerning concatenation and union are proved in the same way as for regular sets of finite words. For later use we show closure of Büchi recognizable sets under intersection. Suppose L_i is recognized by $\mathscr{A}_i = (Q_i, q_i, \Delta_i, F_i)$ and L_2 by $\mathscr{A}_2 = (Q_2, q_2, \Delta_2, F_2)$. A Büchi automaton recognizing $L_1 \cap L_2$ is of the form $\mathscr{A} = (Q_1 \times Q_2 \times \{0, 1, 2\}, (q_1, q_2, 0), \Delta, F)$, where the transition relation Δ copies Δ_1 and Δ_2 in the first two components of states, and changes the third component from 0 to 1 when an F_1-state occurs in the first component, from 1 to 2 when subsequently an F_2-state occurs in the second component and back to 0 immediately afterwards. Then 2 occurs infinitely often as third component in a run iff some F_1-state and some F_2-state occur infinitely often in the first two components. Hence with $F = Q_1 \times Q_2 \times \{2\}$ we obtain a Büchi automaton as desired. □

A representation of an ω-language in the form $L = \bigcup_{i=1}^{n} U_i \cdot V_i^\omega$, where the U_i, V_i are given by regular expressions, is called an *ω-regular expression*. Since the constructions in Lemma 1.2 are effective, the conversion of ω-regular expressions into Büchi automata and vice versa can be carried out effectively. Hence, Büchi recognizable ω-languages are called *regular ω-languages*; other terms used in the literature are *ω-regular*, *rational*, *ω-rational*.

The formalism of ω-regular expressions can also be applied to *relations* over finite words instead of languages. In the case of binary relations, say for $S_1, S_2 \subseteq A^* \times B^*$, one defines $S_1 \cdot S_2^\omega \subseteq A^\omega \times B^\omega$ by applying the operations of product and ω-power componentwise. The *rational ω-relations* are obtained as finite unions of relations $S_1 \cdot S_2^\omega$ where S_1 and S_2 are rational relations over finite words (i.e., generated via usual regular expressions, cf. [5] or [27]). The rational ω-relations are characterized by a multitape version of Büchi automata with one reading head for each tape (Gire and Nivat [38]). However, as for rational relations over finite words, certain closure and decidability results cannot be transferred from regular ω-languages to the case of ω-relations (for instance, Theorems 2.1 and 2.3 below). In the sequel we restrict ourselves to ω-languages.

As is clear from equation (1.1), a Büchi automaton $\mathscr{A}$ accepts some ω-word iff $\mathscr{A}$ reaches some final state (say via the word u) which can then be revisited by a loop (say via the word v). The existence of a reachable final state which is located in a loop of $\mathscr{A}$ can be checked by an effective procedure. Hence, we obtain the following theorem.

A word that is accepted by a Büchi automaton MUST contain a sequence that is repeated over and over.

1.3. THEOREM. (a) *Any nonempty regular ω-language contains an ultimately periodic ω-word (i.e., an ω-word of the form $uvvv\ldots$).*

(b) *The emptiness problem for Büchi automata is decidable.*

Does the Büchi automaton accept a word at all?

Vardi and Wolper [131] show that the nonemptiness-problem for Büchi automata ("$L(\mathscr{A}) \neq \emptyset$?") is logspace complete for NLOGSPACE, and Sistla, Vardi and Wolper [107] prove that the nonuniversality problem for Büchi automata ("$L(\mathscr{A}) \neq A^\omega$?") is logspace complete for PSPACE.

Does the Büchi automaton accept not all possible infinite words?

2. Congruences and complementation

Closure of Büchi recognizable ω-languages under complement is nontrivial and involves an interesting combinatorial argument. As will be seen in Section 4, it is not possible to work with a reduction to deterministic Büchi automata.

Büchi recognizable languages are closed under complementation

2.1. THEOREM (Büchi [9]). *If $L \subseteq A^\omega$ is Büchi recognizable, so is $A^\omega - L$. Moreover, from a Büchi automaton recognizing L one can construct one recognizing $A^\omega - L$.*

For the proof we shall represent both L and $A^\omega - L$ as finite unions of sets $U.V^\omega$ where U and V are regular sets of a special kind, namely classes of a certain congruence relation over A^* of finite index. (A congruence is here an equivalence relation compatible with concatenation.) Let $L = L(\mathscr{A})$ where $\mathscr{A} = (Q, q_0, \Delta, F)$ is a Büchi automaton. Write $s \xrightarrow{F}_w s'$ if there is a run of $\mathscr{A}$ on w from state s to state s' such that at least one of the states in the run (including s and s') belongs to F. The set

$$W^F_{ss'} := \{w \in A^* \mid s \xrightarrow{F}_w s'\}$$

The language of all the finite words that, if run on a Büchi automaton, make it visit at least one accepting state.

is regular. Now define the equivalence relation $\sim_{\mathscr{A}}$ over A^* as follows:

$$u \sim_{\mathscr{A}} v \text{ iff } \forall s, s' \in Q\,(s \xrightarrow{}_u s' \Leftrightarrow s \xrightarrow{}_v s' \text{ and } s \xrightarrow{F}_u s' \Leftrightarrow s \xrightarrow{F}_v s').$$

Two words are equivalent if they result in the same start and end state and either do or do not make the automaton visit at least one accepting state.

The relation $\sim_{\mathscr{A}}$ is a congruence over A^*, which is of finite index by finiteness of Q. Hence each $\sim_{\mathscr{A}}$-class is regular. (The $\sim_{\mathscr{A}}$-class $[w]$ containing the word w is the intersection of the sets $W_{ss'}$ and $W^F_{ss'}$ and $A^* - W_{ss'}$ and $A^* - W^F_{ss'}$ containing w.)

A representation of $L(\mathscr{A})$ and $A^\omega - L(\mathscr{A})$ in terms of the $\sim_{\mathscr{A}}$-classes will be provided by the following lemma:

2.2. LEMMA. (a) *Let $\mathscr{A}$ be a Büchi automaton. For any $\sim_{\mathscr{A}}$-classes U, V: If $U.V^\omega \cap L(\mathscr{A}) \neq \emptyset$, then $U.V^\omega \subseteq L(\mathscr{A})$. (Hence: if $U.V^\omega \cap (A^\omega - L(\mathscr{A})) \neq \emptyset$, then $U.V^\omega \subseteq A^\omega - L(\mathscr{A})$.)*

(b) *Let $\sim$ be a congruence over A^* of finite index. For any ω-word $\alpha \in A^\omega$ there are $\sim$-classes U, V (even with $V.V \subseteq V$) such that $\alpha \in U.V^\omega$.*

Before the proof let us apply the lemma. Part (a) states a "saturation" property of $\sim_{\mathscr{A}}$ with respect to $L(\mathscr{A})$ and $A^\omega - L(\mathscr{A})$. By definition, a congruence $\sim$ over A^* *saturates* an ω-language $L \subseteq A^\omega$ if

$$U.V^\omega \cap L \neq \emptyset \text{ implies } U.V^\omega \subseteq L \quad \text{for all } \sim\text{-classes } U, V.$$

If $\sim$ saturates L and also is of finite index, then

$$L = \bigcup \{U.V^\omega \mid U, V \sim\text{-classes}, U.V^\omega \cap L \neq \emptyset\};$$

the inclusion "$\supseteq$" holds by saturation, and "$\subseteq$" follows from Lemma 2.2(b). Moreover, by the assumption that $\sim$ has finite index the $\sim$-classes are regular and the union is finite; so L is a regular ω-language. For the congruence $\sim_{\mathcal{A}}$, which saturates $A^\omega - L(\mathcal{A})$ by Lemma 2.2(a) and which is of finite index, we obtain that $A^\omega - L(\mathcal{A})$ is regular. Note that emptiness of $U \cdot V^\omega \cap L(\mathcal{A})$ (and hence also nonemptiness of $U \cdot V^\omega \cap (A^\omega - L(\mathcal{A}))$ can be decided effectively by Lemma 1.2(c) and Theorem 1.3. Hence, using Theorem 1.1, a Büchi automaton recognizing $A^\omega - L(\mathcal{A})$ can be constructed effectively from the given automaton $\mathcal{A}$. So Lemma 2.2 suffices to prove Theorem 2.1.

PROOF OF LEMMA 2.2. Let $\mathcal{A} = (Q, q_0, \Delta, F)$. Suppose $\alpha = uv_1v_2 \ldots$ where $u \in U$ and $v_i \in V$ for $i > 0$, and assume further that there is a successful run of $\mathcal{A}$ on α. From this run we obtain states $s_1, s_2, \ldots$ such that

$$q_0 \xrightarrow{u} s_1 \xrightarrow{v_1} s_2 \xrightarrow{v_2} s_3 \to \cdots$$

where we even have

$$s_i \xrightarrow{\frac{F}{v_i}} s_{i+1} \text{ for infinitely many } i.$$

Let $\beta \in U \cdot V^\omega$ be arbitrary. We show that $\beta \in L(\mathcal{A})$. We have $\beta = u'v_1'v_2' \ldots$ where $u' \in U$ and $v_i' \in V$ for $i > 0$. Since U, V are $\sim_{\mathcal{A}}$-classes and hence $u \sim_{\mathcal{A}} u'$, $v_i \sim_{\mathcal{A}} v_i'$, we obtain

$$q_0 \xrightarrow{u'} s_1 \xrightarrow{v_1'} s_2 \xrightarrow{v_2'} s_3 \to \cdots$$

and

$$s_i \xrightarrow{\frac{F}{v_i'}} s_{i+1} \text{ for infinitely many } i.$$

This yields a run of $\mathcal{A}$ on β in which some F-state occurs infinitely often. Hence, $\beta \in L(\mathcal{A})$.

(b) Let $\sim$ be a congruence of finite index over A^*. Given $\alpha \in A^\omega$, two positions k, k' are said to *merge at position* m (where $m > k, k'$) if $\alpha(k, m) \sim \alpha(k', m)$. In this case write $k \cong_\alpha k'(m)$. Note that then also $k \cong_\alpha k'(m')$ for any $m' > m$ (because $\alpha(k, m) \sim \alpha(k', m)$ implies $\alpha(k, m)\alpha(m, m') \sim \alpha(k', m)\alpha(m, m')$). Write $k \cong_\alpha k'$ if $k \cong_\alpha k'(m)$ for some m. The relation $\cong_\alpha$ is an equivalence relation of finite index over ω (because $\sim$ is of finite index). Hence there is an infinite sequence $k_0, k_1, \ldots$ of positions which all belong to the same $\cong_\alpha$-class. By passing to a subsequence (if necessary), we can assume $k_0 > 0$ and that for $i > 0$ the segments $\alpha(k_0, k_i)$ all belong to the same $\sim$-class V. Let U be the $\sim$-class of $\alpha(0, k_0)$. We obtain

$$\exists k_0 \, (\alpha(0, k_0) \in U \wedge \exists^\omega k \, (\alpha(k_0, k) \in V \wedge \exists m \, k_0 \cong_\alpha k(m))). \tag{2.1}$$

We shall show that (2.1) implies $\alpha \in U \cdot V^\omega$ and $V \cdot V \subseteq V$ (which completes the proof of (b)). Suppose that k_0 and a sequence $k_1, k_2, \ldots$ are given as guaranteed by (2.1). Again by passing to an infinite subsequence, we may assume that for all $i \geq 0$, the positions $k_0, \ldots, k_i$ merge at some $m < k_{i+1}$ and hence at k_{i+1}. We show $\alpha(k_i, k_{i+1}) \in V$ for $i \geq 0$. From (2.1) it is clear that $\alpha(k_0, k_1) \in V$. By induction assume that $\alpha(k_j, k_{j+1}) \in V$ for $j < i$. We know $\alpha(k_0, k_{i+1}) \in V$ and that k_0, k_i merge at k_{i+1}. Thus $\alpha(k_i, k_{i+1}) \in V$ and hence $\alpha \in U \cdot V$.

Finally, in order to verify the claim $V.V \subseteq V$, it suffices to show $V.V \cap V \neq \emptyset$ (since V is a class of a congruence). But this is clear since $\alpha(k_0, k_i)$, $\alpha(k_i, k_{i+1})$ and $\alpha(k_0, k_{i+1})$ belong to V for any $i > 0$. □

The use of the merging relation $\cong_\alpha$ in the preceding proof can be avoided if Ramsey's Theorem is invoked (as done in the original proof by Büchi [9]): One notes that $\sim$ induces a finite partition of the set $\{(i,j) \mid i<j\}$ by defining that (i,j) and (i',j') belong to the same class iff $\alpha(i,j) \sim \alpha(i',j')$. Now Ramsey's Theorem (in the version for countable sets) states that there is an infinite "homogeneous" set, i.e., a set $\{i_0, i_1, \ldots\}$ such that all segments $\alpha(i_k, i_l)$ with $k<l$ are in one $\sim$-class, in particular all $\alpha(i_k, i_{k+1})$ are in this class. Define V to be this $\sim$-class and let U be the $\sim$-class of $\alpha(0, i_0)$. Then $\alpha \in U.V^\omega$. We gave the above self-contained proof because condition (2.1) will be used again in Section 4.

By Lemma 1.2 and Theorem 2.1, the regular ω-languages are effectively closed under Boolean operations. Hence the inclusion test and the equivalence test for Büchi automata reduce to the emptiness test (using $L_1 \subseteq L_2$ iff $L_1 \cap \sim L_2 = \emptyset$).

L1 doesn't contain anything that L2 doesn't contain.

2.3. THEOREM. *The inclusion problem and the equivalence problem for Büchi automata are decidable.*

Let us consider the complexity of the complementation process and the equivalence test. Given a Büchi automaton with n states, there are n^2 different pairs (s,s') and hence $O(2^{2n^2})$ different $\sim_{\mathcal{A}}$-classes. This leads to a size bound of $O(2^{4n^2})$ states for the complement automaton [83, 107]. An improved and essentially optimal bound of $2^{O(n \log n)}$ is given in [99]. In [107] it is shown that the equivalence problem for Büchi automata is logspace complete for PSPACE.

The equivalence problem has also been studied in terms of equations between ω-regular expressions, building on work of Salomaa for classical regular expressions. A sound and complete axiom system for deriving these equations is given in [133]; for further references see [110].

Lemma 2.2 above not only shows complementation for regular ω-languages but also serves as a starting point for an investigation of these ω-languages in terms of finite semigroups. Recall that a language $W \subseteq A^*$ is regular iff there is a finite monoid M and a monoid homomorphism $f: A^* \to M$ such that W is a union of sets $f^{-1}(m)$ where $m \in M$. Since $A^*/\sim_{\mathcal{A}}$ is a finite monoid (for any Büchi automaton $\mathcal{A}$), we obtain, from Theorem 1.1 and Lemma 2.2, the following theorem.

What is a monoid?

2.4. THEOREM. *An ω-language $L \subseteq A^\omega$ is regular iff there is a finite monoid M and a monoid homomorphism $f: A^* \to M$ such that L is a union of sets $f^{-1}(m).(f^{-1}(e))^\omega$ with $m, e \in M$ and where e can be assumed to be idempotent (i.e., satisfying $e.e = e$).*

As will be shown in Theorem 2.6 below, there is a canonical minimal monoid with this property. In other words, there is a coarsest congruence $\approx_L$ over A^* which saturates L. We introduce $\approx_L$ here together with another natural congruence associated with an ω-language L. Given $L \subseteq A^\omega$, define for $u, v \in A^*$

$$u \sim_L v \quad \text{iff} \quad \forall \alpha \in A^\omega (u\alpha \in L \Leftrightarrow v\alpha \in L)$$

(a right congruence, cf. [124, 108]);

$$u \approx_L v \quad \text{iff} \quad \forall x, y, z \in A^* (xuyz^\omega \in L \Leftrightarrow xvyz^\omega \in L \text{ and } x(yuz)^\omega \in L \Leftrightarrow x(yvz)^\omega \in L)$$

(cf. [2]). The congruence $\approx_L$ regards two finite words as equivalent iff they cannot be distinguished by L as corresponding segments of ultimately periodic ω-words. Regularity of L implies that $\sim_L$ and $\approx_L$ are of finite index (since they are refined by the finite congruence $\sim_{\mathcal{A}}$ if $\mathcal{A}$ is a Büchi automaton that recognizes L). We note that the converse fails.

2.5. REMARK (Trakhtenbrot [124]) *There are nonregular sets $L \subseteq A^\omega$ such that $\sim_L$ and $\approx_L$ are of finite index.*

PROOF. For given $\beta \in A^\omega$, let $L(\beta)$ contain all ω-words that have a common suffix with β. Then any two words u, v are $\sim_{L(\beta)}$-equivalent since, for two ω-words $u\alpha$, $v\alpha$, membership in $L(\beta)$ does not depend on u, v. So there is only one $\sim_{L(\beta)}$-class. If we choose β to be not ultimately periodic, $L(\beta)$ is not regular (by Theorem 1.3). Furthermore, in this latter case also $\approx_{L(\beta)}$ has only one congruence class. □

We now show the mentioned maximality property of $\approx_L$.

2.6. THEOREM (Arnold [2]). *An ω-language L is regular iff $\approx_L$ is of finite index and saturates L; moreover, $\approx_L$ is the coarsest congruence saturating L.*

PROOF. If $\approx_L$ is of finite index and saturates L, then $L = \bigcup \{U \cdot V^\omega \mid U, V \text{ are } \approx_L\text{-classes}, U \cdot V^\omega \cap L \neq \emptyset\}$ and L hence is regular (see the remark following Lemma 2.2). Conversely, suppose L is regular; then (as seen before Remark 2.5) $\approx_L$ is of finite index. We show that $\approx_L$ saturates L, i.e. $U \cdot V^\omega \cap L \neq \emptyset$ implies $U \cdot V^\omega \subseteq L$ for any $\approx_L$-classes U, V. Since $U \cdot V^\omega \cap L$ is regular, we can assume that there is an ultimately periodic ω-word xy^ω in $U \cdot V^\omega \cap L$. In a decomposition of xy^ω into a U-segment and a sequence of V-segments, we find two V-segments which start after the same prefix y_1 of period y; so we obtain $w := xy^m y_1 \in U \cdot V^r$ and $z := y_2 y^n y_1 \in V^s$ for some m, n, r, s and $y_1 y_2 = y$, so that $xy^\omega = wz^\omega$. Denote by $[w]$ and $[z]$ the $\approx_L$-classes of w and z. Since $[w] \cap U \cdot V^r \neq \emptyset$ we have $U \cdot V^r \subseteq [w]$; similarly, $V^s \subseteq [z]$, and hence $U \cdot V^\omega \subseteq [w] \cdot [z]^\omega$. It remains to prove $[w] \cdot [z]^\omega \subseteq L$. For a contradiction, assume there is $\alpha \in [w] \cdot [z]^\omega - L$, say $\alpha = w_0 z_1 z_2 \ldots$ where $w_0 \approx_L w, z_i \approx_L z$. Since α may be assumed again to be ultimately periodic, we obtain p, q with

$$\alpha = w_0 z_1 \ldots z_p (z_{p+1} \ldots z_{p+q})^\omega.$$

But then, from $wz^\omega = xy^\omega \in L$, we know $wz^p (z^q)^\omega \in L$, so

$$w_0 z_1 \ldots z_p (z_{p+1} \ldots z_{p+q})^\omega \in L$$

by definition of $\approx_L$ and thus $\alpha \in L$, a contradiction.

It remains to show that $\approx_L$ is the coarsest among the congruences $\sim$ saturating L. So assume $\sim$ is such a congruence and suppose $u \sim v$ (or: $\langle u \rangle = \langle v \rangle$ for the $\sim$-classes of

u and v). We verify $u \approx_L v$. We have $xuyz^\omega \in L$ iff $\langle xuy\rangle\langle z\rangle^\omega \subseteq L$ (since $\sim$ saturates L) iff $\langle xvy\rangle\langle z\rangle^\omega \subseteq L$ (since $u \sim v$) iff $xvyz^\omega \in L$. Similarly, one obtains $x(yuz)^\omega \in L$ iff $x(yvz)^\omega \in L$; thus $u \approx_L v$. □

The preceding result justifies calling $A^*/\approx_L$ the *syntactic monoid of* L, with concatenation of classes as the product. It allows us to classify the regular ω-languages by reference to selected varieties of monoids, extending the classification theory for regular sets of finite words (cf. [87]). Examples will be mentioned in Section 6.

3. The sequential calculus

One motivation for considering automata on infinite sequences was the analysis of the "sequential calculus", a system of monadic second-order logic for the formalization of properties of sequences. Büchi [9] showed the surprising fact that any condition on sequences that is written in this calculus can be reformulated as a statement about acceptance of sequences by an automaton.

For questions of logical definability, an ω-word $\alpha \in A^\omega$ is represented as a model-theoretic structure of the form $\underline{\alpha} = (\omega, 0, +1, <, (Q_a)_{a\in A})$, where $(\omega, 0, +1, <)$ is the structure of the natural numbers with zero, successor function, and the usual ordering, and where $Q_a = \{i \in \omega \mid \alpha(i) = a\}$ (for $a \in A$). The corresponding first-order language contains variables $x, y, \ldots$ for natural numbers, i.e. for the positions in ω-words. Typical atomic formulas are "$x+1<y$" ("the position following x comes before y") or "$x \in Q_a$" ("position x carries letter a"). In this framework, the example set $L_1 \subseteq \{a, b, c\}^\omega$ of Section 1 (containing the ω-words where after any letter a there is eventually a letter b) can be defined by the sentence

p. 11

We shall also allow variables $X, Y, \ldots$ for *sets* of natural numbers and quantifiers ranging over them. For example, they occur in a definition of the ω-language L_2 of Section 1 (containing the ω-words where between any two succeeding occurrences of letter a there is an even number of letters b, c):

$$\varphi_2: \quad \forall x \forall y (x \in Q_a \wedge y \in Q_a \wedge x < y \wedge \neg \exists z (x < z \wedge z < y \wedge z \in Q_a) \rightarrow \exists X (x \in X \wedge \forall z (z \in X \leftrightarrow \neg(z+1 \in X)) \wedge \neg(y \in X))).$$

Note that the set quantifier postulates a set containing every second position starting with position x; this ensures that the number of letters between positions x and y is even. The sequential calculus consists of all the conditions on ω-words which can be written in this logical language.

One also calls this framework S1S for "second-order theory of *one* successor". (Below, in Theorem 3.1, it will be seen that $<$ is second-order definable in terms of successor and hence inessential). It is a system of *monadic second-order logic*, due to the quantification over sets, which are unary relations and hence "monadic second-order objects".

DEFINITION. The interpreted system $S1S_A$ (*second-order theory of one successor* over A) is built up as follows: *Terms* are constructed from the constant 0 and the variables $x, y, \ldots$ by applications of "$+1$" (successor function). *Atomic formulas* are of the form $t=t'$, $t<t'$, $t\in X$, $t\in Q_a$ (for $a\in A$) where t, t' are terms and X is a set variable, and $S1S_A$-*formulas* are constructed from atomic formulas using the connectives $\neg$, $\vee$, $\rightarrow$, $\leftrightarrow$ and the quantifiers $\exists, \forall$ acting on either kind of variables. We write $\varphi(X_1, \ldots, X_n)$ to indicate that at most the variables $X_1, \ldots, X_n$ occur free in φ (i.e., are not in the scope of a quantifier). Formulas without free variables are called *sentences*. Given $\alpha\in A^\omega$ and an $S1S_A$-sentence φ, we write $\underline{\alpha}\models\varphi$ if φ is satisfied in $\underline{\alpha}$ under the canonical interpretation (over the domain ω of $\underline{\alpha}$, as described above). The ω-language defined by an $S1S_A$-sentence φ is $L(\varphi)=\{\alpha\in A^\omega \mid \underline{\alpha}\models\varphi\}$. □

For instance, if $\alpha = abcaabcaaabc\ldots$ and φ_1 is as before, we have $\underline{\alpha}\models\varphi_1$, and $L(\varphi_1)$ is the ω-language L_1 of the example in Section 1.

