Electronic Colloquium on Computational Complexity, Report No. 94 (2007)

Efficient Algorithms for Membership in Boolean Hierarchies of Regular Languages

Christian Glaßer* Heinz Schmitz† Victor Selivanov‡

August 2, 2007

Abstract

The purpose of this paper is to provide *efficient* algorithms that decide membership for classes of several Boolean hierarchies for which efficiency (or even decidability) were previously not known. We develop new forbidden-chain characterizations for the single levels of these hierarchies and obtain the following results:

- The classes of the Boolean hierarchy over level Σ_1 of the dot-depth hierarchy are decidable in NL (previously only the decidability was known). The same remains true if predicates mod d for fixed d are added.
- If modular predicates for arbitrary d are allowed, then the classes of the Boolean hierarchy over level Σ_1 are decidable.
- For the restricted case of a two-letter alphabet, the classes of the Boolean hierarchy over level Σ_2 of the Straubing-Thérien hierarchy are decidable in NL. This is the first decidability result for this hierarchy.
- The membership problems for all mentioned Boolean-hierarchy classes are logspace many-one hard for NL.
- The membership problems for quasi-aperiodic languages and for d-quasi-aperiodic languages are logspace many-one complete for PSPACE.

1 Introduction

The study of decidability and complexity questions for classes of regular languages is a central research topic in automata theory. Its importance stems from the fact that finite automata are fundamental to many branches of computer science, e.g., databases, operating systems, verification, and hardware and software design.

There are many examples for decidable classes of regular languages (e.g., locally testable languages), while the decidability of other classes is still a challenging open question (e.g., dot-depth

*Julius-Maximilians-Universität Würzburg, Theoretische Informatik, Am Hubland, 97074 Würzburg, Germany. EMail: glasser@informatik.uni-wuerzburg.de

†Fachhochschule Trier, Fachbereich Informatik, Schneidershof, 54293 Trier, Germany. EMail: schmitz@informatik.fh-trier.de

‡A. P. Ershov Institute of Informatics Systems, Siberian Division of the Russian Academy of Sciences, Russia. EMail: vseliv@nspu.ru. This work was done during a stay of the third author at the University of Würzburg, supported by DFG Mercator program and by RFBR grant 07-01-00543a.

ISSN 1433-8092

two, generalized star-height). Moreover, among the decidable classes there is a broad range of complexity results. For some of them, e.g., the class of piecewise testable languages, efficient algorithms are known that work in nondeterministic logarithmic space (NL) and hence in polynomial time. For other classes, a membership test needs more resources, e.g., deciding the membership in the class of star-free languages is PSPACE-complete.

The purpose of this paper is to provide *efficient* algorithms that decide membership for classes of several Boolean hierarchies for which efficiency (or even decidability) were not known previously. Many of the known efficient decidability results for classes of regular languages are based on so-called forbidden-pattern characterizations. Here a language belongs to a class of regular languages if and only if its deterministic finite automaton does *not* have a certain subgraph (the forbidden pattern) in its transition graph. Usually, such a condition can be checked efficiently, e.g., in nondeterministic logarithmic space [Ste85a, CPP93, GS00a, GS00b].

However, for the Boolean hierarchies considered in this paper, the design of efficient algorithm is more involved, since here no forbidden-pattern characterizations are known. More precisely, wherever decidability is known, it is obtained from a characterization of the corresponding class in terms of *forbidden* alternating chains of word extensions. Though the latter also is a forbidden property, the known characterizations are not efficiently checkable in general (Exceptions are the special 'local' cases $\Sigma_1^{\varrho}(n)$ and $\mathcal{C}_k^1(n)$ where decidability in NL is known [SW98, Sch01].) To overcome these difficulties, we first develop alternative forbidden-chain characterizations (they essentially ask only for certain reachability conditions in transition graphs). With our new characterizations at hand, we explicitly provide efficient algorithms for membership tests in NL and prove their correctness. For two of the considered Boolean hierarchies, these are the first decidable characterizations at all, i.e., for the classes $\Sigma_2^{\varrho}(n)$ for the alphabet $A = \{a, b\}$, and for the classes $\Sigma_1^{\tau}(n)$).

Definitions. We sketch the definitions of the Boolean hierarchies considered in this paper; formal definitions can be found in section 2. Σ_1^{ϱ} denotes the class of languages definable by first-order Σ_1-sentences of the signature $\varrho = \{\leq, Q_a, \ldots\}$ where for every letter $a \in A$, $Q_a(i)$ is true if and only if the letter a appears at the i-th position in the word. Σ_1^{ϱ} equals level 1/2 of the Straubing-Thérien hierarchy (STH for short) [Str81, Thé81, Str85, PP86]. Σ_2^{ϱ} is the class of languages definable by similar first-order Σ_2-sentences; this class equals level 3/2 of the Straubing-Thérien hierarchy. Let σ be the signature obtained from ϱ by adding constants for the minimum and maximum positions in words and adding functions that compute the successor and the predecessor of positions. Σ_1^{σ} denotes the class of languages definable by first-order Σ_1-sentences of the signature σ; this class equals level 1/2 of the dot-depth hierarchy (DDH for short) [CB71, Tho82]. Let τ_d be the signature obtained from σ by adding the unary predicates $P_d^0, \ldots, P_d^{d-1}$ where $P_d^j(i)$ is true if and only if $i \equiv j (\text{mod } d)$. Let τ be the union of all τ_d. $\Sigma_1^{\tau_d}(n)$ (resp., $\Sigma_1^{\tau}(n)$) is the class of languages definable by first-order Σ_1-sentences of the signature τ_d (resp., τ). $\mathcal{C}_k^d$ is the generalization of Σ_1^{ϱ} where neighborhoods of $k+1$ consecutive letters and distances modulo d are expressible (Definition 2.5). For a class $\mathcal{D}$ (in our case one of the classes Σ_1^{ϱ}, Σ_1^{σ}, $\mathcal{C}_k^d$, $\Sigma_1^{\tau_d}$, Σ_1^{τ}, and Σ_2^{ϱ} for $|A| = 2$), the *Boolean hierarchy over* $\mathcal{D}$ is the family of classes

$$\mathcal{D}(n) \overset{df}{=} \{L \mid L = L_1 - (L_2 - (\ldots - L_n)) \text{ where } L_1, \ldots, L_n \in \mathcal{D} \text{ and } L_1 \supseteq L_2 \supseteq \cdots \supseteq L_n\}.$$

The Boolean hierarchies considered in this paper are illustrated in Figure 1.

Our Contribution. The paper contributes to the understanding of Boolean hierarchies of regular languages in two ways:

1. For the classes $\Sigma_1^{\sigma}(n)$, $\Sigma_1^{\tau_{\mathrm{d}}}(n)$, and $\Sigma_2^{\varrho}(n)$ for the alphabet $A = \{a, b\}$ we prove new characterizations in terms of forbidden alternating chains. In case of $\Sigma_2^{\varrho}(n)$ for the alphabet $A = \{a, b\}$, this is the first characterization of this class.

2. For the classes $\Sigma_1^{\sigma}(n)$, $\mathcal{C}_k^d(n)$, $\Sigma_1^{\tau_{\mathrm{d}}}(n)$, and $\Sigma_2^{\varrho}(n)$ for the alphabet $A = \{a, b\}$ we construct the first efficient algorithms for testing membership in these classes. In particular, this yields the decidability of the classes $\Sigma_1^{\tau}(n)$, and of $\Sigma_2^{\varrho}(n)$ for the alphabet $A = \{a, b\}$.

We also show that the membership problems for all mentioned Boolean-hierarchy classes are logspace many-one hard for NL. An overview of the obtained decidability and complexity results can be found in Table 1. Moreover, we prove that the membership problems for quasi-aperiodic languages and for d-quasi-aperiodic languages are logspace many-one complete for PSPACE.

Boolean hierarchies can also be seen as fine-grain measures for regular languages in terms of descriptional complexity. Note that all Boolean hierarchies considered in this paper do not collapse [Shu98, SS00, Sel04]. Moreover, all these hierarchies either are known or turn out to be decidable (see Table 1 for the attribution of these results). If in addition the Boolean closure of the base class is decidable, then we can even exactly compute the Boolean level of a given language. By known results (summarized in Theorems 2.1 and 2.7), one can do this exact computation of the level for the Boolean hierarchies over Σ_1^{ϱ}, Σ_2^{ϱ} (for alphabet $A = \{a, b\}$), $\mathcal{C}_k$, Σ_1^{σ}, and Σ_1^{τ}. To achieve the same for the Boolean hierarchies over $\mathcal{C}_k^d$ and $\Sigma_1^{\tau_{\mathrm{d}}}$ we need the decidability of their Boolean closures which is not known.

Related Work. Due to the many characterizations of regular languages there are several approaches to attack decision problems on subclasses of regular languages: Among them there is the algebraic, the automata-theoretic, and the logical approach. In this paper we mainly use the logical approach which has a long tradition starting with the early work of Trakhtenbrot [Tra58] and Büchi [Büc62]. Decidability questions for Boolean hierarchies over classes of concatenation hierarchies were previously studied by [SW98, Sch01, GS01a, Sel04]. Enrichments of the first-order logics related to the dot-depth hierarchy and the Straubing-Thérien hierarchy were considered in [BCST92, Str94, MPT00, Sel04, CPS06]. For more background on regular languages, starfree languages, concatenation hierarchies, and their decidability questions we refer to the survey articles [Brz76, Pin95, Pin96a, Pin96b, Yu96, PW02, Wei04].

Paper Outline. The paper is organized as follows. After the preliminaries (section 2), we explain the general idea of an efficient membership algorithm for the classes $\mathcal{C}_k^d(n)$ (section 3). This easy example shows how a suitable characterization of a Boolean hierarchy can be turned into an efficient membership test. Later we will construct similar, but more complicated algorithms for other Boolean hierarchies. In section 4 we develop new alternating-chain characterizations for the Boolean hierarchies over Σ_1^{σ}, $\Sigma_1^{\tau_{\mathrm{d}}}$, and Σ_2^{ϱ} for the alphabet $A = \{a, b\}$. In section 5 we exploit these characterizations and design efficient algorithms for testing the membership in these classes. In particular, we obtain the decidability of the classes $\Sigma_1^{\tau}(n)$ and $\Sigma_2^{\varrho}(n)$ for the alphabet $A = \{a, b\}$. Finally, section 6 provides lower bounds for the complexity of the considered decidability problems. As a consequence (with the exception of $\Sigma_1^{\tau}(n)$) the membership problems of all considered Boolean levels are logspace many-one complete for NL. In contrast, the membership problems of the general classes FO_{τ} and FO_{τ_d} are logspace many-one complete for PSPACE and hence are strictly more complex.

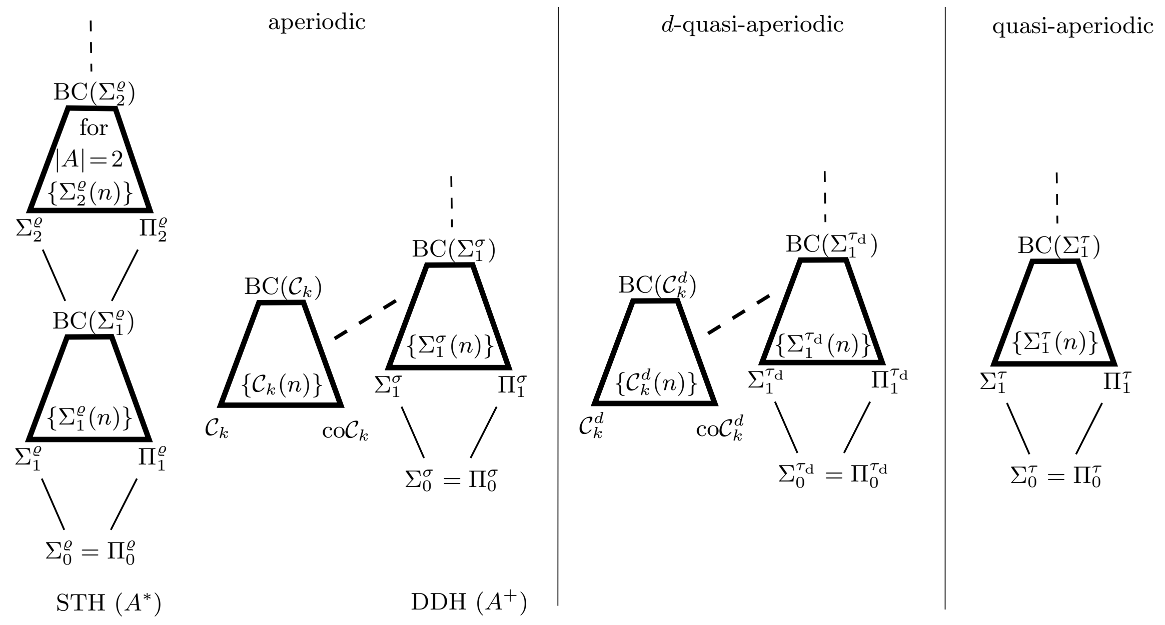


Figure 1: Boolean hierarchies considered in this paper.

Boolean hierarchy classes	decidability	complexity
$\Sigma_1^{\varrho}(n)$	[SW98]	NL-complete [SW98]
$\mathcal{C}_k^1(n)$	[GS01a, Sel01]	NL-complete [Sch01]
$\Sigma_1^{\sigma}(n)$	[GS01a]	NL-complete [this paper]
$\mathcal{C}_k^d(n)$	[Sel04]	NL-complete [this paper]
$\Sigma_1^{\tau_{\mathrm{d}}}(n)$	[Sel04]	NL-complete [this paper]
$\Sigma_1^{\tau}(n)$	[this paper]	no efficient bound known (see Remark 5.6)
$\Sigma_2^{\varrho}(n)$ for $\|A\| = 2$	[this paper]	NL-complete [this paper]

Table 1: Overview of decidability and complexity results.

2 Preliminaries

In this section we recall definitions and results that are needed in later proofs. If not stated otherwise, A denotes some finite alphabet with $|A| \geq 2$. Let A^* and A^+ be the sets of finite (resp., of finite non-empty) words over A. If not stated otherwise, variables range over the set of natural numbers. We use $[m, n]$ as abbreviation for the interval $\{m, m+1, \dots, n\}$. For a deterministic finite automaton $M = (A, Z, \delta, s_0, F)$ (dfa for short), the number of states is denoted by $|M|$ and the accepted language is denoted by $L(M)$. Moreover, for words x and y we write $x \equiv_M y$ if and only if $\delta(s_0, x) = \delta(s_0, y)$. For a class of languages $\mathcal{C}$, $\mathrm{BC}(\mathcal{C})$ denotes the Boolean closure of $\mathcal{C}$, i.e., the closure under unions, intersection, and complementation.

All hardness and completeness results in this paper are with respect to logspace many-one reductions, i.e., whenever we refer to NL-complete sets (resp., PSPACE-complete sets) then we mean sets that are logspace many-one complete for NL (resp., PSPACE).

2.1 The Logical Approach to Regular Languages

Relate to any alphabet $A = \{a, \dots\}$ the signatures $\varrho = \{\leq, Q_a, \dots\}$ and $\sigma = \{\leq, Q_a, \dots, \bot, \top, p, s\}$, where $\leq$ is a binary relation symbol, Q_a (for any $a \in A$) is a unary relation symbol, $\bot$ and $\top$ are constant symbols, and p, s are unary function symbols. A word $u = u_0 \dots u_n \in A^+$ may be considered as a structure $\mathbf{u} = (\{0, \dots, n\}; \leq, Q_a, \dots)$ of signature σ, where $\leq$ has its usual meaning, $Q_a (a \in A)$ are unary predicates on $\{0, \dots, n\}$ defined by $Q_a(i) \Leftrightarrow u_i = a$, the symbols $\bot$ and $\top$ denote the least and the greatest elements, while p and s are respectively the predecessor and successor functions on $\{0, \dots, n\}$ (with $p(0) = 0$ and $s(n) = n$). Similarly, a word $v = v_1 \dots v_n \in A^*$ may be considered as a structure $\mathbf{v} = (\{1, \dots, n\}; \leq, Q_a, \dots)$ of signature ϱ. For a sentence ϕ of σ (resp., ϱ), let $L_\phi = \{u \in A^+ | \mathbf{u} \models \phi\}$ (resp., $L_\phi = \{v \in A^* | \mathbf{v} \models \phi\}$). Sentences ϕ, ψ are treated as equivalent when $L_\phi = L_\psi$. A language is FO_σ-definable (resp., FO_ϱ-definable) if it is of the form L_ϕ, where ϕ ranges over first-order sentences of σ (resp., ϱ). We denote by Σ_k^σ (resp., Π_k^σ) the class of languages that can be defined by a sentence of σ having at most $k-1$ quantifier alternations, starting with an existential (resp., universal) quantifier. The classes Σ_k^ϱ and Π_k^ϱ are defined analogously with respect to ϱ.

It is well-known that the class of FO_σ-definable languages (as well as the class of FO_ϱ-definable languages) coincides with the class of *regular aperiodic languages* which are also known as the *star-free languages*. Moreover there is a levelwise correspondence to concatenation hierarchies, i.e., the classes Σ_k^ϱ, Π_k^ϱ, and $\mathrm{BC}(\Sigma_k^\varrho)$ coincide with the classes of the Straubing-Thérien hierarchy [PP86], while the classes Σ_k^σ, Π_k^σ, and $\mathrm{BC}(\Sigma_k^\sigma)$ coincide with the classes of the dot-depth hierarchy [Tho82].

