Going higher in the First-order Quantifier Alternation Hierarchy on Words*

Thomas Place and Marc Zeitoun

LaBRI, Université de Bordeaux, France

Abstract. We investigate the quantifier alternation hierarchy in first-order logic on finite words. Levels in this hierarchy are defined by counting the number of quantifier alternations in formulas. We prove that one can decide membership of a regular language to the levels $\mathcal{B}\Sigma_2$ (boolean combination of formulas having only 1 alternation) and Σ_3 (formulas having only 2 alternations beginning with an existential block). Our proof works by considering a deeper problem, called separation, which, once solved for lower levels, allows us to solve membership for higher levels.

The connection between logic and automata theory is well known and has a fruitful history in computer science. It was first observed when Büchi, Elgot and Trakhtenbrot proved independently that the regular languages are exactly those that can be defined using a monadic second-order logic (MSO) formula. Since then, many efforts have been made to investigate and understand the expressive power of relevant fragments of MSO. In this field, the yardstick result is often to prove *decidable characterizations*, *i.e.*, to design an algorithm which, given as input a regular language, decides whether it can be defined in the fragment under investigation. More than the algorithm itself, the main motivation is the insight given by its proof. Indeed, in order to prove a decidable characterization, one has to consider and understand *all* properties that can be expressed in the fragment.

The most prominent fragment of MSO is first-order logic (FO) equipped with a predicate "$<$" for the linear-order. The expressive power of FO is now well-understood over words and a decidable characterization has been obtained. The result, Schützenberger's Theorem [20,10], states that a regular language is definable in FO if and only if its syntactic monoid is aperiodic. The syntactic monoid is a finite algebraic structure that can effectively be computed from any representation of the language. Moreover, aperiodicity can be rephrased as an equation that needs to be satisfied by all elements of the monoid. Therefore, Schützenberger's Theorem can indeed be used to decide definability in FO.

In this paper, we investigate an important hierarchy inside FO, obtained by classifying formulas according to the number of quantifier alternations in their prenex normal form. More precisely, an FO formula is Σ_i if its prenex normal form has at most $(i-1)$ quantifier alternations and starts with a block of existential quantifiers. The hierarchy also involves the classes $\mathcal{B}\Sigma_i$ of boolean combinations of Σ_i formulas, and the classes Δ_i of languages that can be defined

* Supported by ANR 2010 BLAN 0202 01 FREC

by both a Σ_i and the negation of a Σ_i formula. The quantifier alternation hierarchy was proved to be strict [6,31]: $\Delta_i \subsetneq \Sigma_i \subsetneq \mathcal{B}\Sigma_i \subsetneq \Delta_{i+1}$. In the literature, many efforts have been made to find decidable characterizations of levels of this well-known hierarchy.

Despite these efforts, only the lower levels are known to be decidable. The class $\mathcal{B}\Sigma_1$ consists exactly of all piecewise testable languages, *i.e.*, such that membership of a word only depends on its subwords up to a fixed size. These languages were characterized by Simon [21] as those whose syntactic monoid is $\mathcal{J}$-trivial. A decidable characterization of Σ_2 (and hence of Δ_2 as well) was proven in [3]. For Δ_2, the literature is very rich [27]. For example, these are exactly the languages definable by the two variable restriction of FO [29]. These are also those whose syntactic monoid is in the class DA [14]. For higher levels in the hierarchy, getting decidable characterizations remained an important open problem. In particular, the case of $\mathcal{B}\Sigma_2$ has a very rich history and a series of combinatorial, logical, and algebraic conjectures have been proposed over the years. We refer to [12,2,11,13] for an exhaustive bibliography. So far, the only known effective result was partial, working only when the alphabet is of size 2 [25]. One of the main motivations for investigating this class in formal language theory is its ties with two other famous hierarchies defined in terms of regular expressions. In the first one, the *Straubing-Thérien hierarchy* [23,28], level i corresponds exactly to the class $\mathcal{B}\Sigma_i$ [30]. In the second one, the *dot-depth hierarchy* [7], level i corresponds to adding a predicate for the successor relation in $\mathcal{B}\Sigma_i$ [30]. Proving decidability for $\mathcal{B}\Sigma_2$ immediately proves decidability of level 2 in the Straubing-Thérien hierarchy, but also in the dot-depth hierarchy using a reduction by Straubing [24].

In this paper, we prove decidability for $\mathcal{B}\Sigma_2$, Δ_3 and Σ_3. These new results are based on a deeper decision problem than decidable characterizations: the separation problem. Fix a class Sep of languages. The Sep-separation problem amounts to decide whether, given two input regular languages, there exists a third language in Sep containing the first language while being disjoint from the second one. This problem generalizes decidable characterizations. Indeed, since regular languages are closed under complement, testing membership in Sep can be achieved by testing whether the input is Sep-separable from its complement. Historically, the separation problem was first investigated as a special case of a deep problem in semigroup theory, see [1]. This line of research gave solutions to the problem for several classes. However, the motivations are disconnected from our own, and the proofs rely on deep, purely algebraic arguments. Recently, a research effort has been made to investigate this problem from a different perspective, with the aim of finding new and self-contained proofs relying on elementary ideas and notions from language theory only [8,16,19,17]. This paper is a continuation of this effort: we solve the separation problem for Σ_2, and use our solution as a basis to obtain decidable characterizations for $\mathcal{B}\Sigma_2$, Δ_3 and Σ_3.

Our solution works as follows: given two regular languages, one can easily construct a monoid morphism $\alpha : A^* \to M$ that recognizes both of them. We then design an algorithm that computes, inside the monoid M, enough Σ_2-related information to answer the Σ_2-separation question for *any* pair of languages that are recognized by α. It turns out that it is also possible (though much more difficult) to use this information to obtain decidability of $\mathcal{B}\Sigma_2$, Δ_3 and Σ_3. This information amounts to the notion of Σ_2-chain, our main tool in the paper. A Σ_2-chain is an *ordered sequence* $s_1, \dots, s_n \in M$ that witnesses a property of α wrt. Σ_2. Let us give some intuition in the case $n = 2$ – which is enough to make the link with Σ_2-separation. A sequence s_1, s_2 is a Σ_2-chain if any Σ_2 language containing all words in $\alpha^{-1}(s_1)$ intersects $\alpha^{-1}(s_2)$. In terms of separation, this means that $\alpha^{-1}(s_1)$ is *not* separable from $\alpha^{-1}(s_2)$ by a Σ_2 definable language.

This paper contains three main separate and difficult new results: (1) an algorithm to compute Σ_2-chains – hence Σ_2-separability is decidable (2) decidability of Σ_3 (decidability of Δ_3 is an immediate consequence), and (3) decidability of $\mathcal{B}\Sigma_2$. Computing Σ_2-chains is achieved using a fixpoint algorithm that starts with trivial Σ_2-chains such as $s, s, \dots, s$, and iteratively computes more Σ_2-chains until a fixpoint is reached. Note that its completeness proof relies on the Factorization Forest Theorem of Simon [22]. This is not surprising, as the link between this theorem and the quantifier alternation hierarchy was already observed in [14,4].

For Σ_3, we prove a decidable characterization via an equation on the syntactic monoid of the language. This equation is parametrized by the set of Σ_2-chains of length 2. In other words, we use Σ_2-chains to abstract an infinite set of equations into a single one. The proof relies again on the Factorization Forest Theorem of Simon [22] and is actually generic to all levels in the hierarchy. This means that for any i, we define a notion of Σ_i-chain and characterize Σ_{i+1} using an equation parametrized by Σ_i-chains of length 2. However, decidability of Σ_{i+1} depends on our ability to compute the Σ_i-chains of length 2, which we can only do for $i = 2$.

Our decidable characterization of $\mathcal{B}\Sigma_2$ is the most difficult result of the paper. As for Σ_3, it is presented by two equations parametrized by Σ_2-chains (of length 2 and 3). However, the characterization is this time specific to the case $i = 2$. This is because most of our proof relies on a deep analysis of our algorithm that computes Σ_2-chains, which only works for $i = 2$. The equations share surprising similarities with the ones used in [5] to characterize a totally different formalism: boolean combination of open sets of infinite trees. In [5] also, the authors present their characterization as a set of equations parametrized by a notion of "chain" for open sets of infinite trees (although their "chains" are not explicitly identified as a separation relation). Since the formalisms are of different nature, the way these chains and our Σ_2-chains are constructed are completely independent, which means that the proofs are also mostly independent. However, once the construction analysis of chains has been done, several combinatorial arguments used to make the link with equations are analogous. In particular, we reuse and adapt definitions from [5] to present these combinatorial arguments in our proof. One could say that the proofs are both (very different) setups to apply similar combinatorial arguments in the end.

Organization. We present definitions on languages and logic in Sections 1 and 2 respectively. Section 3 is devoted to the presentation of our main tool: Σ_i-chains.

In Section 4, we give our algorithm computing Σ_2-chains. The two remaining sections present our decidable characterizations, for Σ_3 and Δ_3 in Section 5 and for $\mathcal{B}\Sigma_2$ in Section 6. Due to lack of space, proofs can be found in [18].

1 Words and Algebra

Words and Languages. We fix a finite alphabet A and we denote by A^* the set of all words over A. If u, v are words, we denote by $u \cdot v$ or uv the word obtained by concatenation of u and v. If $u \in A^*$ we denote by $\mathsf{alph}(u)$ its alphabet, *i.e.*, the smallest subset B of A such that $u \in B^*$. A *language* is a subset of A^*. In this paper we consider regular languages: these are languages definable by *nondeterministic finite automata*, or equivalently by *finite monoids*. In the paper, we only work with the monoid representation of regular languages.

Monoids. A *semigroup* is a set S equipped with an associative multiplication denoted by '$\cdot$'. A *monoid* M is a semigroup in which there exists a neutral element denoted 1_M. In the paper, we investigate classes of languages, such as Σ_i, that are not closed under complement. For such classes, it is known that one needs to use *ordered monoids*. An ordered monoid is a monoid endowed with a partial order '$\leqslant$' which is compatible with multiplication: $s \leqslant t$ and $s' \leqslant t'$ imply $ss' \leqslant tt'$. Given any finite semigroup S, it is well known that there is a number $\omega(S)$ (denoted by ω when S is understood from the context) such that for each element s of S, s^ω is an idempotent: $s^\omega = s^\omega \cdot s^\omega$.

