Efficient Algorithms for Membership in Boolean Hierarchies of Regular Languages
Christian Glasser, Heinz Schmitz, Victor Selivanov

To cite this version:

HAL Id: hal-00227550
https://hal.archives-ouvertes.fr/hal-00227550
Submitted on 31 Jan 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
EFFICIENT ALGORITHMS FOR MEMBERSHIP IN BOOLEAN HIERARCHIES OF REGULAR LANGUAGES

C. GLÄSSER 1, H. SCHMITZ 2, AND V. SELIVANOV 3

1 Universität Würzburg, Germany. E-mail address: glasser@informatik.uni-wuerzburg.de
2 Fachhochschule Trier, Germany. E-mail address: schmitz@informatik.fh-trier.de
3 A. P. Ershov Institute of Informatics Systems, Russia. E-mail address: vseliv@nspu.ru

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 Σ1 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 allowed.
• If modular predicates for arbitrary d are allowed, then the classes of the Boolean hierarchy over level Σ1 are decidable.
• For the restricted case of a two-letter alphabet, the classes of the Boolean hierarchy over level Σ2 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.

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

Over and the predecessor of positions. Σ minimum and maximum positions in words and adding functions that compute the successor Σ. Let Thérien hierarchy. Let our case one of the classes Σ̺ for short) [CB71, Tho82]. Let τ Σ of the signature definable by similar first-order Σ̺-sentences; this class equals level 3/2 of the Straubing-Thérien hierarchy. Σ̺ of the signature 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 Σ̺(n) and C̺ 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). From our new characterizations we obtain efficient algorithms for membership tests in NL. For two of the considered Boolean hierarchies, these are the first decidable characterizations at all, i.e., for the classes Σ̺(n) for the alphabet A = {a, b}, and for the classes Σ̺ 1(n)).

Definitions. We sketch the definitions of the Boolean hierarchies considered in this paper. Σ̺ 1 denotes the class of languages definable by first-order Σ̺ 1-sentences over the signature ϕ = {≤, Qa, ...} where for every letter a ∈ A, Qa(i) is true if and only if the letter a appears at the i-th position in the word. Σ̺ 1 equals level 1/2 of the Straubing-Thérien hierarchy (STH for short) [Str81, Thé81, Str85, P P86]. Σ̺ 1 is the class of languages definable by similar first-order Σ̺ 1-sentences; this class equals level 3/2 of the Straubing-Thérien hierarchy. Let σ be the signature obtained from ϕ by adding constants for the minimum and maximum positions in words and adding functions that compute the successor and the predecessor of positions. Σ̺ 1 denotes the class of languages definable by first-order Σ̺ 1-sentences of the signature σ; this class equals level 1/2 of the dot-depth hierarchy (DDH for short) [B77, Ih82]. Let τ̺ d be the signature obtained from σ by adding the unary predicates P̺ d, ..., P̺ d−1 where P̺ d(i) is true if and only if i ≡ j(mod d). Let τ be the union of all τ̺ d. Σ̺ 1 (resp., Σ̺ 1) is the class of languages definable by first-order Σ̺ 1-sentences of the signature τ̺ d (resp., τ). C̺ k is the generalization of Σ̺ 1 where neighborhoods of k + 1 consecutive letters and distances modulo d are expressible (Definition 1.2). For a class D (in our case one of the classes Σ̺ 1, Σ̺ 2, C̺ k, Σ̺ 1, Σ̺ 1, and Σ̺ 1 for |A| = 2), the Boolean hierarchy over D is the family of classes

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

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 $\Sigma^2_\sigma(n)$, $\Sigma^{d_1}_\tau(n)$, and $\Sigma^2_\rho(n)$ for the alphabet $A = \{a, b\}$ we prove new characterizations in terms of forbidden alternating chains. In case of $\Sigma^2_\rho(n)$ for the alphabet $A = \{a, b\}$, this is the first characterization of this class.