We will consider also some enrichments of the signature σ. Namely, for any positive integer d let τ_d be the signature $\sigma \cup \{P_d^0, \dots, P_d^{d-1}\}$, where P_d^r is the unary predicate true on the positions of a word which are equivalent to r modulo d. By FO_{τ_d}-definable language we mean any language of the form L_ϕ, where ϕ is a first-order sentence of signature τ_d. Note that signature τ_1 is essentially the same as σ because P_1^0 is the valid predicate. In contrast, for $d > 1$ the FO_{τ_d}-definable languages need not to be aperiodic. E.g., the sentence $P_2^1(\top)$ defines the language L consisting of all words of even length which is known to be non-aperiodic. We are also interested in the signature $\tau = \bigcup_d \tau_d$. Barrington et al. [BCST92, Str94] defined quasi-aperiodic languages and showed that this class coincides with the class of FO_τ-definable languages. With the same proof we obtain the equality of the class of d-quasi-aperiodic languages and the class of FO_{τ_d}-definable languages [Sel04]. It was observed in the same paper that $\Sigma_n^\tau = \bigcup_d \Sigma_n^{\tau_\mathrm{d}}$ for each $n > 0$, where Σ_n with an upper index denotes the class of regular languages defined by Σ_n-sentences of the corresponding signature in the upper index.

Theorem 2.1 *For the following classes $\mathcal{D}$ it is decidable whether a given dfa M accepts a language in $\mathcal{D}$.*

1. $\mathrm{BC}(\Sigma_1^\varrho)$ *[Sim75]*
2. $\mathrm{BC}(\Sigma_2^\varrho)$ *for* $|A| = 2$ *[Str88]*
3. $\mathrm{BC}(\Sigma_1^\sigma)$ *[Kna83]*
4. $\mathrm{BC}(\Sigma_1^\tau)$ *[MPT00]*

Proof $\mathrm{BC}(\Sigma_1^\varrho)$ is the class of piecewise testable languages (or equivalently, the first level of the Straubing-Thérien hierarchy) which was shown to be decidable by Simon [Sim75]. $\mathrm{BC}(\Sigma_2^\varrho)$ is the second level of the Straubing-Thérien hierarchy which is known to be decidable only for the alphabet $\{a, b\}$ [Str88]. $\mathrm{BC}(\Sigma_1^\sigma)$ is the first level of the dot-depth hierarchy shown to be decidable by Knast [Kna83]. The decidability of $\mathrm{BC}(\Sigma_1^\tau)$ is obtained as follows: Let $\Sigma_1^{\varrho\cup\mathrm{REG}}$ denote the class of languages definable by first-order Σ_1-sentences of ϱ that additionally may use arbitrary regular numerical predicates. By [MPT00, Corollary 10 and Theorem 12] and [Sel04, Theorem 3.4], $\mathrm{BC}(\Sigma_1^{\varrho\cup\mathrm{REG}}) = \mathrm{BC}(\Sigma_1^\tau)$. Chaubard, Pin, and Straubing [CPS06] credit [MPT00] with the decidability of $\mathrm{BC}(\Sigma_1^{\varrho\cup\mathrm{REG}})$. $\square$

We do not know a reference for the decidability of $\mathrm{BC}(\Sigma_1^{\tau_\mathrm{d}})$ which is likely to be a generalization of Knast's proof.

2.2 Preliminaries on Boolean Hierarchies

Here we recall some definitions and facts on Boolean hierarchies. Let S be any set. By a *base in* S we mean any class $\mathcal{C}$ of subsets of S which is closed under union and intersection and contains $\emptyset$ and S as elements. It is easy to see that, e.g., Σ_1^ϱ is a base in A^* and Σ_1^σ, $\Sigma_1^{\tau_\mathrm{d}}$ are bases in A^+.

For any base $\mathcal{C}$ and $n \geq 1$, let $\mathcal{C}(n)$ be the class of all sets of the form $\bigcup_i (L_{2i} \setminus L_{2i+1})$, where $L_0 \supseteq L_1 \supseteq \cdots$ is a descending sequence of sets from $\mathcal{C}$ and $L_i = \emptyset$ for $i \geq n$. The sequence $\{\mathcal{C}(n)\}_{n\geq 1}$ is known as the *Boolean hierarchy (BH) over* $\mathcal{C}$. As is well-known, $\mathcal{C}(n) \cup \mathrm{co}(\mathcal{C}(n)) \subseteq \mathcal{C}(n+1)$ for every n, and the class $\bigcup_n \mathcal{C}(n)$ coincides with the Boolean closure $\mathrm{BC}(\mathcal{C})$ of $\mathcal{C}$. A BH has several nice characterizations (see e.g. [KSW87, CGH$^+$88]), in particular $\mathcal{C}(n)$ coincides with the class of languages L that can be written as $L = L_1 - (L_2 - (\ldots - L_n))$ for some $L_1, L_2, \ldots, L_n \in \mathcal{C}$ with $L_1 \supseteq L_2 \supseteq \cdots \supseteq L_n$.

If any partial order $S = (S; \leq)$ is given we observe that the class $\mathcal{C}$ consisting of all upper sets of S is a base in S. Recall that a set $L \subseteq S$ is upper if $x \in L$ and $x \leq y$ imply $y \in L$. By an *alternating chain* of length n for a set $L \subseteq S$ we mean a sequence $(x_0, \ldots, x_n)$ of elements of S such that $x_0 \leq \cdots \leq x_n$ and $x_i \in L \Leftrightarrow x_{i+1} \notin L$ for every $i < n$. Such a chain is called a 1-alternating chain if $x_0 \in L$, otherwise it is called a 0-alternating chain. Variants of the following fact frequently appear when treating BH's of different kind [Sel04, Proposition 4.6].

Proposition 2.2 *Let $S = (S; \leq)$ be a partial order and $\mathcal{C}$ the base of upper sets in S. For all $L \subseteq S$ and $n \geq 1$, $L \in \mathcal{C}(n)$ if and only if L has no 1-alternating chains of length n.*

So there is a generic method to prove characterizations of classes of a BH in terms of *forbidden* alternating chains: One has to identify a partial order such that the respective base is just the class of all upper sets. Even more, the latter can be carried out over a different base. Let us first observe the following assertion which essentially coincides with [Sel04, Proposition 4.1].

Proposition 2.3 *Let $\mathcal{C}$ a base in S, $\mathcal{M}$ a base in T and $f : T \to S$ be a surjection such that $f(M) \in \mathcal{C}$ for all $M \in \mathcal{M}$. Moreover, let $L \in \mathcal{C}(n)$ for some $n \geq 1$ and $f^{-1}(L) \in \mathcal{M}(n)$, i.e., $f^{-1}(L) = \bigcup_i (M_{2i} \setminus M_{2i+1})$ for some $M_i \in \mathcal{M}$ with $M_0 \supseteq M_1 \supseteq \cdots$ and $M_i = \emptyset$ for $i \geq n$. Then $L = \bigcup_i (f(M_{2i}) \setminus f(M_{2i+1}))$, $f(M_0) \supseteq f(M_1) \supseteq \cdots$, $f(M_n) = \emptyset$ and therefore $L \in \mathcal{C}$.*

Now we can identify a condition which is sufficient to characterize membership in the classes of a BH over one base in terms of alternating chains over a different base. It follows immediately from Propositions 2.2 and 2.3.

Theorem 2.4 *Let $\mathcal{C}$ be a base in S and $L \subseteq S$. Moreover, let $T = (T; \leq)$ be some partial order, $\mathcal{M}$ be the base of upper sets in T and $f : T \to S$ be a surjection. If $f(M) \in \mathcal{C}$ for all $M \in \mathcal{M}$ and $f^{-1}(L) \in \mathcal{M}$ for all $L \in \mathcal{C}$ then for all $n \geq 1$ it holds that*

$$L \in \mathcal{C}(n) \Leftrightarrow f^{-1}(L) \text{ has no 1-alternating chain of length } n \text{ in } (T; \leq).$$

Observe that $(T; \leq)$ may depend on L. One way to apply the theorem is to prove that $(T; \leq)$ is a well partial order (i.e., it has neither infinite descending chains nor infinite antichains). Then the upper sets M in $(T; \leq)$ can be finitely represented which helps to show that $f(M) \in \mathcal{C}$ for all $M \in \mathcal{M}$. This approach has been carried out for the BH over Σ_1^{σ} [GS01a] and $\Sigma_1^{\tau_d}$ [Sel04]. We recall these results in the following subsections.

2.3 Preliminaries on the Classes $\mathcal{C}_k^d(n)$

In section 3 we will also refer to 'local' versions of the BH's over the classes Σ_1^{σ} and $\Sigma_1^{\tau_d}$ which were considered in [Ste85a, GS01a, Sel01, Sel04]. For any $k \geq 0$ the following partial order on non-empty words was studied in [Ste85a, GS01a, Sel01]: $u \leq_k v$, if $u = v \in A^{\leq k}$ or $u, v \in A^{>k}$, $p_k(u) = p_k(v)$, $s_k(u) = s_k(v)$, and there is a k-embedding $f : u \to v$. Here $p_k(u)$ ($s_k(u)$) is the prefix (resp., suffix) of u of length k, and the k-embedding f is a monotone injective function from $\{0. \ldots, |u| - 1\}$ to $\{0. \ldots, |v| - 1\}$ such that $u(i) \cdots u(i + k) = v(f(i)) \cdots v(f(i) + k)$ for all $i < |u| - k$. Note that the relation $\leq_0$ is just the subword relation.

In [Sel04] the following generalization of this partial order was considered.

Definition 2.5 ([Sel04]) *Let $k \geq 0$ and $d > 0$.*

1. *We say that a k-embedding $f : u \to v$ is a (k, d)-embedding, if $P_d^r(i)$ implies $P_d^r(f(i))$ for all $i < |u|$ and $r < d$.*

2. *For all $u, v \in A^+$, let $u \leq_k^d v$ mean that $u = v \in A^{\leq k}$ or $u, v \in A^{>k}$, $p_k(u) = p_k(v)$, $s_k(u) = s_k(v)$, and there is a (k,d)-embedding $f : u \to v$.*
3. *With $\mathcal{C}_k^d$ we denote the class of all upper sets in $(A^+; \leq_k^d)$.*

Note that for $d = 1$ the order $\leq_k^d$ coincides with $\leq_k$. From Proposition 2.2 we obtain the following characterization of the classes $\mathcal{C}_k^d(n)$, first established in [GS01a, Sel01, Sel04].

Proposition 2.6 *For all $L \subseteq A^+$ and $n \geq 1$, $L \in \mathcal{C}_k^d(n)$ if and only if L has no 1-alternating chains of length n in $(A^+; \leq_k^d)$.*

Moreover, it is shown in these papers that $(A^+; \leq_k^d)$ is a well partial order and that $\Sigma_1^{\tau_\mathrm{d}} = \bigcup_k \mathcal{C}_k^d$ and hence $\Sigma_1^{\tau} = \bigcup_{k,d} \mathcal{C}_k^d$.

Theorem 2.7 ([Ste85a]) *It is decidable whether a given dfa M accepts a language in* $\mathrm{BC}(\mathcal{C}_k)$.

For $d > 1$ it is not known whether $\mathrm{BC}(\mathcal{C}_k^d)$ is decidable. However, we expect that this can be shown by generalizing the proof in [Ste85a].

2.4 Preliminaries on the Classes Σ_1^{σ} and $\Sigma_1^{\tau_\mathrm{d}}$

In case of the Boolean hierarchy over Σ_1^{σ} a more sophisticated characterization of the single levels is known [GS01a]. Let $M = (A, Z, \delta, s_0, F)$ be a dfa. A word $u \in A^+$ is *M-idempotent* if $\delta(s, u) = \delta(s, uu)$ for all states $s \in Z$. Define $c_M = (n+1)^{(n+1)^{n+1}}$ where $n = |M|$. The *set of structured words* is defined as

$$\mathcal{A}_M \overset{\mathit{df}}{=} \{ w_0 \boxed{u_1} w_1 \boxed{u_2} w_2 \cdots \boxed{u_m} w_m \mid u_i, w_i \in A^+,\ |w_i|, |u_i| \leq c_M \text{ and each } u_i \text{ is } M\text{-idempotent} \}.$$

For a structured word $x = w_0 \boxed{u_1} w_1 \boxed{u_2} w_2 \cdots \boxed{u_m} w_m \in \mathcal{A}_M$ let $\overline{x} \overset{\mathit{df}}{=} w_0 u_1 w_1 \cdots u_m w_m$. It is shown in [GS01a] that every word can be structured. So $f : \mathcal{A}_M \to A^+$ with $f(x) \overset{\mathit{df}}{=} \overline{x}$ is a surjection.

For $x, y \in \mathcal{A}_M$ we write $x \preceq_M y$ if there exist words $u_i \in A^+$ and $x_i, z_i \in \mathcal{A}_M$ such that

$$\begin{aligned} x &= x_0 \boxed{u_1} \quad x_1 \boxed{u_2} \quad x_2 \cdots \boxed{u_m} \quad x_m \quad \text{and} \\ y &= x_0 \boxed{u_1} z_1 \boxed{u_1} x_1 \boxed{u_2} z_2 \boxed{u_2} x_2 \cdots \boxed{u_m} z_m \boxed{u_m} x_m. \end{aligned}$$

Theorem 2.8 ([GS01a]) *Let M be a dfa, $L = L(M) \subseteq A^+$ and $n \geq 1$. Then $L \in \Sigma_1^{\sigma}(n)$ if and only if $f^{-1}(L)$ has no 1-alternating chain of length n in $(\mathcal{A}_M; \preceq_M)$.*

Next we recall a generalization of the last result to the BH over $\Sigma_1^{\tau_\mathrm{d}}$ for any $d > 1$ as already observed in [Sel04]. Let $\mathcal{A}_M^d$ denote the set of structured words $w_0 \boxed{u_1} w_1 \boxed{u_2} w_2 \cdots \boxed{u_m} w_m$ such that $0 < |w_i|, |u_j| \leq c_{\mathcal{A}} \cdot d$ for all $i \in [0, n], j \in [1, n]$ and $|w_i| \equiv |u_j| \equiv 0 \pmod d$ for all $i \in [0, n-1], j \in [1, n]$. The relation $\preceq_M$ and the function $f : \mathcal{A}_M^d \to A^+$ are defined exactly as above. Then small modifications of the proofs in [GS01a] yield the conditions of f from Theorem 2.4 and hence the following result (for details see the journal version of [Sel04] submitted to Theoretical Informatics and Applications).

Theorem 2.9 ([Sel04]) *Let M be a dfa, $L = L(M) \subseteq A^+$ and $d, n \geq 1$. Then $L \in \Sigma_1^{\tau_\mathrm{d}}(n)$ if and only if $f^{-1}(L)$ has no 1-alternating chain of length n in $(\mathcal{A}_M^d; \preceq_M)$.*

3 Efficient Algorithms for $\mathcal{C}_k^d(n)$

The main objective of this paper is the design of *efficient* algorithms deciding membership for particular Boolean hierarchies. For this, two things are needed: First, we need to prove suitable characterizations for the single levels of these hierarchies. This gives us certain criteria that can be used for testing membership. Second, we need to construct algorithms that efficiently apply these criteria. If both steps are successful, then we obtain an efficient membership test.

Based on known ideas for membership tests for $\mathcal{C}_0^1(n)$ [SW98][1] and $\mathcal{C}_k^1(n)$ [Sch01], in this section we explain the construction of a nondeterministic, logarithmic-space membership algorithm for the classes $\mathcal{C}_k^d(n)$. This is the first efficient membership test for this general case. Our explanation has an introductory character, since it shows how a suitable characterization of a Boolean hierarchy can be turned into an efficient membership test. Later (in section 5) we will use similar, but more complicated constructions.

We start with the easiest case $k = 0$ and $d = 1$, i.e., with the classes $\mathcal{C}_0^1(n)$. By Proposition 2.6,

$$L \notin \mathcal{C}_0^1(n) \quad \Leftrightarrow \quad L \text{ has a 1-alternating } \leq_0\text{-chain of length } n. \tag{1}$$

We argue that for a given L, represented by a finite automaton M, the condition on the right-hand side can be verified in nondeterministic logarithmic space. So we have to test whether there exists a chain $w_0 \leq_0 \cdots \leq_0 w_n$ such that $w_i \in L$ if and only if i is even. This is done by the following algorithm.

```
0   // On input of a deterministic, finite automaton $\mathtt{M} = (\mathtt{A}, \mathtt{Z}, \delta, \mathtt{z_0}, \mathtt{F})$
       the algorithm tests whether $\mathtt{L(M)} \in \mathcal{C}_0^1(n)$.
1   let $\mathtt{s_0} = \cdots = \mathtt{s_n} = \mathtt{z_0}$
2   do
3      nondeterministically choose $\mathtt{a} \in \mathtt{A}$ and $\mathtt{j} \in [\mathtt{0}, \mathtt{n}]$
4      for $\mathtt{i} = \mathtt{j}$ to n
5         $\mathtt{s_i} = \delta(\mathtt{s_i}, \mathtt{a})$        // stands for the imaginary command:  $\mathtt{w_i} = \mathtt{w_i a}$
6      next i
7   until $\forall \mathtt{i}, [\mathtt{s_i} \in \mathtt{F} \Leftrightarrow$ i is even$]$
8   accept
```

The algorithm guesses the words $w_0, \ldots, w_n$ in parallel. However, instead of constructing these words in the memory, it guesses the words letter by letter and stores only the states $s_i = \delta(z_0, w_i)$. More precisely, in each pass of the loop we choose a letter a and a number j, and we interpret this choice as appending a to the words $w_j, \ldots, w_n$. Simultaneously, we update the states $s_0, \ldots, s_n$ appropriately. By doing so, we guess all possible chains $w_0 \leq_0 \cdots \leq_0 w_n$ in a way such that we know the states $s_i = \delta(z_0, w_i)$. This allows us to easily verify the right-hand side of (1) in line 7. Hence, testing non-membership in $\mathcal{C}_0^1(n)$ is in NL. By NL = coNL [Imm88, Sze87], also the membership test belongs to NL.