Let L be a language and M be a monoid. We say that L *is recognized by* M if there exists a monoid morphism $\alpha : A^* \to M$ and an *accepting set* $F \subseteq M$ such that $L = \alpha^{-1}(F)$. It is well known that a language is regular if and only if it can be recognized by a *finite monoid*.

Syntactic Ordered Monoid of a Language. The *syntactic preorder* $\leqslant_L$ of a language L is defined as follows on pairs of words in A^*: $w \leqslant_L w'$ if for all $u, v \in A^*$, $uwv \in L \Rightarrow uw'v \in L$. Similarly, we define $\equiv_L$, the *syntactic equivalence* of L as follows: $w \equiv_L w'$ if $w \leqslant_L w'$ and $w' \leqslant_L w$. One can verify that $\leqslant_L$ and $\equiv_L$ are compatible with multiplication. Therefore, the quotient M_L of A^* by $\equiv_L$ is an ordered monoid for the partial order induced by the preorder $\leqslant_L$. It is well known that M_L can be effectively computed from L. Moreover, M_L recognizes L. We call M_L the *syntactic ordered monoid of* L and the associated morphism the *syntactic morphism*.

Separation. Given three languages L, L_0, L_1, we say that L *separates* L_0 from L_1 if $L_0 \subseteq L$ and $L_1 \cap L = \emptyset$. Set X as a class of languages, we say that L_0 is X*-separable* from L_1 if some language in X separates L_0 from L_1. Observe that when X is not closed under complement, the definition is not symmetrical: L_0 could be X-separable from L_1 while L_1 is not X-separable from L_0.

When working on separation, we consider as input two regular languages L_0, L_1. It will be convenient to have a *single* monoid recognizing both of them, rather than having to deal with two objects. Let M_0, M_1 be monoids recognizing L_0, L_1 together with the morphisms α_0, α_1, respectively. Then, $M_0 \times M_1$

equipped with the componentwise multiplication $(s_0, s_1) \cdot (t_0, t_1) = (s_0t_0, s_1t_1)$ is a monoid that recognizes both L_0 and L_1 with the morphism $\alpha : w \mapsto (\alpha_0(w), \alpha_1(w))$. From now on, we work with such a single monoid recognizing both languages.

Chains and Sets of Chains. Set M as a finite monoid. A *chain* for M is a word over the alphabet M, *i.e.*, an element of M^*. A remark about notation is in order here. A word is usually denoted as the concatenation of its letters. Since M is a monoid, this would be ambiguous here since st could either mean a word with 2 letters s and t, or the product of s and t in M. To avoid confusion, we will write $(s_1, \dots, s_n)$ a chain of length n on the alphabet M.

In the paper, we will consider both sets of chains (denoted by $\mathcal{T}, \mathcal{S}, \dots$) and sets of sets of chains (denoted by $\mathfrak{T}, \mathfrak{S}, \dots$). In particular, if $\mathfrak{T}$ is a set of sets of chains, we define $\downarrow\mathfrak{T}$, the *downset* of $\mathfrak{T}$, as the set:

$$\downarrow\mathfrak{T} = \{\mathcal{T} \mid \exists \mathcal{S} \in \mathfrak{T},\ \mathcal{T} \subseteq \mathcal{S}\}.$$

We will often restrict ourselves to considering only chains of a given fixed length. For $n \in \mathbb{N}$, observe that M^n, the set of chains of length n, is a monoid when equipped with the componentwise multiplication. Similarly the set 2^{M^n} of sets of chains of length n is a monoid for the operation: $\mathcal{S} \cdot \mathcal{T} = \{\bar{s}\bar{t} \in M^n \mid \bar{s} \in \mathcal{S} \quad \bar{t} \in \mathcal{T}\}$.

2 First-Order Logic and Quantifier Alternation Hierarchy

We view words as logical structures made of a sequence of positions labeled over A. We denote by $<$ the linear order over the positions. We work with first-order logic FO using unary predicates P_a for all $a \in A$ that select positions labeled with an a, as well as a binary predicate for the linear order $<$. The *quantifier rank* of an FO formula is the length of its longest sequence of nested quantifiers.

One can classify first-order formulas by counting the number of alternations between $\exists$ and $\forall$ quantifiers in the prenex normal form of the formula. Set $i \in \mathbb{N}$, a formula is said to be Σ_i (resp. Π_i) if its prenex normal form has $i-1$ quantifier alternations (*i.e.*, i blocks of quantifiers) and starts with an $\exists$ (resp. $\forall$) quantification. For example, a formula whose prenex normal form is

$$\forall x_1 \forall x_2 \exists x_3 \forall x_4\ \varphi(x_1, x_2, x_3, x_4) \quad \text{(with } \varphi \text{ quantifier-free)}$$

is Π_3. Observe that a Π_i formula is by definition the negation of a Σ_i formula. Finally, a $\mathcal{B}\Sigma_i$ formula is a boolean combination of Σ_i formulas. For $X = \text{FO}, \Sigma_i, \Pi_i$ or $\mathcal{B}\Sigma_i$, we say that a language L is X-definable if it can be defined by an X-formula. Finally, we say that a language is Δ_i-definable if it can be defined by *both* a Σ_i and a Π_i formula. It is known that this gives a strict infinite hierarchy of classes of languages as represented in Figure 1.

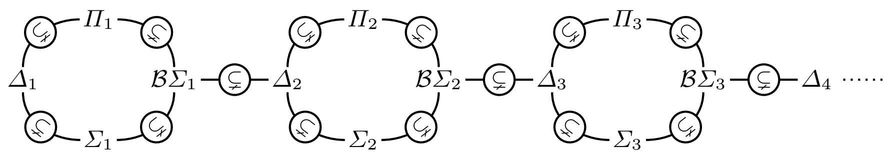

Fig. 1. Quantifier Alternation Hierarchy

Preorder for Σ_i. Let $w, w' \in A^*$ and $k, i \in \mathbb{N}$. We write $w \lesssim_i^k w'$ if any Σ_i formula of quantifier rank k satisfied by w is also satisfied by w'. Observe that since a Π_i formula is the negation of a Σ_i formula, we have $w \lesssim_i^k w'$ iff any Π_i formula of quantifier rank k satisfied by w' is also satisfied by w. One can verify that $\lesssim_i^k$ is a preorder for all k, i. Moreover, by definition, a language L can be defined by a Σ_i formula of rank k iff L is saturated by $\lesssim_i^k$, *i.e.*, for all $w \in L$ and all w' such that $w \lesssim_i^k w'$, we have $w' \in L$.

3 Σ_i-Chains

We now introduce the main tool of this paper: Σ_i*-chains*. Fix a level i in the quantifier alternation hierarchy and $\alpha : A^* \to M$ a monoid morphism. A Σ_i*-chain* for α is a chain $(s_1, \dots, s_n) \in M^*$ such that for arbitrarily large $k \in \mathbb{N}$, there exist words $w_1 \lesssim_i^k \cdots \lesssim_i^k w_n$ mapped respectively to $s_1, \dots, s_n$ by α. Intuitively, this contains information about the limits of the expressive power of the logic Σ_i with respect to α. For example, if (s_1, s_2) is a Σ_i-chain, then any Σ_i language that contains all words of image s_1 must also contain at least one word of image s_2.

In this section, we first give all definitions related to Σ_i-chains. We then present an immediate application of this notion: solving the separation problem for Σ_i can be reduced to computing the Σ_i-chains of length 2.

3.1 Definitions

Σ_i-Chains. Fix i a level in the hierarchy, $k \in \mathbb{N}$ and $B \subseteq A$. We define $\mathcal{C}_i^k[\alpha]$ (resp. $\mathcal{C}_i^k[\alpha, B]$) as the *set of $\Sigma_i[k]$-chains for α* (resp. for (α, B)) and $\mathcal{C}_i[\alpha]$ (resp. $\mathcal{C}_i[\alpha, B]$) as the *set of Σ_i-chains for α* (resp. for (α, B)). For $i = 0$, we set $\mathcal{C}_i[\alpha] = \mathcal{C}_i^k[\alpha] = M^*$. Otherwise, let $\bar{s} = (s_1, \dots, s_n) \in M^*$. We let

- $\bar{s} \in \mathcal{C}_i^k[\alpha]$ if there exist $w_1, \dots, w_n \in A^*$ verifying $w_1 \lesssim_i^k w_2 \lesssim_i^k \cdots \lesssim_i^k w_n$ and for all j, we have $\alpha(w_j) = s_j$. Moreover, $\bar{s} \in \mathcal{C}_i^k[\alpha, B]$ if the words w_j can be chosen so that they satisfy additionally $\mathsf{alph}(w_j) = B$ for all j.
- $\bar{s} \in \mathcal{C}_i[\alpha]$ if for all k, we have $\bar{s} \in \mathcal{C}_i^k[\alpha]$. That is, $\mathcal{C}_i[\alpha] = \bigcap_k \mathcal{C}_i^k[\alpha]$. In the same way, $\mathcal{C}_i[\alpha, B] = \bigcap_k \mathcal{C}_i^k[\alpha, B]$.

One can check that if $i \geqslant 2$, then $\mathcal{C}_i^k[\alpha] = \bigcup_{B \subseteq A} \mathcal{C}_i^k[\alpha, B]$, since the fragment Σ_i can detect the alphabet (*i.e.*, for $i \geqslant 2$, $w \lesssim_i^k w'$ implies $\mathsf{alph}(w) = \mathsf{alph}(w')$). Similarly for $i \geqslant 2$, the set of Σ_i-chains for α is $\mathcal{C}_i[\alpha] = \bigcup_{B \subseteq A} \mathcal{C}_i[\alpha, B]$. Observe that all these sets are closed under subwords. Therefore, by Higman's lemma, we get the following fact.

Fact 1 *For all $i, k \in \mathbb{N}$ and $B \subseteq A$, $\mathcal{C}_i[\alpha, B]$ and $\mathcal{C}_i^k[\alpha, B]$ are regular languages.*

Fact 1 is interesting but essentially useless in our argument, as Higman's lemma provides no way for actually computing a recognizing device for $\mathcal{C}_i[\alpha, B]$.