Sometimes it is convenient to cancel the predicate symbols Q_a and use free set variables X_k in their place. We denote the resulting formalism by S1S. In this case we consider formulas $\varphi(X_1, \ldots, X_n)$ without the symbols Q_a and interpret them in ω-words over the special alphabet $\{0,1\}^n$. In $\alpha\in(\{0,1\}^n)^\omega$, the formula $x\in X_k$ says that the xth letter of α has 1 in its kth component. As an example, consider the following sequence $\alpha\in(\{0,1\}^2)^\omega$, where the letters from $\{0,1\}^2$ are written as columns:

$$\alpha:\ \begin{matrix}1&0&1&0&1&0&1&0&1\\1&1&1&0&0&0&0&0&0\end{matrix}\ \cdots$$

Since in α there are infinitely many letters with first component 1 and second component 0, we have

$$\underline{\alpha}\models\forall x\,\exists y\,(x<y\wedge y\in X_1\wedge\neg(y\in X_2)).$$

Formally, we represent $\alpha\in(\{0,1\}^n)^\omega$ by the structure $\underline{\alpha}=(\omega, 0, +1, <, P_1, \ldots, P_n)$ where $P_k=\{i \mid (\alpha(i))_k=1\}$, writing $\underline{\alpha}\models\varphi(X_1, \ldots, X_n)$ iff φ holds in $\underline{\alpha}$ with P_k as interpretation of X_k. For an S1S-formula $\varphi=\varphi(X_1, \ldots, X_n)$ define $L(\varphi)=\{\alpha\in(\{0,1\}^n)^\omega \mid \underline{\alpha}\models\varphi(X_1, \ldots, X_n)\}$.

By embedding a given alphabet A into a set $\{0,1\}^n$ for suitable n, $S1S_A$-sentences can be reformulated as S1S-formulas $\varphi(X_1, \ldots, X_n)$. (Instead of "$x\in Q_a$", write the corresponding conjunction consisting of formulas "$x\in X_k$" and "$\neg\, x\in X_k$".) Depending on the alphabet under consideration, we shall call an ω-language L simply *definable in* S1S if for some sentence φ with the symbols Q_a, respectively for some formula $\varphi=\varphi(X_1, \ldots, X_n)$ without the Q_a, we have $L=L(\varphi)$.

3.1. BÜCHI'S THEOREM (cf. [9]). *An ω-language is definable in* S1S *iff it is regular.*

PROOF. The direction from right to left is easy: Let $\mathcal{A}=(Q, q_0, \Delta, F)$ be a Büchi automaton over the alphabet A, and assume $Q=\{0, \ldots, m\}$ and $q_0=0$. The existence of a successful run on an ω-word $\alpha\in A^\omega$ can be expressed by the existence of a suitable $(m+1)$-tuple of sets $Y_0, \ldots, Y_m$; Y_i contains the positions where the run assumes state i

$L(\mathcal{A})$ is defined by the sentence

$$\exists Y_0 \ldots \exists Y_m \Big(\bigwedge_{i \neq j} \neg \exists y (y \in Y_i \wedge y \in Y_j) \wedge 0 \in Y_0 \wedge \forall x \bigvee_{(i,a,j) \in \Delta} (x \in Y_i \wedge x \in Q_a \wedge x+1 \in Y_j) \wedge \bigvee_{i \in F} \forall x \exists y (x < y \wedge y \in Y_i) \Big) \tag{3.1}$$

We shall prove the converse for S1S-formulas $\varphi(X_1, \ldots, X_n)$ interpreted in ω-words over $\{0,1\}^n$ as explained above. We proceed in two steps: First S1S is reduced to a simpler formalism $S1S_0$ where *only* second-order variables X_i occur and the atomic formulas are of the form $X_i \subseteq X_j$("X_i is a subset of X_j") and $\mathrm{Succ}(X_i, X_j)$ ("X_i, X_j are singletons $\{x\}, \{y\}$ where $x+1=y$"). In a second step an induction over $S1S_0$-formulas $\varphi(X_1, \ldots, X_n)$ shows that $L(\varphi)$ is Büchi recognizable.

(1) *Reduction of* S1S *to* $S1S_0$. Carry out the following steps, starting with a given S1S-formula:

(i) Eliminate superpositions of "$+1$" by rewriting, e.g.,

"$(x+1)+1 \in X$" as "$\exists y \exists z (x+1=y \wedge y+1=z \wedge z \in X)$".

(ii) Eliminate the symbol 0 and then the symbol $<$ by rewriting, e.g.,

"$0 \in X$" as "$\exists x (x \in X \wedge \neg \exists y (y < x))$",

"$x < y$" as "$\forall X (x+1 \in X \wedge \forall z (z \in X \rightarrow z+1 \in X) \rightarrow y \in X)$".

We arrive at a formula with atomic formulas of type $x=y$, $x+1=y$, and $x \in X$ only. For the remaining step we use the shorthands

"$X=Y$" for "$X \subseteq Y \wedge Y \subseteq X$", "$X \neq Y$" for "$\neg X=Y$",

"Sing X" ("X is a singleton")

for "$\exists Y (Y \subseteq X \wedge Y \neq X \wedge \neg \exists Z (Z \subseteq X \wedge Z \neq X \wedge Z \neq Y))$"

("there is exactly one proper subset of X").

(iii) Eliminate first-order variables, by rewriting, e.g.,

"$\forall x \exists y (x+1=y \wedge y \in Z)$"

as "$\forall X (\mathrm{Sing}\, X \rightarrow \exists Y (\mathrm{Sing}\, Y \wedge \mathrm{Succ}(X, Y) \wedge Y \subseteq Z))$".

We obtain an $S1S_0$-formula equivalent to the given S1S-formula.

(2) *Büchi recognizability of* $S1S_0$*-definable sets*. By induction over $S1S_0$-formulas we show that for any $S1S_0$-formula $\varphi(X_1, \ldots, X_n)$ there is a Büchi automaton $\mathcal{A}$ over $\{0,1\}^n$ with $L(\mathcal{A}) = L(\varphi)$. As typical examples of atomic formulas, consider $X_1 \subseteq X_2$ and $\mathrm{Succ}(X_1, X_2)$. Corresponding Büchi automata are given by the state graphs shown in Fig. 4.

For the induction step it suffices to treat $\neg$, $\vee$, and $\exists$ (since $\wedge$, $\rightarrow$, $\leftrightarrow$, $\forall$ are expressible in terms of $\neg$, $\vee$, $\exists$). Cases $\neg$ and $\vee$ are clear by closure of the regular ω-languages under complement and union. Concerning $\exists$, we have to show closure of the regular

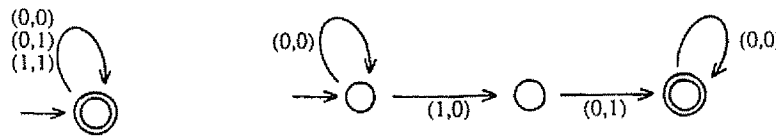


Fig 4

ω-languages under *projection*: Assume, for example, that, for the $S1S_0$-formula $\varphi(X_1, X_2)$, a Büchi automaton $\mathcal{A}$ over $\{0,1\}^2$ exists with $L(\mathcal{A}) = L(\varphi)$; we have to find an automaton $\mathcal{A}'$ over $\{0,1\}$ for the formula $\varphi'(X_1) = \exists X_2\, \varphi(X_1, X_2)$. We obtain $\mathcal{A}'$ by changing the letters of the transitions of $\mathcal{A}$ from (1, 0), (1, 1) to 1, respectively from (0,0), (0, 1) to 0. A successful run of $\mathcal{A}'$ thus "guesses" a second component for the given input $\alpha \in \{0,1\}^\omega$, and for the resulting sequence from $(\{0,1\}^2)^\omega$ it is a successful run of $\mathcal{A}$. Thus $\mathcal{A}'$ recognizes $L(\varphi')$. □

Büchi's Theorem shows that "global" properties of sequences, as formulated in S1S-conditions by means of quantifiers over arbitrary elements and sets, can be given a strictly "operational" meaning, represented by the stepwise working of a Büchi automaton. From a dual point of view, Büchi automata are, despite their conceptual simplicity, a very powerful formalism for the specification of sequence properties. In this connection we note that the strength of Büchi automata is also reflected by their equivalence with extended models, e.g. the *two-way Büchi automaton* [82, 128] and *alternating ω-automata* [64].

With minor modifications Büchi's Theorem also holds for sets of *finite* words. Thus "regular" is equivalent to "monadic second-order definable" for languages as well as for ω-languages. In a finite word w of length k, the variables $x, y, \ldots$ refer to the positions of w (from 1 to k) and the variables $X, Y, \ldots$ to subsets of $\{1, \ldots, k\}$. Moreover, the successor function has to be redefined for the maximal element k; we may set, for instance, $k+1 = k$. Also a suitable convention for the empty word is adopted (it should satisfy universal sentences but not existential sentences). With the resulting notion of monadic second-order definability for sets $W \subseteq A^*$ one obtains the following equivalence result.

3.2. THEOREM (Büchi [8], Elgot [28]). *A set $W \subseteq A^*$ is regular iff it is definable in* S1S *(interpreted over finite word models).*

Looking back at the projection step of Theorem 3.1, one notes that it works also for a formula $\varphi' = \exists X_1\ \varphi(X_1)$, which is a *sentence* of S1S, i.e. a formula without free variables. The resulting automaton $\mathcal{A}'$ has unlabelled transitions and in this sense works "input-free". $\mathcal{A}'$ admits a successful run iff the existential sentence φ' is true over the domain ω of the natural numbers. Since the existence of a successful run is effective (see Theorem 1.3), it can be decided whether an S1S-sentence holds in $(\omega, 0, +1, <)$:

3.3. THEOREM (Büchi [9]). *Truth of sentences of* S1S *is decidable.*

The decision problem for S1S was a main motivation in Büchi's investigations. Automata represented a normal form of S1S-formulas (namely, (3.1) in the proof of Theorem 3.1) which was simple enough to be decided effectively.

If set quantification refers to finite sets only, one speaks of the *weak monadic theory of successor*, denoted WS1S. Decidability was shown earlier for WS1S than for S1S (in connection with the characterization of regular sets of finite words, by Büchi [8] and Elgot [28]); it also follows from Theorem 3.3 by an interpretation of WS1S in S1S (since finiteness of sets is definable in S1S).

The decidability result on S1S has been extended in many directions. One of them, Rabin's Tree Theorem (stating decidability of the monadic theory S2S of two successor functions, i.e. of the binary tree) will be discussed in Section 11 below. Another kind of extension is concerned with addition of further number-theoretic relations or functions to the theory S1S (besides successor and order). Elgot and Rabin [29] showed that one may add the unary predicates "is a factorial" or "is a power of k" (for $k \geq 2$) without destroying decidability. However, S1S enriched by the function $x \mapsto 2x$ already allows to interpret full first-order arithmetic and thus is undecidable. More recent results are discussed in [102].

Büchi [10] extended his proof of Theorem 3.3 to transfinite ordinals, using automata over ordinal numbers, and showed that the monadic second-order theory of the ordinal $(\omega_1, <)$ is decidable. Siefkes [104], and Büchi and Siefkes [14] presented axiomatizations of the monadic theories of $(\omega, <)$ and $(\omega_1, <)$. In subsequent work of Gurevich, Magidor and Shelah it became clear that from ω_2 onwards the monadic second-order theory of an ordinal depends on set-theoretic hypotheses (cf. [41]).

A fundamental progress in the study of monadic theories was made by Shelah [103]. He developed a model-theoretic technique for obtaining decidability results, which does not refer to automata and is applicable to a larger class of structures. A central idea in this approach is to "compose" a finite fragment of the theory of an ordering (given, say, by the formulas up to some quantifier depth) from the corresponding theory fragments of suborderings and the way these suborderings are arranged. In a series of papers Gurevich and Shelah applied the method to show strong decidability results, in particular concerning dense orderings and trees. However, again based on [103], they also showed that the monadic second-order theory of $(R, <)$, the ordering of the real numbers, is undecidable. For an exhaustive presentation of the subject, see Gurevich's survey [41].

Recently, the automata-theoretic aspects of the monadic theory of the *integers*, i.e. of the ordering $(Z, <)$, have been investigated. This theory is closely related to problems in ergodic theory and symbolic dynamics. Words over Z ("*bi-infinite words*") lack a distinguished position, like the first position of an ω-word. Thus two Z-words are identified if they can be transformed into each other by a finite shift; and a natural automaton model does not refer to some "start position" but works on Z-words from left to right, "coming from infinity" and "going to infinity". A development of the theory of regular Z-languages, establishing results analogous to the case of ω-languages, is given in [76, 89].

4. Determinism and McNaughton's Theorem

A simple argument, given in Example 4.2 below, shows that deterministic Büchi automata are not closed under complement and hence strictly weaker than Büchi

automata in general. Nevertheless, by refining the notion of acceptance, it is possible to define a type of deterministic automaton which characterizes the regular ω-languages. In the present section we discuss this fundamental determinization theorem, due to McNaughton [62].

If a deterministic finite automaton $\mathscr{A}$, which recognizes the set $W \subseteq A^*$, is used as a Büchi automaton, it accepts an ω-word α iff infinitely many prefixes of α lead $\mathscr{A}$ to a final state, i.e. belong to W. Collecting these α, we obtain the set

$$\overrightarrow{W} := \{\alpha \in A^\omega \mid \exists^\omega n\, \alpha(0, n) \in W\}.$$

By the mentioned conversion of finite automata into deterministic Büchi automata and vice versa we immediately have the following result.

4.1. REMARK. *An ω-language $L \subseteq A^\omega$ is recognized by a deterministic Büchi automaton iff $L = \overrightarrow{W}$ for some regular set $W \subseteq A^*$.*

The following example shows that not every regular ω-language is of this form; at the same time we see that closure under complement fails for deterministic Büchi automata.

4.2. EXAMPLE. *Let $A = \{a, b\}$ and $L := \{\alpha \in A^\omega \mid \exists^{<\omega} n\, \alpha(n) = a\}$* (i.e., $L = A^\omega - \overrightarrow{(b^*a)^*}$). *Then L is not of the form $\overrightarrow{W}$ with $W \subseteq A^*$.*

PROOF. Assuming $L = \overrightarrow{W}$ one obtains a contradiction as follows: For some n_1, $b^{n_1} \in W$ (because $b^\omega \in L = \overrightarrow{W}$). For this n_1 there is some n_2 such that $b^{n_1}ab^{n_2} \in W$ (because $b^{n_1}ab^\omega \in L = \overrightarrow{W}$). Proceeding in this way, one obtains a sequence of words $b^{n_1}ab^{n_2} \ldots ab^{n_k} \in W$ $(k = 1, 2, \ldots)$. Hence the ω-word $b^{n_1}ab^{n_2} \ldots$ is in $\overrightarrow{W}$ and thus in L, contradicting the definition of L. □

A suitably generalized acceptance condition for deterministic automata on ω-words which captures the power of Büchi automata was defined by Muller [71] (in connection with a problem in asynchronous switching theory).

Special acceptance condition: all the infinitely often visited states must be accepting states.

DEFINITION. A *Muller automaton* over the alphabet A is of the form $\mathscr{A} = (Q, q_0, \delta, \mathscr{F})$ with finite state set Q, initial state $q_0 \in Q$, transition function $\delta: Q \times A \to Q$, and a collection $\mathscr{F} \subseteq 2^Q$ of final state sets. A run σ of $\mathscr{A}$ is *successful* if $\mathrm{In}(\sigma) \in \mathscr{F}$, i.e. the states that $\mathscr{A}$ assumes infinitely often in σ form a set from $\mathscr{F}$. $\mathscr{A}$ *accepts* an ω-word α if the unique run σ of $\mathscr{A}$ on α is successful. An ω-language $L \subseteq A^\omega$ is called *Muller recognizable* if it consists of all ω-words accepted by some Muller automaton over A.

deterministic version

In the nondeterministic version of Muller automaton, a transition relation $\Delta \subseteq Q \times A \times Q$ replaces δ, and acceptance of α means existence of a run σ on α with $\mathrm{In}(\sigma) \in \mathscr{F}$. The ω-languages recognized by nondeterministic Muller automata are definable in S1S (by the same idea as in Theorem 3.1) and hence, obviously coincide with the regular ω-languages. In the sequel we consider only the deterministic version.

Each deterministic Büchi automaton $\mathscr{A} = (Q, q_0, \delta, F)$ is equivalent to a Muller automaton, namely to the automaton $\mathscr{A}' = (Q, q_0, \delta, \mathscr{F})$ where $\mathscr{F}$ consists of all subsets

Easy to complement deterministic Muller automata.

of Q having a nonempty intersection with F. Furthermore, the Muller recognizable ω-languages are closed under Boolean operations: If $\mathscr{A}=(Q, q_0, \delta, \mathscr{F})$ recognizes $L \subseteq A^\omega$, then $(Q, q_0, \delta, 2^Q - \mathscr{F})$ recognizes $A^\omega - L$. Given $\mathscr{A}=(Q, q_0, \delta, \mathscr{F})$ and $\mathscr{A}'=(Q', q_0', \delta', \mathscr{F}')$ recognizing L and L' respectively, $L \cup L'$ is recognized by the product automaton of $\mathscr{A}$ and $\mathscr{A}'$, where the collection of final state sets contains $\{(q_1, q_1'), \ldots, (q_n, q_n')\}$ iff $\{(q_1, \ldots, q_n\} \in \mathscr{F}$ or $\{q_1', \ldots, q_n'\} \in \mathscr{F}'$. These observations, together with Remark 4.1, yield the direction from right to left of the following lemma.

4.3. LEMMA. *An ω-language $L \subseteq A^\omega$ is Muller recognizable iff L is a Boolean combination (over A^ω) of sets $\overrightarrow{W}$ with regular $W \subseteq A^*$.*

PROOF. For the direction from left to right, consider a Muller automaton $\mathscr{A}=(Q, q_0, \delta, \mathscr{F})$ recognizing $L \subseteq A^\omega$; write W_q for the set of finite words recognized by the finite automaton $(Q, q_0, \delta, \{q\})$. By definition, $\mathscr{A}$ accepts α iff, for some $F \in \mathscr{F}$, α belongs to $\overrightarrow{W_q}$ for all $q \in F$ and α does not belong to $\overrightarrow{W_q}$ for all $q \in Q - F$. Thus we get the desired representation (writing $\sim$ for the complement w.r.t. A^ω):

$$L = \bigcup_{F \in \mathscr{F}} \left(\bigcap_{q \in F} \overrightarrow{W_q} \cap \bigcap_{q \in Q - F} \sim \overrightarrow{W_q} \right) \quad \square$$

We now show that deterministic Muller automata and nondeterministic Büchi automata are equivalent in recognition power. Note that this reproves closure of Büchi recognizable sets under complement. The main difficulty lies in the fact that a membership test "$\alpha \in U.V^\omega$?" as performed by a Büchi automaton, i.e. the test whether α can be split into segments $uv_1v_2v_3\ldots$ with $u \in U$ and $v_i \in V$, involves "unlimited guessing" in choosing the segments v_i; this has to be reduced to a deterministic procedure which should depend only on uniformly bounded information about finite prefixes of α.

4.4. MCNAUGHTON'S THEOREM (cf. [62]). *An ω-language is regular (i.e., Büchi recognizable) iff it is Muller recognizable.*

The direction from right to left follows from Lemma 4.3 and the closure properties for Büchi automata. For the converse, it will suffice, again by Lemma 4.3, to show the following theorem.

4.5. THEOREM. *Every Büchi recognizable ω-language $L \subseteq A^\omega$ is a finite union of sets of the form $\overrightarrow{W} \cap \sim\overrightarrow{W'}$ where $W, W' \subseteq A^*$ are regular.*

Before turning to the proof, we mention that Theorem 4.5 motivates also the following modified version of Muller automaton, introduced by Rabin [95].

DEFINITION. A (sequential) *Rabin automaton* over the alphabet A is of the form $\mathscr{A}=(Q, q_0, \delta, \Omega)$ where Q, q_0, δ are as for Muller automata and $\Omega=\{(L_1, U_1), \ldots, (L_n, U_n)\}$ is a collection of *accepting pairs* (L_i, U_i) with $L_i, U_i \subseteq Q$. A run σ of $\mathscr{A}$ is *successful* if for some $i \in \{1, \ldots, n\}$ we have $\mathrm{In}(\sigma) \cap L_i = \emptyset$ and $\mathrm{In}(\sigma) \cap U_i \neq \emptyset$, i.e. the

L_i-states occur only finitely often but some U_i-state occurs infinitely often in σ. $\mathcal{A}$ *accepts* an ω-word $\alpha \in A^\omega$ if the unique run σ of $\mathcal{A}$ on α is successful in this sense.

Let $\mathcal{A}$ be a Rabin automaton as in the above definition. If W_i (respectively W_i') is the regular language recognized by the finite automaton (Q, q_0, δ, U_i) (resp. (Q, q_0, δ, L_i)), then the Rabin automaton (Q, q_0, δ, Ω) recognizes the ω-language

$$\bigcup_{i=1}^{n} (\overrightarrow{W_i} \cap \sim \overrightarrow{W_i'}),$$

hence a language as required in Theorem 4.5. From Lemma 4.3 and Theorem 4.5, it follows that any Rabin automaton is equivalent to a Muller automaton and vice versa.

PROOF OF THEOREM 4.5 Let $\mathcal{A}$ be a Büchi automaton. By Lemma 2.2 it suffices to consider the case $L = U.V^\omega$ with $\sim_{\mathcal{A}}$-classes U, V and $V.V \subseteq V$. We use the notation introduced in the proof of Lemma 2.2, in particular the merging relation $\cong_\alpha$. (By $k \cong_\alpha k'(m)$ we mean here that $\alpha(k, m) \sim_{\mathcal{A}} \alpha(k', m)$.) In Lemma 2.2, the condition $\alpha \in U.V^\omega$ was shown to be equivalent to (cf. Fig. 5)

$$\exists k_0 (\alpha(0, k_0) \in U \wedge \exists^\omega k (\alpha(k_0, k) \in V \wedge \exists m\, k_0 \cong_\alpha k(m))). \tag{4.1}$$

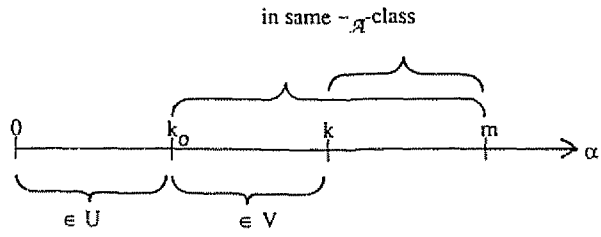


Fig. 5.

Call a segment $\alpha(k_0, m)$ a *V-witness* if, for some k with $k_0 < k < m$, m is the *smallest* position such that $\alpha(k_0, k) \in V$ and $k_0 \cong_\alpha k(m)$. It is not difficult to verify that the set $W_V \subseteq A^*$ of V-witnesses is regular. Since for any k as described in (4.1) there is a unique V-witness, we obtain that (4.1) is equivalent to

$$\exists k_0 (\alpha(0, k_0) \in U \wedge \exists^\omega m\, \alpha(k_0, m) \text{ is a } V\text{-witness}), \tag{4.2}$$

in other words, $\alpha \in U.\overrightarrow{W_V}$.

The aim is to rewrite (4.2) as a Boolean combination of conditions "$\exists^\omega m\, \alpha(0, m) \in W$" (with regular W); so we want to exchange the quantifiers $\exists k_0$ and $\exists^\omega m$ in (4.2). In the desired condition of the form "$\exists^\omega m \exists k_0 \ldots$" we have to ensure that k_0 may be chosen fixed (independent of the m). The idea is to postulate infinitely many prefixes $\alpha(0, m)$ which admit a decomposition $\alpha(0, m) = \alpha(0, k_0)\alpha(k_0, m)$ with $\alpha(0, k_0) \in U$ and $\alpha(k_0, m) \in W_V$, and to guarantee that only finitely many choices of k_0 occur while m increases. A simple approach would be to refer always to the smallest $k_0 < m$ with $\alpha(0, k_0) \in U$.