The algorithm can be modified such that it works for $\mathcal{C}_0^d(n)$ where d is arbitrary: For this we have to make sure that the guessed $\leq_0$-chain is even a $\leq_0^d$-chain, i.e., the word extensions must be such that the lengths of single insertions are divisible by d. This is done by (i) introducing new variables l_i that count the current length of w_i modulo d and (ii) by making sure that

[1]For all n, the classes $\mathcal{C}_0^1(n)$ and $\Sigma_1^e(n)$ coincide up to the empty word, i.e., $\mathcal{C}_0^1(n) = \{L \cap A^+ \mid L \in \Sigma_1^e(n)\}$.

$l_i = l_{i+1}$ whenever $j \le i < n$ (i.e., letters that appear in both words, w_i and w_{i+1}, must appear at equivalent positions modulo d). So also the membership test for $\mathcal{C}_0^d(n)$ belongs to NL.

Finally, we adapt the algorithm to make it work for $\mathcal{C}_k^d(n)$ where d and k are arbitrary. So we have to make sure that the guessed $\le_0^d$-chain is even a $\le_k^d$-chain. This means that for a single insertion $u_1u_2 \le_0^d u_1vu_2$ it must additionally hold that the length k prefixes of u_2 and vu_2 are equal. The latter is called the *prefix condition.* The algorithm can test this condition by introducing new variables v_i that contain a guessed preview of the next k letters in w_i. Each time a letter is appended to w_i, (i) we verify that this letter is consistent with the preview v_i and (ii) we update v_i by removing the first letter and by appending a new guessed letter. In this way the modified algorithm carries the length k previews of the w_i with it and it makes sure that guessed letters are consistent with these previews. Moreover, we modify the algorithm such that whenever $j \le i < n$, then the condition $v_i = v_{i+1}$ is tested. The latter makes sure that single insertions $u_1u_2 \le_0^d u_1vu_2$ satisfy the prefix condition and hence the involved words are even in $\le_k^d$ relation. This modified algorithm shows the following.

Theorem 3.1 *For $k \ge 0$ and $d \ge 1$,*

$$\{M \mid M \text{ is a deterministic finite automaton and } L(M) \in \mathcal{C}_k^d(n)\} \in \mathrm{NL}.$$

We now explain why the above idea does not immediately lead to a nondeterministic, logarithmic-space membership algorithm for the classes $\Sigma_1^\sigma(n)$, although an alternating chain characterization for $\Sigma_1^\sigma(n)$ is known from Theorem 2.8. Note that the described algorithm for $\mathcal{C}_k^d(n)$ stores the following types of variables in logarithmic space.

1. variables s_i that contain states of M
2. variables l_i that contain numbers from $[0, d-1]$
3. variables v_i that contain words of length k

However, the characterization of the classes $\Sigma_1^\sigma(n)$ (Theorem 2.8) is unsuitable for our algorithm: In order to verify the condition on the right-hand side of Theorem 2.8, we have to guess a sequence of structured words $x_1, \ldots, x_n$ and have to make sure that $x_i \preceq_M x_{i+1}$. In particular, we have to make sure that certain parts of x_i and x_{i+1} are M-idempotent. Again we would try to guess the words x_i letter by letter, but now we have to make sure that (larger) parts u of these words are M-idempotent. We do not know how to verify the latter condition in logarithmic space. (Note however, that in *linear space* the membership test is possible: Just guess a word u letter by letter and simultaneously store/update for each state s the state $\delta(s, u)$.)

In a similar way one observes that the characterization of the classes $\Sigma_1^{\tau_d}(n)$ at the end of the previous section cannot be used for the construction of an efficient membership test. So new characterizations of $\Sigma_1^\sigma(n)$ and $\Sigma_1^{\tau_d}(n)$ are needed in order to obtain efficient membership algorithms.

4 New Characterizations of Boolean-Hierarchy Classes

In this section we develop new alternating-chain characterizations that allow the design of efficient algorithms deciding membership for the Boolean hierarchies over Σ_1^σ, $\Sigma_1^{\tau_d}$, and Σ_2^ϱ for $|A| = 2$. We begin with the introduction of *marked words* and related partial orders which turn out to be crucial for the design of our algorithms. In subsection 4.2 we prove characterizations for $\Sigma_1^\sigma(n)$ and $\Sigma_1^{\tau_d}(n)$ in terms of alternating chains of marked words. Subsection 4.3 shows a way to exploit these results for the classes $\Sigma_1^\tau(n)$. Finally, subsection 4.4 develops an alternating-chain characterization for $\Sigma_2^\varrho(n)$ restricted to an alphabet that contains exactly two letters.

4.1 Marked Words

For a fixed finite alphabet A, let $\mathcal{A} \overset{df}{=} \{ [a,u] \mid a \in A, u \in A^* \}$ be the corresponding marked alphabet. Words over $\mathcal{A}$ are called marked words. For $w \in \mathcal{A}^*$ with $w = [a_1,u_1] \cdots [a_m,u_m]$ let $\overline{w} \overset{df}{=} a_1 \cdots a_m \in A^*$ be the corresponding unmarked word. Sometimes we use the functional notation $f_i(w) = a_1 u_1^i \cdots a_m u_m^i$, i.e., $f_0(w) = \overline{w}$. Clearly, $f_0 : \mathcal{A}^* \to A^*$ is a surjection. For $x = x_1 \cdots x_m \in A^+$ and $u \in A^*$ we define $[x,u] \overset{df}{=} [x_1,\varepsilon] \cdots [x_{m-1},\varepsilon][x_m,u]$.

Next we define a relation on marked words. For $w, w' \in \mathcal{A}^*$ we write $w \preceq w'$ if and only if there exist $m \geq 0$, marked words $x_i, z_i \in \mathcal{A}^*$, and marked letters $b_i = [a_i,u_i] \in \mathcal{A}$ where $u_i \in A^+$ such that

$$\begin{array}{rcllllllll} w & = & x_0 b_1 & & x_1 b_2 & & x_2 & \cdots & b_m & & x_m, \text{ and} \\ w' & = & x_0 b_1 & z_1 b_1 & x_1 b_2 & z_2 b_2 & x_2 & \cdots & b_m & z_m b_m & x_m. \end{array}$$

We call b_i the context letter of the insertion $z_i b_i$. We write $w \preceq^d w'$ if $w \preceq w'$ and $|f_0(z_i b_i)| \equiv 0 \pmod{d}$ for all i. Note that $\preceq^1$ coincides with $\preceq$ and observe that $\preceq^d$ is a transitive relation.

For a dfa $M = (A, Z, \delta, s_0, F)$ and $s, t \in Z$ we write $s \xrightarrow[M]{w} t$, if $\delta(s, \overline{w}) = t$ and for all i, $\delta(s, a_1 \cdots a_i) = \delta(s, a_1 \cdots a_i u_i)$. So $s \xrightarrow[M]{w} t$ means that the marked word w leads from s to t in a way such that the labels of w are consistent with loops in M. We say that w is M-consistent, if for some $t \in Z$, $s_0 \xrightarrow[M]{w} t$ and denote by $\mathcal{B}_M$ the set of marked words that are M-consistent. Every M-consistent word has the following nice property.

Proposition 4.1 *For $w = [c_1,u_1] \cdots [c_m,u_m] \in \mathcal{B}_M$ it holds that*

$$\forall j \geq 0, \quad f_0(w) \equiv_M c_1 u_1^j \cdots c_m u_m^j.$$

We will frequently use this fact without further reference.

4.2 New Characterization of the Classes $\Sigma_1^\sigma(n)$ and $\Sigma_1^{\tau_d}(n)$

We extend Theorem 2.9 and add a characterization in terms of alternating chains on M-consistent marked words. Because we can also restrict the length of the labels u_i, denote by $\mathcal{B}_M^c$ for any $c > 0$ the set of marked words $[a_0,u_0] \cdots [a_n,u_n]$ that are M-consistent and satisfy $|u_i| \leq c$ for all $i \leq n$.

Theorem 4.2 *Let $d, n \geq 1$, M be a dfa, $c = |M|^{|M|}$, and $L = L(M) \subseteq A^+$. Then the following assertions are equivalent:*

(1) $L \in \Sigma_1^{\tau_\mathrm{d}}(n)$

(2) $f_0^{-1}(L)$ *has no 1-alternating chain of length n in* $(\mathcal{B}_M; \preceq^d)$

(3) $f_0^{-1}(L)$ *has no 1-alternating chain of length n in* $(\mathcal{B}_M^c; \preceq^d)$

Proof "(1) $\Rightarrow$ (2)" We check the contraposition. Let $f_0^{-1}(L)$ have a 1-alternating chain $(x_0, \ldots, x_n)$ in $(\mathcal{B}_M; \preceq^d)$. Since in the definition of $\preceq^d$ we suppose that the loops u_i are non-empty, just as in [GS01a] we obtain that $(f_{kd}(x_0), \ldots, f_{kd}(x_n))$ is a 1-alternating chain for L in $(A^+; \leq_k^d)$. By Lemma 2.6, $L \notin \mathcal{C}_k^d(n)$. Since k was arbitrary and $\Sigma_1^{\tau_\mathrm{d}} = \bigcup_k \mathcal{C}_k^d$, we obtain $L \notin \Sigma_1^{\tau_\mathrm{d}}(n)$.

"(2) $\Rightarrow$ (3)" Obvious.

"(3) $\Rightarrow$ (1)" Assume that $L \notin \Sigma_1^{\tau_\mathrm{d}}(n)$. Then by Theorem 2.9, $f^{-1}(L)$ has a 1-alternating chain $(x_0, \ldots, x_n)$ in $(\mathcal{A}_M^d; \preceq_M)$ with

$$x_i \in f^{-1}(L) \iff \overline{x_i} \in L \iff i \text{ is even.} \tag{2}$$

We turn structured words into marked words via $g : \mathcal{A}_M^d \to \mathcal{A}^+$ with

$$g(w_0 \boxed{u_1} w_1 \boxed{u_2} w_2 \cdots \boxed{u_m} w_m) \overset{df}{=} [w_0, \varepsilon][u_1, u_1][w_1, \varepsilon] \cdots [u_m, u_m][w_m, \varepsilon].$$

Observe that

$$\forall x \in \mathcal{A}_M,\ \overline{x} = f_0(g(x)). \tag{3}$$

By the definition of structured words, all u_i are M-idempotent and hence

$$\forall x \in \mathcal{A}_M,\ g(x) \text{ is } M\text{-consistent.} \tag{4}$$

Moreover, from the definition of $\mathcal{A}_M^d$, $\preceq_M$ and $\preceq_M^d$ we obtain

$$\forall x, y \in \mathcal{A}_M^d,\ [x \preceq_M y \Rightarrow g(x) \preceq^d g(y)]. \tag{5}$$

Consider $(w_0, \ldots, w_n)$ with $w_i \overset{df}{=} g(x_i)$. By (4) and definition of $\mathcal{A}_M$ we have $w_i \in \mathcal{B}_M^c$. By (5),(3) and (2) the sequence $(w_0, \ldots, w_n)$ is a 1-alternating chain for $f_0^{-1}(L)$ in $(\mathcal{B}_M; \preceq^d)$.

We show how to modify the sequence $(w_0, \ldots, w_n)$ to obtain a 1-alternating chain of length n for $f_0^{-1}(L)$ in $(\mathcal{B}_M^c; \preceq^d)$ (then the negation of statement 3 holds, as desired). It suffices to show that for any $u \in A^+$ there is a $v \in A^+$ such that $|v| \leq c$ and $\delta^u = \delta^v$ where $\delta^u : Z \to Z$ is the function defined by $\delta^u(z) = \delta(z, u)$. Indeed, then we simply replace any label $u \in A^+$ in the words $w_0, \ldots, w_n$ by label $v \in A^+$ with $|v| \leq c$ and $\delta^u = \delta^v$. Consequently, the resulting sequence $(w'_0, \ldots, w'_n)$ is a 1-alternating chain of length n for $f_0^{-1}(L)$ in $(\mathcal{B}_M^c; \preceq^d)$).

To obtain such a label, it suffices to show that for any $u \in A^+$, $|u| > c$, there is a $v \in A^+$ such that $|v| < |u|$ and $\delta^u = \delta^v$. So assume $|u| > c$ and consider the functions δ^{u_1}, for all prefixes $u_1 \sqsubseteq u$ of u. Since there are strictly more than c such prefixes but only c functions from Z to Z, there must exist prefixes $u_1 \sqsubset u_2 \sqsubseteq u$ with $\delta^{u_1} = \delta^{u_2}$. Let u_3 be the word satisfying $u = u_2 u_3$, then $\delta^u = \delta^{u_2}\delta^{u_3} = \delta^{u_1}\delta^{u_3} = \delta^{u_1 u_3}$, hence we can take $v = u_1 u_3$. □

In the special case $d = 1$ we have in particular an alternative to Theorem 2.8.

Theorem 4.3 *Let M be a dfa, $L = L(M) \subseteq A^+$ and $n \geq 1$. Then $L \in \Sigma_1^{\sigma}(n)$ if and only if $f_0^{-1}(L)$ has no 1-alternating chain of length n in $(\mathcal{B}_M; \preceq)$.*

4.3 An Upper Bound on d for languages in $\Sigma_1^{\tau}(n)$

For $s \in A^+$, a marked word $w = [a_0, u_0] \cdots [a_n, u_n]$ and $i \leq n$, by an *i-insertion in w* we mean a marked word of the form $[a_0, u_0] \cdots [a_i, u_i][s, \varepsilon][a_{i+1}, u_{i+1}] \cdots [a_n, u_n]$, where $s = u_i^k$ for some $k > 0$. A *variant of w* is a marked word that is obtained from w be several such (simultaneous) insertions. Obviously, such insertions do not change the property that a marked word w is M-consistent and that w has labels of bounded length.

Lemma 4.4 *If $w \in \mathcal{B}_M^c$ and w' is a variant of w then $w' \in \mathcal{B}_M^c$ and $f_0(w) \equiv_M f_0(w')$.*

The next lemma is the main technical fact of this subsection.

Lemma 4.5 *Let M be a dfa, $c = |M|^{|M|}$, $d = c!$, e a non-zero multiple of d, and $(w_0, \ldots, w_n)$ a chain in $(\mathcal{B}_M^c; \preceq^d)$. Then there exists a chain $(y_0, \ldots, y_n)$ in $(\mathcal{B}_M^c; \preceq^e)$ such that $f_0(w_i) \equiv_M f_0(y_i)$ for all $i \leq n$.*

Proof The marked words $y_0, y_1, \ldots$ are defined by induction (actually, for simplicity we explain the construction of a couple of elements in this sequence). Set $y_0 = w_0$.

Since $w_0 \preceq^d w_1$, there are representations

$$\begin{array}{rclllllllll} w_0 & = & x_0 b_1 & & x_1 b_2 & & x_2 & \cdots & b_m & & x_m \text{ and} \\ w_1 & = & x_0 b_1 & z_1 b_1 & x_1 b_2 & z_2 b_2 & x_2 & \cdots & b_m & z_m b_m & x_m \end{array}$$

where $m \geq 0$, $x_i, z_i \in \mathcal{A}^*$, $b_i = [a_i, v_i] \in \mathcal{A}$, $v_i \in A^+$ and $|f_0(z_i)| \equiv -1 \pmod{d}$ for all i. (Similar representations exist for any pair (w_i, w_{i+1}) of the chain $(w_0, \ldots, w_n)$, and we work actually with a fixed sequence of such representations.) Since d is a multiple of any $|v_j|$, we have $|f_0(z_j)| + 1 = |v_j| \cdot q_j$ for some $q_1, \ldots, q_m \geq 1$. Let $k_j = q_j(e - 1)$.

Let y_1 be obtained by insertion of $[v_j^{k_j}, \varepsilon]$ in w_1 after the first entry of b_j in the representation of w_1, for all $j = 1, \ldots, m$. Consider the induced representation of y_1. By the choice of k_j, $y_0 \preceq^e y_1$. By Lemma 4.4, $y_1 \in \mathcal{B}_M^c$ and $f_0(w_1) \equiv_M f_0(y_1)$.

In case $n > 1$, we construct y_2 as follows. The above representation of (w_0, w_1) induces an embedding of words $g_0 : w_0 \to w_1$ that sends any x_i in w_0 to the same x_i in w_1 and any b_i in w_0 to the first entry of b_i in w_1. A similar embedding $g_i : w_i \to w_{i+1}$ exists for the representation of (w_i, w_{i+1}) fixed above, for each $i < n$. Let w_2' be the marked word obtained from w_2 by the insertion of $[v_j^{k_j}, \varepsilon]$ after $g_1(g_0(b_j))$, for each $j = 1, \ldots, m$. Then $y_1 \preceq^d w_2'$ (w.r.t. the representation of (y_1, w_2') induced by the representation of (w_1, w_2)). Let y_2 be obtained from (y_1, w_2') by the same construction as y_1 was obtained from (w_0, w_1). Then $y_1 \preceq^e y_2$ and, by Lemma 4.4, $y_2 \in \mathcal{B}_M^c$ and $f_0(w_2) \equiv_M f_0(y_2)$.

In case $n > 2$ we proceed similarly to the previous paragraph. $\square$

Theorem 4.6 *In notation of Lemma 4.5, $L(M) \in \Sigma_1^{\tau}(n)$ implies $L(M) \in \Sigma_1^{\tau_d}(n)$.*

Proof We prove the contraposition. Let $L(M) \notin \Sigma_1^{\tau_d}(n)$. By Theorem 4.2, there is an 1-alternating chain $(w_0, \ldots, w_n)$ for $L = L(M)$ in $(\mathcal{B}_M^c; \preceq^d)$. We have to show that $L(M) \notin \Sigma_1^{\tau}(n)$. By [Sel04], it suffices to show that $L(M) \notin \Sigma_1^{\tau_e}(n)$ for all non-zero multiples e of d. Fix such an e. By Lemma 4.5, there is a chain $(y_0, \ldots, y_n)$ in $(\mathcal{B}_M^c; \preceq^e)$ such that $f_0(w_i) \equiv_M f_0(y_i)$ for all $i \le n$. Then $(y_0, \ldots, y_n)$ is an alternating chain for L. By Theorem 4.2, $L(M) \notin \Sigma_1^{\tau_e}(n)$. □

4.4 Characterization of the Classes $\Sigma_2^{\varrho}(n)$ for $|A| = 2$

In this subsection we prove an alternating-chain characterization for the classes of the Boolean hierarchy over Σ_2^{ϱ} for the special case $|A| = 2$. Together with the later provided algorithm this establishes the first decidability result for this hierarchy. Note that only in case $|A| = 2$ decidability of $\mathrm{BC}(\Sigma_2^{\varrho})$ [Str88] and Σ_3^{ϱ} [GS01b] is known.