For any fixed $n \in \mathbb{N}$, we let $\mathcal{C}_{i,n}^k[\alpha, B]$ be the set of $\Sigma_i[k]$-chains of length n for α, B, *i.e.*, $\mathcal{C}_{i,n}^k[\alpha, B] = \mathcal{C}_i^k[\alpha, B] \cap M^n$. We define $\mathcal{C}_{i,n}[\alpha, B], \mathcal{C}_{i,n}^k[\alpha]$ and $\mathcal{C}_{i,n}[\alpha]$ similarly. The following fact is immediate.

Fact 2 *If $B, C \subseteq A$, then $\mathcal{C}_{i,n}^k[\alpha, B] \cdot \mathcal{C}_{i,n}^k[\alpha, C] \subseteq \mathcal{C}_{i,n}^k[\alpha, B \cup C]$. In particular, $\mathcal{C}_{i,n}^k[\alpha]$ and $\mathcal{C}_{i,n}[\alpha]$ (resp. $\mathcal{C}_{i,n}^k[\alpha, B]$ and $\mathcal{C}_{i,n}[\alpha, B]$) are submonoids (resp. subsemigroups) of M^n.*

This ends the definition of Σ_i-chains. However, in order to define our algorithm for computing Σ_2-chains and state our decidable characterization of $\mathcal{B}\Sigma_2$, we will need a slightly refined notion: *compatible sets of chains*.

Compatible Sets of Σ_i-Chains. In some cases, it will be useful to know that several Σ_i-chains with the same first element can be 'synchronized'. For example take two Σ_i-chains (s, t_1) and (s, t_2) of length 2. By definition, for all k there exist words w_1, w_1', w_2, w_2' whose images under α are s, t_1, s, t_2 respectively, and such that $w_1 \lesssim_i^k w_1'$ and $w_2 \lesssim_i^k w_2'$. In some cases (but not all), it will be possible to choose $w_1 = w_2$ for all k. The goal of the notion of compatible sets of chains is to record the cases in which this is true.

Fix i a level in the hierarchy, $k \in \mathbb{N}$ and $B \subseteq A$. We define two sets of sets of chains: $\mathfrak{C}_i^k[\alpha, B]$, the *set of compatible sets of $\Sigma_i[k]$-chains for* (α, B), and $\mathfrak{C}_i[\alpha, B]$, the *set of compatible sets of Σ_i-chains for* (α, B). Let $\mathcal{T}$ be a set of chains, all having the same length n and the same first element s_1.

- $\mathcal{T} \in \mathfrak{C}_i^k[\alpha, B]$ if there exists $w \in A^*$ such that $\mathsf{alph}(w) = B$, $\alpha(w) = s_1$, and for all chains $(s_1, \dots, s_n) \in \mathcal{T}$, there exist $w_2, \dots, w_n \in A^*$ verifying $w \lesssim_i^k w_2 \lesssim_i^k \cdots \lesssim_i^k w_n$, and for all $j = 2, \dots, n$, $\alpha(w_j) = s_j$, and $\mathsf{alph}(w_j) = B$.
- $\mathcal{T} \in \mathfrak{C}_i[\alpha, B]$ if $\mathcal{T} \in \mathfrak{C}_i^k[\alpha, B]$ for all k.

As before we set $\mathfrak{C}_i^k[\alpha]$ and $\mathfrak{C}_i[\alpha]$ as the union of these sets for all $B \subseteq A$. Moreover, we denote by $\mathfrak{C}_{i,n}^k[\alpha, B], \mathfrak{C}_{i,n}[\alpha, B], \mathfrak{C}_{i,n}^k[\alpha]$ and $\mathfrak{C}_{i,n}[\alpha]$ the restriction of these sets to sets of chains of length n (*i.e.*, subsets of 2^{M^n}).

Fact 3 *If $B, C \subseteq A$, then $\mathfrak{C}_{i,n}^k[\alpha, B] \cdot \mathfrak{C}_{i,n}^k[\alpha, C] \subseteq \mathfrak{C}_{i,n}^k[\alpha, B \cup C]$. In particular, $\mathfrak{C}_{i,n}^k[\alpha]$ and $\mathfrak{C}_{i,n}[\alpha]$ (resp. $\mathfrak{C}_{i,n}^k[\alpha, B]$ and $\mathfrak{C}_{i,n}[\alpha, B]$) are submonoids (resp. subsemigroups) of 2^{M^n}.*

3.2 Σ_i-Chains and Separation

We now state a reduction from the separation problem by Σ_i and by Π_i-definable languages to the computation of Σ_i-chains of length 2.

Theorem 4. *Let L_1, L_2 be regular languages and $\alpha : A^* \to M$ be a morphism into a finite monoid recognizing both languages with accepting sets $F_1, F_2 \subseteq M$. Set $i \in \mathbb{N}$. Then the following properties hold:*

1. *L_1 is Σ_i-separable from L_2 iff for all $s_1, s_2 \in F_1, F_2$, $(s_1, s_2) \notin \mathcal{C}_i[\alpha]$.*
2. *L_1 is Π_i-separable from L_2 iff for all $s_1, s_2 \in F_1, F_2$, $(s_2, s_1) \notin \mathcal{C}_i[\alpha]$.*

The proof of Theorem 4, which is parametrized by Σ_i-chains, is standard and identical to the corresponding theorems in previous separation papers, see *e.g.,* [19]. In Section 4, we present an algorithm computing Σ_i-chains of length 2 at level $i = 2$ of the alternation hierarchy (in fact, our algorithm needs to compute the more general notion of sets of compatible Σ_2-chains). This makes Theorem 4 effective for Σ_2 and Π_2.

4 Computing Σ_2-Chains

In this section, we give an algorithm for computing all Σ_2-chains and sets of compatible Σ_2-chains of a given fixed length. We already know by Theorem 4 that achieving this for length 2 suffices to solve the separation problem for Σ_2 and Π_2. Moreover, we will see in Sections 5 and 6 that this algorithm can be used to obtain decidable characterizations for Σ_3, Π_3, Δ_3 and $\mathcal{B}\Sigma_2$. Note that in this section, we only provide the algorithm and intuition on its correctness.

For the remainder of this section, we fix a morphism $\alpha : A^* \to M$ into a finite monoid M. For any fixed $n \in \mathbb{N}$ and $B \subseteq A$, we need to compute the following:

1. the sets $\mathcal{C}_{2,n}[\alpha, B]$ of Σ_2-chains of length n for α.
2. the sets $\mathfrak{C}_{2,n}[\alpha, B]$ of compatible subsets of $\mathcal{C}_{2,n}[\alpha, B]$.

Our algorithm directly computes the second item, *i.e.*, $\mathfrak{C}_{2,n}[\alpha, B]$. More precisely, we compute the map $B \mapsto \mathfrak{C}_{2,n}[\alpha, B]$. Observe that this is enough to obtain the first item since by definition, $\bar{s} \in \mathcal{C}_{2,n}[\alpha, B]$ iff $\{\bar{s}\} \in \mathfrak{C}_{2,n}[\alpha, B]$. Note that going through compatible subsets is necessary for the technique to work, even if we are only interested in computing the map $B \mapsto \mathcal{C}_{2,n}[\alpha, B]$.

Outline. We begin by explaining what our algorithm does. For this outline, assume $n = 2$. Observe that for all $w \in A^*$ such that $\mathsf{alph}(w) = B$, we have $\{(\alpha(w), \alpha(w))\} \in \mathfrak{C}_{2,n}[\alpha, B]$. The algorithm starts from these trivially compatible sets, and then saturates them with two operations that preserve membership in $\mathfrak{C}_{2,n}[\alpha, B]$. Let us describe these two operations. The first one is multiplication: if $\mathcal{S} \in \mathfrak{C}_{2,n}[\alpha, B]$ and $\mathcal{T} \in \mathfrak{C}_{2,n}[\alpha, C]$ then $\mathcal{S} \cdot \mathcal{T} \in \mathfrak{C}_{2,n}[\alpha, B \cup C]$ by Fact 3. The main idea behind the second operation is to exploit the following property of Σ_2:

$$\forall k\ \exists \ell \quad w \lesssim_2^k u, w \lesssim_2^k u' \text{ and } \mathsf{alph}(w') = \mathsf{alph}(w) \quad \Longrightarrow \quad w^{2\ell} \lesssim_2^k u^\ell w' u'^\ell.$$

This is why compatible sets are needed: in order to use this property, we need to have a single word w such that $w \lesssim_2^k u$ and $w \lesssim_2^k u'$, which is information that is not provided by Σ_2-chains. This yields an operation that states that whenever $\mathcal{S}$ belongs to $\mathfrak{C}_{2,n}[\alpha, B]$, then so does $\mathcal{S}^\omega \cdot \mathcal{T} \cdot \mathcal{S}^\omega$, where $\mathcal{T}$ is the set of chains $(1_M, \alpha(w'))$ with $\mathsf{alph}(w') = B$. Let us now formalize this procedure and generalize it to arbitrary length.

Algorithm. As we explained, our algorithm works by fixpoint, starting from trivial compatible sets. For all $n \in \mathbb{N}$ and $B \subseteq A$, we let $\mathfrak{I}_n[B]$ be the set $\mathfrak{I}_n[B] = \big\{\{(\alpha(w), \dots, \alpha(w))\} \mid \mathsf{alph}(w) = B\big\} \subseteq 2^{M_n}$. Our algorithm will start from the function $f_0 : 2^A \to 2^{2^{M^n}}$ that maps any $C \subseteq A$ to $\mathfrak{I}_n[C]$.

Our algorithm is defined for any fixed length $n \geqslant 1$. We use a procedure Sat_n taking as input a mapping $f : 2^A \to 2^{2^{M^n}}$ and producing another such mapping. The algorithm starts from f_0 and iterates Sat_n until a fixpoint is reached.