2. For the classes $\Sigma^2_\sigma(n)$, $C^d_k(n)$, $\Sigma^{d_1}_\tau(n)$, and $\Sigma^2_\rho(n)$ 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 $\Sigma^2_\tau(n)$, and of $\Sigma^2_\rho(n)$ 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 the 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 Theorem 1.1), one can do this exact computation of the level for the Boolean hierarchies over $\Sigma^2_\sigma$, $\Sigma^2_\rho$ (for alphabet $A = \{a, b\}$), $C^d_k$, $\Sigma^2_\tau$, and $\Sigma^2_\rho$. To achieve the same for the Boolean hierarchies over $C^d_k$ and $\Sigma^{d_1}_\tau$ 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...
Table 1: Overview of decidability and complexity results.

<table>
<thead>
<tr>
<th>Boolean hierarchy classes</th>
<th>decidability</th>
<th>complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Sigma_1^\varpi(n) )</td>
<td>[SW98]</td>
<td>NL-complete [SW98]</td>
</tr>
<tr>
<td>( C_1^k(n) )</td>
<td>[GS01a, Sel01]</td>
<td>NL-complete [Sch01]</td>
</tr>
<tr>
<td>( \Sigma_1^\sigma(n) )</td>
<td>[GS01a]</td>
<td>NL-complete [this paper]</td>
</tr>
<tr>
<td>( \Sigma_1^d(n) )</td>
<td>[Sel04]</td>
<td>NL-complete [this paper]</td>
</tr>
<tr>
<td>( \Sigma_2^\tau_1(n) )</td>
<td>[Sel04]</td>
<td>no efficient bound known (see Remark 4.3)</td>
</tr>
<tr>
<td>( \Sigma_2(n) ) for (</td>
<td>A</td>
<td>= 2 )</td>
</tr>
</tbody>
</table>

we mainly use the logical approach which has a long tradition starting with the early work of Trakhtenbrot [Tra58] and Büchi [Büc60]. 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.** After the preliminaries, we explain the general idea of an efficient membership algorithm for the classes \( C_1^k(n) \) (section 2). This easy example shows how a suitable characterization of a Boolean hierarchy can be turned into an efficient membership test. The algorithms for the other Boolean hierarchies are similar, but more complicated. Section 3 provides new alternating-chain characterizations for the Boolean hierarchies over \( \Sigma_1^\tau_1 \), \( \Sigma_1^d \), and \( \Sigma_2^\varpi \) for the alphabet \( A = \{a, b\} \). In section 4 we exploit these characterizations and obtain efficient algorithms for testing the membership in these classes. In particular, we obtain the decidability of the classes \( \Sigma_1^\tau_1(n) \) and \( \Sigma_2^\varpi(n) \) for the alphabet \( A = \{a, b\} \). Finally, section 5 provides lower bounds for the complexity of the considered decidability problems. As a consequence (with the exception of \( \Sigma_1^\tau_1(n) \)) the membership problems of all considered Boolean levels are logspace many-one complete for NL. In contrast, the membership problems of the general classes \( \text{FO}_\tau \) and \( \text{FO}_{\tau_d} \) are logspace many-one complete for \( \text{PSPACE} \) and hence are strictly more complex.

Detailed proofs are available in the technical report [GSS07].

1. Preliminaries

In this section we recall definitions and results that are needed later in the paper. 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 \( \{m, m+1, \ldots, n\} \). 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 \( \mathcal{C} \), \( \text{BC}(\mathcal{C}) \) denotes the Boolean closure of \( \mathcal{C} \), i.e., the closure under union, 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).