As it turns out for this Boolean hierarchy, it is neither clear how to use an approach via structured words nor how to apply Theorem 2.4 with marked words. Instead, we use a different approach from [GS01b] which connects Σ_k^{ϱ} with Σ_{k-1}^{σ} and apply it to the interleaving Boolean-hierarchy classes. This translation finally allows us to exploit our new characterization of $\Sigma_1^{\tau}(n)$ (Theorem 4.3) for the construction of the needed characterization of $\Sigma_2^{\varrho}(n)$.

4.4.1 Formal Language Classes

As mentioned in the introduction, the classes Σ_k^{ϱ}, Π_k^{ϱ}, and $\mathrm{BC}(\Sigma_k^{\varrho})$ coincide with the classes of the Straubing-Thérien hierarchy [PP86], while the classes Σ_k^{σ}, Π_k^{σ}, and $\mathrm{BC}(\Sigma_k^{\sigma})$ coincide with the classes of the dot-depth hierarchy [Tho82]. We state this correspondence more precisely.

For a class $\mathcal{C}$ of languages over A let $\mathrm{Pol}(\mathcal{C})$ be the class of languages of A^* that can be written as a finite, possibly empty union of languages $L_0 a_1 L_1 \cdots a_m L_m$ with $m \ge 0$, $L_i \in \mathcal{C}$ and $a_i \in A$, and denote by $\mathrm{BC}(\mathcal{C})$ its Boolean closure (complements w.r.t. A^*). Then $\Sigma_0^{\varrho} = \{\emptyset, A^*\}$ and for $k \ge 0$ it holds that $\Sigma_{k+1}^{\varrho} = \mathrm{Pol}(\mathrm{BC}(\Sigma_k^{\varrho}))$. Analogously, let $\mathrm{Pol}^+(\mathcal{C})$ be the class of languages that of A^+ can be written as a finite, possibly empty union of languages $u_0 L_1 u_1 L_2 \cdots L_m u_m$ with $m \ge 0$, $L_i \in \mathcal{C}$ and $u_i \in A^*$ (if $m = 0$ then $u_0 \neq \varepsilon$), and denote by $\mathrm{BC}(\mathcal{C})$ the Boolean closure (complements w.r.t. A^+) of $\mathcal{C}$. Then $\Sigma_0^{\sigma} = \{\emptyset, A^+\}$ and for $k \ge 0$ it holds that $\Sigma_{k+1}^{\sigma} = \mathrm{Pol}^+(\mathrm{BC}(\Sigma_k^{\sigma}))$. Note that we take complements in $\mathrm{BC}(\mathcal{C})$ w.r.t. A^+ (resp. A^*) if $\mathcal{C}$ is a class of languages of A^+ (resp. A^*).

To ease notations, we first observe that the membership in a class of the BH over Σ_2^{ϱ} does not depend on ε. This can be easily generalized to higher levels.

Proposition 4.7 *Let A be some finite alphabet, $n \ge 1$ and $L \subseteq A^*$.*

1. *If $L \in \Sigma_2^{\varrho}(n)$ then $L \cap A^+ \in \Sigma_2^{\varrho}(n)$ and $L \cup \{\varepsilon\} \in \Sigma_2^{\varrho}(n)$.*
2. *If $L \in \Pi_2^{\varrho}(n)$ then $L \cap A^+ \in \Pi_2^{\varrho}(n)$ and $L \cup \{\varepsilon\} \in \Pi_2^{\varrho}(n)$.*

Proof Note that $A^+ = \bigcup_{a \in A} A^* a A^* \in \Sigma_1^{\varrho} \subseteq \mathrm{BC}(\Sigma_1^{\varrho})$ and $\{\varepsilon\} \in \Pi_1^{\varrho} \subseteq \mathrm{BC}(\Sigma_1^{\varrho})$. Since $\Sigma_2^{\varrho} = \Sigma_2^{\varrho}(1)$ and $\Pi_2^{\varrho} = \Pi_2^{\varrho}(1)$ are closed under union and intersection the proposition is true for

$n = 1$. We argue for the first statement by induction on n. If $n+1$ is even, then $L \in \Sigma_2^\varrho(n+1)$ can be written as $L = L_1 \cap L_2$ with $L_1 \in \Sigma_2^\varrho(n)$ and $L_2 \in \Pi_2^\varrho$ [CGH+88]. So also $L \cap A^+ = L_1 \cap L_2 \cap A^+ \in \Sigma_2^\varrho(n+1)$, since $L_2 \cap A^+ \in \Pi_2^\varrho$. Moreover, $L \cup \{\varepsilon\} = (L_1 \cup \{\varepsilon\}) \cap (L_2 \cup \{\varepsilon\}) \in \Sigma_2^\varrho(n+1)$, since $L_1 \cup \{\varepsilon\} \in \Sigma_2^\varrho(n)$ by hypothesis and $L_2 \cup \{\varepsilon\} \in \Pi_2^\varrho$. If $n+1$ is odd, then $L \in \Sigma_2^\varrho(n+1)$ can be written as $L = L_1 \cup L_2$ with $L_1 \in \Sigma_2^\varrho(n)$ and $L_2 \in \Sigma_2^\varrho$ [CGH+88]. So also $L \cap A^+ = (L_1 \cap A^+) \cup (L_2 \cap A^+) \in \Sigma_2^\varrho(n+1)$, since $L_1 \cap A^+ \in \Sigma_2^\varrho(n)$ by hypothesis and $L_2 \cap A^+ \in \Sigma_2^\varrho$. Moreover, $L \cup \{\varepsilon\} = L_1 \cup L_2 \cup \{\varepsilon\} \in \Sigma_2^\varrho(n+1)$ since $L_2 \cup \{\varepsilon\} \in \Sigma_2^\varrho$. The second statement can be shown with dual arguments. □

4.4.2 *A*-Factorizations

We fix $A = \{a, b\}$ for the remainder of this section and recall some definitions from [GS01b]. Let $A_r^a \overset{df}{=} \{a_1, \ldots, a_r\}$, $A_r^b \overset{df}{=} \{b_1, \ldots, b_r\}$ and $A_r \overset{df}{=} A_r^a \cup A_r^b$ for $r \geq 1$. Every word $x \in A^*$ can be decomposed into maximal blocks of equal letters, i.e., $x = x_1 x_2 \cdots x_k$ for some $k \geq 1$ and for words x_i of maximal length such that $x_i \in \{a\}^+ \cup \{b\}^+$. We call this the A-factorization of x and observe that it is unique due to the maximality condition. We say that a letter $c \in A_r$ is of type a if $c \in A_r^a$, and it is of type b if $c \in A_r^b$. For every $r \geq 1$ we define a mapping $h_r : A^* \to (A_r)^*$ which maps every x_i to a single letter from A_r. More precisely, let $r \geq 1$, $x \in A^*$ and let $x = x_1 x_2 \cdots x_k$ for some $k \geq 1$ be the A-factorization of x. Then $h_r(x) \overset{df}{=} c_1 c_2 \cdots c_k \in (A_r)^*$ where

$$c_i \overset{df}{=} \begin{cases} a_{\min\{l,r\}} & : \quad \text{if } x_i = a^l \text{ for some } l \geq 1 \\ b_{\min\{l,r\}} & : \quad \text{if } x_i = b^l \text{ for some } l \geq 1. \end{cases}$$

Moreover, for $L \subseteq A^*$ and $L' \subseteq (A_r)^*$ we define $h_r(L) \overset{df}{=} \bigcup_{x \in L} \{h_r(x)\}$ and $h_r^{-1}(L') \overset{df}{=} \{\, x \in A^* \mid h_r(x) \in L' \,\}$. It is important to notice that not all words from $(A_r)^*$ can appear in the range of h_r. The maximality condition in A-factorizations ensures that the type of letters in every word $h_r(x)$ alternates between a and b. We call such words well-formed and set $W \overset{df}{=} h_r(A^*)$.

Proposition 4.8 *It holds that* $W = (A_r)^* \setminus \big((A_r)^* A_r^a A_r^a (A_r)^* \ \cup \ (A_r)^* A_r^b A_r^b (A_r)^*\big)$.

In particular, the levels of the BH over Σ_1^σ and the BH over Σ_2^ϱ are closely related via h_r^{-1}.

Lemma 4.9 *Let* $A = \{a, b\}$, $n \geq 1$ *and* $L \subseteq (A_r)^+$ *for some* $r \geq 1$.

1. *If* $L \in \Sigma_1^\sigma(n)$ *then* $h_r^{-1}(L) \in \Sigma_2^\varrho(n)$.
2. *If* $L \in \Pi_1^\sigma(n)$ *then* $h_r^{-1}(L) \in \Pi_2^\varrho(n)$.

Proof Let $L \subseteq (A_r)^+$ for some $r \geq 1$ and let $n \geq 1$. To see the first statement suppose $L \in \Sigma_1^\sigma(n)$. So there exist $L_1, L_2, \ldots, L_n \in \Sigma_1^\sigma$ with $L_1 \supseteq L_2 \supseteq \cdots \supseteq L_n$ such that $L = L_1 - (L_2 - (\ldots - L_n))$. We set $L_i' \overset{df}{=} h_r^{-1}(L_i)$ and claim that

1. $L_i' \in \Sigma_2^\varrho$,
2. $L_1' \supseteq L_2' \supseteq \cdots \supseteq L_n'$, and
3. $h_r^{-1}(L) = L_1' - (L_2' - (\ldots - L_n'))$.

To see the first claim observe that

$$L'_i = h_r^{-1}((L_i \cap W) \cup (L_i \cap \overline{W})) = h_r^{-1}(L_i \cap W) \cup h_r^{-1}(L_i \cap \overline{W}) = h_r^{-1}(L_i \cap W),$$

since $h_r^{-1}(L_i \cap \overline{W}) = \emptyset$. In [GS01b, Claim 5] it is shown that $h_r^{-1}(L_i \cap W) \in \Sigma_2^{\varrho}$. Claim 2 is immediate since $L_1 \supseteq L_2 \supseteq \cdots \supseteq L_n$. For claim 3, let $x \in h_r^{-1}(L)$, so $h_r(x) \in L \subseteq L_1$ and hence $x \in h_r^{-1}(L_1) = L'_1$. To complete the inclusion from left to right observe that $h_r(x) \in L_{2j}$ implies $h_r(x) \in L_{2j+1}$, since $h_r(x) \in L$, so $x \in L'_{2j}$ implies $x \in L'_{2j+1}$. For the reverse inclusion let $x \in A^* \backslash h_r^{-1}(L)$. If $x \notin L'_1$ we are done. Otherwise we see that $x \in L'_{2j+1}$ implies $x \in L'_{2j+2}$ with the same argument as before.

For the second statement of the lemma let $L \in \Pi_1^{\sigma}(n)$. Then $(A_r)^+ \backslash L \in \Sigma_1^{\sigma}(n)$ and by the first statement of the lemma we have $h_r^{-1}((A_r)^+ \backslash L) \in \Sigma_2^{\varrho}(n)$. Since $\Pi_2^{\varrho}(n)$ is closed under intersection with A^+ (Proposition 4.7) it holds that

$$L' \overset{df}{=} A^+ \cap \ \left(A^* \backslash h_r^{-1}((A_r)^+ \backslash L)\right) \quad \in \Pi_2^{\varrho}(n).$$

We claim that $h_r^{-1}(L) = L'$. For the inclusion from left to right observe that every $x \in h_r^{-1}(L)$ is non-empty and that $h_r(x) \in L \Rightarrow h_r(x) \notin (A_r)^+ \backslash L$, so $x \notin h_r^{-1}((A_r)^+ \backslash L)$. Conversely, let x be a non-empty word from $A^* \backslash h_r^{-1}((A_r)^+ \backslash L)$ and assume that $h_r(x) \notin L$. Then $h_r(x) \in (A_r)^+ \backslash L$, so $x \in h_r^{-1}((A_r)^+ \backslash L)$ which is a contradiction. □

Note that also this lemma can be generalized to Boolean hierarchies over higher levels.

4.4.3 Alternating-chain Characterization

In the following we will make use of a certain padding operation.

Lemma 4.10 *Let $L(M) \subseteq A^*$ for some dfa M and let $(w_0, \ldots, w_n)$ be a chain in $(\mathcal{B}_M, \preceq)$. For every dfa M' with $L(M') \subseteq A^*$ there exists a chain $(w'_0, \ldots, w'_n)$ in $(\mathcal{B}_{M'}, \preceq)$ such that $f_0(w_i) \equiv_M f_0(w'_i)$.*

Proof Let $L(M') \subseteq A^*$ for some dfa M'. To obtain w'_i from w_i let $r \overset{df}{=} |M'|$ and substitute each letter $[a, u]$ of w_i with $u \neq \varepsilon$ by $[a, \varepsilon][u^{r!}, u^{r!}]$. Since $w_i \in \mathcal{B}_M$ we have $f_0(w_i) \equiv_M f_0(w'_i)$. To see that $w'_i \preceq w'_{i+1}$ holds, note that it suffices to look at each insertion separately. So assume that $w_i = xby$ and $w_{i+1} = xbzby$ with $x, y, z \in \mathcal{A}^*$ and context letter $b = [a, u]$. Then

$$\begin{array}{rclccccc} w'_i & = & x' & [a,\varepsilon][u^{r!},u^{r!}] & & & y' \\ \preceq \ w'_{i+1} & = & x' & [a,\varepsilon][u^{r!},u^{r!}] & z' & [a,\varepsilon][u^{r!},u^{r!}] & y'. \end{array}$$

Finally observe that every w'_i is M'-consistent, since $u^{r!}$ is M'-idempotent. □

Now we give a characterization in terms of marked words that obey an additional constraint. For $u \in A^*$ let $\alpha(u)$ be the set of letters in u. We say that a marked word $w = [c_1, u_1] \cdots [c_m, u_m]$ satisfies the *alphabet condition* if for all $u_i \neq \epsilon$ it holds that $\alpha(u_i) = A$.

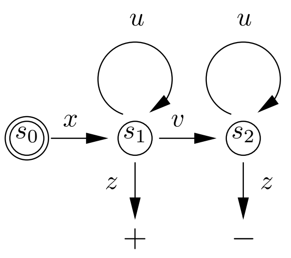


Figure 2: A dfa $M = (A, Z, \delta, s_0, F)$ has pattern P if and only if there exist $x, u, v, z \in A^*$ and $s_1, s_2 \in Z$ such that $s_1 = \delta(s_0, x) = \delta(s_0, xu)$, $s_2 = \delta(s_0, xv) = \delta(s_0, xvu)$, $\delta(s_1, z) \in F$, $\delta(s_2, z) \notin F$, and $\alpha(v) \subseteq \alpha(u)$.

Theorem 4.11 *Let $A = \{a, b\}$, $n \geq 1$ and let $L(M) \subseteq A^*$ for some dfa M such that $L = L(M)$ is a star-free language. Then $L \in \Sigma_2^{\varrho}(n)$ if and only if $f_0^{-1}(L)$ has no 1-alternating chain $(w_0, \ldots, w_n)$ in $(\mathcal{B}_M; \preceq)$ such that all w_i satisfy the alphabet condition.*

Proof "$\Rightarrow$": Let $L = L_1 - (L_2 - (\ldots - L_n))$ for some $L_i \in \Sigma_2^{\varrho}$ with $L_1 \supseteq L_2 \supseteq \cdots \supseteq L_n$. If $M_1, \ldots, M_n$ denote the minimal dfa's that accept the languages $L_1, \ldots, L_n$, respectively, we define $e = (\max_{1 \leq i \leq n} |M_i|)!$ and set $g(w) = f_e(w)$ for notational convenience. Assume to the contrary that $(w_0, \ldots, w_n)$ is a 1-alternating chain for $f_0^{-1}(L)$ in $(\mathcal{B}_M; \preceq)$ such that all w_i satisfy the alphabet condition. We first show that

$$\forall 1 \leq k, l \leq n, \ \ [g(w_{k-1}) \in L_l \Rightarrow g(w_k) \in L_l]. \tag{6}$$

Assume that for some k, l this is not true and let

$$\begin{array}{rcllllll} w_{k-1} & = & x_0 c_1 & & x_1 & \cdots & x_{m-1} c_m & & x_m \text{ and} \\ w_k & = & x_0 c_1 & z_1 c_1 & x_1 & \cdots & x_{m-1} c_m & z_m c_m & x_m \end{array}$$

for suitable $m \geq 0$, $x_i, z_i \in \mathcal{A}^*$ and $c_i = [d_i, u_i] \in \mathcal{A}$ where $u_i \in A^+$ and $\alpha(u_i) = A$. So we have

$$\begin{array}{rcllllll} g(w_{k-1}) & = & g(x_0) d_1 u_1^e & & g(x_1) & \cdots & g(x_{m-1}) d_m u_m^e & & g(x_m) \text{ and} \\ g(w_k) & = & g(x_0) d_1 u_1^e & g(z_1) d_1 u_1^e & g(x_1) & \cdots & g(x_{m-1}) d_m u_m^e & g(z_m) d_m u_m^e & g(x_m). \end{array}$$

For $1 \leq i \leq m$ let v_i be the word that is obtained from $g(w_k)$ by deleting the factors $g(z_j) d_j u_j^e$ for all $j > i$. Note that $v_0 = g(w_{k-1}) \in L_l$ and by assumption, $v_m = g(w_k) \notin L_l$. So there must exist some j with $1 \leq j \leq m-1$ such that $v_j \in L_l$ but $v_{j+1} \notin L_l$.