When $n \geqslant 2$, the procedure Sat_n is parametrized by $\mathcal{C}_{2,n-1}[\alpha, B]$, the sets of Σ_2-chains of length $n-1$, for $B \subseteq A$. This means that in order to use Sat_n, one needs to have previously computed the Σ_2-chains of length $n-1$ with Sat_{n-1}.

We now define the procedure Sat_n. If $\mathcal{S}$ is a set of chains of length $n-1$ and $s \in M$, we write $(s, \mathcal{S})$ for the set $\{(s, s_1, \dots, s_{n-1}) \mid (s_1, \dots, s_{n-1}) \in \mathcal{S}\}$, which consists of chains of length n. Let $f : 2^A \to 2^{2^{M^n}}$ be a mapping, written $f = (C \mapsto \mathfrak{T}_C)$. For all $B \subseteq A$, we define a set $Sat_n[B](f)$ in 2^{M^n}. That is, $B \mapsto Sat_n[B](f)$ is again a mapping from 2^A to $2^{2^{M^n}}$. Observe that when $n = 1$, there is no computation to do since for all B, $\mathfrak{C}_{2,1}[\alpha, B] = \mathfrak{I}_1[B]$ by definition. Therefore, we simply set $Sat_1[B](C \mapsto \mathfrak{T}_C) = \mathfrak{T}_B$. When $n \geqslant 2$, we define $Sat_n[B](C \mapsto \mathfrak{T}_C)$ as the set $\mathfrak{T}_B \cup \mathfrak{M}_B \cup \mathfrak{O}_B$ with

$$\mathfrak{M}_B = \bigcup_{C \cup D = B} (\mathfrak{T}_C \cdot \mathfrak{T}_D) \tag{1}$$

$$\mathfrak{O}_B = \big\{\mathcal{T}^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[\alpha, B]) \cdot \mathcal{T}^\omega \mid \mathcal{T} \in \mathfrak{T}_B\big\} \tag{2}$$

This ends the description of the procedure Sat_n. We now formalize how to iterate it. For any mapping $f : 2^A \to 2^{M^n}$ and any $B \subseteq A$, we set $Sat_n^0[B](f) = f(B)$. For all $j \geqslant 1$, we set $Sat_n^j[B](f) = Sat_n[B](C \mapsto Sat_n^{j-1}[C](f))$. By definition of Sat_n, for all $j \geqslant 0$ and $B \subseteq A$, we have $Sat_n^j(f)[B] \subseteq Sat_n^{j+1}(f)[B] \subseteq 2^{M^n}$. Therefore, there exists j such that $Sat_n^j[B](f) = Sat_n^{j+1}[B](f)$. We denote by $Sat_n^*[B](f)$ this set. This finishes the definition of the algorithm. Its correctness and completeness are stated in the following proposition.

Proposition 5. *Let $n \geqslant 1$, $B \subseteq A$ and $\ell \geqslant 3|M| \cdot 2^{|A|} \cdot n \cdot 2^{2^{2|M|^n}}$. Then*

$$\mathfrak{C}_{2,n}[\alpha, B] = \mathfrak{C}_{2,n}^\ell[\alpha, B] = {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C]).$$

Proposition 5 states correctness of the algorithm (the set ${\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ *only* consists of compatible sets of Σ_2-chains) and completeness (this set contains *all* such sets). It also establishes a bound ℓ. This bound is a byproduct of the proof of the algorithm. It is of particular interest for separation and Theorem 4.

Indeed, one can prove that for any two languages that are Σ_2-separable and recognized by α, the separator can be chosen with quantifier rank ℓ (for $n = 2$).

We will see in Sections 5 and 6 how to use Proposition 5 to get decidable characterizations of Σ_3, Π_3, Δ_3 and $\mathcal{B}\Sigma_2$. We already state the following corollary as a consequence of Theorem 4.

Corollary 6. *Given as input two regular languages L_1, L_2 it is decidable to test whether L_1 can be Σ_2-separated (resp. Π_2-separated) from L_2.*

5 Decidable Characterizations of Σ_3, Π_3, Δ_3

In this section we present our decidable characterizations for Δ_3, Σ_3 and Π_3. We actually give characterizations for all classes Δ_i, Σ_i and Π_i in the quantifier alternation hierarchy. The characterizations are all stated in terms of equations on the syntactic monoid of the language. However, these equations are parametrized by the Σ_{i-1}-chains of length 2. Therefore, getting *decidable* characterizations depends on our ability to compute the set of Σ_{i-1}-chains of length 2, which we are only able to do for $i \leqslant 3$. We begin by stating our characterization for Σ_i, and the characterizations for Π_i and Δ_i will then be simple corollaries.

Theorem 7. *Let L be a regular language and $\alpha : A^* \to M$ be its syntactic morphism. For all $i \geqslant 1$, L is definable in Σ_i iff M satisfies the following property:*

$$s^\omega \leqslant s^\omega t s^\omega \quad \textit{for all } (t,s) \in \mathcal{C}_{i-1}[\alpha]. \tag{3}$$

It follows from Theorem 7 that it suffices to compute the Σ_{i-1}-chains of length 2 in order to decide whether a language is definable in Σ_i. Also observe that when $i = 1$, by definition we have $(t, 1_M) \in \mathcal{C}_0[\alpha]$ for all $t \in M$. Therefore, (3) can be rephrased as $1_M \leqslant t$ for all $t \in M$, which is the already known equation for Σ_1, see [14]. Similarly, when $i = 2$, (3) can be rephrased as $s^\omega \leqslant s^\omega t s^\omega$ whenever t is a 'subword' of s, which is the previously known equation for Σ_2 (see [14,4]).

The proof of Theorem 7 is done using Simon's Factorization Forest Theorem and is actually a generalization of a proof of [4] for the special case of Σ_2. Here, we state characterizations of Π_i and Δ_i as immediate corollaries. Recall that a language is Π_i-definable if its complement is Σ_i-definable, and that it is Δ_i-definable if it is both Σ_i-definable and Π_i-definable.

Corollary 8. *Let L be a regular language and let $\alpha : A^* \to M$ be its syntactic morphism. For all $i \geqslant 1$, the following properties hold:*

- *L is definable in Π_i iff M satisfies $s^\omega \geqslant s^\omega t s^\omega$ for all $(t,s) \in \mathcal{C}_{i-1}[\alpha]$.*
- *L is definable in Δ_i iff M satisfies $s^\omega = s^\omega t s^\omega$ for all $(t,s) \in \mathcal{C}_{i-1}[\alpha]$.*

We finish the section by stating decidability for the case $i = 3$. Indeed by Proposition 5, one can compute the Σ_2-chains of length 2 for any morphism. Therefore, we get the following corollary.

Corollary 9. *Definability of a regular language in Δ_3, Σ_3 or Π_3 is decidable.*

6 Decidable Characterization of $\mathcal{B}\Sigma_2$

In this section we present our decidable characterization for $\mathcal{B}\Sigma_2$. In this case, unlike Theorem 7, the characterization is specific to the case $i = 2$ and does not generalize as a non-effective characterization for all levels. The main reason is that both the intuition and the proof of the characterization rests on a deep analysis of our algorithm for computing Σ_2-chains, which is specific to level $i = 2$. The characterization is stated as two equations that must be satisfied by the syntactic morphism of the language. The first one is parametrized by Σ_2-chains of length 3, and the second one by sets of compatible Σ_2-chains of length 2 through a more involved relation that we define below.

Alternation Schema. Let $\alpha : A^* \to M$ be a monoid morphism and let $B \subseteq A$. A *B-schema* for α is a triple $(s_1, s_2, s_2') \in M^3$ such that there exist $\mathcal{T} \in \mathfrak{C}_2[\alpha, B]$ and $r_1, r_1' \in M$ verifying $s_1 = r_1 r_1'$, $(r_1, s_2) \in \mathcal{C}_2[\alpha, B] \cdot \mathcal{T}^\omega$ and $(r_1', s_2') \in \mathcal{T}^\omega \cdot \mathcal{C}_2[\alpha, B]$. Intuitively, the purpose of B-schemas is to abstract a well-known property of Σ_2 on elements of M: one can prove that if (s_1, s_2, s_2') is a B-schema, then for all $k \in \mathbb{N}$, there exist $w_1, w_2, w_2' \in A^*$, mapped respectively to s_1, s_2, s_2' under α, and such that for all $u \in B^*$, $w_1 \lesssim_2^k w_2 u w_2'$.

Theorem 10. *Let L be a regular language and $\alpha : A^* \to M$ be its syntactic morphism. Then L is definable in $\mathcal{B}\Sigma_2$ iff M satisfies the following properties:*

$$\begin{array}{c} s_1^\omega s_3^\omega = s_1^\omega s_2 s_3^\omega \\ s_3^\omega s_1^\omega = s_3^\omega s_2 s_1^\omega \end{array} \quad \text{for } (s_1, s_2, s_3) \in \mathcal{C}_2[\alpha] \tag{4}$$

$$\begin{array}{c} (s_2 t_2)^\omega s_1 (t_2' s_2')^\omega = (s_2 t_2)^\omega s_2 t_1 s_2' (t_2' s_2')^\omega \\ \text{for } (s_1, s_2, s_2') \text{ and } (t_1, t_2, t_2') \text{ } B\text{-schemas for some } B \subseteq A \end{array} \tag{5}$$

The proof of Theorem 10 is far more involved than that of Theorem 7. However, a simple consequence is decidability of definability in $\mathcal{B}\Sigma_2$. Indeed, it suffices to compute Σ_2-chains of length 3 and the B-schemas for all $B \subseteq A$ to check validity of both equations. Computing this information is possible by Proposition 5, and therefore, we get the following corollary.

Corollary 11. *Definability of a regular language in $\mathcal{B}\Sigma_2$ is decidable.*

7 Conclusion

We solved the separation problem for Σ_2 using the new notion of Σ_2-chains, and we used our solution to prove decidable characterizations for $\mathcal{B}\Sigma_2$, Δ_3, Σ_3 and Π_3. The main open problem in this field remains to lift up these results to higher levels in the hierarchy. In particular, we proved that for any natural i, generalizing our separation solution to Σ_i (*i.e.*, being able to compute the Σ_i-chains of length 2) would yield a decidable characterization for Σ_{i+1}, Π_{i+1} and Δ_{i+1}.