However, this k_0 might not have the property that there are infinitely many m with $\alpha(k_0, m) \in W_V$: it may happen that only some greater k_1 with $\alpha(0, k_1) \in U$, which does not merge with k_0, has this property. Suppose there are exactly r $\cong_\alpha$-classes with elements k such that $\alpha(0, k) \in U$, i.e. there are r positions, say $k_1, \ldots, k_r$, in α which do not merge pairwise and satisfy $\alpha(0, k_i) \in U$ ("case r"). Then k_0 may be chosen as one of these positions. So in "case r" we require for infinitely many m the existence of $k_1, \ldots, k_r < m$ which do not merge pairwise at m and satisfy $\alpha(0, k_i) \in U$. We express that for increasing m these k_i stay below a fixed finite bound (so that only finitely many choices of k_0 are possible) by saying that for only finitely many m a new choice of $k_1, \ldots, k_r$ (or just of the maximal position k_r) is necessary

To give a precise formulation, let us call k *r-appropriate for* m iff k is the smallest number less than m such that k is the last element k_r of an r-tuple $(k_1, \ldots, k_r)$ with $0 < k_1 < \cdots < k_r$, $\alpha(0, k_i) \in U$ for all k_i, and not $k_i \cong_\alpha k_j(m)$ for $k_i \neq k_j$. If in addition $\alpha(k_i, m) \in W_V$ for some k_i, we say that k is r-appropriate for m *via* W_V. Finally, a *new* k r-appropriate for m is a number which is r-appropriate for m and fails to be r-appropriate for any $m' < m$. Then, assuming "case r", (4.2) amounts to the following:

$$\exists^\omega m\,[\text{there exists a } k \text{ } r\text{-appropriate for } m \text{ via } W_V] \wedge \exists^{<\omega} m\,\{\text{there exists a new } k \text{ } r\text{-appropriate for } m\}. \tag{4.3}_r$$

Both conditions $[\ldots]$ and $\{\ldots\}$ above depend only on the segment $\alpha(0, m)$; moreover, one can construct two finite automata accepting exactly the words $\alpha(0, m)$ satisfying these requirements. (As an alternative, one may note definability of $[\ldots]$ and $\{\ldots\}$ in WS1S and apply Theorem 3.2.) Denote by W_r and W_r' the (regular) sets of words $\alpha(0, m)$ satisfying $[\ldots]$, respectively $\{\ldots\}$. Then $(4.3)_r$ says that $\alpha \in \overrightarrow{W_r} \cap \sim \overrightarrow{W_r'}$.

The disjunction of the conditions $(4.3)_r$ over all possible r is equivalent to (4.2). Since r cannot exceed the finite number n of different $\sim_\alpha$-classes, we obtain

$$\bigcup_{r=1}^{n} (\overrightarrow{W_r} \cap \sim \overrightarrow{W_r'})$$

as the desired representation of $U.V^\omega$. □

McNaughton's Theorem has many interesting consequences, among them further characterizations of the regular ω-languages. For example, parts (c) and (d) of the following result give a simple inductive construction of the regular sequence sets, and show the surprising fact that S1S and the weak theory WS1S have the same expressive power.

4.6. THEOREM. *For an ω-language $L \subseteq A^\omega$ the following conditions are equivalent:*

(a) *L is regular.*

(b) *L is a finite union of sets $U.\overrightarrow{W}$ where $U, W \subseteq A^*$ are regular.*

(c) *L can be obtained from A^ω by finitely many applications of union, complement (w.r.t. A^ω) and concatenation with a regular set $W \subseteq A^*$ on the left.*

(d) *L is definable in* WS1S.

PROOF. Implication (a) → (b) is clear from the proof of Theorem 4.5 (cf. condition (4.2))

For (b) → (c) it suffices to represent $\overline{W}$ as stated in (c), supposing that W is regular. Let (Q, q_0, δ, F) be a finite automaton recognizing W; set $W_{pq} := \{w \in A^* \mid \delta(p, w) = q\}$. We have $\overline{W} = \sim L$ where L contains all ω-words which have only finitely many prefixes in W, i.e. no prefix in W or a last prefix in W. Note that $\alpha(0, m)$ is a last prefix of α in W iff for some state $p \in F$, $\delta(q_0, \alpha(0, m)) = p$ and there is no $n > m$ and no $q \in F$ with $\delta(p, \alpha(m, n)) = q$. This yields the equation

$$L = \sim(W.A^\omega) \cup \bigcup_{p \in F} \left(W_{q_0 p} . \sim \left(\bigcup_{q \in F} (W_{pq} \cap A^+).A^\omega \right) \right).$$

So L and hence $\overline{W}$ are represented as desired.

Implications (c) → (a) and (d) → (a) are obvious; so it remains to show (a) → (d). By McNaughton's Theorem, a regular ω-language L is a Boolean combination of sets $\overline{W}$ with regular W. From a WS1S-formula $\varphi(\bar{X})$ which defines W by Theorem 3.2, we obtain a WS1S-formula $\psi(\bar{X}, y)$ which expresses over ω-words that the prefix up to position y satisfies $\varphi(\bar{X})$ (use relativization to the positions $< y$). Now L is definable in WS1S by a Boolean combination of formulas $\forall x \exists y (x < y \wedge \psi(\bar{X}, y))$. □

There are several fully worked-out expositions of McNaughton's Theorem; we mention [95, 10, 125, 15, 27]. The above proof of Theorem 4.5 follows [119]. An elegant automaton construction, giving an essentially optimal size bound for the constructed deterministic automaton, was recently found by Safra [99]: for any nondeterministic Büchi automaton with n states there is an equivalent deterministic Rabin automaton with $2^{O(n \log n)}$ states and n accepting pairs. This result has applications in obtaining good upper complexity bounds for several logics of programs [32, 23, 137].

5. Acceptance conditions and Borel classes

Definability of sequence sets is a traditional subject in set-theoretic topology and descriptive set theory. Automata on ω-words constitute a special type of "finitary" definability of sequence sets and lead to combinatorial problems not considered in the classical mathematical literature. (McNaughton's Theorem is an example.) Conversely, several basic topological notions, in particular in connection with the Borel hierarchy, have a natural meaning in the context of automata and help to systematize the acceptance conditions. In the present section we introduce the classification of acceptance modes for automata in a topological setting, and indicate applications to sequence properties which arise in distributed systems. For a more complete discussion, see the surveys [49, 110].

We refer to the *Cantor topology* on the set A^ω, where A is a finite alphabet. This topology may be characterized in several equivalent ways:

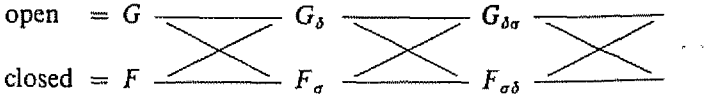
- as the product topology of the discrete topology on A,
- by declaring as *open sets* all ω-languages $W.A^\omega$ with $W \subseteq A^*$, or the sets $\{w\}.A^\omega$, as a basis,

– by the *metric* $d: A^\omega \times A^\omega \to R$, given by

$$d(\alpha,\beta)=\begin{cases}0 & \text{if } \alpha=\beta,\\ 1/2^n \text{ with } n=\min\{i \mid \alpha(i)\neq\beta(i)\} & \text{else}\end{cases}$$

Taking the example $A=\{0,1\}$, the elements of the space A^ω may be considered as paths through the infinite binary tree (where "0" means "branch left" and "1" means "branch right"). Two paths α,β are close to each other if they have a long common initial segment. The so-called Cantor discontinuum is a linearly ordered representation of the space, corresponding to the left to right ordering of the paths in the binary tree.

Following classical terminology we denote by G the class of open ω-languages and by F ("fermé") the class of closed ω-languages. The *Borel hierarchy* is obtained by taking alternately countable intersections and unions starting with ω-languages in G or F. By G_δ (F_σ) one denotes the countable intersections (resp. unions) of sets in G (resp. F), by $G_{\delta\sigma}$ ($F_{\sigma\delta}$) the countable unions (resp. intersections) of sets in G_δ (resp. F_σ), etc. Then a hierarchy of the form



results, where each line indicates a proper inclusion. A set L belongs to some class in the hierarchy (say $L \in G_\delta$) iff its complement is in the dual class (in the example, $\sim L \in F_\sigma$).

Our aim is to locate the regular ω-languages in the Borel hierarchy. As a preparation we note two useful facts about open, respectively closed sets. First, an open set $W.A^\omega$ is equal to $W_0.A^\omega$ where W_0 contains all words from W with no proper prefix in W. Any two words in W_0 are then incomparable w.r.t. the prefix relation; if this holds and $W.A^\omega = W_0.A^\omega$, we say that W_0 is a *minimal basis* for $W.A^\omega$. The property of being a minimal basis will not be destroyed if we replace, for some n, a word $w \in W_0$ by all words in $\{w\}.A^n$. Using thus longer and longer words in minimal bases, it is possible to represent a sequence $W_1.A^\omega, W_2.A^\omega, \ldots$ of open sets by a sequence of pairwise disjoint minimal bases W'_i.

Secondly, note that the closed sets in the Cantor topology do not coincide with the sets $\overline{W} \subseteq A^\omega$. The topological *closure* of a set L consists of all sequences that have arbitrary long common prefixes with ω-words in L. Thus the set $L=\overline{a^*bb^*}$ is not closed since a^ω is in the closure of L but does not belong to L itself. The sets $\overline{W}$ are more general than closed sets in the following sense.

5.1. REMARK. *$L=\overline{W}$ for some $W \subseteq A^*$ iff L is in G_δ.*

PROOF. Assume first $L \in G_\delta$, i.e. $L=\bigcap_{n\geq 0} W_n.A^\omega$. Without loss of generality, we have $W_0 \supseteq W_1 \supseteq \cdots$ (if not, replace W_n by $\bigcap_{i\leq n} W_i$). Choose a sequence $W'_0, W'_1, \ldots$ of disjoint minimal bases (as explained above) of $W_0.A^\omega, W_1.A^\omega, \ldots$ and set $W:=\bigcup_{n\geq 0} W'_n$. Then $\alpha \in \overline{W}$ iff α has a prefix in W'_n for infinitely many n iff α has a prefix in W_n for infinitely many n iff (by monotonicity) $\alpha \in \bigcap_{n\geq 0} W_n.A^\omega$, i.e. $\alpha \in L$. Conversely, suppose

that $L = \overline{W}$. Denote by $A^{\geq n}$ the set of words of length $\geq n$ over A. We have

$$\overline{W} = \bigcap_{n \geq 0} (W \cap A^{\geq n}).A^{\omega}$$

and hence $L \in G_\delta$. □

For a discussion of the topology induced by the operation $\overline{\ }$ as topological closure see [97]. By Remark 5.1 and McNaugthon's Theorem, the regular ω-languages occur very low in the Borel hierarchy, as expressed in the following theorem.

5.2. THEOREM. *Every regular ω-language L is a Boolean combination of G_δ-(and/or F_σ-)sets; in particular, $L \in G_{\delta\sigma} \cap F_{\sigma\delta}$.*

We now discuss refinements of this result due to Landweber [55], which yield a characterization of the regular sets $L \in A^\omega$ occurring on the levels G, F, G_δ, F_σ of the Borel hierarchy, and establish effective procedures for deciding whether a regular set is of one of these types.

DEFINITION. Let $\mathscr{A} = (Q, q_0, \delta, F)$ be a deterministic finite automaton over A and $\alpha \in A^\omega$. $\mathscr{A}$ *1-accepts* α iff $\exists n\ \delta(q_0, \alpha(0, n)) \in F$, and $\mathscr{A}$ *2-accepts* α iff $\exists^\omega n\ \delta(q_0, \alpha(0, n)) \in F$. An ω-language $L \subseteq A^\omega$ is called 1-(resp. 2-)*recognizable* if L consists of the ω-words 1-(resp. 2-)accepted by some deterministic finite automaton.

So 2-recognizability is Büchi recognizability by deterministic automata. Since L is 1-(resp. 2-)recognizable iff $L = W.A^\omega$ (resp. $L = \overline{W}$) for some regular set W, one calls such ω-languages regular-open (resp. regular-G_δ). The following result shows in particular that regular ω-languages which are open are in fact regular-open, and similarly for G_δ.

5.3. THEOREM (Landweber [55]). (a) *A regular ω-language is in G iff it is 1-recognizable.*
(b) *A regular ω-language is in G_δ iff it is 2-recognizable.*
(c) *It is decidable whether a regular ω-language (represented, say, by a Muller automaton) is 1-recognizable, resp. 2-recognizable.*

For the theorem it suffices to show the following lemma.

5.4. LEMMA. *There are effective procedures transforming any Muller automaton $\mathscr{A}$ into a Muller automaton $\mathscr{A}_1$, resp. $\mathscr{A}_2$, such that $L(\mathscr{A}_1)$ is 1-recognizable, $L(\mathscr{A}_2)$ is 2-recognizable, and*

$$L(\mathscr{A}) \in G \text{ iff } \mathscr{A} = \mathscr{A}_1; \qquad L(\mathscr{A}) \in G_\delta \text{ iff } \mathscr{A} = \mathscr{A}_2.$$

PROOF. Assume $\mathscr{A} = (Q, q_0, \delta, \mathscr{F})$ is a Muller automaton, w.l.o.g., such that each $q \in Q$ is reachable from q_0 (i.e., there is $w \in A^*$ with $\delta(q_0, w) = q$). By a "loop of $\mathscr{A}$" we shall mean a strongly connected subset of $\mathscr{A}$ (considering $\mathscr{A}$ as a directed graph).

(a) Call a state q of $\mathscr{A}$ an $\mathscr{F}$-state iff q is located in a loop of $\mathscr{A}$ which forms a set from $\mathscr{F}$. Define $\mathscr{A}_1 := (Q, q_0, \delta, \mathscr{F}_1)$ by extending $\mathscr{F}$ to $\mathscr{F}_1$ such that $\mathscr{F}_1$ contains all loops of $\mathscr{A}$ which are reachable from some $\mathscr{F}$-state of $\mathscr{A}$. Let F_1 be the union of the sets $F \in \mathscr{F}_1$. $\mathscr{A}_1$ is equivalent to (Q, q_0, δ, F_1) with 1-acceptance. Namely, some state $q \in F_1$ is reached by (Q, q_0, δ, F_1) on input α iff on α $\mathscr{A}_1$ assumes ultimately the states of a loop from $\mathscr{F}_1$. It remains to show that $L(\mathscr{A}) \in G$ implies $\mathscr{A} = \mathscr{A}_1$. For this we have to verify $\mathscr{F}_1 \subseteq \mathscr{F}$. Let $F \in \mathscr{F}_1$; so there is an $\mathscr{F}$-state q of $\mathscr{A}$ from which the loop F can be reached. Since q is an $\mathscr{F}$-state, we can choose a sequence α inducing $\mathscr{A}$ to assume a loop from $\mathscr{F}$ in which q is located. By assumption, $L(\mathscr{A}) = W.A^\omega$ for suitable W; so some prefix w of α is in W. Since *all* sequences $w\beta$ are in $L(\mathscr{A})$, any loop reachable from q, and hence F, is realized via some ω-word in $L(\mathscr{A})$. Thus $F \in \mathscr{F}$.

(b) $\mathscr{A}_2$ is constructed from $\mathscr{A}$ by enlarging $\mathscr{F}$ to $\mathscr{F}_2$ as follows: for any state q in a loop forming a set $F \in \mathscr{F}$, and any set E forming some loop containing q, add $F \cup E$ to $\mathscr{F}_2$. $L(\mathscr{A}_2)$ is 2-recognizable: for this we construct a Büchi automaton working as $\mathscr{A}$ does, but further equipped with a "state memory" S which collects the visited states and is set back to $\emptyset$ whenever S includes some $\mathscr{F}$-set. So the states are of form $(q, S) \in Q \times 2^Q$, and from (q, S) the Büchi automaton passes to state $(\delta(q, a), S \cup \{\delta(q, a)\})$ by letter a if $S \cup \{\delta(q, a)\}$ does not include a set from $\mathscr{F}$; otherwise, the transition goes to $(\delta(q, a), \emptyset)$. The initial state is $(q_0, \emptyset)$, and the states $(q, \emptyset)$ are the final ones. 2-Acceptance for this automaton means that some loop extending a loop forming an $\mathscr{F}$-set is ultimately assumed on the given input, i.e. that $\mathscr{A}_2$ accepts. Suppose now $L(\mathscr{A}) \in G_\delta$, say $L(\mathscr{A}) = \overline{W}$, and consider two loops F and E of $\mathscr{A}$ as above with common state q. We show $F \cup E \in \mathscr{F}$ (and hence $\mathscr{F} = \mathscr{F}_2$). Pick w such that $\delta(q_0, w) = q$. Via the loop F we can extend w to a sequence α of $L(\mathscr{A})$ $(= \overline{W})$ and hence reach some finite prefix of α in W, say $wu_1 \in W$. From wu_1 we may complete the loop F back to state q (say via v_1) and continue further through the loop E, again back to q (say via w_1). Repeating the process, we obtain a sequence $wu_1v_1w_1u_2v_2w_2 \ldots$ which is in $\overline{W}$ and causes $\mathscr{A}$ to assume ultimately the states in $F \cup E$. Hence $F \cup E \in \mathscr{F}$. □

A characterization similar to Theorem 5.3 can also be given for the regular ω-languages in the Borel class F (closed sets), respectively F_σ: in these cases one refers to nondeterministic finite automata with the acceptance condition that in some run all states (resp. almost all states) are final. Deterministic automata can be used when the final state set F is replaced by a system $\mathscr{F}$ of final state sets (as in Muller automata).

A more refined system of structural invariants for regular ω-languages is given in [133]. It allows for example, to estimate the length of the Boolean expressions arising in McNaughton's Theorem: Consider representations of a regular set $L \subseteq A^\omega$ by Boolean combinations $\bigcup_{i=1}^n (\overline{W_i} \cap \sim \overline{W_i'})$ of G_δ-sets, and define the *Rabin index* of L to be the smallest n such that L is obtained in this form with regular W_i, W_i'. Wagner [133] (and independently Kaminski [50]) show that this index induces a strict hierarchy; moreover, the Rabin index of a regular ω-language is effectively computable. For a rather complete survey of further results in this field see [110].

Restricted acceptance conditions have been investigated for various other machine models (besides finite automata), for example, pushdown automata [17], deterministic pushdown automata [18, 58], Petri nets [126], and Turing machines [19, 109]. A

unified treatment (referring to the general notion of storage type) is given by Engelfriet and Hoogeboom [138]. Generalized acceptance conditions for finite automata that lead beyond the regular ω-languages are considered in [134].

Recently, the topological approach has also been applied in the classification of sequence properties that arise in distributed systems. Intuitively, one calls a property of state sequences of a system a *safety property* if it ensures that nothing "bad" (like deadlock) happens at any time instance. Similarly, a *liveness property* guarantees that, given any time instance, something "good" (like entering a critical section) will eventually happen. For a systematic treatment of such sequence properties (leading to specific verification strategies), several exact definitions for describing these intuitive notions have been proposed, e.g. by Alpern and Schneider [1], Lichtenstein, Pnueli and Zuck [57], and Manna and Pnueli [60]. These proposals agree in identifying safety properties with closed sets (in the Cantor topology). Liveness properties, however, have been defined in different says. Alpern and Schneider [1] suggest to consider them as *dense* sets in the Cantor space. (A set $L \subseteq A^\omega$ is dense iff every $w \in A^*$ is extendible to a sequence $w\alpha \in L$.) In this case, a topological fact ("every set of the space is the intersection of a closed set with a dense set") can be applied to obtain a decomposition of correctness proofs into two parts, one establishing a safety property and the other showing a liveness property.

In [57, 60], liveness properties are connected with G_δ-sets. The latter paper sets up a correspondence between the Borel hierarchy levels F, G, F_σ, G_δ and the sequence properties of type "safety", "guarantee", "persistence", and "recurrence" respectively. Proof principles for the verification of such properties are presented, using the framework of temporal logic.

An ω-language of type $(\sim \vec{U}) \cup \vec{V}$ (which is in the Boolean closure of G_δ) can be regarded as a so-called *fairness condition*: it represents a sequence property which states that infinitely many instances of event U (as prefix of a sequence) imply infinitely many instances of V. For example, this can mean that (in a state sequence of a system) an action which is "enabled" again and again in fact happens again and again. In this way, fairness conditions may serve to ensure the absence of "starvation" in concurrent programs (where a process is enabled infinitely often but does not continue). A corresponding language-theoretical operation ("fair merge" of two ω-languages) is considered by Park [81]. "Fair" acceptance conditions for finite automata over ω-sequences are studied by Priese, Rehrmann and Willecke-Klemme [92]. It is shown there that nonregular ω-languages can be defined when the fairness constraints refer to unbounded future segments of ω-sequences (instead of initial segments that refer to the past). The study of fairness notions remains an important topic of current research. For a general exposition, see the monograph by Francez [34]; as a recent contribution which characterizes regular ω-languages by fairness conditions in the calculus SCCS (synchronous CCS) we mention [40].

6. Star-free ω-languages and temporal logic

An interesting class of regular ω-languages is obtained when the defining monadic second-order formalism S1S is restricted to first-order logic; in this case quantification

is allowed only over elements (i.e., positions in sequences). The first-order definable ω-languages are closely related to the class of *star-free languages* and to the *propositional temporal logic of linear time*. In this section we discuss both aspects. For a more detailed treatment of temporal logic see Emerson's survey [30] (in this Handbook).

Recall that a language $W \subseteq A^*$ is *star-free* if it can be generated from finite sets of words by repeated application of the Boolean operations and concatenation (see [87]). The resulting star-free expressions are very similar to first-order formulas, by the close correspondence between the operations $\sim$, $\cup$, . and the connectives $\neg$, $\vee$, $\exists$. For example, the star-free language $A^*.(a \cup c).\sim(A^*.b.A^*)$ over $A = \{a, b, c\}$ is defined by the first-order sentence

$$\exists x((x \in Q_a \vee x \in Q_c) \wedge \neg \exists y(x < y \wedge y \in Q_b)).$$

An easy induction shows that each star-free language is first-order definable. In particular, if U, V are defined by the first-order formulas φ, ψ, then $U.V$ is defined by $\exists x(\varphi'(x) \wedge \psi'(x))$ where $\varphi'(x)$ and $\psi'(x)$ are the relativizations of φ, ψ to the elements $\leqslant x$ and $> x$ respectively (assuming here for simplicity that $\varepsilon \notin U$). More difficult is the converse translation, which yields the following result.

6.1. THEOREM. (McNaughton and Papert [63]). *A language $W \subseteq A^*$ is star-free iff it is first-order definable (in the signature with $<$ and the unary predicates Q_a for $a \in A$)*

PROOF. For the translation from first-order logic into star-free expressions we use induction over quantifier depth of formulas (i.e., the maximum number of nested quantifiers). In the induction step, we consider the case of the existential quantifier, here for a formula $\exists x\, \varphi(x)$ of quantifier depth $n+1$, assuming that sentences of quantifier depth n define star-free sets. (The general situation $\exists x\, \varphi(x, \bar{y})$, with free variables $\bar{y}$ as parameters, is a little more technical.) We shall reduce the statement $\exists x\, \varphi(x)$ to statements of the form $\exists x(\varphi_{<x} \wedge x \in Q_a \wedge \varphi_{>x})$ where $\varphi_{<x}, \varphi_{>x}$ speak only about the elements $< x$, resp. $> x$. If $\varphi_{<x}$ and $\varphi_{>x}$ are of quantifier depth n, then such a formula describes a language $U.a.V$ where U, V are star-free by induction hypothesis.

As a preparation we introduce an equivalence relation $\equiv_n$ over A^*: define, for $u, v \in A^*$,

$$u \equiv_n v \text{ iff } u \text{ and } v \text{ satisfy the same sentences of quantifier depth } n.$$

Also an extended version of this definition is needed for formulas with free variables, in our case for formulas $\varphi(x)$ with one free variable x. The corresponding models are words with some distinguished position, of the form (u, r) where $1 \leqslant r \leqslant |u|$. We write $(u, r) \equiv_{n,1} (v, s)$ if (u, r) and (v, s) satisfy the same formulas $\varphi(x)$ of quantifier depth n. An induction over quantifier depth shows the basic fact that there are, for any $n \geqslant 1$, only finitely many equivalence classes of $\equiv_n$ and $\equiv_{n,1}$, and that any such class W of words u, resp. class $\underline{W}$ of word models (u, r), can be defined by a sentence φ_W, resp. formula $\varphi_{\underline{W}}(x)$, of quantifier depth n. It follows that

> any formula $\varphi(x)$ of quantifier depth n is equivalent to a finite disjunction of formulas $\varphi_{\underline{W}}(x)$ (namely those $\varphi_{\underline{W}}(x)$ where $\underline{W}$ contains some (u, r) satisfying $\varphi(x)$). (6.1)

Next we note a fact which (unlike (6.1)) depends on the present choice of signature with ordering and unary predicates only: The relations $\equiv_n$ and $\equiv_{n,1}$ are congruences. Namely,

$$\text{if } u \equiv_n v \text{ and } u' \equiv_n v', \quad \text{then } uu' \equiv_n vv'; \tag{6.2}$$

$$\text{if } u \equiv_n v, a \in A, \text{ and } u' \equiv_n v', \text{ then } (uav, |u|+1) \equiv_{n,1} (u'av', |u'|+1). \tag{6.2$_1$}$$

(A convenient way of showing (6.2) and (6.2)$_1$ uses a characterization of $\equiv_n$ and $\equiv_{n,1}$ by the Ehrenfeucht–Fraissé Game. For more details on this game, see [98].) It turns out that (6.2) and (6.2)$_1$ do not depend on the fact that u, v are finite; indeed these statements hold for any linear orderings expanded by unary predicates.