1.1. The Logical Approach to Regular Languages

We relate to an arbitrary alphabet $A = \{a,\ldots\}$ the signatures $\varrho = \{\leq, Q_a, \ldots\}$ and $\sigma = \{\leq, Q_a, \ldots, \bot, \top, p, s\}$, 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 $u = u_0 \ldots u_n \in A^+$ may be considered as a structure $u = (\{0, \ldots, n\}; \leq, Q_a, \ldots)$ of signature $\sigma$, where $\leq$ has its usual meaning, $Q_a(a \in A)$ are unary predicates on $\{0, \ldots, n\}$ defined by $Q_a(i) \iff u_i = a$, the symbols $\bot$ and $\top$ denote the least and the greatest elements, while $p$ and $s$ are respectively the predecessor and successor functions on $\{0, \ldots, n\}$ (with $p(0) = 0$ and $s(n) = n$). Similarly, a word $v = v_1 \ldots v_n \in A^*$ may be considered as a structure $v = (\{1, \ldots, n\}; \leq, Q_a, \ldots)$ of signature $\varrho$. For a sentence $\phi$ of $\sigma$ (resp., $\varrho$), let $L_\phi = \{u \in A^+ | u \models \phi\}$ (resp., $L_\phi = \{v \in A^* | v \models \phi\}$). Sentences $\phi, \psi$ are treated as equivalent when $L_\phi = L_\psi$. A language is $\text{FO}_\sigma$-definable (resp., $\text{FO}_\varrho$-definable) if it is of the form $L_\phi$, where $\phi$ ranges over first-order sentences of $\sigma$ (resp., $\varrho$). We denote by $\Sigma_k^\sigma$ (resp., $\Pi_k^\varrho$) the class of languages that can be defined by a sentence of $\sigma$ having at most $k - 1$ quantifier alternations, starting with an existential (resp., universal) quantifier. $\Sigma_k^\sigma$ and $\Pi_k^\varrho$ are defined analogously.

It is well-known that the class of $\text{FO}_\sigma$-definable languages (and $\text{FO}_\varrho$-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: The classes $\Sigma_k^\sigma$, $\Pi_k^\varrho$, and $\text{BC}(\Sigma_k^\sigma)$ coincide with the classes of the Straubing-Thérien hierarchy \cite{PST}, while the classes $\Sigma_k^\varrho$, $\Pi_k^\varrho$, and $\text{BC}(\Sigma_k^\varrho)$ coincide with the classes of the dot-depth hierarchy \cite{Hof}. We will consider also some enrichments of the signature $\sigma$. Namely, for any positive integer $d$ let $\tau_d$ be the signature $\sigma \cup \{P_d^0, \ldots, P_d^{d-1}\}$, where $P_d^i$ is the unary predicate true on the positions of a word which are equivalent to $i$ modulo $d$. By $\text{FO}_{\tau_d}$-definable language we mean any language of the form $L_\phi$, where $\phi$ is a first-order sentence of signature $\tau_d$. Note that signature $\tau_1$ is essentially the same as $\sigma$ because $P_1^0$ is the valid predicate. In contrast, for $d > 1$ the $\text{FO}_{\tau_d}$-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. \cite{BCST93,Str94} defined quasi-aperiodic languages and showed that this class coincides with the class of $\text{FO}_\tau$-definable languages. With the same proof we obtain the equality of the class of $\tau$-quasi-aperiodic languages and the class of $\text{FO}_{\tau_d}$-definable languages \cite{Sel04}. It was observed in the same paper that $\Sigma_n^\sigma = \bigcup_d \Sigma_n^{\tau_d}$ for each $n > 0$, where $\Sigma_n$ with an upper index denotes the class of regular languages defined by $\Sigma_n$-sentences of the corresponding signature in the upper index.

**Theorem 1.1.** For the following classes $\mathcal{D}$ it is decidable whether a given dfa $M$ accepts a language in $\mathcal{D}$: $\text{BC}(\Sigma_2^\sigma)$ \cite{Sim73}, $\text{BC}(\Sigma_3^\sigma)$ for $|A| = 2$ \cite{Str88}, $\text{BC}(\Sigma_4^\sigma)$ \cite{Kna83}, $\text{BC}(\Sigma_5^\sigma)$ \cite{MPT00}.

We do not know the decidability of $\text{BC}(\Sigma_1^\sigma)$. However, it is likely to be a generalization of Knast’s proof \cite{Kna83}.
1.2. Preliminaries on the Classes $C_k^d(n)$

We will also refer to ‘local’ versions of the BH’s over $\Sigma_1^*$ and $\Sigma_1^{\tau d}$ [Ste85a, GS01a, Sel01, Sel04]. For any $k \geq 0$ the following partial order on $\Sigma^+$ was studied in [Ste85a, GS01a, Sel01, Sel04]: $u \leq_k v$, 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 $f : u \rightarrow v$. Here $p_k(u)$ (resp., $s_k(u)$) 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 $\leq_0$ is the subword relation.