Our algorithm for computing Σ_2-chains cannot be directly generalized for higher levels. An obvious reason for this is the fact that it considers Σ_2-chains parametrized by sub-alphabets. This parameter is designed to take care of the alternation between levels 1 and 2, but is not adequate for higher levels. However, this is unlikely to be the only problem. In particular, we do have an algorithm that avoids using the alphabet, but it remains difficult to generalize. We leave the presentation of this alternate algorithm for further work.

Another open question is to generalize our results to logical formulas that can use a binary predicate $+1$ for the successor relation. In formal languages, this corresponds to the well-known *dot-depth hierarchy* [7]. It was proved in [24] and [15] that decidability of $\mathcal{B}\Sigma_2(<,+1)$ and $\Sigma_3(<,+1)$ is a consequence of our results for $\mathcal{B}\Sigma_2(<)$ and $\Sigma_3(<)$. However, while the reduction itself is simple, its proof rely on deep algebraic arguments. We believe that our techniques can be generalized to obtain direct proofs of the decidability of $\mathcal{B}\Sigma_2(<,+1)$ and $\Sigma_3(<,+1)$.

References

1. J. Almeida. Some algorithmic problems for pseudovarieties. *Publ. Math. Debrecen*, 54:531–552, 1999. Proc. of Automata and Formal Languages, VIII.
2. J. Almeida and O. Klíma. New decidable upper bound of the 2nd level in the Straubing-Thérien concatenation hierarchy of star-free languages. *DMTCS*, 2010.
3. M. Arfi. Polynomial operations on rational languages. In *STACS'87*, 1987.
4. M. Bojanczyk. Factorization forests. In *DLT'09*, pages 1–17, 2009.
5. M. Bojanczyk and T. Place. Regular languages of infinite trees that are boolean combinations of open sets. In *ICALP'12*, pages 104–115, 2012.
6. J. Brzozowski and R. Knast. The dot-depth hierarchy of star-free languages is infinite. *J. Comp. Syst. Sci.*, 16(1):37–55, 1978.
7. R. S. Cohen and J. Brzozowski. Dot-depth of star-free events. *J. Comp. Syst. Sci.*, 5:1–16, 1971.
8. W. Czerwinski, W. Martens, and T. Masopust. Efficient separability of regular languages by subsequences and suffixes. In *ICALP'13*, pages 150–161, 2013.
9. M. Kufleitner. The height of factorization forests. In *MFCS'08*, 2008.
10. R. McNaughton and S. Papert. *Counter-Free Automata*. MIT Press, 1971.
11. J.-E. Pin. Bridges for concatenation hierarchies. In *ICALP'98*, 1998.
12. J.-E. Pin. Theme and variations on the concatenation product. In *4th Int. Conf. on Algebraic Informatics*, pages 44–64. Springer, 2011.
13. J.-E. Pin and H. Straubing. Monoids of upper triangular boolean matrices. In *Semigroups. Structure and Universal Algebraic Problems*, volume 39 of *Colloquia Mathematica Societatis Janos Bolyal*, pages 259–272. North-Holland, 1985.
14. J.-E. Pin and P. Weil. Polynomial closure and unambiguous product. *Theory of Computing Systems*, 30(4):383–422, 1997.
15. J.-E. Pin and P. Weil. The wreath product principle for ordered semigroups. *Communications in Algebra*, 30:5677–5713, 2002.
16. T. Place, L. van Rooijen, and M. Zeitoun. Separating regular languages by piecewise testable and unambiguous languages. In *MFCS'13*, pages 729–740, 2013.
17. T. Place, L. van Rooijen, and M. Zeitoun. Separating regular languages by locally testable and locally threshold testable languages. In *FSTTCS'13*, LIPIcs, 2013.

18. T. Place and M. Zeitoun. Going higher in the first-order quantifier alternation hierarchy on words. *Arxiv*, 2014.
19. T. Place and M. Zeitoun. Separating regular languages with first-order logic. In *CSL-LICS'14*, 2014.
20. M. P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8:190–194, 1965.
21. I. Simon. Piecewise testable events. In *2nd GI Conference on Automata Theory and Formal Languages*, pages 214–222, 1975.
22. I. Simon. Factorization forests of finite height. *TCS*, 72(1):65–94, 1990.
23. H. Straubing. A generalization of the Schützenberger product of finite monoids. *TCS*, 1981.
24. H. Straubing. Finite semigroup varieties of the form $V * D$. *J. Pure App. Algebra*, 36:53–94, 1985.
25. H. Straubing. Semigroups and languages of dot-depth two. *TCS*, 1988.
26. H. Straubing. *Finite Automata, Formal Logic and Circuit Complexity*. 1994.
27. P. Tesson and D. Therien. Diamonds are forever: The variety DA. In *Semigroups, Algorithms, Automata and Languages*, pages 475–500. World Scientific, 2002.
28. D. Thérien. Classification of finite monoids: the language approach. *TCS*, 1981.
29. D. Thérien and T. Wilke. Over words, two variables are as powerful as one quantifier alternation. In *STOC'98*, pages 234–240. ACM, 1998.
30. W. Thomas. Classifying regular events in symbolic logic. *J. Comp. Syst. Sci.*, 1982.
31. W. Thomas. A concatenation game and the dot-depth hierarchy. In *Computation Theory and Logic*, pages 415–426. 1987.

Appendix

We divide this appendix into several sections. In Appendix A, we define the main tools we will use for our proofs: Ehrenfeucht-Fraïssé games and factorization forests. In Appendix B, we complete Section 4 by proving the correctness and completeness of our algorithm for computing Σ_2-chains. In Appendix C, we prove Theorem 7, i.e. our characterization of $\Sigma_i(<)$ (which is decidable for $i \leqslant 3$). The remaining appendices are then devoted to the proof of Theorem 10, i.e. our decidable characterization of $\mathcal{B}\Sigma_2(<)$. In Appendix D we define *Chains Trees* which are our main tool for proving the difficult direction of the characterization. In Appendix E we give an outline of the proof. Finally, Appendix F and Appendix G are devoted to proving the two most difficult propositions in the proof.

A Tools

In this appendix we define Ehrenfeucht-Fraïssé games and factorization forests. Both notions are well-known and we will use them several times in our proofs.

A.1 Ehrenfeucht-Fraïssé Games

It is well known that the expressive power of logics can be expressed in terms of games. These games are called Ehrenfeucht-Fraïssé games. We define here the game tailored to the quantifier alternation hierarchy.

Before we give the definition, a remark is in order. There are actually two ways to define the class of $\Sigma_i(<)$-definable languages. First, one can consider all first-order formulas and say that a formula is $\Sigma_i(<)$ if it has at most i blocks of quantifiers once rewritten in prenex normal form. This is what we do. However, one can also restrict the set of allowed formulas to those that are already in prenex form and have at most i blocks of quantifiers. While this does not change the class of $\Sigma_i(<)$-definable languages as a whole, this changes the set of formulas of quantifier rank k for a fixed k. Therefore, this changes the preorder $\lesssim_i^k$. This means that there is a version of the Ehrenfeucht-Fraïssé game for each definition. In this paper, we use the version that corresponds to the definition given in the main part of the paper (*i.e.*, the one considering all first-order formulas).

Ehrenfeucht-Fraïssé games. Set i a level in the quantifier alternation hierarchy. We define the game for $\Sigma_i(<)$. The board of the game consists of two words $w, w' \in A^*$ and there are two players called *Spoiler and Duplicator*. Moreover, there exists a distinguished word among w, w' that we call the *active word*. The game is set to last a predefined number k of rounds. When the game starts, both players have k pebbles. Moreover, there are two parameters that get updated during the game, the active word and a counter c called the *alternation counter*. Initially, c is set to 0.

At the start of each round j, Spoiler chooses a word, either w or w'. Spoiler can always choose the active word, in which case both c and the active word remain unchanged. However, Spoiler can only choose the word that is not active when $c < i-1$, in which case the active word is switched and c is incremented by 1 (in particular this means that the active word can be switched at most $i-1$ times). If Spoiler chooses w (resp. w'), he puts a pebble on a position x_j in w (resp. x'_j in w').

Duplicator must answer by putting a pebble at a position x'_j in w' (resp. x_j in w). Moreover, Duplicator must ensure that all pebbles that have been placed up to this point verify the following condition: for all $\ell_1, \ell_2 \leqslant j$, the labels at positions x_{ℓ_1}, x'_{ℓ_1} are the same, and $x_{\ell_1} < x_{\ell_2}$ iff $x'_{\ell_1} < x'_{\ell_2}$.

Duplicator wins if she manages to play for all k rounds, and Spoiler wins as soon as Duplicator is unable to play.

Lemma 12 (Folklore). *For all $k, i \in \mathbb{N}$ and $w, w' \in A^*$, $w \lesssim_i^k w'$ iff Duplicator has a winning strategy for playing k rounds in the $\Sigma_i(<)$ game played on w, w' with w as the initial active word.*

Note that we will often use Lemma 12 implicitly and alternate between the original and the game definition of $\lesssim_i^k$. We now give a few classical lemmas on Ehrenfeucht-Fraïssé games that we reuse several times in our proofs. We begin with a lemma stating that $\lesssim_i^k$ is a pre-congruence, *i.e.* that it is compatible with the concatenation product.

Lemma 13. *Let $i \in \mathbb{N}$ and let $w_1, w_2, w'_1, w'_2 \in A^*$ such that $w_1 \lesssim_i^k w_2$ and $w'_1 \lesssim_i^k w'_2$. Then $w_1 w'_1 \lesssim_i^k w_2 w'_2$.*

Proof. By Lemma 12, Duplicator has winning strategies in the level i games between w_1, w_2 and w'_1, w'_2, with w_1, w'_1 as initial active words respectively. These strategies can be easily combined into a strategy for the level i game between $w_1 w'_1$ and $w_2 w'_2$ with $w_1 w'_1$ as intial active word. We conclude that $w_1 w'_1 \lesssim_i^k w_2 w'_2$. ☐

The second property concerns full first-order logic.