We now can find the desired star-free expression for $\exists x\, \varphi(x)$: by (6.1) it suffices to treat the case of a formula $\exists x\, \varphi_{\underline{W}}(x)$ since

$$\exists x\, \varphi(x) \leftrightarrow \exists x \bigvee_{\underline{W}} \varphi_{\underline{W}}(x) \leftrightarrow \bigvee_{\underline{W}} \exists x\, \varphi_{\underline{W}}(x).$$

Consider now a triple (U, a, V) where U, V are $\equiv_n$-classes and $a \in A$. If there are $u_0 \in U$, $v_0 \in V$ such that $(u_0 a v_0, |u_0|+1) \in \underline{W}$, then, by (6.2)$_1$, for *all* words uav with $u \in U$, $v \in V$, we have that $(uav, |u|+1) \in \underline{W}$. Hence all words from $U.a.V$ satisfy $\exists x\, \varphi_{\underline{W}}(x)$. Thus $\exists x\, \varphi_{\underline{W}}(x)$ defines the union of the sets $U.a.V$ taken over all triples (U, a, V) where U, V are $\equiv_n$-classes and contain words u_0, v_0 as above. Since U, V are star-free by induction hypothesis, $\exists x\, \varphi_{\underline{W}}(x)$ defines a star-free set. □

The correspondence between star-free expressions and first-order formulas is even tighter than expressed in the statement of Theorem 6.1: the classification of star-free languages by dot-depth (= number of alternations between concatenation and Boolean operations) coincides with the classification of first-order definable word-sets in terms of quantifier alternation depth (cf. [120, 116]).

The proof of Theorem 6.1 provides the main prerequisites which are needed for a development of a theory of star-free (or first-order) ω-languages in close analogy to the regular case. It suffices essentially to repeat the proofs in Sections 2 and 4 (in particular, Lemma 2.2, and Theorems 4.5 and 4.6), replacing the congruences $\sim_{\mathscr{A}}$ by the congruences $\equiv_n$ and using the fact that $\equiv_n$-classes are star-free.

6.2 THEOREM (Ladner [53], Thomas [118, 119]). *For an ω-language $L \subseteq A^\omega$, the following conditions are equivalent:*

(a) *L is first-order definable (in the signature $<, Q_a$ for $a \in A$).*
(b) *L is a finite union of sets $U.V^\omega$ where $U, V \subseteq A^*$ are star-free and $V.V \subseteq V$.*
(c) *L is a finite union of sets $\vec{U} \cap \sim\vec{V}$ where $U, V \subseteq A^*$ are star-free.*
(d) *L is obtained from A^ω by repeated application of Boolean operations and concatenation with star-free sets $W \subseteq A^*$ on the left.*

An ω-language satisfying one of the conditions above is called *star-free*. The notion was introduced by Ladner [53], who referred to condition (d) and showed that the star-free ω-languages form a proper subclass of the regular ones. A short and self-contained proof of Theorems 6.1 and 6.2(a)↔(d) is given in [88]. Perrin [86] shows

that the equivalence between (b) and (c) remains true for any class of regular languages which is associated with a variety of semigroups that is closed under the Schützenberger product.

Further aspects of the star-free ω-languages are revealed when one considers their syntactic monoid as introduced in Theorem 2.6. Referring to this monoid, it is possible to extend Schützenberger's Theorem from languages to ω-languages. Recall that this theorem states that a regular language $W \subseteq A^*$ is star-free iff its syntactic monoid is group-free. Via condition (b) of Theorem 6.2, this implies the following equivalence.

6.3 COROLLARY (Perrin [85]). *A regular ω-language $L \subseteq A^\omega$ is star-free iff its syntactic monoid $A^*/\approx_L$ is group-free.*

This result yields an effective test deciding whether a given regular ω-language is star-free. Also one can use this characterization to exhibit regular ω-languages that are not star-free. For this purpose one observes that a nontrivial group exists in $A^*/\approx_L$ iff there are words $u, x, y, z \in A^*$ such that, for infinitely many n, $xu^n yz^\omega \in L$ and, for infinitely many n, $xu^n yz^\omega \notin L$ (or analogously for $x(yu^n z)^\omega$). So the example language L_2 of Section 1 ("between any two a's there is an even number of b, c's") is not star-free, as can be seen by taking $x, y, z = a$ and $u = b$.

As in the theory of regular languages of finite words, the group-free monoids are just a first example of a variety of semigroups that characterizes an interesting language class. Further cases in the domain of ω-languages have been studied by Pécuchet [84].

Much of the recent interest in the star-free ω-languages rests on their connection with propositional temporal logic of linear time (interpreted over ω-models).

DEFINITION. The system PLTL of *propositional linear-time logic* is built up from atomic propositions $p_1, p_2, \ldots$ by means of the Boolean connectives, the unary temporal operators X ("next"), F ("eventually", "*f*inally"), G ("henceforth", "*g*lobally"), and the binary operator U ("*u*ntil").

PLTL-formulas φ with atomic propositions $p_1, \ldots, p_n$ are interpreted in ω-sequences $\alpha \in (\{0,1\}^n)^\omega$. The satisfaction relation $\alpha \models \varphi$ is defined inductively by the following clauses (we skip the Boolean connectives):

$\alpha \models p_i$	iff	$\alpha(0)$ has a 1 in its ith component,	
$\alpha \models \mathrm{X}\varphi$	iff	$\alpha(1, \omega) \models \varphi$	("φ holds next time"),
$\alpha \models \mathrm{F}\varphi$	iff	there is an $i \geqslant 0$ s.t. $\alpha(i, \omega) \models \varphi$	("φ holds eventually"),
$\alpha \models \mathrm{G}\varphi$	iff	for all $i \geqslant 0$, $\alpha(i, \omega) \models \varphi$	("φ holds henceforth"),
$\alpha \models \varphi \mathrm{U} \psi$	iff	there is an $i \geqslant 0$ s.t. $\alpha(i, \omega) \models \psi$ and $\alpha(j, \omega) \models \varphi$ for $0 \leqslant j < i$	("φ holds until eventually ψ holds").

It is straightforward to translate PLTL-formulas (over atomic propositions $p_1, \ldots, p_n$) into formulas of first-order logic (with unary predicate symbols $X_1, \ldots, X_n$ and using the interpretation described in Section 3). For example, the PLTL-

formula

$$G(p_1 \to X((\neg p_2)\ U\ p_1))$$

is equivalent (over ω-sequences $\alpha \in (\{0,1\}^2)^\omega$) to the first-order formula

$$\forall x(x \in X_1 \to \exists y(x < y \wedge y \in X_1 \wedge \forall z(x < z \wedge z < y \to \neg z \in X_2))).$$

Thus PLTL may be considered as a system of first-order logic with only implicit use of variables. Quantification in PLTL-formulas refers to segments that are unbounded to the right, with the only exception of the bounded quantification involved in the until-operator. Hence in PLTL it tends to be hard to express statements about finite segments of sequences. Nevertheless, first-order sentences can be written as PLTL-formulas as stated in the following theorem.

6.4. THEOREM (Kamp [51], Gabbay, Pnueli, Shelah and Stavi [35]). *Propositional temporal logic PLTL (with atomic propositions $p_1, \ldots, p_n$) is expressively equivalent to first-order logic over ω-sequences (in the signature with $<$ and n unary predicates).*

The translation from first-order logic to PLTL is difficult and will not be described here. It involves a nonelementary blow-up in the length of the formulas, as can be seen from the fact that satisfiability of PLTL-formulas is PSPACE-complete [106], while satisfiability over ω-words is nonelementary for star-free expressions or for first-order formulas in the given signature [111].

An extension of PLTL which allows to define exactly the regular sets of ω-sequences was proposed by Wolper [135]; it is called ETL ("extended temporal logic"). The idea is to admit an infinity of temporal operators, each of them associated with a regular grammar (or an automaton). The standard operators of PLTL are included in this set-up as simple examples. Complexity issues for several versions of ETL are studied by Vardi and Wolper [131], Safra and Vardi [141]. Vardi [128] presents another formalism equivalent to ETL, in which the temporal operators (except "next" and "previous") are defined in terms of least and greatest fixed points.

Temporal logic has attracted attention as a framework for the specification and verification of concurrent programs. Temporal logic formulas are well-suited for this purpose since arguing about concurrent programs is primarily concerned with their ongoing behavior in time and not so much with a relation between initial input and final output (to which the formulas of Hoare logic correspond more directly). From the extensive literature on the subject, we select a few aspects connected with Büchi automata and ω-language theory. (The reader will find more on the topic in the surveys [30, 60, 90].)

We refer to a model of computation with shared variables (cf. [30]). A *concurrent program* P consists of, say, n modules $P_1, \ldots, P_n$ where each of the P_i is a sequential program composed of labelled instructions; both shared variables and variables private to the P_i are admitted. The possible computations of P are defined as the interleavings of the computations of the P_i. A state of the program is identified with an n-tuple of instruction labels from $P_1, \ldots, P_n$ and a tuple of values for the program variables. Since the desired properties of such a program (like deadlock freedom, fairness etc.) are typically concerned with the flow of control and not so much with an infinity of possible

values for variables, it is often appropriate to assume that the number of essentially different states is *finite*. In this case one speaks of a *finite-state program*.

Suppose that for a specification of the program P the properties $p_1, \ldots, p_m$ of states are relevant. Then the program can be represented as an annotated directed graph, where nodes are states and arrows represent transitions between states in one step. The state s is annotated by those p_i which are true in s. Formally, the graph is a *Kripke structure* $\mathcal{M}_P = (S, R, \Phi)$ where S is the set of states, $R \subseteq S \times S$ the transition relation between states, and $\Phi: S \to 2^{\{p_1, \ldots, p_m\}}$ a truth valuation. Note that each $\Phi(s)$ can be regarded as an m-bit vector from $\{0,1\}^m$; thus any computation $\sigma = s_0 s_1 \ldots \in S^\omega$ induces a corresponding sequence $\alpha = \Phi(s_0)\Phi(s_1)\ldots$ from $(\{0,1\}^m)^\omega$, containing the relevant information about σ with respect to the properties $p_1, \ldots, p_m$.

In this framework, the *correctness problem* (whether a program P is in accordance with a PLTL-specification φ) is the following question: do all sequences $\alpha \in (\{0,1\}^m)^\omega$ that are given by paths through $\mathcal{M}_P$ satisfy the PLTL-formula φ? In this case one says that "$\mathcal{M}_P$ is a model of φ"; so the correctness problem has been rephrased as a *model checking* problem.

For finite-state programs and PLTL- (or ETL-) specifications, model checking can be carried out effectively. There are several approaches to obtain efficient model checking procedures. Lichtenstein and Pnueli [56] use tableaux (which are extensions of the annotations of $\mathcal{M}_P$ by arbitrary subformulas of the specification) and they obtain a procedure using linear time in the size of the state space and exponential time in the length of the formula. Vardi and Wolper [130] suggest to apply the theory of Büchi automata for the programs *and* for the specifications: First it is possible to view the Kripke structure $\mathcal{M}_P$ as a Büchi automaton $\mathcal{A}_P$ such that $L(\mathcal{A}_P)$ contains the sequences $\alpha \in (\{0,1\}^m)^\omega$ given by $\mathcal{M}_P$. Secondly a PLTL-formula φ defines a regular ω-language (by Theorems 6.2, 6.3) and thus is also representable by a Büchi automaton $\mathcal{A}_\varphi$. Hence the correctness problem reduces to the containment problem for ω-languages "$L(\mathcal{A}_P) \subseteq L(\mathcal{A}_\varphi)$?" or, in negated form, to the intersection problem "$L(\mathcal{A}_P) \cap L(\mathcal{A}_{\neg\varphi}) \neq \emptyset$?". Alpern and Schneider [1], and Manna and Pnueli [59] suggest using Büchi automata as a genuine specification formalism, taking advantage of the fact that a pictoral graph representation can be more transparent than logical formulas. In [59] a variant of Büchi automata is used (called "$\forall$-automata", equivalent to Büchi automata), which involves a "universal" acceptance condition on runs instead of an "existential" one, and is hence better suited for the question whether *all* $\mathcal{A}_P$-runs satisfy the specification.

Further applications of ω-language theory are based on normal form theorems, in particular the representation of regular ω-languages as unions of sets $\bigcup_i U_i \cap \sim V_i$ in McNaughton's Theorem and in Theorem 6.2 (see, e.g., [57, 90, 60]). The determinization of Büchi automata is applied to the verification of probabilistic finite-state programs by Vardi [127], and Courcoubetis and Yannakakis [23].

7. Context-free ω-languages

In this section we give a short account on the use of grammars for the generation of ω-words. A natural approach is to allow infinite leftmost derivations. As an example,

consider the following grammar:

$$G_0: \qquad x_1 \to x_2 x_1, \qquad x_2 \to a x_2 b \mid ab.$$

The infinite derivation

$$x_1 \vdash x_2 x_1 \vdash abx_1 \vdash abx_2 x_1 \vdash ababx_1 \vdash ababx_2 x_1 \vdash \cdots \tag{7.1}$$

generates the ω-word $(ab)^\omega$ from left to right, and

$$x_1 \vdash x_2 x_1 \vdash abx_1 \vdash abx_2 x_1 \vdash abax_2 bx_1 \vdash abaax_2 bbx_1 \vdash abaaax_2 bbbx_1 \vdash \cdots \tag{7.2}$$

yields from left to right the ω-word aba^ω.

Derivation (7.2) will be excluded if we impose the condition that both variables x_1, x_2 be used infinitely often (as left-hand side of applied rules). Thus there are two variants of context-free generation of ω-languages, depending on whether arbitrary (leftmost) derivations are admitted or the variables used infinitely often are also specified.

In the sequel, a context-free grammar G over the alphabet A and with variables (nonterminals) $x_1, \ldots, x_n$ is given by an n-tuple $(G_1, \ldots, G_n)$ of finite sets $G_i \subseteq (A \cup \{x_1, \ldots, x_n\})^*$, where $w \in G_i$ means that $x_i \to w$ is a rule of G. As start symbol the variable x_1 is used. A leftmost derivation

$$y_0 \vdash u_1 y_1 v_1 \vdash u_2 y_2 v_2 \vdash \cdots$$

with $y_0 = x_1, u_i \in A^*$, and $y_i \in \{x_1, \ldots, x_n\}$ generates either an ω-word (the unique common extension of the u_i) or a finite word (if for some n, $u_n = u_{n+1} = \cdots$). Of course, a finite word can also be generated by a terminating derivation. Hence, an appropriate domain for the present discussion is A^∞ instead of A^ω.

DEFINITION. (a) A ∞-language $L \subseteq A^\infty$ is *algebraic* if there is a context-free grammar G such that L consists of all words of A^∞ that are generated from x_1 by a leftmost derivation of G.

(b) A ∞-language $L \subseteq A^\infty$ is *context-free* if there is a context-free grammar G and a system $\mathcal{F}$ of sets of variables of G such that L consists of the words of A^∞ that are generated from x_1 by a leftmost derivation of G where the variables used infinitely often form a set in $\mathcal{F}$.

Algebraic and context-free *ω-languages* are defined analogously (using A^ω instead of A^∞). When part (b) of the definition is applied to right-linear grammars, one obtains the *regular ∞-languages*. Note that in this case the derivations can be viewed as (possibly terminating) runs of nondeterministic Muller automata. Similarly, the ∞-languages defined by right-linear grammars in the sense of (a) are characterized by Büchi automata in which each state is final.

Let us first note that the regular, algebraic, and context-free ω-languages (and hence ∞-languages) form a proper hierarchy.

7.1 THEOREM (Cohen, Gold [17]). *The class of regular ω-languages is properly contained in the class of algebraic ω-languages which itself is properly contained in the class of context-free ω-languages.*

PROOF. It is obvious that any algebraic ω-language is context-free. To show that regular ω-languages are algebraic, we refer to the representation of regular ω-languages in the form $L=\bigcup_{i=1}^{n} U_i . V_i^{\omega}$ where U_i, V_i are regular. The proof will be clear when the case $L=V^{\omega}$ (assuming $\varepsilon \notin V$) is settled. Let G_V be a left-linear grammar which generates the regular set V (say with start symbol y_1). Then G_V extended by the rule $x_1 \to y_1 x_1$ generates ω-words by leftmost derivations of the form

$$x_1 \vdash y_1 x_1 \vdash^* v_1 x_1 \vdash v_1 y_1 x_1 \vdash^* v_1 v_2 x_1 \vdash \cdots$$

and hence defines the ω-language V^{ω}. (Note that if G_V were right-linear, one would obtain the ω-words in $V^* . \overrightarrow{\text{pref } V}$ which in general is different from V^{ω}.)

We indicate properness of the inclusions by definition of suitable example languages: The algebraic ω-language

$$\{a^{n_1}b^{n_1}a^{n_2}b^{n_2}\ldots \mid n_i \geq 1\} \cup \{a^{n_1}b^{n_1}\ldots a^{n_r}b^{n_r}a^{\omega} \mid r \geq 1, n_1, \ldots, n_r \geq 1\}$$

is generated by the example grammar G_0 above and easily shown to be nonregular (by an adaptation of the pumping lemma for nondeterministic finite automata). On the other hand, the first part of the above union is context-free since it is generated by G_0 with the system $\mathscr{F}=\{\{x_1, x_2\}\}$ of one designated variable set (see the above derivation examples (7.1,2)). It turns out that this ω-language is indeed not algebraic. □

The above proof for regular ω-languages can be extended to a characterization of the context-free ω-languages. If $\mathscr{L}$ is a class of languages ($\subseteq A^*$), the *ω-Kleene closure* of $\mathscr{L}$ is the class of all ω-languages which are finite unions of sets $U . V^{\omega}$ with $U, V \in \mathscr{L}$.

7.2. THEOREM (Cohen, Gold [17]). *An ω-language is context-free iff it belongs to the ω-Kleene closure of the class of context-free languages.*

Some standard results on context-free grammars fail when considered in the domain of ∞-languages. We discuss an interesting example of this kind, the reduction to Greibach normal form (i.e., to grammars $G=(G_1, \ldots, G_n)$ where each G_i is contained in $A . (A \cup \{x_1, \ldots, x_n\})^*$). Call an ∞-language *Greibach-algebraic* if it is given as in the definition of "algebraic" but referring to grammars in Greibach form. We shall show that the Greibach algebraic ∞-languages do not cover the class of regular ∞-languages. For the proof one extends the topology over A^{ω} (as introduced in Section 5) to a topology over A^{∞}, by introduction of a new symbol $\Omega \notin A$ and representation of finite words w as sequences $w . \Omega^{\omega}$. We shall verify that Greibach algebraic ∞-languages are closed in this topology. Since there are nonclosed regular ∞-languages (for example, $\overrightarrow{a^*bb^*}$, as shown before Remark 5.1), the Greibach algebraic ∞-languages form a proper subclass of the algebraic ones which is incomparable with the class of regular ∞-languages.

The topological closure $\text{cl}(L)$ of a set $L \subseteq A^{\infty}$ contains the sequences that have arbitrary long common prefixes with sequences from L. In other words, the set

$$\text{adh}(L) := \overrightarrow{\text{pref } L},$$

the *adherence of* L, is the set of accumulation points of L, and we have $\text{cl}(L) = L \cup \text{adh}(L)$.

In order to show that Greibach algebraic ω-languages are closed, let $\alpha \in A^\omega$ be an accumulation point of the Greibach algebraic set L, i.e., for infinitely many prefixes u of α, there is a $\beta \in A^\infty$ with $u\beta \in L$. We have to verify $\alpha \in L$. Consider all leftmost derivations (by a Greibach grammar G for L) which generate a sequence $u\beta \in A^\infty$ where u is prefix of α. We organize the finite initial parts of such derivations in tree form. We obtain a finitely branching tree of finite derivations which has the zero-step derivation x_1 as root and is infinite by assumption on α. By König's Lemma there is an infinite path, i.e. an infinite derivation. This derivation must describe a generation of α (and not only of a finite prefix of α) because G is Greibach.

Boasson and Nivat [7] proved a sharpened form of this result, including also a converse statement.

7.3. THEOREM (Boasson, Nivat [7]). *A context-free ω-language $L \subseteq A^\infty$ is Greibach algebraic iff it is the (topological) closure of a context-free language $W \subseteq A^*$.*

A common framework for characterizations of the Greibach algebraic, algebraic and context-free ω-languages has been developed by Niwinski [77], continuing previous work of Nivat, Park, and others. Here a grammar $G=(G_1, \ldots, G_n)$ is regarded as a *fixed point operator*. The operator maps n-tuples of ω-languages to n-tuples of ω-languages. The first component of its (well-defined) greatest fixed point is, by definition, the ω-ianguage defined by G. More precisely, any word w from $(A \cup \{x_1, \ldots, x_n\})^*$ defines a map F_w that sends an n-tuple $(L_1, \ldots, L_n)$, where $L_i \subseteq A^\infty$, to the ω-language which results from w by substituting L_i for x_i. (For cases like $w = x_1 x_2$, certain conventions concerning concatenation are used, such as $L_1 \cdot L_2 = L_1$ for $L_1 \subseteq A^\omega$, $L_2 \subseteq A^\infty$.) Now with a set $G_i \subseteq (A \cup \{x_1, \ldots, x_n\})^*$ one associates the map which for $(L_1, \ldots, L_n)$ yields the union of the sets $F_w(L_1, \ldots, L_n)$ with $w \in G_i$.

In this way G induces an operator $\bar{G}$ mapping n-tuples to n-tuples of ω-languages. Since $\bar{G}$ is monotone (w.r.t. set inclusion taken componentwise), the Knaster–Tarski Theorem guarantees a greatest fixed point $(K_1, \ldots, K_n)$. (The n-tuple $(K_1, \ldots, K_n)$ is obtained from $(A^\infty, \ldots, A^\infty)$ by a β-fold iteration of $\bar{G}$ for some ordinal β, possibly greater than ω.) Note that in this set-up the sets G_i need not be finite. For a class $\mathscr{L}$ of languages $L \subseteq A^*$, denote by GFP($\mathscr{L}$) the class of ω-languages that are obtained from an operator $\bar{G}$ as first component of its greatest fixed point, where the components G_i of G are in $\mathscr{L}$. For $\mathscr{L}$ = FIN (the finite languages), resp. REG (the regular languages), and CF (the context-free languages), we arrive at the mentioned characterization.

7.4. THEOREM (Niwinski [77]). *For any ω-language L:*

(a) *L is Greibach algebraic iff $L \in$ GFP(FIN),*
(b) *L is algebraic iff $L \in$ GFP(REG),*
(c) *L is context-free iff $L \in$ GFP(CF).*

A much more complicated classification of context-free ω- (or ∞-)languages arises when we consider definitions by deterministic or nondeterministic *pushdown automata* with various modes of acceptance. Cohen and Gold [17] characterize the context-free ω-languages by pushdown automata with a Muller acceptance condition, and the

algebraic ω-languages by pushdown automata with a weakened acceptance mode, requiring a run such that all its states are in one of the designated state sets. A detailed analysis of ω-language classes induced by deterministic pushdown automata is given in [18, 58].

In the above treatment of grammars, a certain asymmetry is manifested in the convention that only left to right generation of ω-words is considered and terminal symbols are ignored when not reached from the left within ω steps. Dropping this restriction, the derivation example (7.2) of the beginning of this section would be regarded as producing the *generalized word* $abaaa\ldots\ldots bbb$. Formally, generalized words over the alphabet A are identified with A-labelled countable orderings (where the ordering $(\omega, <)$ occurs as a special case). Recent references on derivations of generalized words and finite expressions for their description are [24, 122].

There are further devices of computation that have been investigated in connection with ω-languages but cannot be treated in detail here. We mention two such models: *Turing machines* (studied in [19, 109]), and *Petri nets*. From a Petri net, a language and an ω-language can be extracted essentially by collecting all sequences of transitions that describe admissible firing sequences. In [126] a language-theoretical characterization of Petri net ω-languages is established close to the representation of regular ω-sets as unions of sets $U \cdot V^{\omega}$ with regular U, V. Parigot and Pelz [80] describe a logical formalism (extending Büchi's theory S1S) which characterizes Petri net (ω-)languages; they refer to existential formulas in a signature where a primitive for comparison of finite cardinalities has been added.