**Definition 1.2** ([Sel04]). Let $k \geq 0$ and $d > 0$.

1. We say that a $k$-embedding $f : u \rightarrow v$ 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 $f : u \rightarrow v$.

3. With $C_k^d$ we denote the class of all upper sets in $(A^+: \leq_k^d)$.

Note that for $d = 1$ the order $\leq_k^1$ coincides with $\leq_k$. By an alternating $\leq_k^d$-chain of length $n$ for a set $L$ we mean a sequence $(x_0, \ldots, x_n)$ such that $x_0 \leq_k^d x_1 \cdots \leq_k^d x_n$ and $x_i \in L$ if and only if $x_{i+1} \not\in L$ for every $i < n$. The chain is called $1$-alternating if $x_0 \in L$, otherwise it is called $0$-alternating.

**Proposition 1.3** ([GS01a, Sel01, Sel04]). For all $L \subseteq A^+$ and $n \geq 1$, $L \in C_k^d(n)$ if and only if $L$ has no $1$-alternating chain of length $n$ in $(A^+: \leq_k^d)$.

Moreover, $(A^+: \leq_k^d)$ is a well partial order, $\Sigma_1^{\tau d} = \bigcup_k C_k^d$, and $\Sigma_1^d = \bigcup_{k,d} C_k^d$ [GS01a, Sel01, Sel04].

**Theorem 1.4** ([Ste85a]). It is decidable whether a given dfa accepts a language in $BC(C_k^d)$. For $d > 1$ it is not known whether $BC(C_k^d)$ is decidable. However, we expect that this can be shown by generalizing the proof in [Ste85a].

2. Efficient Algorithms for $C_k^d(n)$

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 $C_k^0(n)$ [SW93] and $C_k^1(n)$ [Sch01], in this section we explain the construction of a nondeterministic, logarithmic-space membership algorithm for the classes $C_k^d(n)$. This is the first efficient membership test for this general case. Our explanation has an exemplary character, since it shows how a suitable characterization of a Boolean hierarchy can be turned into an efficient membership test. Our results in later sections use similar, but more complicated constructions.

\footnote{For all $n$, the classes $C_k^0(n)$ and $\Sigma_1^d(n)$ coincide up to the empty word, i.e., $C_k^0(n) = \{L \cap A^+ \mid L \in \Sigma_1^d(n)\}$.}
We start with the easiest case \( k = 0 \) and \( d = 1 \), i.e., with the classes \( C_0^1(n) \). By Proposition 1.3,

\[
L \not\in C_0^1(n) \iff L \text{ has a 1-alternating } \leq_0 \text{-chain of length } n. \tag{2.1}
\]