Lemma 14. *Let $k, k_1, k_2 \in \mathbb{N}$ be such that $k_1, k_2 \geqslant 2^k - 1$. Let $v \in A^*$. Then*

$$\forall i \in \mathbb{N}, \quad v^{k_1} \lesssim_i^k v^{k_2}.$$

Proof. This is well known for full first-order logic (see [26] for details). ☐

We finish with another classical property, which is this time specific to $\Sigma_i(<)$.

Lemma 15. *Let $i \in \mathbb{N}$, let $k, \ell, r, \ell', r' \in \mathbb{N}$ be such that $\ell, r, \ell', r' \geqslant 2^k$ and let $u, v \in A^*$ such that $u \lesssim_i^k v$. Then we have:*

$$v^\ell v^r \lesssim_{i+1}^k v^{\ell'} u v^{r'}.$$

Proof. Set $w = v^\ell v^r$ and $w' = v^{\ell'} u v^{r'}$. We prove that $w \lesssim_{i+1}^k w'$ using an Ehrenfeucht-Fraïssé argument: we prove that Duplicator has a winning strategy for the game in k rounds for $\Sigma_{i+1}(<)$ played on w, w' with w as initial active word.

The proof goes by induction on k. We distinguish two cases depending on the value, 0 or 1, of the alternation counter c after Spoiler has played the first round.

Case 1: $c = 1$. In this case, by definition of the game, it suffices to prove that $w' \lesssim_i^k w$. From our hypothesis we already know that $u \lesssim_i^k v$. Moreover, it follows from Lemma 14 that $v^{\ell'} \lesssim_i^k v^{\ell}$ and $v^{r'} \lesssim_i^k v^{r-1}$. It then follows from Lemma 13 that $w' \lesssim_i^k w$.

Case 2: $c = 0$. By definition, this means that Spoiler played on some position x in w. Therefore x is inside a copy of the word v. Since w contains more than 2^{k+1} copies of v, by symmetry we can assume that there are at least 2^k copies of v to the right of x. We now define a position x' inside w' that will serve as Duplicator's answer. We choose x' so that it belongs to a copy of v inside w' and is at the same relative position inside this copy as x is in its own copy of v. Therefore, to fully define x', it only remains to define the copy of v in which we choose x'. Let n be the number of copies of v to the *left* of x in w, that is, x belongs to the $(n+1)$-th copy of v starting from the left of w. If $n < 2^{k-1} - 1$, then x' is chosen inside the $(n+1)$-th copy of v starting from the left of w'. Otherwise, x' is chosen inside the 2^{k-1}-th copy of v starting from the left of w'. Observe that these copies always exist, since $\ell' \geqslant 2^k$.

Set $w = w_p v w_q$ and $w' = w'_p v w'_q$, with the two distinguished v factors being the copies containing the positions x, x'. By definition of the game, it suffices to prove that $w_p \lesssim_{i+1}^{k-1} w'_p$ and $w_q \lesssim_{i+1}^{k-1} w'_q$ to conclude that Duplicator can play for the remaining $k-1$ rounds. If $n < 2^{k-1} - 1$, then by definition, $w_p = w'_p$, therefore it is immediate that $w_p \lesssim_{i+1}^{k-1} w'_p$. Otherwise, both w_p and w'_p are concatenations of at least $2^{k-1} - 1$ copies of v. Therefore $w_p \lesssim_{i+1}^{k-1} w'_p$ follows Lemma 14. Finally observe that by definition $w_q = v^{\ell_1} v^r$ and $w'_q = v^{\ell'_1} u v^{r'}$ with $\ell_1 + r \geqslant 2^k$ and $\ell'_1, r' \geqslant 2^{k-1}$. Therefore, it is immediate by induction on k that $w_q \lesssim_{i+1}^{k-1} w'_q$. □

A.2 Simon's Facorization Forests Theorem

In this appendix, we briefly recall the definition of factorization forests and state the associated theorem. Proofs and more detailed presentations can be found in [9,4]

Let M be a finite monoid and $\alpha : A^* \to M$ a morphism. An *α-factorization forest* is an ordered unranked tree with nodes labeled by words in A^* and such that for any inner node x with label w, if $x_1, \dots, x_n$ are its children listed from left to right with labels $w_1, \dots, w_n$, then $w = w_1 \cdots w_n$. Moreover, all nodes x in the forest must be of the three following kinds:

- *leaf nodes* which are labeled by either a single letter or the empty word.
- *binary nodes* which have exactly two children.
- *idempotent nodes* which have an arbitrary number of children whose labels $w_1, \dots, w_n$ verify $\alpha(w_1) = \cdots = \alpha(w_n) = e$ for some idempotent $e \in M$.

If $w \in A^*$, an *α-factorization forest for w* is an α-factorization forest whose root is labeled by w.

Theorem 16 (Factorization Forest Theorem of Simon [22,9]). *For all $w \in A^*$, there exists an α-factorization forest for w of height smaller than $3|M|-1$.*

B Appendix to Section 4: Proving the Algorithm

In this appendix, we prove Proposition 5, that it is the correctness and completeness of our algorithm which computes sets of compatible Σ_2-chains. Recall that our algorithm works by fixpoint. Given as input a morphism $\alpha : A^* \to M$ into a finite monoid M and a natural $n \in \mathbb{N}$, it applies iteratively the procedure Sat_n, starting from the application $C \mapsto \mathfrak{I}_n[C]$, where $\mathfrak{I}_n[C]$ is the set of trivial sets of compatible Σ_2-chains of length n for α, C. The fixpoint is a collection of sets indexed by subalphabets B, denoted by $Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$.

We have to show that when the algorithm reaches its fixpoint, the computed set $\downarrow Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$ consists exactly of all compatible sets of Σ_2-chains of length n. This is formulated in Proposition 5, which we restate. In addition, it states that for every length n, one can compute a rank $\ell(n)$ that suffices to capture all sets of compatible sets of Σ_2-chains of length n. In the following, we let

$$\ell(n) = 3|M| \cdot 2^{|A|} \cdot n \cdot 2^{2^{2|M|^n}}.$$

Proposition 5. *Let $n \geqslant 1$, $B \subseteq A$ and $\ell \geqslant \ell(n)$. Then*

$$\mathfrak{C}_{2,n}[\alpha, B] = \mathfrak{C}^{\ell}_{2,n}[\alpha, B] = \downarrow Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]).$$

We proceed by induction on n. Observe that when $n = 1$ all three sets are by definition equal to $\mathfrak{I}_n[B]$, therefore the result is immediate. Assume now that $n \geqslant 2$. Using our induction hypothesis we have the following fact.

Fact 17 *Let $B \subseteq A$, then $\mathfrak{C}_{2,n-1}[\alpha, B] = \mathfrak{C}^{\ell(n-1)}_{2,n-1}[\alpha, B]$. Moreover, it follows that $\mathcal{C}_{2,n-1}[\alpha, B] = \mathcal{C}^{\ell(n-1)}_{2,n-1}[\alpha, B]$.*

For all $B \subseteq A$, we prove the following inclusions: $\mathfrak{C}_{2,n}[\alpha, B] \subseteq \mathfrak{C}^{\ell}_{2,n}[\alpha, B] \subseteq \downarrow Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]) \subseteq \mathfrak{C}_{2,n}[\alpha, B]$. Observe that $\mathfrak{C}_{2,n}[\alpha, B] \subseteq \mathfrak{C}^{\ell}_{2,n}[\alpha, B]$ is immediate by definition. Therefore, we have two inclusions to prove:

- $\downarrow Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]) \subseteq \mathfrak{C}_{2,n}[\alpha, B]$, this corresponds to correctness of the algorithm: all computed sets are indeed sets of compatible Σ_2-chains.
- $\mathfrak{C}^{\ell}_{2,n}[\alpha, B] \subseteq \downarrow Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$, this corresponds to completeness of the algorithm: all sets of compatible Σ_2-chains are computed.

We give each proof its own subsection. Note that Fact 17 (i.e., induction on n) is only used in the completeness proof.

B.1 Correctness of the Algorithm

In this subsection, we prove that for all $B \subseteq A$, $\downarrow Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]) \subseteq \mathfrak{C}_{2,n}[\alpha, B]$. This is a consequence of the following proposition.

Proposition 18. *Set $B \subseteq A$, for all $k \in \mathbb{N}$, $Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]) \subseteq \mathfrak{C}^k_{2,n}[\alpha, B]$.*

Before proving Proposition 18, we explain how it is used to prove correctness. By definition, for all B, $\mathfrak{C}_{2,n}[\alpha, B] = \bigcap_{k\in\mathbb{N}} \mathfrak{C}^k_{2,n}[\alpha, B]$. Therefore, it is immediate from the proposition that $Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]) \subseteq \mathfrak{C}_{2,n}[\alpha, B]$. Moreover, by definition, $\downarrow \mathfrak{C}_{2,n}[\alpha, B] = \mathfrak{C}_{2,n}[\alpha, B]$. We conclude that $\downarrow Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]) \subseteq \mathfrak{C}_{2,n}[\alpha, B]$ which terminates the correctness proof. It now remains to prove Proposition 18.

Let $k \in \mathbb{N}$, $B \subseteq A$ and $\mathcal{R} \in Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$. We need to prove that $\mathcal{R} \in \mathfrak{C}^k_{2,n}[\alpha, B]$. By definition, $\mathcal{R} \in Sat^j_n[B](C \mapsto \mathfrak{I}_n[C])$ for some j. We proceed by induction on j. If $j = 0$, this is immediate since $\mathcal{R} \in \mathfrak{I}_n[B] \subseteq \mathfrak{C}^k_{2,n}[\alpha, B]$.