Observe that for suitable words $y, z \in \mathcal{A}^*$ we have

$$\begin{array}{rclll} v_j & = & y \ d_{j+1} u_{j+1}^e & & z \\ v_{j+1} & = & \underbrace{y \ d_{j+1} u_{j+1}^e}_{x \overset{df}{=}} & \underbrace{g(z_{j+1}) d_{j+1} u_{j+1}^e}_{v \overset{df}{=}} & z. \end{array}$$

Since $u \overset{df}{=} u_{j+1}^e$ is M_j-idempotent and $\alpha(u) = A$ we see that M_j has pattern P in its transition graph, which is forbidden for languages in Σ_2^{ϱ} (see Figure 2, [PW97]). Hence (6) holds.

For $0 \leq k \leq n$, let $r_k = |\{1 \leq l \leq n \mid g(w_k) \in L_l\}|$. Since all w_i are M-consistent we have $f_0(w_i) \equiv_M g(w_i)$. From $f_0(w_0) \in L$ we get $g(w_0) \in L \subseteq L_1$ and so $r_0 \geq 1$. It follows from (6) that

$$1 \leq r_0 \leq r_1 \leq \cdots \leq r_n \leq n.$$

In particular, there must be some k with $r_{k-1} = r_k$. By (6) it holds that $(g(w_{k-1}) \in L_l \Rightarrow g(w_k) \in L_l)$ for all l. So $(g(w_{k-1}) \in L \Leftrightarrow g(w_k) \in L)$ and hence due to M-consistency $(f_0(w_{k-1}) \in L \Leftrightarrow f_0(w_k) \in L)$. The latter contradicts our assumption that $(w_0, \ldots, w_n)$ alternates.

" $\Leftarrow$ ": We need to show that if $L \notin \Sigma_2^{\varrho}(n)$, then $f_0^{-1}(L)$ has a 1-alternating chain $(w_0, \ldots, w_n)$ in $(\mathcal{B}_M; \preceq)$ such that all w_i satisfy the alphabet condition. We may assume that $L = L(M)$ where $M = (A, Z, \delta, s_0, F)$ is a minimal dfa. To see this, suppose for the moment that we have already shown the existence of a 1-alternating chain $(w_0, \ldots, w_n)$ in $(\mathcal{B}_M; \preceq)$ with alphabet condition. With Lemma 4.10 we obtain marked words $w_0', \ldots, w_n' \in \mathcal{B}_{M'}$ having the same properties for every dfa M' with $L(M') = L$. Note that the padding operation in this lemma does not violate the alphabet condition.

Since L is a star-free language we know that M is a permutation-free dfa [Sch65, MP71]. So for $r \overset{df}{=} |M|$ it holds that $\delta(s, c^r) = \delta(s, c^{r+1})$ for letters $c \in A$ and all $s \in Z$. Hence words and their A-factorizations are related as follows:

$$\forall x, y \in A^*, [h_r(x) = h_r(y) \Rightarrow x \equiv_M y]. \tag{7}$$

We distinguish two cases.

Case 1: $n \geq 1$ is even. In light of Proposition 4.7 we may assume w.l.o.g. that $\varepsilon \notin L$. Since for all marked words w in a chain $f_0(w) \neq \varepsilon$, Lemma 4.10 provides M'-consistent words if M' is some dfa with $L(M') = L \cup \{\varepsilon\}$.

Now let $L' \overset{df}{=} h_r(L) \subseteq (A_r)^+$ and observe that $h_r^{-1}(L') = L$. The inclusion from right to left is immediate, for the reverse inclusion let $x \in h_r^{-1}(L')$ so $h_r(x) \in h_r(L)$, i.e., there is some $y \in L$ with $h_r(y) = h_r(x)$ and from (7) we get $x \in L$. By Lemma 4.9 it holds that

$$L = h_r^{-1}(L') \notin \Sigma_2^{\varrho}(n) \Rightarrow L' \notin \Sigma_1^{\sigma}(n).$$

So from our assumption it follows that $L' \notin \Sigma_1^{\sigma}(n)$.

Recall from [GS01b, Lemma 1] that $h_r(L)$ is a regular language: If M is given one can easily construct some dfa which behaves on input a_j (or b_j) for $1 \leq j \leq r$ in the same way as M on input a^j (resp., b^j) and which additionally rejects all non-well-formed inputs from $(A_r)^+$. So let M' be some dfa with $L(M') = L'$. Then by Theorem 4.3 there is a 1-alternating chain $(w_0, \ldots, w_n)$ for L' in $((\mathcal{B}_r)_{M'}; \preceq)$. Here $(\mathcal{B}_r)_{M'}$ denotes the set of all M'-consistent marked words over $\mathcal{A}_r$.

Suppose $w \in (\mathcal{B}_r)^*_{M'}$ is M'-consistent with $f_0(w) \in L' \subseteq W$ and let $w' \preceq w$. Then also $f_0(w') \in W$ because every two consecutive letters of $f_0(w')$ also appear in $f_0(w)$ (recall that the context letter of an insertion is repeated at the end of the insertion). Since n is even, we have $f_0(w_n) \in L'$ and hence

$$\forall 0 \leq k \leq n, \ \ [f_0(w_k) \in W]. \tag{8}$$

Moreover, $f_i(w_n) \in L' \subseteq W$ for all $i \geq 0$. So every label $u \neq \varepsilon$ in each letter $[c, u]$ of w_n consists of letters of both types a and b. Since all marked letters of any w_k also appear in w_n we have

$$\forall 0 \leq k \leq n \ \forall \text{ letters } [c, u] \text{ of } w_k, \ \ [u \neq \varepsilon \Rightarrow (u \text{ has letters of both types } a \text{ and } b)]. \tag{9}$$

$$\begin{tikzcd}
\text{basic alphabets} & A = \{a, b\} & A_r = \{a_1, \dots a_r, b_1, \dots, b_r\} \\
 & A^* \arrow[r, shift left, "h_r"] & (A_r)^* \arrow[l, shift left, "g_r"] \\
\text{marked words} & \mathcal{A}^* \arrow[u, "f_0"] & (\mathcal{A}_r)^* \arrow[l, "g_r"] \arrow[u, "f_0"'] \\
\text{M-consistent marked words} & \mathcal{B}_M & (\mathcal{B}_r)_M
\end{tikzcd}$$

Figure 3: Overview of alphabets and mappings used in the proof of Theorem 4.11.

Next we define a mapping g_r from $(A_r)^*$ back to A^*. For $a_j \in A_r^a$ (resp., $b_j \in A_r^b$) with $1 \le j \le r$ let $g_r(a_j) \overset{df}{=} a^j$ (resp., $g_r(b_j) \overset{df}{=} b^j$), and for $x_i \in A_r$ let $g_r(x_1 x_2 \cdots x_m) \overset{df}{=} g_r(x_1) g_r(x_2) \cdots g_r(x_m)$. Then it holds that

$$\forall x \in W,\ \ [h_r(g_r(x)) = x]. \tag{10}$$

We extend g_r to marked alphabets and define for $[c, u] \in \mathcal{A}_r$ that $g_r([c, u]) \overset{df}{=} [g_r(c), g_r(u)] \in \mathcal{A}^*$ and $g_r([c_1, u_1] \cdots [c_m, u_m]) \overset{df}{=} g_r([c_1, u_1]) \cdots g_r([c_m, u_m])$. Note that for all $w \in (\mathcal{A}_r)^+$ it holds that $f_0(g_r(w)) = g_r(f_0(w))$. See Figure 3 for an overview of alphabets and mappings used in this proof.

Let $v_k \overset{df}{=} g_r(w_k)$ for $0 \le k \le n$. We claim that $(v_0, \dots, v_n)$ is a 1-alternating chain with alphabet condition for $f_0^{-1}(L)$ in $(\mathcal{A}^*, \preceq)$, i.e.,

1. every v_k satisfies the alphabet condition,
2. $f_0(v_k) \in L$ if and only if k is even, and
3. $v_0 \preceq v_1 \preceq \cdots \preceq v_n$.

The first statement is immediate from (9) together with the definition of g_r. To see the second statement we show that

$$\forall 0 \le k \le n,\ \ [f_0(w_k) \in L' \ \Leftrightarrow\ f_0(v_k) \in L]. \tag{11}$$

If $f_0(w_k) \in L'$, then there exists some $x \in L$ such that $h_r(x) = f_0(w_k)$. By (8) and (10) we have

$$h_r(f_0(v_k)) = h_r(f_0(g_r(w_k))) = h_r(g_r(f_0(w_k))) = f_0(w_k) = h_r(x).$$

It follows from (7) that also $f_0(v_k) \in L$. Conversely, if $f_0(v_k) = f_0(g_r(w_k)) = g_r(f_0(w_k)) \in L$, then $f_0(w_k) = h_r(g_r(f_0(w_k))) \in h_r(L) = L'$. Together, this shows (11).

It remains to argue that the third statement holds. Again, we consider each insertion in every $w_{k-1} \preceq w_k$ for $1 \le k \le n$ separately. So assume that $w_{k-1} = xdy$ and $w_k = xdzdy$ with $x, y, z \in (\mathcal{A}_r)^*$ and $d = [c, u] \in \mathcal{A}_r$. W.l.o.g. let $c = a_j$. Then v_{k-1} and v_k can be written as follows:

$$\begin{array}{lclllll}
v_{k-1} & = & g_r(x)\ \ g_r(d) & & & & g_r(y) \\
 & = & g_r(x)\ \ [a,\varepsilon]\cdots[a,\varepsilon] & [a,g_r(u)] & & & g_r(y) \\
v_k & = & \underbrace{g_r(x)\ \ [a,\varepsilon]\cdots[a,\varepsilon]}_{x' \overset{\mathrm{df}}{=}} & \underbrace{[a,g_r(u)]}_{d' \overset{\mathrm{df}}{=}} & \underbrace{g_r(z)\ \ [a,\varepsilon]\cdots[a,\varepsilon]}_{z' \overset{\mathrm{df}}{=}} & \underbrace{[a,g_r(u)]}_{d'} & \underbrace{g_r(y)}_{y' \overset{\mathrm{df}}{=}}
\end{array}$$

So $v_{k-1} = x'd'y' \preceq x'd'z'd'y' = v_k$ with $x', y', z' \in \mathcal{A}^*$ and context letter $d' \in \mathcal{A}$.

Note that the marked words $v_0, \ldots, v_n$ need not to be M-consistent yet. To finally obtain M-consistent words $v_0', \ldots, v_n'$ with the same properties 1.-3. as above, we start the reasoning with a padded version $(w_0', \ldots, w_n')$ of the chain $(w_0, \ldots, w_n)$ provided by Theorem 4.3. As in Lemma 4.10 we substitute in w_k each non-empty label u by $u^{r!}$ in a way such that each such label is preceded in $f_0(w_k)$ by the M-idempotent factor $u^{r!}$. More precisely, each letter $[c, u]$ where $u \neq \varepsilon$ is mapped to $[c \cdot u^{r!}, u^{r!}]$. Due to the homomorphic definition of g_r we get consistency with M after mapping with g_r. As observed before, this operation does not violate the alphabet condition.

Case 2: $n \geq 1$ is odd. Again, we may assume w.l.o.g. that $\varepsilon \in L$. Now let $\overline{L} = A^*\backslash L$ and $L' \overset{\mathrm{df}}{=} h_r(\overline{L}) \subseteq (A_r)^+$ and observe that $h_r^{-1}(L') = \overline{L}$ as before. By Lemma 4.9, we have $L' \in \Pi_1^\sigma(n) \Rightarrow \overline{L} = h_r^{-1}(L') \in \Pi_2^\varrho(n)$ so it holds that

$$L \notin \Sigma_2^\varrho(n) \Rightarrow \overline{L} \notin \Pi_2^\varrho(n) \Rightarrow L' \notin \Pi_1^\sigma(n).$$

Let M' be some dfa with $L(M') = L'$. By Theorem 4.3 and duality arguments we obtain a 0-alternating chain $(w_0, \ldots, w_n)$ for L' in $((\mathcal{B}_r)_{M'}; \preceq)$. Since n is odd we have again $f_0(w_n) \in L'$, so we can argue exactly as in case 1 to get a 0-alternating chain $(v_0', \ldots, v_n')$ with alphabet condition for $\overline{L}$ in $(\mathcal{B}_M; \preceq)$. This is a 1-alternating chain with respect to L. $\square$

```
// Algorithm $L_n$ on input $(M,S)$ where $M = (A,Z,\delta,z_0,F)$ is a
   deterministic, finite automaton and S is a set of states of M.
   The algorithm tests whether the states in S share a nonempty loop.
if $|S| > 2^{n+2}$ then reject
let $s_1,\ldots,s_k$ be all states in S and let $t_1 = s_1,\ldots,t_k = s_k$
do
   nondeterministically choose $a \in A$
   for $i = 1$ to $k$
      $t_i = \delta(t_i, a)$
   next i
until $\forall i, s_i = t_i$
accept
```

Figure 4: Algorithm L_n for $n \geq 1$.

5 Decidability and Complexity

In section 4, we developed new alternating-chain characterizations for the classes $\Sigma_1^{\sigma}(n)$, $\Sigma_1^{\tau_d}(n)$, and $\Sigma_2^{\varrho}(n)$ for $|A| = 2$ (Theorems 4.3, 4.2, and 4.11). We now apply these characterizations and design efficient algorithms for testing the membership in these classes. As corollaries we obtain new decidability results: The classes $\Sigma_1^{\tau}(n)$ and $\Sigma_2^{\varrho}(n)$ for $|A| = 2$ are decidable.

We start with the characterization given in Theorem 4.3 and design a nondeterministic, logarithmic-space membership test for $\Sigma_1^{\sigma}(n)$. For this end we construct families of algorithms $\{L_n\}_{n\geq 1}$, $\{R_k\}_{k\geq 0}$, and $\{Q_k\}_{k\geq 0}$ which are shown in the Figures 4–6. Before we formally describe the behavior of these algorithms, we first give some intuition.

Our algorithms have a nontrivial recursion structure. More precisely, Q_k calls R_k and L_k. R_k calls Q_{k-1} and two instances of R_{k-1}. L_n is an elementary procedure that makes no further calls. So the recursion structure of these algorithms looks as follows.

$$\begin{tikzcd}
\vdots & \vdots & \vdots \\
R_3 \arrow[d, shift left] \arrow[d, shift right] \arrow[dr] & Q_3 \arrow[l] \arrow[r] & L_3 \\
R_2 \arrow[d, shift left] \arrow[d, shift right] \arrow[dr] & Q_2 \arrow[l] \arrow[r] & L_2 \\
R_1 \arrow[d, shift left] \arrow[d, shift right] \arrow[dr] & Q_1 \arrow[l] \arrow[r] & L_1 \\
R_0 & Q_0 &
\end{tikzcd}$$

All paths in this picture are finite and hence the number of recursive calls is finite. Later (in the proof of Theorem 5.4) we will inductively show that each single algorithm works in nondeterministic logarithmic space.

```
0   // Algorithm $R_k$ on input $(M, a, S, s_1, \ldots, s_k, t_1, \ldots, t_k)$ where $M = (A, Z, \delta, z_0, F)$
       is a deterministic, finite automaton, $a \in A$, $S$ is a set of states
       of $M$, and $s_i, t_i$ are states of $M$.  (see Lemma 5.1)
1   if $k = 0$ then accept
2   if $|S| > 2^{k+2} - 2$ then reject
3   if $\{\delta(s_1, a), \ldots, \delta(s_k, a), t_1, \ldots, t_k\} \not\subseteq S$ then reject
4   nondeterministically choose a bit d
5   if $d = 0$ then
6      // insertion between $s_1$ and $t_1$
7      nondeterministically choose $S_1, S_2 \subseteq S$ such that $|S_1|, |S_2| \leq 2^{k+1} - 2$
8      nondeterministically choose $s'_1, \ldots, s'_k \in S_1$ and $t'_1, \ldots, t'_k \in S_2$
9      if $\delta(s_1, a) \neq s'_1$ or $\delta(t'_1, a) \neq t_1$ then reject
10     if $R_{k-1}(M, a, S_1, s_2, \ldots, s_k, s'_2, \ldots, s'_k)$ rejects then reject
11     if $R_{k-1}(M, a, S_2, t'_2, \ldots, t'_k, t_2, \ldots, t_k)$ rejects then reject
12     if $Q_{k-1}(M, s'_1, \ldots, s'_k, t'_1, \ldots, t'_k)$ rejects then reject
13  else
14     // no insertion between $s_1$ and $t_1$
15     if $\delta(s_1, a) \neq t_1$ then reject
16     nondeterministically choose $S' \subseteq S$ such that $|S'| \leq 2^{k+1} - 2$
17     if $R_{k-1}(M, a, S', s_2, \ldots, s_k, t_2, \ldots, t_k)$ rejects then reject
18  endif
19  accept
```

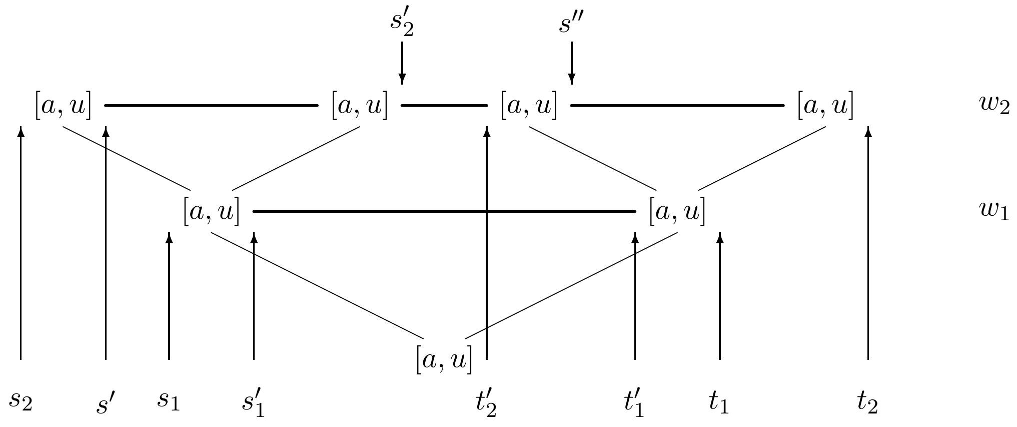
Figure 5: Algorithm R_k for $k \geq 0$.

```
0   // Algorithm $Q_k$ on input $(M, s_0, \ldots, s_k, t_0, \ldots, t_k)$ where $M = (A, Z, \delta, z_0, F)$
       is a deterministic, finite automaton and $s_i, t_i$ are states of $M$.
       (see Lemma 5.1)
1   $s'_0 = s_0, s'_1 = s_1, \ldots, s'_k = s_k$
2   do
3      nondeterministically choose $S \subseteq Z$ such that $|S| \leq 2^{k+2} - 2$
4      if $S = \emptyset$ then
5         // adding a letter marked with the empty word
6         nondeterministically choose $a \in A$
7         for $i = 0$ to $k$
8            $s'_i = \delta(s'_i, a)$
9         next i
10     else
11        // adding a letter marked with a nonempty word
12        if $L_k(M, S)$ rejects then reject
13        nondeterministically choose $a \in A$ and $t'_0, \ldots, t'_k \in S$
14        if $\delta(s'_0, a) \neq t'_0$ then reject
15        if $R_k(M, a, S, s'_1, \ldots, s'_k, t'_1, \ldots, t'_k)$ rejects then reject
16     endif
17     $s'_0 = t'_0, s'_1 = t'_1, \ldots, s'_k = t'_k$
18  until $\forall i, s'_i = t_i$
19  accept
```

Figure 6: Algorithm Q_k for $k \geq 0$.