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.

```plaintext
0 // On input of a deterministic, finite automaton \( M = (A, Z, \delta, z_0, F) \) the algorithm tests whether \( L(M) \in C_0^1(n) \).
1 let \( s_0 = \cdots = s_n = z_0 \)
2 do
3 nondeterministically choose \( a \in A \) and \( j \in [0, n] \)
4 for \( i = j \) to \( n \)
5 \( s_i = \delta(s_i, a) \) // stands for the imaginary command \( w_i := w_i a \)
6 next i
7 until \( \forall i, [s_i \in F \Rightarrow i \text{ is even}] \)
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 \( s_j, \ldots, s_n \) appropriately. By doing so, we guess all possible chains \( w_0 \leq_0 \cdots \leq_0 w_n \) in such a way that we know the states \( s_i = \delta(z_0, w_i) \). This allows us to easily verify the right-hand side of (2.1) in line 7. Hence, testing non-membership in \( C_0^1(n) \) is in NL. By NL = coNL [Imm88, Sze87], the membership test also belongs to NL.

The algorithm can be modified such that it works for \( C_0^d(n) \) where \( d \) is arbitrary: For this we have to make sure that the guessed \( \leq_0 \)-chain is even a \( \leq_d \)-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 \( l_i = l_{i+1} \) whenever \( j \leq i < n \) (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 \( C_0^d(n) \) belongs to NL.

Finally, we adapt the algorithm to make it work for \( C_k^d(n) \) where \( d \) and \( k \) are arbitrary. So we have to make sure that the guessed \( \leq_d^k \)-chain is even a \( \leq_k^d \)-chain. For this, let us consider an extension \( u \leq_k^d w \) where \( u, w \in A^{>k} \). The \((k, d)\)-embedding \( f \) that is used in the definition of \( u \leq_k^d w \) ensures that for all \( i \) it holds that in \( u \) at position \( i \) there are the same \( k + 1 \) letters as in \( w \) at position \( f(i) \). Therefore, a word extension \( u \leq_k^d w \) can be split into a series of elementary extensions of the form \( u_1 u_2 \leq_d^k u_1 v u_2 \) such that the length \( k \) prefixes of \( u_2 \) and \( v u_2 \) are equal. The latter is called the prefix condition. Moreover, we can always make sure that the positions in \( u \) at which the elementary extensions occur form a strictly increasing sequence. This allows us to guess the words in the \( \leq_k^d \)-chain letter by letter. Now the algorithm can test the prefix 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 \( j \leq i < n \), then the condition \( v_i = v_{i+1} \) is tested. The latter makes sure that elementary extensions \( u_1u_2 \leq d_0 \) satisfy the prefix condition and hence the involved words are even in \( \leq d \) relation. This modified algorithm shows the following.

**Theorem 2.1.** \( \{ M \mid M \text{ is a det. finite automaton and } L(M) \in \mathcal{C}_k^d(n) \} \in \text{NL for } k \geq 0, \ d \geq 1. \)

We now explain why the above idea does not immediately lead to a nondeterministic, logarithmic-space membership algorithm for the classes \( \Sigma_1^d(n) \), although an alternating chain characterization for \( \Sigma_1^d(n) \) is known from \[\text{GS01a}\]. 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 states of length \( k \)

However, the characterization of the classes \( \Sigma_1^d(n) \) \[\text{GS01a}\] is unsuitable for our algorithm: In order to verify the forbidden-chain condition, we have to guess a chain of so-called structured words and have to make sure that certain parts \( u \) in these words are \( M \)-idempotent (i.e., \( \delta(s, u) = \delta(s, uu) \) for all states \( s \)). Again we would try to guess the words 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.

In a similar way one observes that the known characterization of the classes \( \Sigma_1^{\alpha_2}(n) \) \[\text{Sel04}\] cannot be used for the construction of an efficient membership test. So new characterizations of \( \Sigma_1^d(n) \) and \( \Sigma_1^{\alpha_d}(n) \) are needed in order to obtain efficient membership algorithms.