Assume now that $j > 0$. For all $D \subseteq A$, we set $\mathfrak{T}_D = Sat^{j-1}_n[D](C \mapsto \mathfrak{I}_n[C])$. By induction hypothesis, for every $D \subseteq A$, every element of $\mathfrak{T}_D$ belongs to $\mathfrak{C}^k_{2,n}[\alpha, D]$. Since $\mathcal{R} \in Sat^j_n[B](C \mapsto \mathfrak{I}_n[C])$, by definition we have $\mathcal{R} \in \mathfrak{T}_B \cup \mathfrak{M}_B \cup \mathfrak{O}_B$ with

$$\mathfrak{M}_B = \bigcup_{C\cup D=B} (\mathfrak{T}_C \cdot \mathfrak{T}_D)$$
$$\mathfrak{O}_B = \{\mathcal{T}^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[\alpha, B]) \cdot \mathcal{T}^\omega \mid \mathcal{T} \in \mathfrak{T}_B\}$$

If $\mathcal{R} \in \mathfrak{T}_B$, it is immediate by induction that $\mathcal{R} \in \mathfrak{C}^k_{2,n}[\alpha, B]$ and we are finished. Assume now that $\mathcal{R} \in \mathfrak{M}_B$. This means that there exist C, D such that $C \cup D = B$, $\mathcal{T}_C \in \mathfrak{T}_C$ and $\mathcal{T}_D \in \mathfrak{T}_D$ such that $\mathcal{R} = \mathcal{T}_C \cdot \mathcal{T}_D$. By induction hypothesis, we have $\mathcal{T}_C \in \mathfrak{C}^k_{2,n}[\alpha, C]$ and $\mathcal{T}_D \in \mathfrak{C}^k_{2,n}[\alpha, D]$. It is then immediate by Fact 3 that $\mathcal{R} = \mathcal{T}_C \cdot \mathcal{T}_D \in \mathfrak{C}^k_{2,n}[\alpha, B]$.

It remains to treat the case when $\mathcal{R} \in \mathfrak{O}_B$. In that case, we get $\mathcal{T} \in \mathfrak{T}_B$ such that $\mathcal{R} = \mathcal{T}^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[\alpha, B]) \cdot \mathcal{T}^\omega$. In the following, we write $h = \omega \times 2^{2k}$ (with ω as $\omega(2^{M^n})$). Note that by definition of the number ω, we have $\mathcal{T}^\omega = \mathcal{T}^h$, and in particular, $\mathcal{R} = \mathcal{T}^h \cdot (1_M, \mathcal{C}_{2,n-1}[\alpha, B]) \cdot \mathcal{T}^h$. Observe first that by induction hypothesis, we know that $\mathcal{T} \in \mathfrak{C}^k_{2,n}[\alpha, B]$. In particular, this means that all chains in $\mathcal{T}$ have the same first element. We denote by t_1 this element. By definition of $\mathfrak{C}^k_{2,n}[\alpha, B]$, we get $u \in A^*$ such that $\mathsf{alph}(u) = B$, $\alpha(u) = t_1$ and for all chains $(t_1, \dots, t_n) \in \mathcal{T}$ there exist $u_2, \dots, u_n \in A^*$ satisfying $u \lesssim^k_2 u_2 \lesssim^k_2 \cdots \lesssim^k_2 u_n$ and for all j, $t_j = \alpha(u_j)$ and $\mathsf{alph}(u_j) = B$.

We now prove that $\mathcal{R} \in \mathfrak{C}^k_{2,n}[\alpha, B]$. Set $w = u^{2h}$ and $r_1 = \alpha(w) = t_1^\omega$, by definition $\mathsf{alph}(w) = B$. Observe that since $\mathcal{R} = \mathcal{T}^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[\alpha, B]) \cdot \mathcal{T}^\omega$, every chain in $\mathcal{R}$ has r_1 as first element. We now prove that for any chain $(r_1, \dots, r_n) \in \mathcal{R}$, there exist $w_2, \dots, w_n \in A^*$ satisfying $w \lesssim^k_2 w_2 \lesssim^k_2 \cdots \lesssim^k_2 w_n$ and for all j, $r_j = \alpha(w_j)$ and $\mathsf{alph}(w_j) = B$. By definition, this will mean that $\mathcal{R} \in \mathfrak{C}^k_{2,n}[\alpha, B]$. Set $(r_1, \dots, r_n) \in \mathcal{R}$. By hypothesis, $(r_1, \dots, r_n) = (t'_1 t''_1, t'_2 s_2 t''_2, \dots, t'_n s_n t''_n)$ with $(t'_1, \dots, t'_n)$, $(t''_1, \dots, t''_n) \in \mathcal{T}^h$ and $(s_2, \dots, s_n) \in \mathcal{C}_{2,n-1}[\alpha, B]$. In particular, $t'_1 = t''_1 = t_1^h = t_1^\omega$. Since $\mathcal{T} \in \mathfrak{C}_{2,n}[\alpha, B]$, we have $\mathcal{T}^h \in \mathfrak{C}_{2,n}[\alpha, B]$, so we get $w'_2, \dots, w'_n, w''_2, \dots, w''_n \in A^*$ such that for all j, $\mathsf{alph}(w'_j) = \mathsf{alph}(w''_j) = B$, $\alpha(w'_j) = t'_j$ and $\alpha(w''_j) = t''_j$ and we have:

$$\begin{array}{c} u^h \lesssim_2^k w'_2 \lesssim_2^k \cdots \lesssim_2^k w'_n \\ u^h \lesssim_2^k w''_2 \lesssim_2^k \cdots \lesssim_2^k w''_n \end{array}$$

On the other hand, using the fact that $(s_2, \dots, s_n) \in \mathcal{C}_{2,n-1}[\alpha, B]$, we get words $v_2, \dots, v_n \in A^*$, mapped to $s_2, \dots, s_n$ by α and all having alphabet B, such that $v_2 \lesssim_2^k \cdots \lesssim_2^k v_n$. For all $j \geqslant 2$, set $w_j = w'_j v_j w''_j$. Observe that for any $j \geqslant 2$, $\mathsf{alph}(w_j) = B$ and $\alpha(w_j) = s_j$. Therefore it remains to prove that $w \lesssim_2^k w_2 \lesssim_2^k \cdots \lesssim_2^k w_n$ to terminate the proof. That $w_2 \lesssim_2^k \cdots \lesssim_2^k w_n$ is immediate by Lemma 13. Recall that $w = u^{2h}$, therefore the last inequality is a consequence of the following lemma.

Lemma 19. $u^h u^h \lesssim_2^k w'_2 v_2 w''_2$

Proof. By Lemma 13, we have $u^h v_2 u^h \lesssim_2^k w'_2 v_2 w''_2$. Therefore, it suffices to prove that $u^h u^h \lesssim_2^k u^h v_2 u^h$ to conclude. Recall that by definition $\mathsf{alph}(v_2) = \mathsf{alph}(u) = B$, therefore, it is straightforward to see that

$$v_2 \lesssim_1^k u^{2^k} \tag{6}$$

Moreover, we chose $h = \omega \times 2^{2k}$. Therefore, it is immediate from Lemma 15 and (6) that $u^h u^h \lesssim_2^k u^h v_2 u^h$. ◻

B.2 Completeness of the Algorithm

We need to prove that for all $B \subseteq A$, we have $\mathfrak{C}^\ell_{2,n}[\alpha, B] \subseteq {\downarrow} Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$ for $\ell \geqslant \ell(n)$, where $\ell(n) = 3|M| \cdot 2^{|A|} \cdot n \cdot 2^{2^{2|M|^n}}$. We do this by proving a slightly more general proposition by induction. To state this proposition, we need more terminology.

Generated Compatible Sets. Set $k \in \mathbb{N}$, $w \in A^*$ and $B = \mathsf{alph}(w)$. We set $\mathcal{G}^k_n(w) \in 2^{M^n}$ as the following set of chains of length n: $(t_1, \dots, t_n) \in \mathcal{G}^k_n(w)$ iff $t_1 = \alpha(w)$ and there exists $w_2, \dots, w_n \in A^*$ satisfying

- for all j, $\alpha(w_j) = t_j$.
- $w \lesssim_2^k w_2 \lesssim_2^k \cdots \lesssim_2^k w_n$.

Observe that the last item implies that all w_j have the same alphabet $\mathsf{alph}(w)$. Therefore, by definition, any $\mathcal{G}^k_n(w)$ is a compatible set of Σ_2-chains of length n: $\mathcal{G}^k_n(w) \in \mathfrak{C}^k_{2,n}[\alpha, \mathsf{alph}(w)]$. Moreover, any compatible set of Σ_2-chains of length n, $\mathcal{T} \in \mathfrak{C}^k_{2,n}[\alpha, B]$ is a subset of $\mathcal{G}^k_n(w)$ for some w of alphabet B. We finish the definition with a decomposition lemma that will be useful in the proof.

Lemma 20. *Let $w_1, \dots, w_{m+1} \in A^*$ and $k \in \mathbb{N}$ with $k > m$, then:*

$$\mathcal{G}^k_n(w_1 \cdots w_{m+1}) \subseteq \mathcal{G}^{k-m}_n(w_1) \cdots \mathcal{G}^{k-m}_n(w_{m+1})$$

Proof. Let $(s_1,\dots,s_n) \in \mathcal{G}_n^k(w_1\cdots w_{m+1})$. By definition, there exists $u_1,\dots,u_n$ such that $u_1 = w_1\cdots w_{m+1}$, for all i, $\alpha(u_i) = s_i$ and $u_1 \lesssim_2^k \cdots \lesssim_2^k u_n$. Using a simple Ehrenfeucht-Fraïssé argument, we obtain that all words u_i can be decomposed as $u_i = u_{i,1}\cdots u_{i,m+1}$ with $u_{1,1} = w_1,\dots,u_{1,m+1} = w_{m+1}$ and for all j: $u_{1,j} \lesssim_2^{k-m} \cdots \lesssim_2^{k-m} u_{n,j}$. For all i,j, set $s_{i,j} = \alpha(u_{i,j})$. By definition, for all j, $(s_{1,j},\dots,s_{n,j}) \in \mathcal{G}_n^{k-m}(w_j)$. Moreover, we have

$$(s_1,\dots,s_n) = (s_{1,1},\dots,s_{n,1})\cdots(s_{1,m+1},\dots,s_{n,m+1}).$$

Therefore, we have $(s_1,\dots,s_n) \in \mathcal{G}_n^{k-m}(w_1)\cdots\mathcal{G}_n^{k-m}(w_{m+1})$ which terminates the proof. ◻

We can now state our inductive proposition and prove that $\mathfrak{C}_{2,n}^{\ell}[\alpha,B] \subseteq {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. Set $\beta : A^* \to M \times 2^A$ defined as $\beta(w) = (\alpha(w),\mathsf{alph}(w))$.