PART II. AUTOMATA ON INFINITE TREES

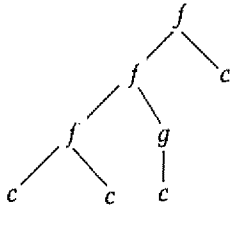
Notation

Given an alphabet A, an A-valued tree t is specified by its set of nodes (the "domain" $\mathrm{dom}(t)$) and a valuation of the nodes in the alphabet A. Formally, a *k-ary A-valued tree* is a map $t: \mathrm{dom}(t) \to A$ where $\mathrm{dom}(t) \subseteq \{0, \ldots, k-1\}^*$ is a nonempty set, closed under prefixes, which satisfies

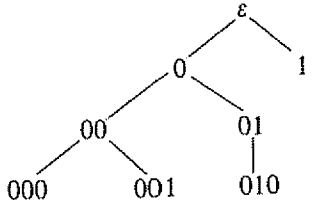
$$wj \in \mathrm{dom}(t),\ i < j \ \Rightarrow\ wi \in \mathrm{dom}(t).$$

As an example over $A = \{f, g, c\}$ consider a finite tree:

t_0:



$\mathrm{dom}(t_0)$:



The *frontier* of t is the set

$$\mathrm{fr}(t) = \{w \in \mathrm{dom}(t) \mid \neg \exists i\ wi \in \mathrm{dom}(t)\},$$

and the *outer frontier* $\mathrm{fr}^+(t)$ contains the points $wi \notin \mathrm{dom}(t)$ where $w \in \mathrm{dom}(t)$ and $i < k$.

We set $\mathrm{dom}^+(t) = \mathrm{dom}(t) \cup \mathrm{fr}^+(t)$. The (proper) prefix relation over $\{0, \ldots, k-1\}^*$ is written $<$. A *path* through t is a maximal subset of $\mathrm{dom}(t)$ linearly ordered by $<$. If π is a path through t, then $t|\pi$ denotes the restriction of the function t to the set π. The *subtree* t_w of t at node $w \in \mathrm{dom}(t)$ is given by $\mathrm{dom}(t_w) = \{v \in \{0, \ldots, k-1\}^* \mid wv \in \mathrm{dom}(t)\}$ *and* $t_w(v) = t(wv)$ for $v \in \mathrm{dom}(t_w)$.

Often trees arise as terms (possibly infinite terms). In this case one refers to a *ranked alphabet* $A = A_0 \cup \cdots \cup A_k$ where A_i contains i-ary function symbols. The example tree t_0 above represents the term $f(f(f(c, c), g(c)), c)$ over the ranked alphabet $A = A_0 \cup A_1 \cup A_2$ with $A_0 = \{c\}$, $A_1 = \{g\}$, $A_2 = \{f\}$.

For easier exposition we shall restrict to *binary trees* in the sequel: By T_A we denote the set of finite binary A-valued trees (where a node has either two sons or no sons), and by T_A^ω the set of infinite A-valued trees with domain $\{0, 1\}^*$. Set $T_A^\infty = T_A \cup T_A^\omega$. Subsets T of T_A or T_A^ω will sometimes be called *tree languages*. Binary trees represent the typical case from which all notions and results to be discussed below are easily transferred to the general situation. For instance, arbitrary k-ary A-valued trees can be represented within binary trees from $T_{A \cup \{\Omega\}}^\omega$, where Ω is a new symbol, via the coding which maps node $n_1 \ldots n_r$ $(n_i < k)$ to node $1^{n_1}0 \ldots 1^{n_r}0$ and associates the value Ω with all nodes of $\{0, 1\}^*$ which are outside the range of this map. In a similar way it is possible to handle countably branching trees in the framework of binary trees.

We now introduce notation concerning *tree concatenation* (defined here in terms of tree substitution). Let $T \subseteq T_A$, $T' \subseteq T_A^\infty$, and $c \in A$. Then $T \cdot_c T'$ contains all trees which result from some $t \in T$ by replacing each occurrence of c on $\mathrm{fr}(t)$ by a tree from T', where different trees are admitted for different occurrences of c. Define a corresponding star operation $*^c$ by

$$T^{*c} = \bigcup_{n \geq 0} T^{nc}$$

where

$$T^{0c} = \{c\}, \qquad T^{(n+1)c} = T^{nc} \cup (T \cdot_c T^{nc}).$$

A tree language $T \subseteq T_A$ is called *regular* iff for some finite set C disjoint from A, T can be obtained from finite subsets of $T_{A \cup C}$ by applications of union, concatenations $\cdot_c$, and star operations $*^c$ where $c \in C$. Note that this notion of regularity generalizes the one for sets of finite words if a word $w = a_1 a_2 \ldots a_r$ over A is considered as a unary tree over $A \cup \{c\}$ of the form $a_1(a_2(\ldots a_r(c)\ldots))$.

We shall also refer to tuples $c = (c_1, \ldots, c_m)$ of concatenation symbols instead of single symbols c. For $T, T_1, \ldots, T_m \subseteq T_A$ let $T \cdot_c (T_1, \ldots, T_m)$ be the set of trees obtained from trees $t \in T$ by substituting, for $i = 1, \ldots, m$, each occurrence of c_i on $\mathrm{fr}(t)$ by some tree in T_i. Furthermore, the set $(T_1, \ldots, T_m)^{\omega_c}$ is defined to consist of all infinite trees obtained by ω-fold iteration of this tree concatenation; more precisely, it contains all trees $t \in T_A^\omega$ for which there are trees $t_0, t_1, \ldots$ such that $t_0 \in \{c_1, \ldots, c_m\}$, $t_{m+1} \in \{t_m\} \cdot_c (T_1, \ldots, T_m)$, and t is the common extension of the trees t'_m which result from the t_m by deleting the symbols c_i at their frontiers.

We shall use expressions like $t_1 \cdot_c t_2$ or $t_0 \cdot_c (t_1, \ldots, t_m)^{\omega_c}$ as shorthands for $\{t_1\} \cdot_c \{t_2\}$, $\{t_0\} \cdot_c (\{t_1\}, \ldots, \{t_m\})^{\omega_c}$ respectively. This notation is extended to *infinite* trees t_i with

domain $\{0, 1\}^*$ by the convention that instead of frontier occurrences of the c_i the "first" occurrences of the c_i are used for replacement, i.e. the occurrences at nodes w such that no $v < w$ exists with a value c_j. We write $t_1 \cdot t_2$ and $t_0 \cdot (t_1, \ldots, t_m)^\omega$ if the symbols c, c are clear from the context.

8. Tree automata

Tree automata generalize sequential automata in the following way: On a given A-valued binary tree, the automaton starts its computation at the root in an initial state and then simultaneously works down the paths of the tree level by level. The transition relation specifies which pairs (q_1, q_2) of states can be assumed at the two sons of a node, given the node's value in A and the state assumed there. The tree automaton accepts the tree if there is a run built up in this fashion which is "successful". A run is successful if all its paths are successful in a sense given by an acceptance condition for sequential automata. It turns out that for infinite trees the reference to Büchi and Muller acceptance leads to nonequivalent types of tree automata. In this section we introduce these tree automata, first studied by Rabin [93, 94].

As a preparation we collect some notions and facts concerning automata over finite trees.

DEFINITION. A (nondeterministic top-down) *tree automaton* over A is of the form $\mathcal{A} = (Q, Q_0, \Delta, F)$, where Q is a finite set of states, $Q_0, F \subseteq Q$ are the sets of initial, respectively final states, and $\Delta \subseteq Q \times A \times Q \times Q$ is the transition relation. A *run* of $\mathcal{A}$ on the finite binary tree t is a tree $r: \mathrm{dom}^+(t) \to Q$ where $r(\varepsilon) \in Q_0$ and $(r(w), t(w), r(w0), r(w1)) \in \Delta$ for each $w \in \mathrm{dom}(t)$; it is *successful* if $r(w) \in F$ for all $w \in \mathrm{fr}^+(t)$. The tree language $T(\mathcal{A})$ recognized by $\mathcal{A}$ consists of all trees t for which there is a successful run of $\mathcal{A}$ on t, and a set $T \subseteq T_A$ is said to be *recognizable* if $T = T(\mathcal{A})$ for some tree automaton $\mathcal{A}$.

Most of the basic results on regular word languages can be reproved for recognizable tree languages, including a Kleene theorem and closure under Boolean operations (cf. Theorem 8.1 below). However, an important difference between the sequential and the tree case appears in the question of *determinism* since deterministic top-down tree automata, where a function $\delta: Q \times A \to Q \times Q$ replaces the transition relation, are strictly weaker than nondeterministic ones. (For instance, any deterministic top-down tree automaton accepting the trees $f(a, b)$ and $f(b, a)$ would have to accept $f(a, a)$ as well; so it does not recognize the finite set $\{f(a, b), f(b, a)\}$, which is clearly recognized by a nondeterministic top-down tree automaton.) Intuitively, tree properties specified by deterministic top-down automata can depend only on path properties. A reduction to determinism is possible when the working direction of tree automata is reversed from "top-down" to "bottom-up". Nondeterministic *bottom-up tree automata* are of the form (Q, Q_0, Δ, F) with $\Delta \subseteq Q \times Q \times A \times Q$ and Q_0, F as before. A successful run on a tree t should have a state from Q_0 at each point of $\mathrm{fr}^+(t)$ and a state from F at the root ε. By an obvious correspondence, nondeterministic bottom-up tree automata are equivalent

to nondeterministic top-down tree automata. However, for the bottom-up version it is possible to carry out the "subset construction" (as for usual finite automata) to obtain equivalent deterministic bottom-up tree automata. Note that the computation of such an automaton, (say, with state set Q), on a term t as input tree may be viewed as an evaluation of t in the finite domain Q. For a detailed treatment, see [37]. In the sequel we refer to the nondeterministic top-down version.

Let us summarize the properties of recognizable tree languages that are needed in the sequel:

8.1. THEOREM (Doner [26], Thatcher and Wright [117]). (a) *The emptiness problem for tree automata over finite trees is decidable (in polynomial time).*

(b) *A tree language $T \subseteq T_A$ is recognizable iff T is regular.*

(c) *The class of recognizable tree languages $T \subseteq T_A$ is closed under Boolean operations and projection.*

PROOF. For (a), the decidability claim is clear from the fact that a tree automaton $\mathcal{A}$, say with n states, accepts some tree iff $\mathcal{A}$ accepts one of the finitely many trees of height $\leqslant n$ (eliminate state repetitions on paths). A polynomial algorithm results from the observation that for the decision it even suffices to build up partial run trees in which each transition of the automaton is used at most once.

Part (b) is shown in close analogy to the proof of Kleene's Theorem for sets of finite words. For details, see [37].

In (c), the step concerning union is straightforward. Closure under complement is shown using the equivalence between nondeterministic top-down and deterministic bottom-up tree automata: for the latter, complementation simply means to change the nonfinal states into final ones and vice versa. For projection, assume $T \subseteq T_{A \times B}$ is recognizable and consider its projection to the A-component, i.e. the set

$$T' = \{s \in T_A \mid \exists t \in T_B \, s \hat{} t \in T\}$$

where $s \hat{} t$ is given by $s \hat{} t(w) = (s(w), t(w))$. If T is recognized by the tree automaton $\mathcal{A}$, then T' is recognized by an automaton which on a tree $s \in T_A$ guesses the B-component and works on the resulting tree as $\mathcal{A}$. □

For nondeterministic automata it is possible to restrict the sets of initial, respectively final states to singletons. For technical reasons we shall henceforth assume that tree automata have a single initial state, which moreover occurs only at the root of run trees (i.e., it does not appear in the third and fourth component of transitions).

We now turn to tree automata over infinite trees. There are two basic types, the Büchi tree automaton (called "special automaton" in [94]) and the Rabin tree automaton, which inherit their acceptance modes from sequential Büchi automata, respectively sequential Rabin automata.

DEFINITION. A *Büchi tree automaton* over the alphabet A is of the form $\mathcal{A} = (Q, q_0, \Delta, F)$ with finite state set Q, initial state $q_0 \in Q$, transition relation $\Delta \subseteq Q \times A \times Q \times Q$, and a set $F \subseteq Q$ of final states. A *run* of $\mathcal{A}$ on a tree $t \in T_A^\omega$ is a map $r: \{0, 1\}^* \to Q$ with $r(\varepsilon) = q_0$

and $(r(w), t(w), r(w0), r(w1)) \in \Delta$ for $w \in \{0, 1\}^*$. The run r is *successful* if on each path some final state occurs infinitely often, i.e.

$$\text{for all paths } \pi, \quad \mathrm{In}(r|\pi) \cap F \neq \emptyset$$

A *Rabin tree automaton* over A has the form $\mathscr{A} = (Q, q_0, \Delta, \Omega)$, where Q, q_0, Δ are as before, and $\Omega = \{(L_1, U_1), \ldots, (L_n, U_n)\}$ is a collection of "accepting pairs" of state sets $L_i, U_i \subseteq Q$. A run r of the Rabin automaton $\mathscr{A}$ is *successful* if

$$\text{for all paths } \pi \text{ there exists an } i \in \{1, \ldots, n\} \text{ with}$$

$$\mathrm{In}(r|\pi) \cap L_i = \emptyset \quad \text{and} \quad \mathrm{In}(r|\pi) \cap U_i \neq \emptyset.$$

A tree $t \in T_A^\omega$ is *accepted* by the Büchi, respectively Rabin tree automaton $\mathscr{A}$ if some run of $\mathscr{A}$ on t is successful in the respective sense. A set $T \subseteq T_A^\omega$ is *Büchi recognizable*, respectively *Rabin recognizable* if it consists of the trees accepted by a Büchi, resp. Rabin tree automaton.

Since any Büchi tree automaton may be regarded as a Rabin tree automaton (set $\Omega = \{(\emptyset, F)\}$), any Büchi recognizable set of infinite trees is Rabin recognizable.

We mention two equivalent variants of Rabin tree automata. In the first, the *Muller tree automaton*, the collection Ω is replaced by a system $\mathscr{F}$ of state sets, and acceptance is defined via existence of a run r such that for each path π, $\mathrm{In}(r|\pi) \in \mathscr{F}$. The second variant, the *Streett automaton* as introduced by Streett [112], is specified as a Rabin tree automaton but uses the negation of the Rabin condition on a given path of a run: a run r of a Streett automaton (Q, q_0, Δ, Ω) is successful if

$$\text{for all paths } \pi \text{ and for all } i \in \{1, \ldots, n\},$$

$$\mathrm{In}(r|\pi) \cap U_i \neq \emptyset \text{ implies } \mathrm{In}(r|\pi) \cap L_i \neq \emptyset.$$

To illustrate the function of Büchi and Rabin tree automata consider an example tree language T_0 over $A = \{a, b\}$:

$$T_0 = \{t \in T_A^\omega \mid \text{some path through } t \text{ carries infinitely many } a\}.$$

A Büchi tree automaton $\mathscr{A}$ which recognizes T_0 may work as follows: by nondeterministic choice, $\mathscr{A}$ guesses a path down the tree and on this path assumes a final state iff letter a is met; on the other parts of the tree only a fixed final state is computed. Then the existence of a successful run amounts to existence of a path in t with infinitely many values a. Thus T_0 is Büchi recognizable (and of course Rabin recognizable). The complement language

$$T_1 = \{t \in T_A^\omega \mid \text{all paths through } t \text{ carry only finitely many } a\}$$

is recognized by a Rabin automaton with one accepting pair $(\{q_a\}, Q)$ where Q is the state set and q_a is computed iff letter a is encountered. Büchi tree automata, however, do not offer in their acceptance condition such a "finiteness test" along paths. Indeed, they cannot recognize T_1, as stated in the following theorem.

8.2. THEOREM (Rabin [94]). *The set T_1 is a tree language which is Rabin recognizable but not Büchi recognizable.*

PROOF. Assume for a contradiction that T_1 is recognized by the Büchi tree automaton $\mathcal{A}=(Q, q_0, \Delta, F)$, say with $n-1$ states. $\mathcal{A}$ accepts all trees t_i which have letter a at the positions $\varepsilon, 1^{m_1}0, \ldots, 1^{m_1}0 \ldots 1^{m_i}0$ where $m_1, \ldots, m_i > 0$, and letter b elsewhere. The figure on the left indicates the nodes with value a in t_2 (see Fig. 6).

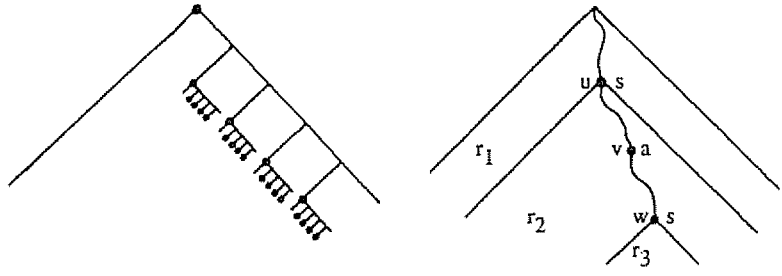


Fig. 6.

Consider a successful run r of $\mathcal{A}$ on t_n. By induction on n one shows that there must be a path in t_n with three nodes $u<v<w$ such that $r(u)=r(w)=s\in F$ and $t_n(v)=a$. Nodes u and w induce a decomposition of r (and t_n) into three parts as follows: Obtain r_1 from r by deleting r_u and setting $r_1(u)=c$. Obtain r_2 similarly from r_u by deleting r_w, and let $r_3=r_w$. Then $r=r_1 \cdot r_2 \cdot r_3$. In the analogous way a decomposition of the underlying tree t_n in the form $t_n=s_1 \cdot s_2 \cdot s_3$ is defined. Now consider $r_1 \cdot r_2^\omega$. It is a successful run of $\mathcal{A}$ on the tree $s_1 \cdot s_2^\omega$. Since by choice of u, v, w this tree contains a path with infinitely many letters a, we obtain a contradiction. □

The following sections present results analogous to Theorem 8.1 for Büchi and Rabin tree automata, excepting only a Kleene-type characterization for Rabin recognizable sets and complementation for Büchi tree automata. For Rabin tree automata, however, closure under Boolean operations and projection holds, which leads to an equivalence with monadic second-order logic as in Theorem 3.1.

9. Emptiness problem and regular trees

In this section the structure of successful runs of Büchi and Rabin tree automata is analyzed.

First we consider Büchi tree automata and show a representation of Büchi recognizable sets in terms of recognizable sets of finite trees.

Let $\mathcal{A}=(Q, q_0, \Delta, F)$ be a Büchi tree automaton, where $F=\{q_1, \ldots, q_m\}$, and let $r: \{0,1\}^* \to Q$ be a successful run of $\mathcal{A}$ on the tree $t\in T_\Delta^\omega$. We claim that r can be built up from finite run trees of $\mathcal{A}$ which are delimited by final states of $\mathcal{A}$. Indeed, the following observation shows that starting from any node u of r the next occurrences of final states enclose a finite tree: Let

$$D_u := \{w \in u.\{0,1\}^* \mid r(v) \notin F \text{ for all } v \text{ with } u<v\leqslant w\}.$$

D_u defines a finitely branching tree which does not contain an infinite path (since r is a successful run). So by König's Lemma D_u is finite, and the nodes of its outer frontier have r-values which are final states. This argument allows to decompose r into "layers" consisting of finite trees: the first layer consists of D_ε and, given the outer frontier F_n of the nth layer, the $(n+1)$st layer is the union of all D_u with $u \in F_n$.

As a consequence we can represent $T(\mathcal{A})$ using recognizable sets of finite trees. We refer to trees from the set $T_{A \cup F}$ which have values from F exactly at the frontier and are otherwise valued in A. (If t is such a tree, $\bar{t}$ results from t by deleting the F-valued frontier.) Now for $q \in Q$, let T_q consist of all such trees t where there is a run of $\mathcal{A}$ on $\bar{t}$ which starts in q and reaches on $\mathrm{fr}^+(\bar{t})$ exactly the states of $\mathrm{fr}(t)$. Each set T_q is recognizable. The argument above shows that an infinite tree is accepted by the Büchi tree automaton $\mathcal{A}$ iff it belongs to

$$T_{q_0} \cdot_q (T_{q_1}, \ldots, T_{q_m})^{\omega q} \tag{9.1}$$

where $q = (q_1, \ldots, q_m)$ is the sequence of all final states in F. Conversely, it is easy to see that for any $(m+1)$-tuple $(T_0, T_1, \ldots, T_m)$ of recognizable sets of finite trees, the expression corresponding to (9.1) defines a Büchi recognizable set of infinite trees:

9.1. THEOREM. *A set $T \subseteq T_A^\omega$ is Büchi recognizable iff there are recognizable sets $T_0, T_1, \ldots, T_m \subseteq T_{A \cup C}$ (where $C = \{c_1, \ldots, c_m\}$) such that $T = T_0 \cdot_c (T_1, \ldots, T_m)^{\omega c}$*

The representation is implicit in Rabin's solution of the emptiness problem [94]; for a stronger statement see [115] (where also rational expressions for tree languages are introduced).

Using the sets T_q above, the emptiness problem for Büchi tree automata is shown to be decidable. For this, we set up an algorithm which eliminates step by step those states of a given Büchi tree automaton $\mathcal{A} = (Q, q_0, \Delta, F)$ that are useless for successful runs. Certainly, a state q cannot appear in a successful run if the set T_q is empty. So eliminate successively those states q from $\mathcal{A}$ where T_q is empty and update the transition relation of $\mathcal{A}$ accordingly. (Note that each T_q is recognizable, so its emptiness can be checked in polynomial time by Theorem 8.1(a).) The elimination procedure stops after at most $|Q|$ steps, delivering a state set Q_0. We claim that $\mathcal{A}$ accepts some infinite tree iff Q_0 still contains the initial state q_0 (which establishes the desired algorithm). To prove this, assume q_0 is not eliminated and let $q_1, \ldots, q_m$ be the final states remaining in Q_0; note that $m \geq 1$ by nonemptiness of T_{q_0} and that also $T_{q_1}, \ldots, T_{q_m}$ are nonempty. So the set (9.1) is nonempty, which is a subset of $T(\mathcal{A})$.

Conversely, it is clear that, in case $T(\mathcal{A}) \neq \emptyset$, the state q_0 will not be eliminated. A closer analysis of the algorithm yields also a polynomial complexity bound:

9.2. THEOREM (Rabin [94], Vardi and Wolper [129]). *The emptiness problem for Büchi tree automata is decidable; moreover, it is logspace complete for* PTIME.

An example tree t in a nonempty Büchi recognizable set T can be obtained by choosing finite trees $t_0, \ldots, t_m$ from the sets $T_{q_0}, \ldots, T_{q_m}$ which the above algorithm produces, and setting $t = t_0 \cdot (t_1, \ldots, t_m)^\omega$. In this infinite tree t the number of distinct

subtrees t_u is bounded by $\Sigma_i |\mathrm{dom}(t_i)|$ and hence finite. Trees satisfying this finiteness condition are called "regular".

DEFINITION. An infinite tree $t \in T_A^\omega$ is said to be *regular* if there are only finitely distinct subtrees t_u in t (where $u \in \{0,1\}^*$).

An equivalent definition states that $t: \{0,1\}^* \to A$ is regular iff there is a finite automaton $\mathcal{A}$ over finite words which "generates t", i.e. whose state set is partitioned into sets Q_a ($a \in A$) such that $\mathcal{A}$ reaches a state in Q_a via input u iff $t(u) = a$. In terms of regular expressions this means that, for each letter $a \in A$, there is a regular expression r_a which defines the language $\{u \in \{0,1\}^* \mid t(u) = a\}$.

Regular trees represent the simplest infinite terms; they are useful in several other areas of computer science, for instance in semantics of program schemes (where they appear as unravellings of finite flowcharts) and in the foundations of logic programming. Reference [20] is a survey which covers the basic theory and applications in semantics.

Extending the above consideration on Büchi recognizable sets, we shall show that also any nonempty Rabin recognizable set contains a regular tree, and thereby see that the emptiness problem for Rabin tree automata is decidable. Since unary regular trees are ultimately periodic ω-words, this generalizes Theorem 1.3 on Büchi automata in a natural way.