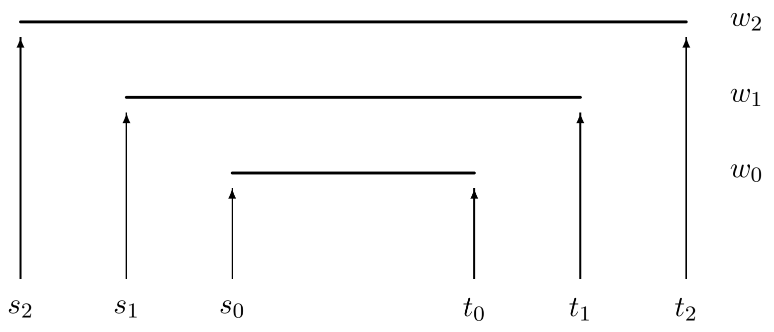
Intuition of the algorithm L_n: Let M be a deterministic finite automaton M and S be a set of at most 2^{n+2} states. The computation $L_k(M, S)$ tests whether the states in S share a nonempty loop. This means it tests whether there exists $u \in A^*$ such that for all $s \in S$, $\delta(s, u) = s$.

Intuition of the algorithm R_k: We consider the computation $R_2(M, a, S, s_1, s_2, t_1, t_2)$ for a deterministic finite automaton $M = (A, Z, \delta, z_0, F)$, a letter a, some $S \subseteq Z$ and states $s_1, s_2, t_1, t_2 \in Z$. This computation tests whether there exists a nonempty word u and a chain of marked words $[a, u] \preceq w_1 \preceq w_2$ such that $s_1 \xrightarrow[M]{w_1} t_1$ and $s_2 \xrightarrow[M]{w_2} t_2$. So we look for the following situation where the arrows mark the positions in the words where particular states are demanded.



Note that the initial letter $[a, u]$ propagates in a binary-tree-like fashion to the words w_1 and w_2. So our algorithm has to make sure that after each letter $[a, u]$, the automaton reaches a state that has a loop with label u. However, we cannot simply store the word u in logarithmic space. So we pursue a different strategy and guess in before (i.e., in algorithm Q_2) all states that are reached after reading one of the shown letters $[a, u]$. These states are collected in the set S, i.e., in our example, $S = \{s'_1, s'_2, t_1, t_2, s', s''\}$. With help of the algorithm L_2 (which is called by Q_2) we can test whether all states in S share some nonempty loop u, i.e., for all $s \in S$, $\delta(s, u) = s$. Therefore, if the algorithm R_2 makes sure that each state that is reached after reading one of the shown $[a, u]$ belongs to S, then all states reached after some $[a, u]$ have a loop with label u. Note that the size of S grows exponentially in the length of the considered chain $[a, u] \preceq w_1 \preceq w_2$, but this length is constant.

Intuition of the algorithm Q_k: We consider $Q_2(M, s_0, s_1, s_2, t_0, t_1, t_2)$ for a deterministic finite automaton $M = (A, Z, \delta, z_0, F)$ and states $s_0, s_1, s_2, t_0, t_1, t_2 \in Z$. This computation tests whether there exists a chain of nonempty marked words $w_0 \preceq w_1 \preceq w_2$ such that $s_0 \xrightarrow[M]{w_0} t_0$, $s_1 \xrightarrow[M]{w_1} t_1$, and $s_2 \xrightarrow[M]{w_2} t_2$. This is illustrated in the following figure.



Now we formally describe the behavior of the algorithms R_k and Q_k.

Lemma 5.1 *Let $k \geq 0$, $M = (A, Z, \delta, z_0, F)$ be a deterministic, finite automaton, $a \in A$, $u \in A^+$, and $s_0, \dots, s_k, t_0, \dots, t_k \in Z$.*

$$\begin{array}{l}\exists S \subseteq Z \text{ s.t. } |S| \leq 2^{k+2}-2, \forall s \in S[\delta(s,u) = s], \\ \text{and } R_k(M, a, S, s_1, \dots, s_k, t_1, \dots, t_k)\end{array} \Leftrightarrow \begin{array}{l}\exists w_1, \dots, w_k \in \mathcal{A}^+, \\ \Big([a,u] \preceq w_1 \preceq \cdots \preceq w_k \wedge \underset{i \in [1,k]}{\forall} [s_i \xrightarrow[M]{w_i} t_i]\Big)\end{array} \tag{12}$$

$$Q_k(M, s_0, \dots, s_k, t_0, \dots, t_k) \Leftrightarrow \exists w_0, \dots, w_k \in \mathcal{A}^+ \Big(w_0 \preceq \cdots \preceq w_k \wedge \underset{i \in [0,k]}{\forall} [s_i \xrightarrow[M]{w_i} t_i]\Big) \tag{13}$$

Proof We prove the equivalences (12) and (13) simultaneously by induction on k. First observe that for $k = 0$, both sides of (12) are always true. So (12) holds for $k = 0$.

Claim 5.2 *If (12) holds for some k, then (13) holds for this k.*

Proof So assume that (12) holds for a particular k. We prove the equivalence (13) for this k.

"⇒" Consider an accepting path of $Q_k(M, s_0, \dots, s_k, t_0, \dots, t_k)$ and let $n \geq 1$ be the number of passes through the loop. Fix some j such that $1 \leq j \leq n$. Below we explain how the j-th pass can be considered as the choice of particular words $w_{j,0}, \dots, w_{j,k} \in \mathcal{A}^+$. For this let s_i^b (resp., s_i^e) denote the value of s_i at the beginning (resp., end) of the j-th pass.

If $S = \emptyset$, then choose $w_{j,0} = \cdots = w_{j,k} = [a, \varepsilon]$ where a is the letter that was chosen in line 6. Note that in this case, by the lines 7–9, for all $i \in [0,k]$, $s_i^b \xrightarrow[M]{w_{j,i}} s_i^e$.

If $S \neq \emptyset$, then by line 12, there exists $u \in A^+$ such that all states in S share the loop u. (Remember that $L_k(M, S)$ simply tests whether the states in S share a nonempty loop.) Let $w_{j,0} = [a, u]$ where a is the letter that was chosen in line 13. By line 15, $R_k(M, a, S, s'_1, \dots, s'_k, t'_1, \dots, t'_k)$ holds. So the left-hand side of (12) holds for $s'_1, \dots, s'_k, t'_1, \dots, t'_k$, and hence, by our assumption, the right-hand side of (12) holds for $s'_1, \dots, s'_k, t'_1, \dots, t'_k$. In particular, there exist corresponding words $w_1, \dots, w_k$. Let $w_{j,i} = w_j$ for $i \geq 1$. Observe with help of the right-hand side of (12) that for all $i \in [0,k]$, $s_i^b \xrightarrow[M]{w_{j,i}} s_i^e$.

So in both cases ($S = \emptyset$ and $S \neq \emptyset$) there exist words $w_{j,0}, \dots, w_{j,k} \in \mathcal{A}^+$ such that $w_{j,0} \preceq \cdots \preceq w_{j,k}$ and for all i, $s_i^b \xrightarrow[M]{w_{j,i}} s_i^e$. So if we define $w_i = w_{1,i} \cdots w_{n,i}$ for $i \in [0,k]$, then $w_0 \preceq \cdots \preceq w_k$ and for all i, $s_i \xrightarrow[M]{w_i} t_i$. This shows the implication from left to right in (13).

"$\Leftarrow$" Let $w_0, \ldots, w_k$ be as in the right-hand side of (13). Let $w_0 = w_{1,0}w_{2,0} \cdots w_{n,0}$ be the decomposition of w_0 into letters from $\mathcal{A}$. This induces a decomposition $w_i = w_{1,i}w_{2,i} \cdots w_{n,i}$ for $i \in [1,k]$ such that the $w_{j,i}$ are nonempty words over $\mathcal{A}$ and for $j \in [1,n]$ it holds that $w_{j,0} \preceq \cdots \preceq w_{j,k}$. Note that if for some j, $|w_{j,k}| = 1$, then $w_{j,0} = \cdots = w_{j,k} \in \mathcal{A}$ and hence we may assume that this letter is marked with the empty word (since there is no insertion at this letter and so the mark is not needed).

We describe an accepting path of $Q_k(M, s_0, \ldots, s_k, t_0, \ldots, t_k)$. More precisely, by induction on j we show that there exists a path on which after the j-th pass of the loop it holds that for all i, $s'_i = \delta(s_i, w_{1,i} \cdots w_{j,i})$. For $j = 0$ this holds trivially by line 1. So let $j > 0$. By induction hypothesis there is a path on which after the $(j-1)$-st pass of the loop it holds that $s'_i = \delta(s_i, w_{1,i} \cdots w_{j-1,i})$. We continue this path and describe the necessary choices in the j-th pass.

If $w_{j,0} = [a, \varepsilon]$, then we are done by choosing $S = \emptyset$ in line 3 and a in line 6. Otherwise, $w_{j,0} = [a,u]$ for some nonempty word u. Hence $[a,u] \preceq w_{j,1} \preceq \cdots \preceq w_{j,k}$. We choose $t'_i = \delta(s'_i, \overline{w_{j,i}})$. Observe that for all $i \in [1,k]$, $s'_i \xrightarrow[M]{w_{j,i}} t'_i$ (otherwise this contradicts $s_i \xrightarrow[M]{w_i} t_i$ which holds by assumption). We apply (12) to $[a,u] \preceq w_{j,1} \preceq \cdots \preceq w_{j,k}$ and $s'_1, \ldots, s'_k, t'_1, \ldots, t'_k$. It follows that there exists $S \subseteq Z$ such that $|S| \leq 2^{k+2} - 2$, $\forall s \in S[\delta(s,u) = s]$, and $R_k(M, a, S, s'_1, \ldots, s'_k, t'_1, \ldots, t'_k)$. So if we choose $t'_0, \ldots, t'_k$ and S in this way, then we pass the lines 12, 14, and 15. Therefore, after the j-th pass, $s'_i = \delta(s_i, w_{1,i} \cdots w_{j,i})$. This completes the induction and shows that after the n-th pass, $s'_i = \delta(s_i, w_{1,i} \cdots w_{n,i}) = \delta(s_i, w_i) = t_i$. In this way we find an accepting path of $Q_k(M, s_0, \ldots, s_k, t_0, \ldots, t_k)$. This proves Claim 5.2. □

Claim 5.3 *If (12) and (13) hold for some k, then (12) holds for $k+1$.*

Proof We prove the equivalence (12) for $k+1$.

"$\Rightarrow$" Let S be as in the left-hand side of (12) with respect to $k+1$. We consider an accepting path of $R_{k+1}(M, a, S, s_1, \ldots, s_{k+1}, t_1, \ldots, t_{k+1})$. Note that we must reach line 4 and so we guess a bit d.

Case 1: $d = 1$. Here we reach line 16 and so $\delta(s_1, a) = t_1$. Since for acceptance we must reach line 19, there exists $S' \subseteq S$ such that $|S'| \leq 2^{k+2} - 2$ and $R_k(M, a, S', s_2, \ldots, s_{k+1}, t_2, \ldots, t_{k+1})$ accepts. So the left-hand side of (12) holds with respect to $R_k(M, a, S', s_2, \ldots, s_{k+1}, t_2, \ldots, t_{k+1})$. Hence by our assumption, there exist $w_2, \ldots, w_{k+1} \in \mathcal{A}^+$ such that $[a,u] \preceq w_2 \preceq \cdots \preceq w_{k+1}$ and for all $i \in [2, k+1]$, $s_i \xrightarrow[M]{w_i} t_i$. So with $w_1 = [a,u]$ we obtain $[a,u] \preceq w_1 \preceq \cdots \preceq w_{k+1}$ and for all $i \in [1, k+1]$, $s_i \xrightarrow[M]{w_i} t_i$.

Case 2: $d = 0$. Here we reach line 7 and so we choose S_1, S_2 according to line 7 and $s'_1, \ldots, s'_{k+1}, t'_1, \ldots, t'_{k+1}$ according to line 8. From our assumption that (12) holds with respect to k and from the acceptance of $R_k(M, a, S_1, s_2, \ldots, s_{k+1}, s'_2, \ldots, s'_{k+1})$ in line 10 (this must accept, since we reach line 19) it follows that

$$\exists x_2, \ldots, x_{k+1} \in \mathcal{A}^+ \Big([a,u] \preceq x_2 \preceq \cdots \preceq x_{k+1} \wedge \underset{i \in [2,k+1]}{\forall} [s_i \xrightarrow[M]{x_i} s'_i] \Big). \tag{14}$$

Similarly, from the acceptance of $R_k(M, a, S_2, t'_2, \ldots, t'_{k+1}, t_2, \ldots, t_{k+1})$ in line 11 it follows that

$$\exists z_2, \ldots, z_{k+1} \in \mathcal{A}^+ \Big([a,u] \preceq z_2 \preceq \cdots \preceq z_{k+1} \wedge \underset{i \in [2,k+1]}{\forall} [t'_i \xrightarrow[M]{x_i} t_i] \Big). \tag{15}$$

Finally, from our assumption that (13) holds with respect to k and from the acceptance of $Q_k(M, s'_1, \ldots, s'_{k+1}, t'_1, \ldots, t'_{k+1})$ in line 12 it follows that

$$\exists y_1, \ldots, y_{k+1} \in \mathcal{A}^+ \Big(y_1 \preceq \cdots \preceq y_{k+1} \wedge \underset{i \in [1,k+1]}{\forall} [s'_i \xrightarrow[M]{y_i} t'_i] \Big). \tag{16}$$

Let $x_1 = z_1 = [a, u]$ and let $w_i = x_i y_i z_i$ for $i \in [1, k+1]$. From (14), (15), and (16) we obtain $[a, u] \preceq w_1 \preceq w_2 \preceq \cdots \preceq w_{k+1}$ and for all $i \in [1, k+1]$, $s_i \xrightarrow[M]{w_i} t_i$. This establishes the right-hand side of (12) with respect to $k+1$.

"$\Leftarrow$" Assume that there exist $w_1, \ldots, w_{k+1} \in \mathcal{A}^+$ such that $[a, u] \preceq w_1 \preceq \cdots \preceq w_{k+1}$ and for all $i \in [1, k+1]$, $s_i \xrightarrow[M]{w_i} t_i$.

Case 1: $w_1 = [a, u]$. We describe the choice of S and we describe an accepting path of $R_{k+1}(M, a, S, s_1, \ldots, s_{k+1}, t_1, \ldots, t_{k+1})$. By assumption, (12) holds for k. So we can apply it to the chain $[a, u] \preceq w_2 \preceq \cdots \preceq w_{k+1}$. So there exists an $S' \subseteq Z$ such that $|S'| \leq 2^{k+2} - 2$, $\forall s \in S'[\delta(s, u) = s]$, and $R_k(M, a, S', s_2, \ldots, s_{k+1}, t_2, \ldots, t_{k+1})$ accepts. In particular, $\{\delta(s_2, a), \ldots, \delta(s_{k+1}, a), t_2, \ldots, t_{k+1}\} \subseteq S'$ (by line 3). Let $S = S' \cup \{t_1\}$. From $s_1 \xrightarrow[M]{w_1} t_1$ we obtain $\delta(s_1, a) = t_1$ and so $\{\delta(s_1, a), \ldots, \delta(s_{k+1}, a), t_1, \ldots, t_{k+1}\} \subseteq S$. Therefore, $R_{k+1}(M, a, S, s_1, \ldots, s_{k+1}, t_1, \ldots, t_{k+1})$ passes the lines 2 and 3. At line 4 we choose $d = 1$ and so we reach line 16. Here we chose S' as defined above. This brings us to line 19 where we accept.