### 3. New Characterizations of Boolean-Hierarchy Classes

In this section we develop new alternating-chain characterizations that allow the construction of efficient algorithms deciding membership for the Boolean hierarchies over \( \Sigma_1^d \), \( \Sigma_1^{\alpha_d} \), and \( \Sigma_2^d \) 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 efficient algorithms.

#### 3.1. Marked Words

For a fixed finite alphabet \( A \), let \( \mathcal{A} \triangleq \{ [a, u] \mid a \in A, u \in A^* \} \) be the corresponding marked alphabet. Words over \( \mathcal{A} \) are called marked words. For \( w \in A^* \) with \( w = [a_1, u_1] \cdots [a_m, u_m] \) let \( \overline{w} \triangleq a_1 \cdots a_m \in A^* \) be the corresponding unmarked word. Sometimes we use the functional notation \( f_1(w) = a_1u_1' \cdots a_mu_m' \), i.e., \( f_0(w) = \overline{w} \). Clearly, \( f_0 : A^* \rightarrow A^* \) is a surjection. For \( x = x_1 \cdots x_m \in A^+ \) and \( u \in A^* \) we define \([x, u] \triangleq [x_1, \varepsilon] \cdots [x_{m-1}, \varepsilon][x_m, u] \).

Next we define a relation on marked words. For \( w, w' \in A^* \) we write \( w \preceq w' \) if and only if there exist \( m \geq 0 \), marked words \( x_i, z_i \in A^* \), and marked letters \( b_i = [a_i, u_i] \in \mathcal{A} \) where \( u_i \in A^+ \) s.t.

\[
\begin{align*}
    w &= x_0b_1 x_1b_2 x_2 \cdots b_m x_m, \\
    w' &= x_0b_1 z_1b_1 x_1b_2 z_2b_2 x_2 \cdots b_m z_mb_m x_m.
\end{align*}
\]

We call \( b_i \) the context letter of the insertion \( z_ib_i \). We write \( w \preceq^d w' \) if \( w \preceq w' \) and \(|f_0(z_ib_i)| \equiv 0 \pmod{d} \) 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 $s \xrightarrow{w} M^t$, if $\delta(s, w) = t$ and for all $i, \delta(s, a_1 \cdots a_i) = \delta(s, a_1 \cdots a_i u_i)$. So $s \xrightarrow{w} M^t$ 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 $t \in Z$, $s_0 \xrightarrow{w} M^t$ and denote by $B_M$ the set of marked words that are $M$-consistent. Every $M$-consistent word has the following nice property.

**Proposition 3.1.** For $w = [c_1, u_1] \cdots [c_m, u_m] \in B_M$ and all $j \geq 0$, $f_0(w) \equiv_M c_1 u_1^j \cdots c_m u_m^j$.

### 3.2. New Characterization of the Classes $\Sigma^c_1(n)$ and $\Sigma^τ_1(n)$

We extend the known characterization of the classes $\Sigma^c_1(n)$ [Sel01] and add a characterization in terms of alternating chains on $M$-consistent marked words. Because we can also restrict the length of the labels of 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 3.2.** The following is equivalent for $d, n \geq 1$, a dfa $M$, $c = |M|^{|M|}$, and $L = L(M) \subseteq A^+$.

1. $L \in \Sigma^c_1(n)$
2. $f_0^{-1}(L)$ has no 1-alternating chain of length $n$ in $(B_M; \preceq^d)$
3. $f_0^{-1}(L)$ has no 1-alternating chain of length $n$ in $(B_M; \preceq^d)$

The case $d = 1$ is an alternative to the known characterization of the classes $\Sigma^c_1(n)$ [GS01a].

**Theorem 3.3.** Let $M$ be a dfa, $L = L(M) \subseteq A^+$ and $n \geq 1$. Then $L \in \Sigma^c_1(n)$ if and only if $f_0^{-1}(L)$ has no 1-alternating chain of length $n$ in $(B_M; \preceq)$.

We can give an upper bound on $d$ for languages in $\Sigma^c_1(n)$.

**Theorem 3.4.** For every dfa $M$, $c = |M|^{|M|}$, and $d = c!$, $L(M) \in \Sigma^c_1(n) \Rightarrow L(M) \in \Sigma^τ_1(n)$.

### 3.3. Characterization of the Classes $\Sigma^c_2(n)$ for $|A| = 2$

We obtain an alternating-chain characterization for the classes of the Boolean hierarchy over $\Sigma^c_2$ for the case $|A| = 2$. This allows us to prove the first decidability result for this hierarchy. Note that only in case $|A| = 2$ decidability of $BC(\Sigma^c_2)$ [Str88] and $\Sigma^c_2$ [GS01b] is known.

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 $\alpha(u_i) = A$.

**Theorem 3.5.** 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 $L \in \Sigma^c_2(n)$ if and only if $f_0^{-1}(L)$ has no 1-alternating chain $(w_0, \ldots, w_n)$ in $(B_M; \preceq)$ such that all $w_i$ satisfy the alphabet condition.
4. Decidability and Complexity