9.3. THEOREM (Rabin [95]). (a) *Any nonempty Rabin recognizable set of trees contains a regular tree.*

(b) *The emptiness problem for Rabin tree automata is decidable.*

PROOF. (a) First reduce the problem to "input-free" tree automata with a transition relation $\Delta \subseteq Q \times Q \times Q$. For this, transform a given Rabin tree automaton $\mathcal{A} = (Q, q_0, \Delta, \Omega)$ over A into $\mathcal{A}' = (Q \times A, Q_0, \Delta', \Omega')$ where $\Delta' \subseteq (Q \times A) \times (Q \times A) \times (Q \times A)$ contains a transition $((q,a),(q',a'),(q'',a''))$ iff $(q,a,q',q'') \in \Delta$. Q_0 contains all states (q_0, a), and Ω' contains those pairs of sets of states from $Q \times A$ where the Q-components yield a pair from Ω. Then the successful runs r' of $\mathcal{A}'$ are the pairs $r \wedge t$ where r is a successful run of $\mathcal{A}$ on t; and in this case r is regular provided r' is regular. A corresponding statement holds if $\mathcal{A}'$ has been reduced to an automaton with a single initial state (as mentioned after Theorem 8.1).

So it suffices to show that an imput-free Rabin tree automaton with some successful run admits also a regular sucessful run. Let $\mathcal{A} = (Q, q_0, \Delta, \Omega)$ be a Rabin tree automaton with $\Delta \subseteq Q \times Q \times Q$. Call a state $q \in Q$ *live* if $q \neq q_0$ and the automaton is not forced to stay in q by the single available transition (q,q,q). Using induction on the number of live states of $\mathcal{A}$ we transform a given successful run r into a regular successful run.

If there are no live states, the run r will be stationary from the sons of its root onwards and hence be regular.

In the induction step, distinguish three cases. First, assume that in r some live state q of $\mathcal{A}$ is missing. Then the induction hypothesis can be applied to the automaton where q has been cancelled, and we obtain the desired regular run of $\mathcal{A}$. Secondly,

suppose in r there is a node u such that $r(u)=q$ is live but some live state q' does not appear beyond node u. We find two regular runs r_1, r_2, replacing the r-parts "up to first occurrences of q" and "from first occurrences of q onwards", such that the regular run $r_1 \cdot_q r_2$ is successful for $\mathcal{A}$. Run r_1 is obtained by declaring q as nonlive in $\mathcal{A}$, i.e. we consider the modified automaton where only transition (q, q, q) is available for q, and apply the induction hypothesis. Run r_2 is found from the modified automaton where q is taken as initial state and q' is deleted, again by induction hypothesis

It remains to treat the case that all live states appear in r beyond any given node. Then we may choose a path π_0 through r where all live states appear again and again (and hence no nonlive states can occur). Since r is successful, there is an accepting pair, say (L_1, U_1), such that $\mathrm{In}(r|\pi_0)\cap L_1=\emptyset$ and $\mathrm{In}(r|\pi_0)\cap U_1\neq\emptyset$. Note that L_1 contains only nonlive states since $\mathrm{In}(r|\pi_0)$ is the set of live states. Pick q from $\mathrm{In}(r|\pi_0)\cap U_1$. Again we find two regular runs r_1, r_2, now aiming at the property that the regular run $r_1 \cdot_q r_2^{\omega q}$ is successful for $\mathcal{A}$. Run r_1 is given as in the second case above. Run r_2 is obtained from a modification of $\mathcal{A}$, where q is used in two copies: as initial state and as a nonlive state when revisited the first time; to this modified automaton the induction hypothesis can be applied. In the verification that $r_1 \cdot_q r_2^{\omega q}$ is indeed successful, the interesting case concerns those paths π where q appears infinitely often. We show for such a π that (L_1, U_1) is an appropriate accepting pair: Concerning U_1 it suffices to note that $q\in U_1$. Suppose some L_1-state q' occurs infinitely often on π. Since L_1 contains only nonlive states, q' is nonlive and hence must be the only state occurring infinitely often on π; but this contradicts the fact that already the (live) state q occurs infinitely often.

(b) As in the preceding proof, it is enough to consider input-free automata. Suppose an input-free Rabin automaton $\mathcal{A}$ has to be checked for existence of a successful run. Assume $\mathcal{A}$ has n live states. Only finitely many automata can be obtained from $\mathcal{A}$ by the above-mentioned modifications which reduce the number of live states by 1. Iterating these reductions, we obtain in a constructive way finitely many automata derived from $\mathcal{A}$, where the number i of live states ranges from n to 0. In case $i=0$ (nonlive states only), it is trivial to decide whether a successful run exists. For an automaton with $i+1$ live states, the proof above shows how to decide existence of a successful run, given this information for the automata with i live states. Using n such steps, the answer concerning $\mathcal{A}$ is computed. □

The analysis of the algorithm yields an exponential time bound for its execution. Emerson and Jutla [32] show that the nonemptiness problem for Rabin tree automata is NP-complete.

10. Complementation and determinacy of games

In this survey it is not possible to give a proof of the complementation theorem for Rabin tree automata, the most intricate part of Rabin's paper [93]. We shall explain, however, a treatment of this problem in the framework of "infinite games". This approach was proposed by Büchi [11, 12] and Gurevich, Harrington [42]. It allows to study the complementation problem in the context of descriptive set theory and

clarifies the connections between tree automata and sequential automata. Moreover, some of the earliest problems on ω-automata, the questions of Church [16] on "solvability of sequential conditions", can be settled via these connections.

We consider infinite games as studied by Gale and Stewart [36], and Davis [25].

DEFINITION. Let A and B be alphabets (each with at least two letters) and let $\Gamma \subseteq (A \times B)^\omega$ be an ω-language. The associated *Gale–Stewart game*, again denoted Γ, is played between two players I and II. A single *play* is performed as follows: First I picks some $a_0 \in A$, then II picks some $b_0 \in B$, then I some $a_1 \in A$, and so on in turns. Player I wins the play if the resulting ω-word $(a_0, b_0)(a_1, b_1)\ldots$ is in Γ, otherwise II wins. (If $\alpha = a_0 a_1 \ldots$ and $\beta = b_0 b_1 \ldots$ we shall denote the sequence $(a_0, b_0)(a_1, b_1)\ldots$ by $\alpha\hat{}\beta$.) A *strategy* for I is a function $f: B^* \to A$, telling I to choose $a_n = f(b_0 \ldots b_{n-1})$ if II has chosen $b_0, \ldots, b_{n-1}$. The strategy f induces a transformation $\bar{f}: B^\omega \to A^\omega$; if II builds up β and I plays strategy f, the play $\bar{f}(\beta)\hat{}\beta$ will emerge. We say that f is a *winning strategy* for I if, for all $\beta \in B^\omega$, $\bar{f}(\beta)\hat{}\beta \in \Gamma$. If there is such a strategy for I, we say that I *wins* Γ. Analogous definitions apply to player II (where a strategy for II is a map $g: A^+ \to B$, inducing a transformation $\bar{g}: A^\omega \to B^\omega$).

A game Γ is called *determined* if player I or player II wins Γ.

Determinacy of Γ amounts to an infinitary version of a quantifier law:

$$\neg\, \exists a_0 \forall b_0 \exists a_1 \forall b_1 \ldots (a_0, b_0)(a_1, b_1)\ldots \in \Gamma \qquad \text{("I does not win } \Gamma\text{")}$$
$$\text{iff } \forall a_0 \exists b_0 \forall a_1 \exists b_1 \ldots (a_0, b_0)(a_1, b_1)\ldots \notin \Gamma \quad \text{("II wins } \Gamma\text{")}.$$

But the assumption that all games Γ are determined is a strong set-theoretic hypothesis which contradicts the Axiom of Choice (cf. [66]). Under certain restrictions, however, determinacy has been shown: Martin [61] proved that a game Γ is determined provided Γ belongs to the Borel hierarchy. (See Gurevich [139] for a lucid discussion of the cases that Γ is open or in the Borel class F_σ.)

In the sequel, the relevance of determinacy lies in the fact that it allows to transform the statement

$$\neg\, \exists \text{ strategy } f\ \forall \beta\ \ \bar{f}(\beta)\hat{}\beta \in \Gamma \quad \text{("I does not win } \Gamma\text{")}$$

into the form

$$\exists \text{ strategy } g\ \forall \alpha\ \ \alpha\hat{}\bar{g}(\alpha) \notin \Gamma \quad \text{("II wins } \Gamma\text{")}$$

and hence to write the negation of an existential statement again as an existential statement. Complementation of Rabin tree automata is a natural application, since it requires to express nonexistence of a successful run by one automaton as the existence of a successful run by the complement automaton.

For this purpose the following observation is crucial: *runs of tree automata* (*and, more generally, valued trees*) *are strategies*. Just view a strategy $f: B^* \to A$ as a $|B|$-ary A-valued tree: its nodes represent the words from B^* (the root corresponding to the empty word), and the value $a \in A$ at node w indicates that $f(w) = a$. Similarly, strategies $g: A^+ \to B$ are $|A|$-ary B-valued trees with a default value at the root. If such a tree is *regular* (cf. Section 9), it codes a special kind of strategy: since in this case the value of the tree, say at a node w, is computable by a finite automaton (determined by the state

reached after reading w), the corresponding strategy is "executable by a finite automaton", or shorter: a *finite-state strategy*. Thus for a finite-state strategy f, the choice $f(w)$ depends only on uniformly bounded finite information in w.

We now specialize the games Γ in two ways: First Γ is defined in terms of automata. The key example are the *regular games* $\Gamma \subseteq (A \times B)^\omega$, i.e. games which are recognized by Büchi (or Muller) automata when considered as ω-languages over $A \times B$. Note that in this case Γ belongs to the Borel hierarchy; hence by Martin's result stated above [61] regular games are determined. (Since, by Theorem 5.2, a regular game Γ is even in the Boolean closure of the Borel class F_σ, determinacy may be inferred from an easier result of Davis [25].) Secondly, we impose also corresponding restrictions on the two players: their strategies are now required to be *finite-state*. So the following sharpened questions on determinacy arise:

SOLVABILITY: Given (a presentation of) a regular game Γ, can one decide effectively who wins Γ?

SYNTHESIS: Can one exhibit a finite-state winning strategy for the winner of a regular game Γ?

Both problems were posed by Church [16], referring however to different motivation and terminology. Γ was considered as a "sequential condition" on pairs of sequences (i.e., $\Gamma \subseteq A^\omega \times B^\omega$), expressed in the sequential calculus as a "synthesis requirement" for digital circuits. The circuits should realize a transformation producing for any input sequence β an output sequence α such that $(\alpha, \beta) \in \Gamma$. In recent literature (e.g., [91, 136, 140]) this view is also applied to arbitrary *reactive systems*, i.e. to nonterminating programs (like operating systems), whose purpose is to interact with their environment and to maintain a certain behavior under any influence effected by the environment.

Church's problems were solved positively by Landweber in his thesis (cf. [13]). We sketch here a short proof due to Rabin [95] which exploits the correspondence between strategies and trees.

10.1. THEOREM (Büchi, Landweber [13], see also Trakhtenbrot and Barzdin [125]). *Regular games are determined in the following strong sense: it can be decided effectively who wins, and the winner has a finite-state winning strategy.*

PROOF (Rabin [95]). As a typical example consider the case $A = \{0, 1\}$, $B = \{0, 1\}$. Let $\Gamma \subseteq (\{0, 1\} \times \{0, 1\})^\omega$ be regular, say recognized by the Muller automaton $\mathcal{M} = (Q, q_0, \delta, \mathcal{F})$ over $\{0, 1\} \times \{0, 1\}$. We transform $\mathcal{M}$ into a (deterministic) tree automaton $\mathcal{R} = (Q, q_0, \bar{\delta}, \mathcal{F})$ over $\{0, 1\}$, by defining

$$\bar{\delta}(q, a) = (q', q'') \text{ iff } \delta(q, (a, 0)) = q' \text{ and } \delta(q, (a, 1)) = q'' \quad (a \in \{0, 1\}).$$

$\mathcal{R}$ accepts a tree $t \in T_A^\omega$ iff along all paths β the states assumed infinitely often by $\mathcal{R}$ form a set in $\mathcal{F}$. This means that for all $\beta = d_1 d_2 \ldots$ (where $d_i \in \{0, 1\}$) the sequence $(t(\varepsilon), d_1)(t(d_1), d_2)(t(d_1 d_2), d_3) \ldots$ is accepted by $\mathcal{M}$ and thus in Γ. Hence $\mathcal{R}$ accepts t iff t is a winning strategy for I in Γ. We may assume that $\mathcal{R}$ is (redefined as) a Rabin tree

automaton. The existence of a winning strategy for I can now be decided effectively, by deciding nonemptiness of $T(\mathscr{R})$ (see Theorem 9.3(b)), and a finite-state strategy is guaranteed in this case by Theorem 9.3(a). The case that II wins (which is the only other possibility by the determinacy result of [25]) is handled similarly. □

The complementation problem for Rabin tree automata requires a more general type of game: With any Rabin tree automaton $\mathscr{A}=(Q, q_0, \Delta, \Omega)$ (accepting A-valued trees) and any tree $t \in T_A^\omega$ we associate a game $\Gamma_{\mathscr{A},t} \subseteq (\Delta \times \{0,1\})^\omega$. Thus, player I picks transitions from Δ, and player II picks elements from $\{0,1\}$, i.e. directions building up a path through the tree t. $\Gamma_{\mathscr{A},t}$ contains all sequences $\alpha \hat{} \beta \in (\Delta \times \{0,1\})^\omega$ which "describe a successful path for $\mathscr{A}$ on t". Formally,

$$\alpha \hat{} \beta = ((s_0, a_0, s_0', s_0''), d_1)((s_1, a_1, s_1', s_1''), d_2) \ldots$$

should satisfy $s_0 = q_0$, $a_i = t(d_1 \ldots d_{i-1})$, $s_{i+1} = s_i'$ if $d_{i+1} = 0$, $s_{i+1} = s_i''$ if $d_{i+1} = 1$, and the state sequence $s_0 s_1 s_2 \ldots$ should fulfil the acceptance condition Ω. Then the winning strategies $f : \{0,1\}^* \to \Delta$ for I are in one-to-one correspondence with the successful runs of $\mathscr{A}$ on t, and we have

$$\mathscr{A} \text{ accepts } t \text{ iff I wins } \Gamma_{\mathscr{A},t}. \tag{10.1}$$

Note that the underlying tree t is completely arbitrary and one can no more expect that the winning strategies are finite-state. However, it turns out that *relativized finite-state strategies* (as we call them) can be guaranteed. Such a strategy, say for player II, is executed by a finite automaton $\mathscr{C}$ which is allowed to use an auxiliary tree $t' \in T_{A'}^\omega$ over an alphabet A' in addition to the given tree $t \in T_A^\omega$ (and the transitions from Δ picked by player I). $\mathscr{C}$ works over $\Delta \times A \times A'$ and outputs directions from $\{0,1\}$ (via a partition of its state set into "0-states" and "1-states"). More precisely, consider the game situation

$$\begin{aligned} &\text{I: } \bar{\tau} = \tau_0 \quad \tau_1 \ldots \quad \tau_n \\ &\text{II: } w = d_1 \quad \ldots d_n \end{aligned}$$

and denote by $(t|w)\hat{}(t'|w)$ the sequence of $A \times A'$-values of $t \hat{} t'$ for the nodes visited along w (namely $\varepsilon, d_1, d_1 d_2, \ldots, d_1 \ldots d_n$). Then the state reached by $\mathscr{C}$ after reading the word $\bar{\tau}\hat{}(t|w)\hat{}(t'|w) \in (\Delta \times A \times A')^{n+1}$ fixes the next choice of player II.

Determinacy for the games $\Gamma_{\mathscr{A},t}$ by such relativized finite-state strategies was stated by Büchi [11] with a short proof hint and it was given a detailed exposition in Büchi [12]. A different and simpler proof was given by Gurevich and Harrington in [42]. We state this difficult result here without proof and from it conclude the complementation for Rabin tree automata.

10.2. THEOREM (Büchi [11, 12], Gurevich and Harrington [42]). *Let $\mathscr{A}$ be a Rabin tree automaton over A and $t \in T_A^\omega$. The game $\Gamma_{\mathscr{A},t}$ is determined, and the winner has a relativized finite-state strategy.*

10.3. COROLLARY. *For any Rabin tree automaton $\mathscr{A}$ (by effective construction) there is a Rabin tree automaton $\mathscr{A}'$ recognizing $T_A^\omega - T(\mathscr{A})$.*

PROOF. Let $\mathscr{A}=(Q, q_0, \Delta, \Omega)$ be a Rabin tree automaton over A. We have to find a Rabin tree automaton $\mathscr{A}'$ such that, for all $t \in T_A^\omega$,

$\mathscr{A}$ does not accept t iff $\mathscr{A}'$ accepts t.

By (10.1), $\mathscr{A}$ does not accept t iff I does not win $\Gamma_{\mathscr{A},t}$. By Theorem 10.2, this holds iff II wins $\Gamma_{\mathscr{A},t}$ in the following sense, using a finite automaton $\mathscr{C}$.

for some tree $t' \in T_A^\omega$, over an auxiliary alphabet A',
if $\alpha \in \Delta^\omega$ is chosen by I and $\beta \in \{0,1\}^\omega$ is the response by II (computed by $\mathscr{C}$ from α, t, t'),
then $\alpha^\wedge\beta$ does not describe a successful path for $\mathscr{A}$ on t.

This can be formulated as follows:

(1) for some $t' \in T_{A'}^\omega$,
(2) for all $\beta \in \{0,1\}^\omega$,
(3) for all $\alpha \in \Delta^\omega$, if β results from α, t, t' by output of $\mathscr{C}$, then $\alpha^\wedge\beta$ does not describe a successful path for $\mathscr{A}$ on t.

Note that (3) is a condition on sequences of the form $\beta^\wedge(t|\beta)^\wedge(t'|\beta) \in (\{0,1\} \times A \times A')^\omega$, that (2) is a condition on trees from $T_{A \times A'}^\omega$, and that (1) is a condition on trees from T_A^ω. Since (3) is easily expressed in S1S, it can be defined by a Muller automaton $\mathscr{M}$. Now construct from $\mathscr{M}$ a deterministic tree automaton $\mathscr{R}$ as in the poof of Theorem 10.1. Then $\mathscr{R}$ accepts the trees satisfying (3) on each path β, i.e. all trees $t \in T_{A \times A'}^\omega$ with property (2). Projection to T_A^ω yields a (nondeterministic) Rabin automaton $\mathscr{A}'$, accepting the trees $t \in T_A^\omega$ with property (1). Since these trees were just the trees for which II wins $\Gamma_{\mathscr{A},t}$, $\mathscr{A}'$ is a Rabin tree automaton as desired. □

The complementation problem for Rabin tree automata has been (and continues to be) investigated by several authors. Muchnik [70] presents an elegant proof using an induction over the number of states in the automata. An interesting approach has recently been developed by Muller and Schupp [75], based on the idea of *alternating automata* over trees. In addition to performing nondeterministic choice, these automata are able to pursue several computations simultaneously. The operation of an alternating tree automaton with state set Q is described by the elements of the free lattice over $\{0,1\} \times Q$. The intended meaning of such an element, say

$$((0, q_1) \wedge (1, q_2)) \vee ((1, q_1) \wedge (1, q_2)),$$

is that the automaton proceeds from a given node with q_1 to the left and q_2 to the right, or proceeds with q_1 to the right and also with q_2 to the right. The second possibility is missing in Rabin tree automata. One can now collect all possible *histories* of the alternating automaton $\mathscr{A}$ over t in a *computation tree* $C(\mathscr{A}, t)$, where one history follows one simultaneous realization of states through the levels of the tree. (So a history is a kind of "multirun".) $\mathscr{A}$ accepts t if for some infinite history, i.e. some path in $C(\mathscr{A}, t)$, all state sequences along paths described by it are successful in the sense of Muller acceptance. Complementation for these alternating automata is easy, performed

by dualizing the given automaton (exchange $\wedge$ and $\vee$ in the transitions, and complement the system of final state sets). The hard closure property is projection, which is shown again by an application of Theorem 10.2. An advantage of this approach is that fragments of the monadic theory of the tree which are defined in terms of restricted second-order quantifiers may be handled by suitable restrictions of the projection operation for alternating automata and hence by weakened versions of Theorem 10.2 (cf. [72, 73]).

11. Monadic tree theory and decidability results

For a transfer of the preceding results from automata theory to logic, trees are represented as model-theoretic structures. If A is the alphabet $\{0,1\}^n$, a tree $t \in T_A^\omega$ is coded by a model of the form

$$\underline{t} = (\{0,1\}^*, \varepsilon, \mathrm{succ}_0, \mathrm{succ}_1, <, P_1, \ldots, P_n),$$

where succ_0, succ_1 are the two successor functions over $\{0,1\}^*$ with $\mathrm{succ}_0(w) = w0$, $\mathrm{succ}_1(w) = w1$, $<$ is the proper prefix relation over $\{0,1\}^*$, and $P_1, \ldots, P_n$ are subsets of $\{0,1\}^*$ with $w \in P_i$ iff the ith component of $t(w)$ is 1.

DEFINITION. The interpreted formalism S2S ("*second-order theory of two successors*") contains variables $x, y, \ldots$ and $X, Y, \ldots$ (ranging over elements, respectively subsets of $\{0,1\}^*$). *Terms* are obtained from the individual variables $x, y, \ldots$ and the constant ε by applications of succ_0, succ_1; we write $x0$ and $x1$ instead of $\mathrm{succ}_0(x)$ and $\mathrm{succ}_1(x)$. *Atomic formulas* are of the form $t = t'$, $t < t'$, $t \in X$ where t, t' are terms and X is a set variable; and arbitrary *formulas* are generated from atomic formulas by Boolean connectives and the quantifiers $\exists, \forall$ (ranging over either kind of variables). If $\varphi(X_1, \ldots, X_n)$ is an S2S-formula and $\underline{t}$ a tree model as above, write $\underline{t} \models \varphi(X_1, \ldots, X_n)$ if φ is satisfied in $\underline{t}$ with P_i as interpretation for X_i. Let $T(\varphi) = \{t \in T_A^\omega \mid \underline{t} \models \varphi(\bar{X})\}$. If $T = T(\varphi)$ for some S2S-formula φ, T is called *definable in* S2S.

The "weak" system WS2S is obtained when the set quantifiers range over finite subsets of $\{0,1\}^*$ only. If $T = T(\varphi)$ for some WS2S-formula φ (i.e., some S2S-formula using this "weak interpretation"), T is *definable in* WS2S (or simply: "weakly definable").

The above definitions are analogously applied to finite tree models $\underline{t}$ (where $t \in T_A$). In this case there is no difference between the weak and the strong interpretation.

Note that S1S (as introduced in Section 3) results from S2S by deleting the successor function succ_1 and by restriction of the underlying models to the domain 0^*. Similarly to the case of S1S, the primitive ε and $<$ are definable in terms of succ_0, succ_1 and hence could be cancelled; we use them for easier formalizations. By "the infinite binary tree" (as a model-theoretic structure) we shall mean the structure $(\{0,1\}^*, \mathrm{succ}_0, \mathrm{succ}_1)$.

We list some examples of S2S-formulas (using freely abbreviations such as $x \leqslant y$, $X \subseteq Y$, etc.):

$$\mathrm{Chain}(X): \quad \forall x \forall y (x \in X \wedge y \in X \rightarrow x < y \vee x = y \vee y < x),$$

$\mathrm{Path}(X)$: $\mathrm{Chain}(X) \wedge \neg\exists Y(X \subseteq Y \wedge X \neq Y \wedge \mathrm{Chain}(Y))$,

$x \preccurlyeq y$: $x \leqslant y \vee \exists z\,(z0 \leqslant x \wedge z1 \leqslant y)$
(this is the total lexicographical ordering of $\{0,1\}^*$),

$\mathrm{Fin}(X)$: $\forall Y(Y \subseteq X \wedge Y \neq \emptyset \rightarrow (\exists y$ "y is $\preccurlyeq$-minimal in Y"
$\wedge\, \exists y$ "y is $\preccurlyeq$-maximal in Y"))
(this shows definability of finiteness in S2S; hence WS2S can be interpreted in S2S).

As an example tree language definable in S2S consider the set T_0 of Section 8; it is defined as follows (identifying letters a, b with $1, 0$):

$\varphi(X_1)$: $\exists Y(\mathrm{Path}(Y) \wedge \forall x\,(x \in Y \rightarrow \exists y\,(y \in Y \wedge x < y \wedge y \in X_1)))$.

We now can state the analog of Büchi's Theorem 3.1 for sets of trees.