Case 2: $w_1 \neq [a, u]$. So $w_1 = [a, u] y_1 [a, u]$ for some $y_1 \in \mathcal{A}^*$. This decomposition of w_1 induces for $i \in [2, k+1]$ a decomposition $w_i = x_i y_i z_i$ such that

$$[a, u] \preceq x_2 \preceq \cdots \preceq x_{k+1}, \tag{17}$$
$$y_1 \preceq y_2 \preceq \cdots \preceq y_{k+1}, \text{ and} \tag{18}$$
$$[a, u] \preceq z_2 \preceq \cdots \preceq z_{k+1}. \tag{19}$$

So there exist states $s'_1, \ldots, s'_{k+1}, t'_1, \ldots, t'_{k+1}$ such that for $i \in [1, k+1]$, $s_i \xrightarrow[M]{x_i} s'_i$, $s'_i \xrightarrow[M]{y_i} t'_i$, and $t'_i \xrightarrow[M]{x_i} t_i$ where $x_1 = z_1 = [a, u]$. From (17) and (19) together with our assumption that (12) holds for k we obtain: There exist $S_1, S_2 \subseteq Z$ such that $|S_1|, |S_2| \leq 2^{k+2} - 2$, $\forall s \in S_1[\delta(s, u) = s]$, $\forall s \in S_2[\delta(s, u) = s]$, $R_k(M, a, S_1, s_2, \ldots, s_{k+1}, s'_2, \ldots, s'_{k+1})$ accepts, and $R_k(M, a, S_2, t'_2, \ldots, t'_{k+1}, t_2, \ldots, t_{k+1})$ accepts. Moreover, by (18) and our assumption that (13) holds for k, $Q_k(M, s'_1, \ldots, s'_{k+1}, t'_1, \ldots, t'_{k+1})$ accepts. Let $S = S_1 \cup S_2 \cup \{s'_1, t_1\}$. Hence $|S| \leq 2^{k+3} - 4 + 2 = 2^{k+3} - 2$. Similar to Case 1, we observe that $\delta(s_1, a) = s'_1$, $\delta(t'_1, a) = t_1$, and $\{\delta(s_1, a), \ldots, \delta(s_{k+1}, a), t_1, \ldots, t_{k+1}\} \subseteq S$. So $R_{k+1}(M, a, S, s_1, \ldots, s_{k+1}, t_1, \ldots, t_{k+1})$ passes the lines 2 and 3. At line 4 we choose $d = 0$ and so we reach the lines 7 and 8 where we choose S_1, S_2 and $s'_1, \ldots, s'_{k+1}, t'_1, \ldots, t'_{k+1}$ as described above. By (17)–(19), this lets us pass the lines 9–12 and therefore, we reach line 19 where we accept. This proves Claim 5.3. □

Now Lemma 5.1 immediately follows from the Claims 5.2 and 5.3 by induction on k. This finishes the proof of Lemma 5.1. □

Theorem 5.4 *For every $n \geq 1$,*

$$\{M \mid M \textit{ is a deterministic finite automaton and } L(M) \in \Sigma_1^\sigma(n)\} \in \mathrm{NL}.$$

Proof Let $M = (A, Z, \delta, z_0, F)$ be a deterministic finite automaton such that $\varepsilon \notin L(M)$. By Theorem 4.3 and Lemma 5.1,

$$\begin{aligned} L(M) \notin \Sigma_1^{\sigma}(n) \quad &\Leftrightarrow \quad \text{there exist } t_0, \ldots, t_n \in Z \text{ and } w_0, \ldots, w_n \in \mathcal{A}^+ \text{ such that} \\ &\qquad 1.\ \forall i, [t_i \in F \text{ if and only if } i \text{ is even}], \\ &\qquad 2.\ w_0 \preceq w_1 \preceq \cdots \preceq w_n, \text{ and} \\ &\qquad 3.\ \forall i, z_0 \xrightarrow[M]{w_i} t_i. \\ &\Leftrightarrow \quad \text{there exist } t_0, \ldots, t_n \in Z \text{ such that } Q_n(M, z_0, \ldots, z_0, t_0, \ldots, t_n) \text{ and} \\ &\qquad \text{for all } i, [t_i \in F \text{ if and only if } i \text{ is even}]. \end{aligned}$$

So the following algorithm decides whether $L(M) \notin \Sigma_1^{\sigma}(n)$.

```
1  nondeterministically choose $t_0, \ldots, t_n \in Z$ such that $t_i \in F \Leftrightarrow i \equiv 0(2)$
2  if $Q_n(M, z_0, \ldots, z_0, t_0, \ldots, t_n)$ then reject else accept
```

If the algorithm Q_n accepts a language in NL, then $L(M) \notin \Sigma_1^{\sigma}(n)$ is decidable in $\mathrm{NL}^{\mathrm{NL}} = \mathrm{NL}$. Hence, also $L(M) \in \Sigma_1^{\sigma}(n)$ is decidable in NL. So it remains to show that Q_n accepts a language in NL.

We start with the algorithm L_n shown in Figure 4. On input of a deterministic, finite automaton $M = (A, Z, \delta, z_0, F)$ and a set $S \subseteq Z$ where $|S| \leq 2^{n+2}$ this algorithm tests whether there exists a nonempty word u such that for all $s \in S$, $\delta(s, u) = s$. So it tests whether all states in S share some nonempty loop. The algorithm works in nondeterministic logarithmic space, since $|S| \leq 2^{n+2}$ which is a constant. So L_n accepts a language in NL.

Now let us turn to the algorithms R_k and Q_k shown in the Figures 5 and 6. We show that R_k and Q_k accept sets in NL.

First, let us observe that if R_k accepts a set in NL, then Q_k accepts a sets in NL. The algorithm Q_k has to store $O(k)$ single states of M and a set of at most $2^{k+2} - 2$ states of M. This is possible in logarithmic space, since k is constant. Furthermore, the algorithm uses L_k and R_k as oracles. Hence, if R_k accepts a set in NL, then Q_k accepts a set in $\mathrm{NL}^{\mathrm{NL}} = \mathrm{NL}$.

We proceed by induction on k and show that R_k and Q_k accept sets in NL. This is obvious for R_0 and by the observed implication, it also holds for Q_0. So let $k > 0$. The algorithm R_k has to store $O(k)$ single states and three sets of at most $2^{k+2} - 2$ states. This is a constant number of states, since k is constant. Moreover, R_k uses the algorithms Q_{k-1} and R_{k-1} as oracles. So by induction hypothesis, R_k accepts a set in $\mathrm{NL}^{\mathrm{NL}} = \mathrm{NL}$, and by the observed implication, Q_k accepts a set in NL. □

Theorem 5.5 *For every $n \geq 1$,*

$$\{M \mid M \textit{ is a deterministic finite automaton and } L(M) \in \Sigma_1^{\tau_{\mathrm{d}}}(n)\} \in \mathrm{NL}.$$

Proof The algorithm used in Theorem 5.4 tests for alternating chains in $(\mathcal{B}_M; \preceq)$. We can generalize this algorithm for alternating chains in $(\mathcal{B}_M; \preceq^d)$. By Theorem 4.2, this yields a membership algorithm for the classes $\Sigma_1^{\tau_{\mathrm{d}}}(n)$.

To make this idea work, we have to modify the algorithms Q_k and R_k such that they additionally verify length conditions modulo d. We denote these algorithms by $Q_{k,d}$ and $R_{k,d}$. More precisely, $Q_{k,d}$ obtains an additional parameter l and it makes sure that the number of guessed letters $a \in A$ is equivalent to l modulo d (i.e., the number of iterations of the loop at the lines 2–18 is equivalent l modulo d). Moreover, whenever $R_{k,d}$ calls $Q_{k,d}$, then the additional parameter is set to $l = d - 1$. By doing so we make sure that all insertions have a length equivalent to 0 modulo d. Finally, the following algorithm decides whether $L(M) \notin \Sigma_1^{\tau_d}(n)$.

```
1  nondeterministically choose $t_0, \dots, t_n \in Z$ such that $t_i \in F \Leftrightarrow i \equiv 0(2)$
2  if $Q_{n,d}(M, z_0, \dots, z_0, t_0, \dots, t_n, l)$ for some $l \in [0, d-1]$
   then reject else accept
```

Now the same argument as in the proof of Theorem 5.4 shows that $\Sigma_1^{\tau_d}(n)$ is decidable in NL.
□

Remark 5.6 Unfortunately, we do not obtain NL-decidability for the classes $\Sigma_1^{\tau}(n)$. The reason is that the d in Theorem 4.6 is extremely big, i.e., we only know the upper bound $d \leq (m^m)!$ where m is the size of the automaton. We leave the question for an improved bound open. Note that if d can be bounded polynomially in the size of the automaton, then $\Sigma_1^{\tau}(n)$ is decidable in NL. Although d is very large, it is still computable from the automaton M which implies the decidability of all levels $\Sigma_1^{\tau}(n)$. This settles a question left open in [Sel04].

Theorem 5.7 *For every $n \geq 1$, the following set is decidable.*

$$\{M \mid M \textit{ is a deterministic finite automaton and } L(M) \in \Sigma_1^{\tau}(n)\}$$

Proof Follows from the Theorems 4.6 and 5.5. □

Theorem 5.8 *For every $n \geq 1$,*

$\{M \mid M$ *is a deterministic finite automaton over the alphabet* $\{a, b\}$ *and* $L(M) \in \Sigma_2^{\varrho}(n)\} \in$ NL.

Proof If one compares the characterizations of $\Sigma_1^{\sigma}(n)$ (Theorem 4.3) and $\Sigma_2^{\varrho}(n)$ (Theorem 4.11), then one notices that two new aspects appear in the characterization of $\Sigma_2^{\varrho}(n)$.

1. It assumes chains satisfying the alphabet condition.

2. It works only for star-free languages.

We can easily change the decision algorithm for $\Sigma_1^{\sigma}(n)$ such that it verifies alphabet conditions. For this we only have to modify the algorithm L_n (Figure 4) such that it makes sure that the loop (lines 3–8) guesses at least one letter a and at least one letter b.

Regarding the second item, we cannot simply test for membership in SF, since such a test is PSPACE-complete [CH91]. However, we can get rid of the restriction on star-free languages by testing whether $L(M) \in \Sigma_3^{\varrho}$. Note that $\Sigma_2^{\varrho}(n) \subseteq \Sigma_3^{\varrho} \subseteq$ SF. Moreover, it is known [GS01b] that for the alphabet $A = \{a, b\}$ this test is decidable in NL. □

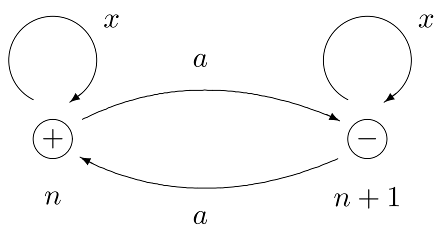


Figure 7: The 2-counting pattern with $|a| = |x| = 1$.

6 Exact Complexity Estimations

We show that with the exception of $\Sigma_1^{\tau}(n)$, the membership problems of all classes of Boolean hierarchies considered in this paper are NL-complete. In contrast, the membership problems of the general classes FO_{τ} and FO_{τ_d} are PSPACE-complete and hence are strictly more complex.

We start with an easy fact that gives lower bounds uniformly for all classes under discussion. Let A be an alphabet with $a, b \in A$.

Proposition 6.1 *Let $\mathcal{C}$ be any class of regular quasi-aperiodic languages over A with $\emptyset \in \mathcal{C}$. Then it is* NL*-hard to decide whether a given dfa M accepts a language in $\mathcal{C}$.*

Proof Let $\mathcal{G}$ be the class of structures $G = ([0, n]; E, 0, n)$ where $n \geq 1$ and $([0, n]; E)$ is a directed acyclic graph in which any vertex has at most 2 out-going edges, 0 has no in-going edges and n has no out-going edges. From the NL-completeness of the reachability problem it follows that the problem $\mathcal{R}$ of deciding whether in a given structure G as above n is reachable from 0 is NL-complete. Hence, it suffices to relate (in logarithmic space) to any $G \in \mathcal{G}$ a dfa $M = (A, Z, \delta, s_0, F)$ such that $L(M) = \emptyset$ for $G \notin \mathcal{R}$ and $L(M)$ is not quasi-aperiodic otherwise.

Set $Z = [0, n + 1]$, $s_0 = 0$, $F = \{n\}$ and define the transition function as follows:

if $q < n$ and q has no out-going edges, set $\delta(q, x) = q$ for all $x \in A$;

if $q < n$ and $E(q, s)$ for exactly one $s \leq n$, set $\delta(q, a) = s$ and $\delta(q, x) = q$ for all $x \in A \setminus \{a\}$;

if $q < n$ and $E(q, s_1)$, $E(q, s_2)$ for two distinct $s_1, s_2 \leq n$, set $\delta(q, a) = s_1$, $\delta(q, b) = s_2$ and $\delta(q, x) = q$ for all $x \in A \setminus \{a, b\}$;

set $\delta(n, a) = n + 1$ and $\delta(n, x) = n$ for all $x \in A \setminus \{a\}$;

set $\delta(n + 1, a) = n$ and $\delta(n + 1, x) = n + 1$ for all $x \in A \setminus \{a\}$.

If $G \notin \mathcal{R}$ then n is not reachable from 0 in M, hence $L(M) = \emptyset$. If $G \in \mathcal{R}$ then n is reachable from 0 by a word in $\{a, b\}^*$, and $\{n, n+1\}$ forms the balanced 2-counting pattern [Gla07, Sel04] shown in Figure 7. By [Sel04], existence of such a reachable pattern implies that $L(M)$ is not quasi-aperiodic. □

The last proposition together with the upper bounds established in the previous sections immediately imply the following exact complexity estimations.

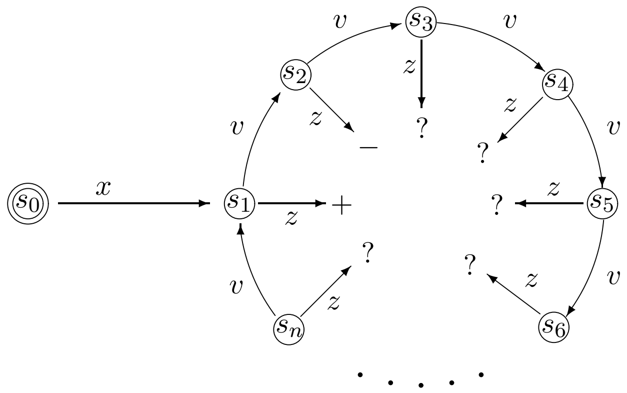


Figure 8: The n-counting pattern with initial state s_0.

Theorem 6.2 *Let $k \geq 0$, $n \geq 1$, $d \geq 1$ and $\mathcal{C}$ is one of the classes $\mathcal{C}_k^d(n)$, $\Sigma_1^\sigma(n)$, $\Sigma_1^{\tau_d}(n)$, or $\Sigma_2^\varrho(n)$ for $|A| = 2$. Then the set $\{M \mid M$ is a deterministic finite automaton and $L(M) \in \mathcal{C}\}$ is* NL*-complete.*

Proof Follows from Proposition 6.1 and the Theorems 3.1, 5.4, 5.5, and 5.8. □

We conclude this section with a corollary of the PSPACE-completeness of deciding FO_σ which was established by Stern [Ste85b] and by Cho and Huynh [CH91]. It shows that the complexity of deciding the classes FO_τ and FO_{τ_d} is strictly higher than the complexity of deciding the classes mentioned in Theorem 6.2. (Note that NL is closed under logspace many-one reductions, $\mathrm{NL} \subseteq \mathrm{DSPACE}(\log^2 n)$ [Sav70] and $\mathrm{DSPACE}(\log^2 n) \subsetneq \mathrm{PSPACE}$ [HS65]. Hence the classes FO_τ and FO_{τ_d} can not be decided in NL.)

Theorem 6.3 *The classes* FO_τ *and* FO_{τ_d} *are* PSPACE*-complete.*

Proof For the upper bound, slight modifications of the proof in [Ste85b] that the class of aperiodic languages is in PSPACE suffice. As is well-known, aperiodicity is equivalent to the non-existence of the n-counting pattern for each $n \geq 2$ (see Figure 8). Since the dfa-minimization is computable in polynomial time, it suffices to consider minimal dfa's. Let us recall the idea of the proof in [Ste85a] that, given a minimal dfa M, it is decidable in PSPACE whether M has a counting pattern, i.e., whether there are a state p of M, a number r and a word v such that $\delta(p, v) \neq p$ and $\delta(p, v^r) = p$. By Savitch's theorem, it suffices to decide the last property in nondeterministic polynomial space.

The idea of the algorithm is as follows (we assume that the set of states of M is $\{1, \ldots, m\}$): Guess subsequent letters of v and compute the array $A[1], \ldots, A[m]$ such that $A[i] = \delta(i, v)$ for all $i \in [1, m]$. Then guess a state p and a number r and check, using the array A, the properties $\delta(p, v) \neq p$ and $\delta(p, v^r) = p$ (see [Ste85b] for additional details).

To show that the question "$L(M) \in \mathrm{FO}_{\tau_d}$" is in PSPACE, we use the following result that is an easy corollary of the characterization of FO_{τ_d}-definable languages from [Sel04]: a regular

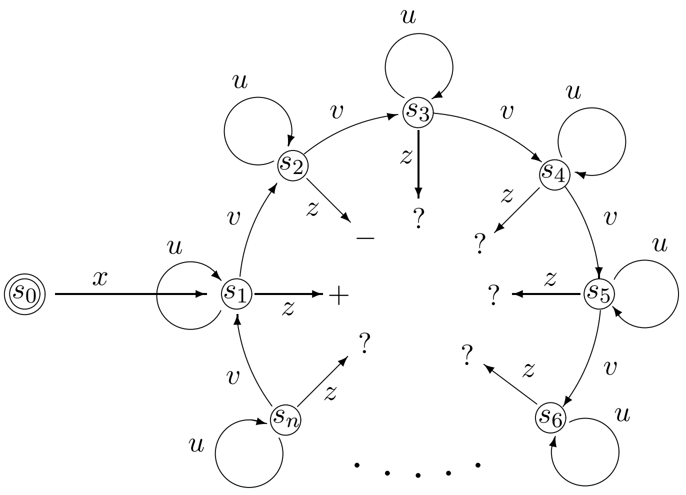


Figure 9: The balanced n-counting pattern with initial state s_0 and $|v| = |u|$.

language L is in FO_{τ_d} if and only if the minimal automaton M of L has no d-balanced counting patterns, i.e., counting patterns as above, but with the additional condition $|v| \equiv 0 \pmod{d}$. It follows that it suffices to modify Stern's algorithm as follows: instead of guessing subsequent letters of v, we have just to guess d subsequent letters at each step.