The alternating-chain characterizations from the last sections can be used for the construction of efficient algorithms for testing the membership in these classes. As corollaries we obtain new decidability results: the classes $\Sigma_1^\sigma(n)$ and $\Sigma_2^\sigma(n)$ for $|A| = 2$ are decidable.

The characterizations given in Theorems 3.2 and 3.3 allow the construction of non-deterministic, logarithmic-space membership tests for $\Sigma_1^\sigma(n)$ and $\Sigma_1^{\tau d}(n)$.

**Theorem 4.1.** For all $n \geq 1$, \( \{ M \mid M \text{ is a det. finite automaton and } L(M) \in \Sigma_1^\sigma(n) \} \in \text{NL} \).

**Theorem 4.2.** For all $n \geq 1$, \( \{ M \mid M \text{ is a det. finite automaton and } L(M) \in \Sigma_1^{\tau d}(n) \} \in \text{NL} \).

**Remark 4.3.** Unfortunately, we do not obtain NL decidability for the classes $\Sigma_1^\sigma(n)$. The reason is that the $d$ in Theorem 3.4 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 $\Sigma_1^\sigma(n)$ is decidable in NL. Although $d$ is very large, it is still computable from the automaton $M$ which implies the decidability of all levels $\Sigma_1^\sigma(n)$. This settles a question left open in [Sel04].

**Theorem 4.4.** For all $n \geq 1$, \( \{ M \mid M \text{ is a det. finite automaton and } L(M) \in \Sigma_1^\sigma(n) \} \) is decidable.

**Theorem 4.5.** For all $n \geq 1$,
\[
\{ M \mid M \text{ is a det. finite automaton over the alphabet } \{a, b\} \text{ and } L(M) \in \Sigma_2^\sigma(n) \} \in \text{NL}.
\]

5. Exact Complexity Estimations

With the exception of $\Sigma_1^\sigma(n)$, 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 $\text{FO}_\tau$ and $\text{FO}_{\tau d}$ are PSPACE-complete and hence are strictly more complex.

**Proposition 5.1.** Let $C$ be any class of regular quasi-aperiodic languages over $A$ with $|A| \geq 2$ and $\emptyset \in C$. Then it is NL-hard to decide whether a given dfa $M$ accepts a language in $C$.

Together with the upper bounds established in the previous sections this immediately implies the following exact complexity estimations.

**Theorem 5.2.** Let $k \geq 0$, $n \geq 1$, $d \geq 1$ and $C$ is one of the classes $\mathcal{C}_k^\sigma(n)$, $\Sigma_1^\sigma(n)$, $\Sigma_1^{\tau d}(n)$, or $\Sigma_2^\sigma(n)$ for $|A| = 2$. Then \( \{ M \mid M \text{ is a det. finite automaton and } L(M) \in C \} \) is NL-complete.

We conclude this section with a corollary of the PSPACE-completeness of deciding $\text{FO}_\tau$ which was established by Stern [Ste85b] and by Cho and Huynh [CH91]. It shows that the complexity of deciding the classes $\text{FO}_\tau$ and $\text{FO}_{\tau d}$ is strictly higher than the complexity of deciding the classes mentioned in Theorem 4.2. (Note that NL is closed under logspace many-one reductions, $\text{NL} \subseteq \text{DSPACE}(\log^2 n)$ [Sav70] and $\text{DSPACE}(\log^2 n) \subseteq \text{PSPACE}$ [IS63]. Hence the classes $\text{FO}_\tau$ and $\text{FO}_{\tau d}$ cannot be decided in NL.)

**Theorem 5.3.** The classes $\text{FO}_\tau$ and $\text{FO}_{\tau d}$ are PSPACE-complete.
6. Conclusions

The results of this paper (as well as several previous facts that appeared in the literature) show that more and more decidable levels of hierarchies turn out to be decidable in NL. One is tempted to strengthen the well-known challenging conjecture of decidability of the dot-depth hierarchy to the conjecture that all levels of reasonable hierarchies of first-order definable regular languages are decidable in NL. At least, it seems instructive to ask this question about any level of such a hierarchy known to be decidable.

In this paper we considered the complexity of classes of regular languages only w.r.t. the representation of regular languages by dfas. Similar questions are probably open for other natural representations of regular languages, like nondeterministic finite automata and propositions of monadic second order, first order or temporal logics.

Acknowledgements

We are grateful to Klaus W. Wagner for many helpful discussions. We would like to thank the anonymous referees for their valuable comments.

References