11.1. THEOREM. (a) (Doner [26], Thatcher and Wright [117]) *A set $T \subseteq T_A$ of finite trees is definable in* (W)S2S iff *T is recognizable.*

(b) (Rabin [93]) *A set $T \subseteq T_A^\omega$ is definable in* S2S *iff T is Rabin recognizable.*

PROOF. Assume $A = \{0,1\}^n$. For the implications from right to left, formalize the acceptance condition for the given tree automaton $\mathscr{A}$: suppose, for (b), that the Rabin tree automaton $\mathscr{A}$ has the states $0, \ldots, m$ and the accepting pairs $(L_1, U_1), \ldots, (L_r, U_r)$. $T(\mathscr{A})$ is defined by an S2S-formula which says (see proof of Theorem 3.1)

$$\exists Y_0 \ldots \exists Y_m \Big(\text{“}Y_0, \ldots, Y_m \text{ represent a run of } \mathscr{A} \text{ on } X_1, \ldots, X_n\text{”}$$
$$\wedge \forall Z \Big(\mathrm{Path}(Z) \rightarrow \bigvee_{1 \leqslant i \leqslant r} \Big(\bigwedge_{j \in L_i} \text{“}\exists^{<\omega} x\,(x \in Z \wedge x \in Y_j)\text{”}$$
$$\wedge \bigvee_{j \in U_i} \text{“}\exists^{\omega} x (x \in Z \wedge x \in Y_j)\text{”} \Big) \Big) \Big).$$

The converse is also shown similarly to Theorem 3.1: First S2S is reduced to a pure second-order formalism $\mathrm{S2S}_0$ with atomic formulas of the form $\mathrm{Succ}_0(X_i, X_j)$, $\mathrm{Succ}_1(X_i, X_j)$, $X_i \subseteq X_j$ only. Induction over $\mathrm{S2S}_0$-formulas $\varphi(X_1, \ldots, X_n)$ shows recognizability, respectively Rabin recognizability of $T(\varphi)$. The steps for $\vee$ and $\exists$ are easy since the (nondeterministic) automata are closed under union and projection. (For part (a) apply Theorem 8.1(c); the proof for Rabin tree automata is similar.) Concerning negation, use again Theorem 8.1(c), respectively the complementation result in Corollary 10.3. □

Büchi tree automata correspond to a proper fragment of S2S which still allows us to express many interesting tree properties; they are also used for a beautiful characterization of WS2S. We state the result here without proof.

11.2. THEOREM (Rabin [94]). *Let* $A=\{0,1\}^*$.

(a) *A set* $T\subseteq T_A^\omega$ *is Büchi recognizable iff* T *is definable by an* S2S*-formula* $\exists Y_1 \ldots \exists Y_m \varphi(Y_1,\ldots,Y_m,X_1,\ldots,X_n)$ *where* φ *is a* WS2S*-formula.*

(b) *A set* $T\subseteq T_A^\omega$ *is definable in* WS2S *iff* T *and* $T_A^\omega - T$ *are Büchi recognizable.*

A direct automata-theoretic characterization of WS2S (in terms of alternating automata over trees) is given by Muller, Saoudi and Schupp [72].

In Theorem 4.6 it was shown that S1S and WS1S are expressively equivalent. From Theorem 11.2(a) and the failure of complementation for Büchi tree automata (cf. Theorem 8.2) it follows that WS2S is strictly less expressive than S2S and even than Büchi tree automata. However, this applies only to formulas $\varphi(X_1,\ldots,X_n)$ which speak about arbitrary sets: Läuchli and Savioz [54] show that any S2S-formula $\varphi(x_1,\ldots,x_n)$, where only individual variables occur free, can be expressed as a WS2S-formula, and Thomas [144] proves an analog for formulas $\varphi(X_1,\ldots,X_n)$ where $X_1,\ldots,X_n$ stand for paths.

An application of the above equivalence theorems is the analysis of variants of Büchi, respectively Rabin tree automata. Let us mention one such automaton model, the subtree automaton of Vardi and Wolper [129].

A *subtree automaton* over A is of the form $\mathcal{A}=(Q,\Delta,f,F)$ where Q,Δ,F are as for Büchi tree automata and $f: A\to Q$. A tree $t\in T_A^\omega$ is accepted by $\mathcal{A}$ if all its nodes x are roots of finite trees accepted by the tree automaton $\mathcal{A}'=(Q,f(x),\Delta,F)$. Hence a set that is recognized by the subtree automaton $\mathcal{A}$ is definable by a formula $\psi=\forall x\,\exists X\,\varphi(x,X)$ where $\varphi(x,X)$ expresses that the finite tree with root x and (finite) frontier X is recognized by $\mathcal{A}'$. Since ψ is a WS2S-formula, subtree automata recognize only WS2S-definable sets and hence are a proper specialization of Büchi tree automata. Subtree automata are tailored for obtaining good upper complexity bounds for program logics. Vardi and Wolper [129] show that for several progam logics the satisfiability problem amounts to a test on existence of certain trees ("Hintikka-trees") which are defined in terms of eventuality properties as defined by ψ above.

We turn to applications of Theorem 11.1 in decision problems. The starting point is the following fundamental result.

11.3. RABIN'S TREE THEOREM (cf. [93]). *The monadic second-order theory of the infinite binary tree is decidable.*

PROOF. For any S2S-sentence φ one can construct, by the proof of Theorem 11.1, an input-free Rabin tree automaton $\mathcal{A}_\varphi$ such that φ is true in the infinite binary tree iff $\mathcal{A}_\varphi$ has a successful run. The latter condition is decided effectively by Theorem 9.3. □

The result is easily generalized to the monadic second-order theory of the full n-ary tree, where n successor functions $\mathrm{succ}_0,\ldots\,\mathrm{succ}_{n-1}$ are allowed in the formulas. Similarly, the monadic second-order theory SωS of the countably branching tree is proved decidable; here one usually refers to the signature with $<$ (prefix relation over ω^*) and $\preccurlyeq$ (lexicographic order over ω^*), because each of the infinitely many successor functions succ_i is definable in terms of $\leqslant$ and $\preccurlyeq$.

Many theories of mathematical logic have been shown to be decidable via Rabin's

Tree Theorem; for some examples see [96]. In the following, we outline a standard application in dynamic logic: the solution of the *satisfiability problem for modal logics of programs*. The method (and refinements of it) have been used for several logics, for example propositional dynamic logic and extensions [112], process logic [47], the calculus L_μ [113], and computation tree logic CTL* [31, 33]. In all instances the satisfiability question: "is there a model $\mathcal{M}$ satisfying the formula φ?" is effectively transformed to a question "is there a tree t satisfying the S2S-sentence φ'?". (Logics allowing this reduction are said to share the *tree model property*.) We present the conceptually simplest form of this translation for the example CTL* of computation tree logic. Further developments of the method (with much better complexity bounds) are surveyed by Emerson [30]. Muller, Saoudi and Schupp [73] present a uniform method to obtain exponential time bounds for logics that involve only quantifications over finite computation paths, based on the alternating automata mentioned at the end of Section 10.

Computation tree logic CTL* is a system of modal logic which allows to specify properties of paths through Kripke structures. From atomic propositions, say $p_1, \ldots, p_n$ for the following discussion, the CTL*-formulas are built up using Boolean connectives, the linear time temporal operators, X, F, G, U (cf. Section 6) and the additional unary operator E. Recall (from Section 6) that a Kripke structure is of the form $\mathcal{M} = (S, R, \Phi)$ where S is a (here at most countable) set of states, $R \subseteq S \times S$ the transition relation, and $\Phi: S \to 2^{\{p_1, \ldots, p_n\}}$ a truth valuation. For simplicity we assume that there is a distinguished start state s_0. The semantics of CTL*-formulas in Kripke structures is based on the usual meaning of X, F, G, U over given state paths and the interpretation of E by "there is an infinite state path". Consider an example: the CTL*-formula

$$p_1 \wedge \mathrm{E}(\mathrm{G}p_2 \wedge \mathrm{FE}\mathrm{F}p_3)$$

says that

> "p_1 is true in s_0, and starting from s_0 there is an infinite path π through the model such that all states on π satisfy p_2, and in some state of π a path π' starts with some state satisfying p_3".

The decision procedure for satisfiability of CTL*-formulas is based on the *unravelling of Kripke structures* in tree form: given the Kripke structure $\mathcal{M} = (S, R, \Phi)$, define the structure $\mathcal{M}' = (S', R', \Phi')$ by

$$S' = s_0 S^*,$$

$$(r_1 \ldots r_m) R' (s_1 \ldots s_k s) \text{ iff } k = m,\ r_i = s_i \text{ for } 1 \leqslant i \leqslant k,\ s_k R s,$$

$$\Phi'(s_1 \ldots s_k) = \Phi(s_k).$$

$\mathcal{M}'$ can be considered as an (at most countably branching) $\{0,1\}^n$-valued tree in which the nodes are finite histories of states from S, the relation "is-father-of" is represented by R, and the valuation by Φ. $\mathcal{M}'$ is encoded over the binary tree by the map $n_1 \ldots n_r \mapsto 10^{n_1} \ldots 10^{n_r}$. We obtain a $\{0,1\}^{n+1}$-valued binary tree $t_{\mathcal{M}}$ where the additional component of the valuation indicates the range of $\mathcal{M}$ under this map. Let $\mu(X, Y_1, \ldots, Y_n)$ be an S2S-formula which says that $X \subseteq (10^*)^*$ is a range of a tree under the coding and that $Y_1, \ldots, Y_n \subseteq X$. Any tree of form $t_{\mathcal{M}}$ satisfies μ; conversely, any tree satisfying μ induces a Kripke model (over the set $(10^*)^*$).

It is straightforward to reformulate a given CTL*-formula φ as an S2S-formula

$\varphi'(X, Y_1, \ldots, Y_n)$ such that it is true over a tree $t_{\mathcal{M}}$ iff φ holds in $\mathcal{M}$. Hence φ is satisfiable iff the S2S-sentence

$$\exists X \exists Y_1 \ldots \exists Y_n (\mu(X, Y_1, \ldots, Y_n) \wedge \varphi'(X, Y_1, \ldots, Y_n))$$

is true over the infinite binary tree. So by Rabin's Tree Theorem, satisfiability of CTL*-formulas is decidable.

The decision procedure induced by the above transformation is nonelementary. (Each level of negation in the given formula requires a corresponding complementation of a Rabin automaton and hence an at least exponential blow-up in the size of the automata.) Better procedures are obtained by incorporating more information than just for the atomic formulas in the tree model $t_{\mathcal{M}}$, e.g. by including the "Fischer–Ladner closure" of the given formula. For more details see [30, 32].

We end this section with the formulation of two interesting generalizations of Rabin's Tree Theorem and some remarks on undecidable extensions of the monadic theory of the binary tree.

Stupp [114], continuing work of Shelah [103], extended Rabin's techniques (in particular concerning the complementation theorem for Rabin tree automata) to "higher-dimensional trees" and similar structures.

11.4. THEOREM (Shelah [103], Stupp [114]). *Let $\mathcal{M} = (M, (R_i)_{i<k})$ be a relational structure (say with binary relations $R_i \subseteq M \times M$). Define the structure $\mathcal{M}^* = (M^*, <, (R_i^*)_{i<k})$ by*

$u < v$ *iff* u *is a proper prefix of* v *(over* M^**)*

$u R_i^* v$ *iff* $\exists m_1, \ldots, m_k, m, m' \in M$ *such that*

$$u = m_1 \ldots m_k m, v = m_1 \ldots m_k m', m R_i m'.$$

If the monadic second-order theory of $\mathcal{M}$ is decidable, so is the monadic second-order theory of $\mathcal{M}^*$.

The special case of Theorem 11.4 which yields Rabin's Tree Theorem concerns the finite structure $\mathcal{M}_0 = (\{0, 1\}, <_0)$ where $<_0$ is the usual order on $\{0, 1\}$ (and which of course has a decidable monadic second-order theory). We have $\mathcal{M}_0^* = (\{0, 1\}^*, <_1, <_0)$ where

$u <_1 v$ iff u is a proper prefix of v,

$u <_0 v$ iff $u = w0, v = w1$ for some $w \in \{0, 1\}^*$.

The functions succ_0, succ_1 are (monadic second-order) definable in terms of $<_1, <_0$ and vice versa. So $\mathcal{M}_0^*$ is essentially the infinite binary tree.

Another extension of Rabin's Tree Theorem is due to Muller and Schupp [74]. They consider infinite directed graphs with labelled edges. Such a graph is said to be "finitely generated" if it has a distinguished vertex v_0 ("origin"), a finite label alphabet A and a fixed finite bound on the degrees of the vertices. The model-theoretic structures which represent these graphs are of the form $\underline{G} = (G, v_0, (R_a)_{a \in A})$ such that $u R_a v$ iff there is an edge labelled a from u to v. A finitely generated graph is called a *context-free graph* if

it is "finitely behaved at infinity" in the following sense: one obtains only finitely many distinct isomorphism types by collecting, for all vertices v, the substructures $\Gamma(v)$ which remain as connected components when the points are deleted with distance to v_0 smaller than between v and v_0. For more information on these graphs and the background from group theory, see the survey of Berstel and Boasson [6] in this Handbook.

11.5. THEOREM (Muller, Schupp [74]). *The monadic second-order theory of any context-free graph is decidable.*

The proof of Theorem 11.5 is based on Rabin's Tree Theorem. The general problem of reducing monadic theories of graphs to theories of trees is further investigated by Seese [101]; he considers the conjecture that *any* decidable monadic theory of a class of graphs has an interpretation in the monadic theory of a class of trees, and shows that this is true for the class of planar graphs. Other results in this direction are given in [21], where "equational graphs" (defined by certain graph rewriting systems) are shown to have a decidable monadic second-order theory.

The theories in Theorems 11.4 and 11.5 seem close to the margin of undecidability. We mention some variants, respectively extensions of the monadic theory of the binary tree which are undecidable. (These results have been shown by several authors; a recent reference is [54]).

The most basic example is the monadic theory of the "grid" $(\omega \times \omega, s_0, s_1)$ where $s_0(m,n)=(m+1,n)$ and $s_1(m,n)=(m,n+1)$. This structure is obtained from the free algebra $(\{0,1\}^*, \mathrm{succ}_0, \mathrm{succ}_1)$ by adding the relation $\mathrm{succ}_0 \circ \mathrm{succ}_1 = \mathrm{succ}_1 \circ \mathrm{succ}_0$.

11.6. THEOREM (a) *The (weak) monadic second-order theory of the grid $(\omega \times \omega, s_0, s_1)$ is undecidable.*

(b) *The (weak) monadic second-order theory of the infinite binary tree extended by the function s with $s(w)=0w$ (for $w \in \{0,1\}^*$) is undecidable.*

(c) *The (weak) monadic second-order theory of the infinite binary tree extended by the "equal-level predicate" E, given by $u\,E\,v$ iff $|u|=|v|$ (for $u,v \in \{0,1\}^*$), is undecidable.*

PROOF (a) The idea is similar as in the undecidability proof for the origin constrained domino problem. For any Turing machine $\mathcal{A}$, construct a sentence $\varphi_{\mathcal{A}}$ in the weak monadic second-order language of the grid which expresses existence of a halting computation of $\mathcal{A}$ when $\mathcal{A}$ is started on the empty tape (the tape is assumed here left-bounded and right-infinite). As in the domino problem, the ith cell of the jth configuration is represented by the point $(i,j) \in \omega \times \omega$. Using existential quantification over auxiliary unary predicates (which code the letters and states of $\mathcal{A}$), it is easy to formalize that a halting configuration is reached. Since only finitely many steps and a finite portion of the tape are involved, weak second-order quantification suffices.

(b) Identify the (weakly definable) subset 0^*1^* of the binary tree with $\omega \times \omega$. Note that $(\omega \times \omega, s_0, s_1)$ is isomorphic to $(0^*1^*, s, \mathrm{succ}_1)$; so (a) can be applied.

(c) Using the predicate E, the function s is weakly definable on 0^*1^* since we have

(for $u, v \in 0^*1^*$)

$$s(u)=v \text{ iff } (u \in 0^* \wedge v=u0) \vee \exists w \in 0^* \exists u' \quad (u \in w1^* \wedge u' \in w01^* \wedge u E u' \wedge u'1=v).$$

Now the claim follows from (b). □

By decidability of S2S one obtains as a corollary that the function s and the relation E over the binary tree are not definable in S2S. In contrast to part (c) of Theorem 11.6, a decidable theory is obtained when the relation E is adjoined to a restricted version of S2S where the second-order variables range over paths only (cf. [144]).

12. Classifications of Rabin recognizable sets

In this final section we give a short overview of the (mostly ongoing) work which studies the "fine structure" of the class of Rabin recognizable sets of trees. The results presented here fall in three categories, depending on the formalism in which tree properties are classified: monadic second-order logic, tree automata, and fixed point calculi.

12.1 Restrictions of monadic second-order logic

Natural subsystems of the monadic second-order formalism S2S are obtained when the range of set quantification is narrowed to special subsets of $\{0, 1\}^*$.

First we consider the question for which restricted set quantifiers the same sentences are true (over the infinite binary tree) as in the case of quantification over arbitrary subsets. Rabin [95] proved (as a corollary to his result given in Theorem 9.3(a)) that the regular subsets of $\{0, 1\}^*$ constitute such a restriction. Siefkes [105] showed that this fails for the recursive subsets of $\{0, 1\}^*$. A related question is the uniformization problem: is there, for any given formula $\varphi(X, Y)$ such that $\forall X \exists Y \varphi(X, Y)$ holds, a "definable choice function", i.e. a (set) function $X \mapsto Y$ defined by a formula $\psi(X, Y)$ such that $\forall X \forall Y(\psi(X, Y) \rightarrow \varphi(X, Y))$? Siefkes [105] proves this for S1S, and Gurevich and Shelah [43] showed that it fails for S2S using a very intricate proof method.

We discuss two further restricted set quantifiers: those ranging over finite sets ("weak quantifiers"), and those ranging over paths through the infinite binary tree.

The weakly definable sets of trees, already considered in the preceding section, have been further classified in [121] (see also [68]): An infinite hierarchy is induced by the alternation depth of "unbounded" quantifiers over finite sets and elements. ("Bounded" set quantifiers are of the form $\exists X \leqslant Y \varphi(X, Y)$, where $X \leqslant Y$ is an abbreviation for $\forall x(x \in X \rightarrow \exists y(y \in Y \wedge x \leqslant y))$.) Over ω-words the analogous hierarchy is finite, because, by McNaughton's Theorem, two unbounded quantifiers suffice for the definition of regular ω-languages (cf. Theorem 4.6).

When set quantifiers refer to chains in trees (i.e., sets linearly ordered by the prefix relation $<$) or to paths (i.e. maximal chains), one obtains *chain logic*, respectively *path logic* over the binary tree. In [123], these systems are characterized in terms of the regular and star-free ω-languages. A close connection between path quantifiers and

systems of branching time logic is set up in [46], where computation tree logic CTL* and path logic are shown expressively equivalent, provided that binary tree models in the signature $<$ are considered.

A general decidability result on path quantifiers over trees was shown by Gurevich and Shelah [44]. They prove that in the language of path logic the theory of arbitrary trees (considered as any partial order where each set $\{y \mid y \leqslant x\}$ is totally ordered) is decidable.

12.2. *Restrictions in Rabin tree automata*

Several authors investigated the possibility of extending Landweber's Theorem 5.3 to Rabin recognizable sets of trees. The notions of 1- and 2-acceptance can be transferred from sequential automata to tree automata (namely, as conditions for all paths of a run). Also the Cantor topology is extended canonically from A^ω to the space T_A^ω (the sets $t.T_A^\omega$, where t is a finite tree, form an open basis). Results on inclusion relations for these acceptance conditions and their topological meaning are presented in [48, 65]. Skurczyński showed in [143] that the Borel hierarchy of Rabin recognizable tree sets is infinite. Mostowski, Skurczyński and Wagner [69] obtain a partial transfer of Theorem 5.3 (including decidability results) to tree languages recognized by deterministic Rabin tree automata. Further results on deterministic tree automata and the power of several acceptance conditions are given in [100].

Niwinski [78] showed that the Rabin index (the number of accepting pairs in Rabin tree automata) defines an infinite hierarchy of sets of trees: for each n there is a set $T_n \subseteq T_A^\omega$ which is recognized only by Rabin tree automata with at least n accepting pairs. The presented example languages T_n belong to the Boolean closure of the Büchi recognizable sets of trees; Hafer [45] proves that this Boolean closure is still properly contained in the class of Rabin recognizable sets. Mostowski [67] presents a "standard form" of Rabin tree automata in which an ordering of the state set enters the acceptance condition. As an application, a calculus of regular-like expressions is set up which allows to define the Rabin recognizable sets.

12.3. *Fixed point calculi*

Niwinski [78] and Takahashi [115] studied the specification of tree properties by a fixed point calculus in which least and greatest fixed points are included. Following [78], we define the *μ-terms* over an alphabet A and with variables $x_1, x_2, \ldots$ by the clauses

- each variable x_i is a μ-term,
- if τ_1, τ_2 are μ-terms and $a \in A$, then $a(\tau_1, \tau_2)$ and $\tau_1 \cup \tau_2$ are μ-terms,
- if τ is a μ-term and x is a variable, then $\mu x \tau$ and $\nu x \tau$ are μ-terms.

Any μ-term $\tau(x_1, \ldots, x_n)$ with free variables $x_1, \ldots, x_n$ defines a function $F_\tau : (T_A^\omega)^n \to T_A^\omega$. For $\tau = x_i$ it is the ith projection. If $\tau(x_1, \ldots, x_n)$ has the form $\tau = \tau_1 \cup \tau_2$, let

$$F_\tau(T_1, \ldots, T_n) = F_{\tau_1}(T_1, \ldots, T_n) \cup F_{\tau_2}(T_1, \ldots, T_n);$$

similarly for $\tau = a(\tau_1, \tau_2)$,

$$F_\tau(T_1, \ldots, T_n) = \{t \in T_A^\omega \mid t(\varepsilon) = a,\ t_0 \in F_{\tau_1}(T_1, \ldots, T_n),\ t_1 \in F_{\tau_2}(T_1, \ldots, T_n)\}.$$

Finally, for $\tau = \mu y \tau_0(y, x_1, \ldots, x_n)$, respectively $\tau = \nu y \tau_0(y, x_1, \ldots, x_n)$, let $F_\tau(T_1, \ldots, T_n)$ be the least, resp. greatest fixed point of the function $T \mapsto F_{\tau_0}(T, T_1, \ldots, T_n)$. (These fixed points exist by the Knaster–Tarski Theorem.)

Each μ-term τ without free variables defines a set $T \subseteq T_A^\omega$, denoted here by $T(\tau)$. As an example over the alphabet $A = \{a, b\}$, consider the μ-term

$$\mu x_1 \nu x_0 (b(x_0, x_0) \cup a(x_1, x_1));$$

it defines the set T_1 of Section 8, containing all trees such that on each path there are only finitely many letters a.

By induction over the μ-terms τ one verifies that the functions F_τ are definable in S2S. It follows that the sets $T(\tau)$ are Rabin recognizable. Niwinski [79] proves also the converse; so the μ-terms have the same expressive power as Rabin tree automata (or S2S). A strict hierarchy of sets of trees is generated by increasing the number of alternations between the least and greatest fixed point operators μ and ν in the defining terms. Moreover, the second level of the hierarchy (given by the terms where all ν-operators precede all μ-operators) characterizes the Büchi recognizable sets of trees (cf. [78, 115, 3]).

Greatest fixed points consisting of finite and infinite trees also arise naturally in the theory of nondeterministic *recursive program schemes* (cf. [4] or [22, Section 8.3]. These schemes and the associated fixed point operators can be considered as tree replacement systems (tree grammars). Since in their derivations a context-free substitution mechanism is involved (which is not present in the μ-terms above), the tree languages generated by recursive program schemes cannot be described by finite-state tree automata. An extended model of "pushdown tree automaton" that is appropriate for this purpose has been introduced in [39, 142].

Acknowledgment

I would like to thank all who contributed comments and corrections to a draft version of this paper; in particular, A. Arnold, H.D. Ebbinghaus, J. Flum, Y. Gurevich, A.W. Mostowski, D. Niwinski, D. Perrin, J.E. Pin, D. Seese, D. Siefkes, L. Staiger and P. Wolper, as well as students and colleagues from the RWTH Aachen (where this paper was written). I also thank Carla Meckler for her endless work in typing and arranging the manuscript.

References

[1] ALPERN, B. and F.B. SCHNEIDER, Recognizing safety and liveness, *Distributed Comput.* 2 (1987) 117–126.

[2] ARNOLD, A., A syntactic congruence for rational ω-languages, *Theoret. Comput. Sci.* 39 (1985) 333–335.

[3] ARNOLD, A., Logical definability of fixed points, *Theoret. Comput. Sci.* 61 (1988) 289–297.