To show that the relation "$L(M) \in \mathrm{FO}_\tau$" is in PSPACE, we use the following result that is an easy corollary of the characterization of FO_τ-definable languages from [Sel04]: a regular language L is in FO_τ if and only if the minimal automaton M of L has no balanced counting patterns (see Figure 9). It suffices to check in PSPACE whether there are a state p of M, a number r and words v, u such that $|u| = |v|$, $\delta(p, v) \neq p$, $\delta(p, v^r) = p$ and any state in the sequence $p, \delta(p, v), \ldots, \delta(p, v^{r-1})$ has a u-loop. This time it suffices to modify Stern's algorithm as follows: Guess synchronously subsequent letters of v and u (thus guaranteeing that $|u| = |v|$) and compute the array $A[1], \ldots, A[m]$ as above and the array $B[1], \ldots, B[m]$ such that $B[i] = \delta(i, u)$ for all $i \in [1, m]$. Then guess a state p and a number r and check the properties $\delta(p, v) \neq p$, $\delta(p, v^r) = p$ (using the array A) and the loop property (using the array B).

For the lower bound, we use the result in [CH91] that the class $\mathrm{SF}(\{a, b\})$ of star-free languages over the binary alphabet $\{a, b\}$ is PSPACE-complete. So it suffices to show that $\mathrm{SF}(\{a, b\})$ is reducible (in polynomial time) to $\mathrm{FO}_\tau(A)$ and $\mathrm{FO}_{\tau_d}(A)$ for any alphabet $A \supseteq \{a, b\}$. The proof is especially simple for $|A| > 2$. Relate to any dfa $M = (\{a, b\}, Z, \delta, s_0, F)$ the dfa $M' = (A, Z, \delta', s_0, F)$ by adjoining the 1-letter loops $\delta'(q, x) = x$ for all $q \in Z$ and $x \in A \setminus \{a, b\}$. If $L(M)$ is star-free, then M has no counting pattern, then M' also has no counting pattern, and hence $L(M')$ is star-free and so is in $\mathrm{FO}_{\tau_d}(A)$ for any d. If $L(M)$ is not star-free, then M has a counting pattern as in Figure 8, then M' has a balanced counting pattern as in Figure 9 (take $u = x^{|v|}$ for some $x \in X \setminus \{a, b\}$). By [Sel04], the existence of such a balanced pattern implies $L(M') \notin \mathrm{FO}_\tau(A)$.

It remains to reduce $\mathrm{SF}(\{a, b\})$ to $\mathrm{FO}_\tau(\{a, b\})$ and $\mathrm{FO}_{\tau_d}(\{a, b\})$. The idea is the same as in the previous paragraph, but this time we use the coding $a \mapsto ba$, $b \mapsto ab$ and organize the 1-letter loops using the remaining 2-letter words aa and bb. Of course, this time the automaton $M' = (\{a, b\}, Z', \delta', s_0', F')$ is a bit more complex as above. We choose now two disjoint copies

Z^a and Z^b of Z (which are also disjoint from Z), a bijection $s \mapsto s^a$ between Z and Z^a, and a bijection $s \mapsto s^b$ between Z and Z^b. Set $Z' = Z^b \cup Z \cup Z^a$, $s'_0 = s_0$, $F' = \{s^b, s, s^a \mid s \in F\}$ and define the transition function $\delta' : Z' \times \{a,b\} \to Z'$ as follows (where $s \in Z$):

$$\delta'(s,a) = s^a,\ \delta'(s,b) = s^b,\ \delta'(s^a,a) = s^a,\ \delta'(s^a,b) = \delta(s,b),\ \delta'(s^b,b) = s^b,\ \delta'(s^b,a) = \delta(s,a).$$

Let $h : \{a,b\}^* \to \{a,b\}^*$ be the monoid morphism such that $h(a) = ba$ and $h(b) = ab$. Since $\delta'(s,ba) = \delta(s,a)$ and $\delta'(s,ab) = \delta(s,b)$, we have $\delta'(s,h(v)) = \delta(s,v)$ for all $s \in Z$ and $v \in \{a,b\}^*$.

Now, assume that M has a counting pattern $(s_1,\dots,s_n)$, $n > 1$, as in Figure 8, for some witnesses $v,x,z \in \{a,b\}^*$. Assume that the first letter of the non-empty word v is a, i.e., $v = a\tilde{v}$ for some $\tilde{v} \in \{a,b\}^*$ (if the first letter of v is b, the argument is symmetric). Then $(s_1,\dots,s_n)$ is also a counting pattern of M', with the witnesses $v' = h(v) = bah(\tilde{v})$, $x' = h(x)$ and $z' = h(z)$. Then the "shift" $(s_1^b,\dots,s_n^b)$ is also a counting pattern of M', with suitable witnesses v'_b, x'_b and z'_b (e.g., as v'_b we can take the cyclic shift $ah(\tilde{v})b$ of v'). Moreover, any of the states $s_1^b,\dots,s_n^b$ has a b-loop. As in the easy case of a bigger alphabet A, M' has a balanced counting pattern.

It remains to show that if M' has a counting pattern $(s_1,\dots,s_n)$, $n > 1$, for some witnesses $v',x',z' \in \{a,b\}^*$, then M also has a counting pattern. Represent v' in the form $v' = a_1^{k_1}\cdots a_m^{k_m}$ where $m, k_i \geq 1$, $a_i \in \{a,b\}$ and $a_{i+1} \neq a_i$ for all $i \in [1,m-1]$. Note that in fact $m \geq 2$, because $a_1^{k_1}$ cannot be a witness for a non-trivial counting pattern by definition of δ'.

We check that all the states $s_1,\dots,s_n$ are in one and the same of the sets Z, Z^a, Z^b. Let first $k_1 = \cdots = k_m = 1$, i.e., v' is a repetition-free word, and assume w.l.o.g. that $a_m = a$. Since $\delta'(q,a) \in Z \cup Z^a$ for all $q \in Z'$, we subsequently get $s_2,\dots,s_n,s_1 \in Z \cup Z^a$. Assume that $a_1 = a$. Then, since $\delta'(q,a) \in Z^a$ for all $q \in Z \cup Z^a$, we subsequently get $s_2,\dots,s_n,s_1 \in Z^a$. Now let $a_1 = b$. In this case, $(s_1 \in Z \Rightarrow s_2,\dots,s_n \in Z)$ and $(s_1 \in Z^a \Rightarrow s_2,\dots,s_n \in Z^a)$. Now let $k_i \geq 2$ for some $i \in [1,m]$. Choose the largest such i and assume w.l.o.g. that $a_i = a$. Since $\delta'(q,a^j) \in Z^a$ for all $q \in Z'$ and $j \geq 2$, $\delta'(s_1, a_1^{k_1}\cdots a_i^{k_i}) \in Z^a$. Arguing as above, we see that in fact $s_2,\dots,s_n,s_1 \in Z \cup Z^a$. If $a_m = a$ we subsequently get $s_2,\dots,s_n,s_1 \in Z^a$. If $a_m = b$ we subsequently get $s_2,\dots,s_n,s_1 \in Z$.

Now assume that $s_1,\dots,s_n \in Z$ and consider the path in M' starting at s_1 and labeled by the word v'. This path may contain one-letter loops (e.g., such a loop can appear in case $k_1 \geq 2$). Let v'' be the word obtained from v' by deleting all labels of such 1-letter loops. Then v'' may also serve as a witness for the counting pattern $(s_1,\dots,s_n)$, because $\delta'(s_i,v'') = \delta'(s_i,v')$ for all $i \in [1,n]$. Moreover, $v'' = h(v)$ for a unique $v \in \{a,b\}^+$, hence $\delta(s_i,v) = s_{i+1}$ for all $i \in [1,n-1]$ and $\delta(s_n,v) = s_1$. The same construction applied to the word x' yields a word $x \in \{a,b\}^*$ such that $\delta'(s_0,h(x)) = s_1$, hence $\delta(s_0,x) = s_1$.

Consider now the word $z' \in \{a,b\}^*$ such that $t_1 = \delta'(s_1,z') \in F'$ and $t_2 = \delta'(s_2,z') \notin F'$; we want to find a word $z \in \{a,b\}^*$ such that $\delta(s_1,z) \in F$ and $\delta(s_2,z) \notin F$. If $z' = \varepsilon$, then take $z = \varepsilon$. Otherwise, represent z' in the form $z' = b_1^{l_1}\cdots b_u^{l_u}$ where $u, l_i \geq 1$, $b_i \in \{a,b\}$ and $b_{i+1} \neq b_i$ for all $i \in [1,m-1]$. Clearly, both t_1 and t_2 are in one and the same of the sets Z, Z^a, Z^b. W.l.o.g. we may even assume that $t_1,t_2 \in Z$, because otherwise we could take the word $b_1^{l_1}\cdots b_{u-1}^{l_{u-1}}$ in place of z'. Now we can apply the same construction as above and find a word $z \in \{a,b\}^*$ such that $t_1 = \delta'(s_1,h(z))$ and $t_2 = \delta'(s_2,h(z))$, hence also $t_1 = \delta(s_1,z)$ and $t_2 = \delta(s_2,z)$. Therefore, $s_1,\dots,s_n,v,x,z$ form a counting pattern in M.

It still remains to consider the case $s_1,\dots,s_n \in Z^a$ (the case $s_1,\dots,s_n \in Z^b$ is symmetric). This case is reduced to the case $s_1,\dots,s_n \in Z$ as follows. We have $s_i = u_i^a$ for unique $u_1,\dots,u_n \in Z$, and the last letter of v' is a. It is easy to see that the sequence $(u_1, \dots, u_n)$, together with suitable "shifts" of v', x', z' is a cycle of M', hence the argument above applies. $\square$

Acknowledgement. We are grateful to Klaus Wagner for many helpful discussions.

References

[BCST92] D. A. Mix Barrington, K. Compton, H. Straubing, and D. Thérien. Regular languages in NC^1. *Journal of Computer and System Sciences*, 44:478–499, 1992.

[Brz76] J. A. Brzozowski. Hierarchies of aperiodic languages. *RAIRO Inform. Theor.*, 10:33–49, 1976.

[Büc62] J. R. Büchi. On a decision method in restricted second-order arithmetic. In *Proceedings Logic, Methodology and Philosophy of Sciences 1960*, Stanford, CA, 1962. Stanford University Press.

[CB71] R. S. Cohen and J. A. Brzozowski. Dot-depth of star-free events. *Journal of Computer and System Sciences*, 5:1–16, 1971.

[CGH+88] J.-Y. Cai, T. Gundermann, J. Hartmanis, L. A. Hemachandra, V. Sewelson, K. W. Wagner, and G. Wechsung. The boolean hierarchy I: Structural properties. *SIAM Journal on Computing*, 17:1232–1252, 1988.

[CH91] S. Cho and D. T. Huynh. Finite-automaton aperiodicity is PSPACE-complete. *Theoretical Computer Science*, 88:99–116, 1991.

[CPP93] R. S. Cohen, D. Perrin, and J. E. Pin. On the expressive power of temporal logic. *Journal of Computer and System Sciences*, 46:271–294, 1993.

[CPS06] L. Chaubard, J. E. Pin, and H. Straubing. First order formulas with modular predicates. In *Proceedings 21th IEEE Symposium on Logic in Computer Science*, pages 211–220. IEEE Computer Society, 2006.

[Gla07] C. Glaßer. Languages polylog-time reducible to dot-depth 1/2. *Journal of Computer and System Sciences*, 73(1):36–56, 2007.

[GS00a] C. Glaßer and H. Schmitz. Languages of dot-depth 3/2. In *Proceedings 17th Symposium on Theoretical Aspects of Computer Science*, volume 1770 of *Lecture Notes in Computer Science*, pages 555–566. Springer Verlag, 2000.

[GS00b] C. Glaßer and H. Schmitz. Decidable hierarchies of starfree languages. In *Proceedings 20th Conference on the Foundations of Software Technology and Theoretical Computer Science*, volume 1974 of *Lecture Notes in Computer Science*, pages 503–515. Springer Verlag, 2000.

[GS01a] C. Glaßer and H. Schmitz. The boolean structure of dot-depth one. *Journal of Automata, Languages and Combinatorics*, 6(4):437–452, 2001.

[GS01b] C. Glaßer and H. Schmitz. Level 5/2 of the Straubing-Thérien hierarchy for two-letter alphabets. In *Preproceedings 5th Conference on Developments in Language Theory*, pages 254–265, 2001.

[HS65] J. Hartmanis and R. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[Imm88] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17:935–938, 1988.

[Kna83] R. Knast. A semigroup characterization of dot-depth one languages. *RAIRO Inform. Theor.*, 17:321–330, 1983.

[KSW87] J. Köbler, U. Schöning, and K. W. Wagner. The difference and the truth-table hierarchies for NP. *RAIRO Inform. Théor.*, 21:419–435, 1987.

[MP71] R. McNaughton and S. Papert. *Counterfree Automata*. MIT Press, Cambridge, 1971.

[MPT00] A. Maciel, P. Péladeau, and D. Thérien. Programs over semigroups of dot–depth one. *Theoretical Computer Science*, 245:135–148, 2000.

[Pin95] J. E. Pin. Finite semigroups and recognizable languages: an introduction. In J. Fountain, editor, *NATO Advanced Study Institute: Semigroups, Formal Languages and Groups*, pages 1–32. Kluwer Academic Publishers, 1995.

[Pin96a] J. E. Pin. Logic, semigroups and automata on words. *Annals of Mathematics and Artificial Intelligence*, 16:343–384, 1996.

[Pin96b] J. E. Pin. Syntactic semigroups. In G. Rozenberg and A. Salomaa, editors, *Handbook of formal languages*, volume I, pages 679–746. Springer, 1996.

[PP86] D. Perrin and J. E. Pin. First-order logic and star-free sets. *Journal of Computer and System Sciences*, 32:393–406, 1986.

[PW97] J. E. Pin and P. Weil. Polynomial closure and unambiguous product. *Theory of computing systems*, 30:383–422, 1997.

[PW02] J. E. Pin and P. Weil. The wreath product principle for ordered semigroups. *Communications in Algebra*, 30:5677–5713, 2002.

[Sav70] W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.

[Sch65] M. P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8:190–194, 1965.

[Sch01] H. Schmitz. *The Forbidden-Pattern Approach to Concatenation Hierarchies*. PhD thesis, Fakultät für Mathematik und Informatik, Universität Würzburg, 2001.

[Sel01] V. L. Selivanov. A logical approach to decidability of hierarchies of regular star-free languages. In *Proceedings 18th Symposium on Theoretical Aspects of Computer Science*, volume 2010 of *Lecture Notes in Computer Science*, pages 539–550. Springer Verlag, 2001.

[Sel04] V. L. Selivanov. Some hierarchies and reducibilities on regular languages. Technical Report 349, Inst. für Informatik, Univ. Würzburg, 2004.

[Shu98] A.G. Shukin. Difference hierarchies of regular languages (in russian). *Computing Systems (Novosibirsk, Institute of Mathematics)*, (161):141–155, 1998.

[Sim75] I. Simon. Piecewise testable events. In *Proceedings 2nd GI Conference*, volume 33 of *Lecture Notes in Computer Science*, pages 214–222. Springer-Verlag, 1975.

[SS00] V. L. Selivanov and A. G. Shukin. On hierarchies of regular star-free languages. Technical Report Preprint 69, A. P. Ershov Institute of Informatics Systems, Novosibirsk, 2000.

[Ste85a] J. Stern. Characterizations of some classes of regular events. *Theoretical Computer Science*, 35:17–42, 1985.

[Ste85b] J. Stern. Complexity of some problems from the theory of automata. *Information and Control*, 66:163–176, 1985.

[Str81] H. Straubing. A generalization of the Schützenberger product of finite monoids. *Theoretical Computer Science*, 13:137–150, 1981.

[Str85] H. Straubing. Finite semigroup varieties of the form $\mathbf{V} * \mathbf{D}$. *J.Pure Appl.Algebra*, 36:53–94, 1985.

[Str88] H. Straubing. Semigroups and languages of dot-depth two. *Theoretical Computer Science*, 58:361–378, 1988.

[Str94] H. Straubing. *Finite automata, formal logic and circuit complexity.* Birkhäuser, Boston, 1994.

[SW98] Heinz Schmitz and Klaus W. Wagner. The Boolean hierarchy over level 1/2 of the Straubing-Thèrien hierarchy. *CoRR*, cs.CC/9809118, 1998. http://arxiv.org/abs/cs.CC/9809118.

[Sze87] R. Szelepcsényi. The method of forcing for nondeterministic automata. *Bull. of the EATCS*, 33:96–100, 1987.

[Thé81] D. Thérien. Classification of finite monoids: the language approach. *Theoretical Computer Science*, 14:195–208, 1981.

[Tho82] W. Thomas. Classifying regular events in symbolic logic. *Journal of Computer and System Sciences*, 25:360–376, 1982.

[Tra58] B. A. Trakhtenbrot. Synthesis of logic networks whose operators are described by means of single-place predicate calculus. *Doklady Akad. Nauk SSSR*, 118:646–649, 1958.

[Wei04] P. Weil. Algebraic recognizability of languages. In *Proceedings 29th Mathematical Foundations of Computer Science*, volume 3153 of *Lecture Notes in Computer Science*, pages 149–175. Springer-Verlag, 2004.

[Yu96] S. Yu. Regular languages. In G.Rozenberg and A.Salomaa, editors, *Handbook of formal languages*, volume I, pages 41–110. Springer, 1996.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/