[4] ARNOLD, A. and M. NIVAT, Formal computations of nondeterministic recursive program schemes, *Math. Systems Theory* 13 (1980) 219–236.

[5] BERSTEL, J., *Transductions and Context-Free Languages* (Teubner, Stuttgart, 1979).
[6] BERSTEL, J. and L. BOASSON, Context-free languages, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (North-Holland, Amsterdam, 1990).
[7] BOASSON, L. and M. NIVAT, Adherences of languages, *J. Comput. System Sci.* 20 (1980) 285–309.
[8] BÜCHI, J.R., Weak second-order arithmetic and finite automata, *Z. Math. Logik Grundlag. Math.* 6 (1960) 66–92.
[9] BÜCHI, J.R., On a decision method in restricted second order arithmetic, in: E. Nagel et al., eds., *Proc. Internat. Congr. on Logic, Methodology and Philosophy of Science* (Stanford Univ. Press, Stanford, CA, 1960) 1–11.
[10] BÜCHI, J.R., The monadic theory of ω_1, in: *Decidable Theories II*, Lecture Notes in Mathematics, Vol. 328 (Springer, Berlin, 1973) 1–127.
[11] BÜCHI, J.R., Using determinacy to eliminate quantifiers, in: M. Karpinski, ed., *Fundamentals of Computation Theory*, Lecture Notes in Computer Science, Vol. 56 (Springer, Berlin, 1977) 367–378.
[12] BÜCHI, J.R., State-strategies for games in $F_{\sigma\delta}\cap G_{\delta\sigma}$, *J. Symbolic Logic* **48** (1983) 1171–1198.
[13] BÜCHI, J.R. and L.H. LANDWEBER, Solving sequential conditions by finite-state strategies, *Trans. Amer. Math. Soc.* **138** (1969) 295–311.
[14] BÜCHI, J.R. and D. SIEFKES, Axiomatization of the monadic second order theory of ω_1, in: *Decidable Theories II*, Lecture Notes in Mathematics, Vol. 328 (Springer, Berlin, 1973) 129–217.
[15] CHOUEKA, Y., Theories of automata on ω-tapes: a simplified approach, *J. Comput. System Sci.* **8** (1974) 117–141.
[16] CHURCH, A., Logic, arithmetic and automata, in: *Proc. Internat. Congress Math.* (1963) 23–35.
[17] COHEN, R.S. and A.Y. GOLD, Theory of ω-languages, *J. Comput. System Sci.* **15** (1977) 169–184 and 185–203.
[18] COHEN, R.S. and A.Y. GOLD, ω-Computations of deterministic pushdown machines, *J. Comput. System Sci.* **16** (1978) 275–300.
[19] COHEN, R.S. and A.Y. GOLD, ω-Computations on Turing machines, *Theoret. Comput. Sci.* **6** (1978) 1–23.
[20] COURCELLE, B., Fundamental properties of infinite trees, *Theoret. Comput. Sci.* **25** (1983) 95–169.
[21] COURCELLE, B., The monadic second-order logic of graphs II: infinite graphs of bounded width, *Math. Systems Theory* **21** (1989) 187–221.
[22] COURCELLE, B., Recursive applicative program schemes, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (North-Holland, Amsterdam, 1990).
[23] COURCOUBETIS, C. and M. YANNAKAKIS, Verifying properties of finite-state probabilistic programs, in: *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science* (1988) 338–345.
[24] DAUCHET, M. and E. TIMMERMAN, Continuous monoids and yields of infinite trees, *RAIRO Inform. Théor. Appl.* **20** (1986) 251–274.
[25] DAVIS, M., Infinite games of perfect information, in: *Advances in Game Theory* (Princeton Univ. Press, Princeton, NJ, 1964) 85–101.
[26] DONER, J., Tree acceptors and some of their applications, *J. Comput. System Sci.* **4** (1970) 406–451.
[27] EILENBERG, S., *Automata, Languages and Machines, Vol. A* (Academic Press, New York, 1974).
[28] ELGOT, C.C., Decision problems of finite automata design and related arithmetics, *Trans. Amer. Math. Soc.* **98** (1961) 21–52.
[29] ELGOT, C.C. and M.O. RABIN, Decidability and undecidability of second (first) order theory of (generalized) successor, *J. Symbolic Logic* **31** (1966) 169–181.
[30] EMERSON, E.A., Temporal and modal logic, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (North-Holland, Amsterdam, 1990).
[31] EMERSON, E.A. and J.Y. HALPERN, "Sometimes" and "Not Never" revisited: On branching time versus linear time, *J. Assoc. Comput. Mach.* **33** (1986) 151–178.
[32] EMERSON, E.A. and C.S. JUTLA, The complexity of tree automata and logics of programs, in: *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science* (1988) 328–337.
[33] EMERSON, E.A. and A.P. SISTLA, Deciding full branching time logic, *Inform. and Control* **61** (1984) 175–201.
[34] FRANCEZ, N., *Fairness* (Springer, Berlin, 1987).

[35] GABBAY, A., A. PNUELI, S. SHELAH, and J. STAVI, On the temporal analysis of fairness, in: *Proc. 7th Ann. ACM Symp. on Principles of Programming Languages* (1980) 163–173.
[36] GALE, D. and F.M. STEWART, Infinite games with perfect information, in: *Contributions to the Theory of Games* (Princeton Univ. Press, Princeton, NJ, 1953) 245–266.
[37] GÉCSEG, F. and M. STEINBY, *Tree Automata* (Akadémiai Kiadó, Budapest, 1984).
[38] GIRE, F. and M. NIVAT, Relations rationelles infinitaires, *Calcolo* **21** (1984) 91–125.
[39] GUESSARIAN, I., Pushdown tree automata, *Math. Systems Theory* **16** (1983) 237–264.
[40] GUESSARIAN, I. and W. NIAR-DINEDANE, Fairness and regularity for SCCS processes, *Inform. Théor. Appl.* **23** (1989) 59–86.
[41] GUREVICH, Y., Monadic second-order theories in: J. Barwise and S. Feferman, eds., *Model-theoretic Logics* (Springer, Berlin, 1985), 479–506.
[42] GUREVICH, Y. and L.A. HARRINGTON, Automata, trees, and games, in: *Proc. 14th Ann. ACM Symp. on the Theory of Computing* (1982) 60–65.
[43] GUREVICH, Y. and S. SHELAH, Rabin's uniformization problem, *J. Symbolic Logic* **48** (1983) 1105–1119.
[44] GUREVICH, Y. and S. SHELAH, The decision problem for branching time logic, *J. Symbolic Logic* **50** (1985) 668–681.
[45] HAFER, T., On the boolean closure of Büchi tree automaton definable sets of ω-trees, Aachener Inform. Ber. Nr. 87–16, RWTH Aachen, 1987.
[46] HAFER, T. and W. THOMAS, Computation tree logic CTL* and path quantifiers in the monadic theory of the binary tree, in: T. Ottmann, ed., *Proc. 14th Internat. Coll. on Automata, Languages and Programming*, Lecture Notes in Computer Science, Vol. 267 (Springer, Berlin, 1987) 269–279.
[47] HAREL, D., D. KOZEN and R. PARIKH, Process logic: expressiveness, decidability, completeness, *J. Comput. System Sci.* **25** (1982) 144–170.
[48] HAYASHI, T. and S. MIYANO, Finite tree automata on infinite trees, *Bull. Inform. Cybernet.* **21** (1985) 71–82.
[49] HOOGEBOOM, H.J. and G. ROZENBERG, Infinitary languages: basic theory and applications to concurrent systems, in: J.W. de Bakker et al., eds., *Current Trends in Concurrency*, Lecture Notes in Computer Science, Vol. 224 (Springer, Berlin, 1986) 266–342.
[50] KAMINSKI, M., A classification of ω-regular languages, *Theoret. Comput. Sci.* **36** (1985) 217–239.
[51] KAMP, H.W., Tense logic and the theory of linear order, Ph.D. Thesis, Univ. of California, Los Angeles, CA, 1968.
[52] KOZEN, D. and J. TIURYN, Logics of programs, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (North-Holland, Amsterdam, 1990) 789–840.
[53] LADNER, R.E., Application of model-theoretic games to discrete linear orders and finite automata, *Inform. and Control* **33** (1977) 281–303.
[54] LÄUCHLI, H. and C. SAVOIZ, C. Monadic second order definable relations on the binary tree, *J. Symbolic Logic* **52** (1987) 219–226.
[55] LANDWEBER, L.H., Decision problems for ω-automata, *Math. Systems Theory* **3** (1969) 376–384.
[56] LICHTENSTEIN, O. and A. PNUELI, Checking that finite-state concurrent programs satisfy their specification, in: *Proc. 12th Ann. ACM Symp. on Principles of Programming Languages* (1985) 97–107.
[57] LICHTENSTEIN, O., A. PNUELI and L. ZUCK, The glory of the past, in: R. Parikh, ed., *Logics of Programs*, Lecture Notes in Computer Science, Vol. 193 (Springer, Berlin, 1985) 196–218.
[58] LINNA, M., On ω-sets associated with context-free languages, *Inform. and Control* **31** (1976) 272–293.
[59] MANNA, Z., and A. PNUELI, Specification and verification of concurrent programs by $\forall$-automata, in: *Proc. 14th Ann. ACM Symp. on Principles of Programming Languages* (1987) 1–12.
[60] MANNA, Z. and A. PNUELI, The anchored version of the temporal framework, in: J.W. de Bakker et al., eds. *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, Lecture Notes in Computer Science, Vol. 345 (Springer, Berlin, 1989) 201–284.
[61] MARTIN, D.A., Borel determinacy, *Ann. Math.* **102** (1975) 363–371.
[62] MCNAUGHTON, R., Testing and generating infinite sequences by a finite automaton, *Inform. and Control* **9** (1966) 521–530.
[63] MCNAUGHTON, R. and S. PAPERT, *Counter-Free Automata* (MIT Press, Cambridge, MA, 1971).
[64] MIYANO, S. and T. HAYASHI, Alternating automata on ω-words, *Theoret. Comput. Sci.* **32** (1984) 321–330.

[65] MORIYA, T., Topological characterizations of infinite tree languages, *Theoret. Comput. Sci.* 52 (1987) 165–171.
[66] MOSCHOVAKIS, Y.N., *Descriptive Set Theory* (North-Holland, Amsterdam, 1980).
[67] MOSTOWSKI, A.W., Regular expressions for infinite trees and a standard form of automata, in: A. Skowron, ed., *Computation Theory*, Lecture Notes in Computer Science, Vol. 208 (Springer, Berlin, 1984) 157–168.
[68] MOSTOWSKI, A.W., Hierarchies of weak monadic formulas for two successors arithmetic, *J. Inform. Process. Cybernet.* 23 (1987) 509–515.
[69] MOSTOWSKI, A.W., J. SKURCZYŃSKI, and K. WAGNER, Deterministic automata on infinite trees and the Borel hierarchy, in: M. Arato, I. Kátai and L. Varga, eds., *Proc. 4th Hungarian Conf. on Computer Science* (1985) 103–115.
[70] MUCHNIK, A.A.,Games on infinite trees and automata with dead-ends: a new proof of the decidability of the monadic theory of two successors, *Semiotics and Information* 24 (1984) 17–40 (in Russian).
[71] MULLER, D.E., Infinite sequences and finite machines, in: *Proc. 4th Ann. IEEE Symp. on Switching Circuit Theory and Logical Design* (1963) 3–16.
[72] MULLER, D.E., A. SAOUDI and P.E. SCHUPP, Alternating automata, the weak monadic theory of the tree, and its complexity, in: L. Kott, ed., *Proc. 13th Internat. Coll. on Automata, Languages and Programming*, Lecture Notes in Computer Science, Vol. 226 (Springer, Berlin, 1986) 275–283.
[73] MULLER, D.E., A. SAOUDI and P.E. SCHUPP, Weak alternating automata give a simple explanation of why most temporal and dynamic logics are decidable in exponential time, in: *Proc. 3rd IEEE Ann. Symp. on Logic in Computer Science* (1988) 422–427.
[74] MULLER, D.E. and P.E. SCHUPP, The theory of ends, pushdown automata, and second-order logic, *Theoret. Comput. Sci.* 37 (1985) 51–75.
[75] MULLER, D.E. and P.E. SCHUPP, Alternating automata on infinite trees, *Theoret. Comput. Sci.* 54 (1987) 267–276.
[76] NIVAT, M. and D. PERRIN, Ensembles reconnaissables de mots biinfinis, *Canad. J. Math.* 38 (1986) 513–537.
[77] NIWINSKI, D., Fixed-point characterization of context-free ω-languages, *Inform. and Control* 63 (1984) 247–276.
[78] NIWINSKI, D., On fixed-point clones, in: L. Kott, ed., *Proc. 13th Internat. Coll. on Automata, Languages and Programming*, Lecture Notes in Computer Science Vol. 226 (Springer, Berlin, 1986) 464–473.
[79] NIWINSKI, D., Fixed points vs. infinite generation, in: *Proc. 3rd Ann. IEEE Symp. on Logic in Computer Science* (1988) 402–409.
[80] PARIGOT, M. and E. PELZ, A logical approach of Petri net languages, *Theoret. Comput. Sci.* 39 (1985) 155–169.
[81] PARK, D., Concurrency and automata on infinite sequences, in: P. Deussen, ed., *Theoretical Computer Science*, Lecture Notes in Computer Science, Vol. 104 (Springer, Berlin, 1981) 167–183.
[82] PÉCUCHET, J.P., Automates boustrophédons et mots infinis, *Theoret. Comput. Sci.* 35 (1985) 115–122.
[83] PÉCUCHET, J.P., On the complementation of Büchi automata, *Theoret. Comput. Sci.* 47 (1986) 95–98.
[84] PÉCUCHET, J.P., Etude syntaxique des parties reconnaissables de mots infinis, in: L. Kott, ed., *Proc. 13th Internat. Coll. on Automata, Languages and Programming*, Lecture Notes in Computer Science, Vol. 226 (Springer, Berlin, 1986) 294–303.
[85] PERRIN, D., Recent results on automata and infinite words, in: M.P. Chytil and V. Koubek, eds., *Mathematical Foundations of Computer Science '84*, Lecture Notes in Computer Science, Vol. 176 (Springer, Berlin, 1984) 134–148.
[86] PERRIN, D., Variétés de semigroupes et mots infinis, *C.R. Acad. Sci. Paris* 295 (1985) 595–598.
[87] PERRIN, D., Finite automata, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (North-Holland, Amsterdam, 1990) 1–57.
[88] PERRIN, D. and J.E. PIN, First order logic and star-free sets, *J. Comput. System Sci.* 32 (1986) 393–406.
[89] PERRIN, D. and P.E. SCHUPP, Automata on the integers, recurrence distinguishability, and the equivalence and decidability of monadic theories, in: *Proc. 1st IEEE Symp. on Logic in Computer Science* (1986) 301–304.
[90] PNUELI, A., Applications of temporal logic to the specification and verification of reactive systems: a survey of current trends, in: J.W. de Bakker et al., eds., *Current Trends in Concurrency*, Lecture Notes in Computer Science, Vol. 224 (Springer, Berlin, 1986) 510–584.

[91] PNUELI, A. and R. ROSNER, On the synthesis of a reactive module, in: *Proc. 16th Symp. Princ. of Prog. Lang.* (1989) 179–190.

[92] PRIESE, L., R. REHRMANN, and U. WILLECKE-KLEMME, An introduction to the regular theory of fairness, *Theoret. Comput. Sci.* **54** (1987) 139–163.

[93] RABIN, M.O., Decidability of second-order theories and automata on infinite trees, *Trans. Amer. Math. Soc.* **141** (1969) 1–35.

[94] RABIN, M.O., Weakly definable relations and special automata in: Y. Bar-Hillel, ed., *Mathematical Logic and Foundations of Set Theory* (North-Holland, Amsterdam, 1970) 1–23.

[95] RABIN, M.O., *Automata on Infinite Objects and Church's Problem* (Amer. Mathematical Soc., Providence, RI, 1972).

[96] RABIN, M.O., Decidable theories, in J. Barwise, ed., *Handbook of Mathematical Logic* (North-Holland, Amsterdam, 1977) 595–629.

[97] REDZIEJOWSKI, R.R., Infinite-word languages and continuous mappings, *Theoret. Comput. Sci.* **43** (1985) 59–79.

[98] ROSENSTEIN, J.G., *Linear Orderings* (Academic Press, New York, 1982).

[99] SAFRA, S., On the complexity of ω-automata, in: *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science* (1988) 319–327.

[100] SAOUDI, A., Variétés d'automates descendants d'arbres infinis, *Theoret. Comput. Sci.* **43** (1986) 315–335.

[101] SEESE, D., The structure of the models of decidable monadic theories of graphs, Akad. d. Wiss. der DDR, Inst. f. Math. Berlin, 1988.

[102] SEMENOV, A.L., Decidability of monadic theories, in: M.P. Chytil and V. Koubek, eds., *Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science, Vol. 176 (Springer, Berlin, 1984) 162–175.

[103] SHELAH, S., The monadic theory of order, *Ann. of Math.* **102** (1975) 379–419.

[104] SIEFKES, D., *Decidable Theories I: Büchi's Monadic Second Order Successor Arithmetic*, Lecture Notes in Mathematics, Vol. 120 (Springer, Berlin, 1970).

[105] SIEFKES, D., The recursive sets in certain monadic second order fragments of arithmetic, *Arch. Math. Logik* **17** (1975) 71–80.

[106] SISTLA, A.P. and E.M. CLARKE, The complexity of propositional linear time logics, *J. Assoc. Comput. Mach.* **32** (1985) 733–749.

[107] SISTLA, A.P., M.Y. VARDI and P. WOLPER, The complementation problem for Büchi automata with applications to temporal logic, *Theoret. Comput. Sci.* **49** (1987) 217–237.

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

[109] STAIGER, L., Hierarchies of recursive ω-languages, *J. Inform. Process. Cybernet.* **22** (1986) 219–241.

[110] STAIGER, L., Research in the theory of ω-languages, *J. Inform. Process. Cybernet.* **23** (1987) 415–439.

[111] STOCKMEYER, L.J. and A.R. MEYER, Word problems requiring exponential time: preliminary report, in: *Proc. 5th Ann. ACM Symp. on the Theory of Computing* (1973) 1–9.

[112] STREETT, R.S., Propositional dynamic logic of looping and converse, *Inform. and Control* **54** (1982) 121–141.

[113] STREETT, R.S. and E.A. EMERSON, The propositional Mu-calculus is elementary, in: J. Paredaens, ed., *Proc. 11th Internat. Coll. on Automata, Languages and Programming*, Lecture Notes in Computer Science, Vol. 172 (Springer, Berlin, 1984) 465–472.

[114] STUPP, J., The lattice model is recursive in the original model, manuscript (The Hebrew University, Jerusalem, 1975).

[115] TAKAHASHI, M., The greatest fixed-points and rational omega-tree languages, *Theoret. Comput. Sci.* **44** (1986) 259–274.

[116] TAKAHASHI, M., Brzozowski hierarchy of ω-languages, *Theoret. Comput. Sci.* **49** (1987) 1–12.

[117] THATCHER, J.W. and J.B. WRIGHT, Generalized finite automata with an application to a decision problem of second-order logic, *Math. Systems Theory* **2** (1968) 57–82.

[118] THOMAS, W., Star-free regular sets of ω-sequences, *Inform. and Control* **42** (1979) 148–156.

[119] THOMAS, W., A combinatorial approach to the theory of ω-automata, *Inform. and Control* **48** (1981) 261–283.

[120] THOMAS, W., Classifying regular events in symbolic logic, *J. Comput. System Sci.* **25** (1982) 360–376.

[121] THOMAS, W., A hierarchy of sets of infinite trees, in: A.B. Cremers and H.P. Kriegel, eds., *Theoretical Computer Science*, Lecture Notes in Computer Science, Vol. 145 (Springer, Berlin, 1982) 335–342.

[122] THOMAS, W., On frontiers of regular trees, *RAIRO Inform. Théor. Appl.* **20** (1986) 371–381.

[123] THOMAS, W., On chain logic, path logic, and first-order logic over infinite trees, in: *Proc. 2nd Ann. IEEE Symp. on Logic in Computer Science* (1987) 245–256.

[124] TRAKHTENBROT, B.A., Finite automata and the logic of one-place predicates, *Siberian Math. J.* **3**, 103–131; English translation in: *AMS Transl.* **59** (1966) 23–55.

[125] TRAKHTENBROT, B.A. and Y.M. BARZDIN, *Finite Automata* (North-Holland, Amsterdam, 1973).

[126] VALK, R., Infinite behaviour of Petri nets, *Theoret. Comput. Sci.* **25** (1983) 311–341.

[127] VARDI, M.Y., Automatic verification of probabilistic concurrent finite-state programs, in: *Proc. 26th Ann. IEEE Symp. on Foundations of Computer Science* (1985) 327–338.

[128] VARDI, M.Y., A temporal fixed point calculus, in: *Proc. 15th Ann. ACM Symp. on Principles of Programming Languages* (1988) 250–259.

[129] VARDI, M.Y. and P. WOLPER, Automata-theoretic techniques for modal logics of programs, *J. Comput. System Sci.* **32** (1986) 183–221.

[130] VARDI, M.Y. and P. WOLPER, An automata theoretic approach to automatic program verification, in: *Proc. 1st Ann. IEEE Symp. on Logic in Computer Science* (1986) 332–334.

[131] VARDI, M.Y. and P. WOLPER, Reasoning about infinite computation paths, to appear.

[132] WAGNER, K., Eine Axiomatisierung der Theorie der regulären Folgenmengen, *J. Inform. Process. Cybernet.* **12** (1976) 337–354.

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

[134] WISNIEWSKI, K., A generalization of finite automata, *Fund. Inform.* **10** (1987) 415–436.

[135] WOLPER, P., Temporal logic can be more expressive, *Inform. and Control* **56** (1983) 72–99.

[136] ABADI, M., L. LAMPORT and P. WOLPER, Realizable and unrealizable specifications of reactive systems, in: G. Ausiello et al., eds, *Proc. 16th Internat. Coll. on Automata, Languages, and Programming*, Lecture Notes in Computer Science, Vol. 372 (Springer, Berlin, 1989) 1–17.

[137] EMERSON, E.A. and C.S. JUTLA, On simultaneously determinizing and complementing ω-automata, in: *Proc. 4th Ann. Symp. on Logic in Computer Science* (1989) 333–342.

[138] ENGELFRIET, J. and H.J. HOOGEBOOM, Automata with storage on infinite words, in: G. Ausiello et al., eds., *Proc. 16th Intern. Coll. on Automata, Languages, and Programming*, Lecture Notes in Computer Science, Vol. 372 (Springer, Berlin, 1989) 289–303.

[139] GUREVICH, Y., Games People Play, in: S. MacLane and D. Siefkes, eds., *The Collected Works of J.R. Büchi* (Springer, Berlin, 1990) 518–524.

[140] PNUELI, A. and R. ROSNER, On the synthesis of an asynchronous reactive module, in: D. Ausiello et al., eds., *Proc. 16th Internat. Coll. on Automata, Languages, and Programing*, Lecture Notes in Computer Science, Vol. 372 (Springer, Berlin, 1989) 652–671.

[141] SAFRA, S. and M.Y. VARDI, On ω-automata and temporal logic, in: *Proc. 21st Ann. Symp. on Theory of Computing* (1989) 127–137.

[142] SAOUDI, A., Pushdown automata on infinite trees and omega-Kleene closure of context-free tree sets, in: A. Kreczmar and G. Mirkowka, eds., *Math. Found. of Comput. Sci. 1989*, Lecture Notes in Computer Science, Vol. 379 (Springer, Berlin, 1989) 445–457.

[143] SKURCZYŃSKI, J., The Borel Hierarchy is infinite in the class of regular sets of trees, in: J. Csirik et al., eds., *Fundamentals of Computation Theory*, Lecture Notes in Computer Science, Vol. 380 (Springer, Berlin, 1989) 416–423.

[144] THOMAS, W., Infinite trees and automaton definable relations over ω-words, in: C. Choffrut and T. Lengauer, eds., *Proc. 7th Ann. Symp. STACS 90*, Lecture Notes in Computer Science, Vol. 415 (Springer, Berlin, 1990) 263–277.

Handbook of Theoretical Computer Science

Volume B

FORMAL MODELS AND SEMANTICS

edited by
JAN VAN LEEUWEN,
Utrecht University, The Netherlands

1990

ELSEVIER
AMSTERDAM • NEW YORK • OXFORD • TOKYO

THE MIT PRESS
CAMBRIDGE, MASSACHUSETTS