Proposition 21. *Let $B \subseteq A$, $j \in \mathbb{N}$ and $w \in A^*$ that admits a β-factorization forest of height h and such that $\mathsf{alph}(w) = B$. Set $k \geqslant h \cdot 2^{2^{2|M|^n}} + \ell(n-1)$, then $\mathcal{G}_n^k(w) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$.*

Before proving Proposition 21, we explain how to use it to terminate our completeness proof. Set $\mathcal{T} \in \mathfrak{C}_{2,n}^{\ell}[\alpha,B]$, by definition, this means that there exists $w \in A^*$ such that $\mathsf{alph}(w) = B$ and $\mathcal{T} \subseteq \mathcal{G}_n^{\ell}(w)$. By Theorem 16, we know that w admits a β-factorization forest of height at most $3|M|2^{|A|}$. Therefore, by choice of ℓ, we can apply Proposition 21 and we obtain $\mathcal{G}_n^k(w) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. By definition of $\downarrow$ it is then immediate that $\mathcal{T} \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ which terminates the proof.

It remains to prove Proposition 21. Note that this is where we use Fact 17 (i.e. induction on n). Set $w \in A^*$ that admits a β-factorization forest of height h, $B = \mathsf{alph}(w)$ and $k \geqslant h \times 2^{2^{3|M|^n}} + \ell(n-1)$. We need to prove that $\mathcal{G}_n^k(w) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$, i.e., to construct $\mathcal{T} \in Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ such that $\mathcal{G}_n^k(w) \subseteq \mathcal{T}$. The proof is by induction on the height h of the factorization forest of w. It works by applying the proposition inductively to the factors given by this factorization forest. In particular, we will use Lemma 20 to decompose $\mathcal{G}_n^k(w)$ according to this factorization forest. Then, once the factors have been treated by induction, we will use the definition of the procedure Sat_n (i.e. Operations (1) and (2)) to conclude. In particular, we will use the following fact several times.

Fact 22 ${\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ *is subsemigroup of* 2^{M^n}*.*

Proof. We prove that $Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ is subsemigroup of 2^{M^n}, the result is then immediate by definition of $\downarrow$. Set $\mathcal{S}_1,\mathcal{S}_2 \in Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. By definition of Sat_n (see Operation (1)), we have $\mathcal{S}_1 \cdot \mathcal{S}_2 \in Sat_n[B](B \mapsto Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])) = Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. ◻

We now start the induction. We distinguish three cases depending on the nature of the topmost node in the β-factorization forest of w.

Case 1: the topmost node is a leaf. In that case, $h = 1$ and w is a single letter word $a \in A$. In particular $B = \mathsf{alph}(w) = \{a\}$. Observe that $k \geqslant 2$, therefore, one can verify that $\mathcal{G}_n^k(a) = \{(\alpha(a), \dots, \alpha(a))\}$. It follows that $\mathcal{G}_n^k(a) \in \mathfrak{I}_n[B]$ which terminates the proof for this case.

Case 2: the topmost node is a binary node. We use induction on h and Operation (1) in the definition of Sat_n. By hypothesis $w = w_1 \cdot w_2$ with w_1, w_2 words admitting β-factorization forests of heights $h_1, h_2 \leqslant h - 1$. Set $B_1 = \mathsf{alph}(w_1)$ and $B_2 = \mathsf{alph}(w_2)$, by definition, we have $B = B_1 \cup B_2$. Moreover, observe that

$$k - 1 \geqslant (h-1) \cdot 2^{2^{2|M|^n}} + \ell(n-1).$$

Therefore, we can apply our induction hypothesis to w_1, w_2 and we obtain $\mathcal{T}_1 \in Sat_n^*[B_1](C \mapsto \mathfrak{I}_n[C])$ and $\mathcal{T}_2 \in Sat_n^*[B_2](C \mapsto \mathfrak{I}_n[C])$ such that $\mathcal{G}_n^{k-1}(w_1) \subseteq \mathcal{T}_1$ and $\mathcal{G}_n^{k-1}(w_2) \subseteq \mathcal{T}_2$. By Operation (1) in the definition of Sat, it is immediate that $\mathcal{T}_1 \cdot \mathcal{T}_2 \in Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. Moreover, by Lemma 20, $\mathcal{G}_n^k(w) \subseteq \mathcal{G}_n^{k-1}(w_1) \cdot \mathcal{G}_n^{k-1}(w_2) \subseteq \mathcal{T}_1 \cdot \mathcal{T}_2$. It follows that $\mathcal{G}_n^k(w) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ which terminates this case.

Case 3: the topmost node is an idempotent node. This is the most difficult case. We use induction on h, Operation (2) in the definition of Sat_n and Fact 22. Note that this is also where Fact 17 (i.e. induction on n in the general proof of Proposition 5) is used. We begin by summarizing our hypothesis: w admits what we call an (e, B)-decomposition.

(e, B)**-Decompositions.** Set $\widetilde{k} = (h-1) \cdot 2^{2^{2|M|^n}} + \ell(n-1)$, $e \in M$ an idempotent and $u \in A^*$. We say that u admits an (e, B)-*decomposition* $u_1, \dots, u_m$ if

a) $u = u_1 \cdots u_m$,
b) for all j, $\mathsf{alph}(u_j) = B$ and $\alpha(u_j) = e$ and
c) for all j, $\mathcal{G}_n^{\widetilde{k}}(w_j) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$.

Note that *b)* means that $\beta(u_j)$ is a constant idempotent, where we recall that $\beta : A^* \to M \times 2^A$ is the morphism defined by $\beta(w) = (\alpha(w), \mathsf{alph}(w))$.

Fact 23 *w admits an (e, B)-decomposition for some idempotent $e \in M$.*

Proof. By hypothesis of Case 3, there exists a decomposition $w_1, \dots, w_m$ of w that satisfies points *a)* and *b)*. Moreover, for all j, w_j admits a β-factorization forest of height $h_j \leqslant h-1$. Therefore point *c)* is obtained by induction hypothesis on the height h. ◻

For the remainder of this case, we assume that the idempotent $e \in M$ and the (e, B)-decomposition $w_1, \dots, w_m$ of w are fixed. We finish the definition, with the following useful fact, which follows from Fact 17.

Fact 24 *Assume that u admits an (e, B)*-decomposition *$u_1, \dots, u_m$ and let $i \leqslant j \leqslant m$. Then, $\mathcal{G}_n^{\ell(n-1)}(u_i \cdots u_j) \subseteq (e, \mathcal{C}_{2,n-1}[B])$.*

Proof. Let $(s_1,\dots,s_n) \in \mathcal{G}_n^{\ell(n-1)}(u_i \cdots u_j)$. Since $\alpha(u_i \cdots u_j) = e$, we have $s_1 = e$. Moreover, it is immediate from Fact 17 that $(s_2,\dots,s_n) \in \mathcal{C}_{n-1}^2[\alpha, B]$. We conclude that $(s_1,\dots,s_n) \in (e, \mathcal{C}_{2,n-1}[B])$. ◻

Recall that we want to prove that $\mathcal{G}_n^k(w) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. In general, the number of factors m in the (e,B)-decomposition of w can be arbitrarily large. In particular, it is possible that $k-(m-1) < \widetilde{k}$. This means that we cannot simply use Lemma 20 as we did in the previous case to conclude that $\mathcal{G}_n^k(w) \subseteq \mathcal{G}_n^{\widetilde{k}}(w_1) \cdots \mathcal{G}_n^{\widetilde{k}}(w_m)$. However, we will partition $w_1,\dots,w_m$ as a bounded number of subdecompositions that we can treat using Operation (2) in the definition of Sat_n. The partition is given by induction on a parameter of the (e,B)-decomposition $w_1,\dots,w_m$ that we define now.

Index of an (e,B)-decomposition. Set $k_n = 2^{|M|^n}$ (the size of the monoid 2^{M^n}). Let $u \in A^*$ that admits an (e,B)-decomposition $u_1,\dots,u_m$ and let $j \in \mathbb{N}$ such that $1 \leqslant j \leqslant m-k_n$ (i.e. j is the index of one of the first $m-k_n$ factors in the decomposition). The *k_n-sequence* occurring at j is the sequence $\mathcal{G}_n^{\tilde{k}}(w_j),\dots,\mathcal{G}_n^{\tilde{k}}(w_{j+k_n}) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. The *index* of $u_1,\dots,u_m$ is the number of k_n-sequences that occur in $u_1,\dots,u_m$. Observe that by definition, there are at most $(k_n)^{k_n+1}$ k_n-sequences. Therefore the index of the decomposition is bounded by $(k_n)^{k_n+1}$. We proceed by induction on the index of the decomposition and state this induction in the following lemma.

Lemma 25. *Let $u \in A^*$ admitting an (e,B)-decomposition $u_1,\dots,u_m$ of index g and set $\widehat{k} \geqslant 2g+2(k_n+1)+\widetilde{k}+\ell(n-1)$. Then $\mathcal{G}_n^{\widehat{k}}(u) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$.*

Before proving this lemma, we use it to conclude Case 3. We know that our (e,B)-decomposition $w_1,\dots,w_m$ has an index $g \leqslant (k_n)^{k_n+1}$. Therefore, it suffices to prove that $k \geqslant 2(k_n)^{k_n+1} + 2(k_n+1) + \widetilde{k} + \ell(n-1)$ to conclude that $\mathcal{G}_n^k(w) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ using Lemma 25. One can verify that $2^{2^{2|M|^n}} \geqslant 2(k_n)^{k_n+1} + 2(k_n+1)$ as soon as $k_n > 2$. It is then immediate that

$$k = h \cdot 2^{2^{2|M|^n}} + \ell(n-1) \geqslant 2^{2^{2|M|^n}} + (h-1) \cdot 2^{2^{2|M|^n}} + \ell(n-1) = 2^{2^{2|M|^n}} + \widetilde{k} + \ell(n-1)$$

Proof (of Lemma 25). The proof goes by induction on the index g. We distinguish two cases depending on whether there exists a k_n-sequence that occurs at two different positions in the (e,B)-decomposition.

Assume first that this is not the case, i.e., all k_n-sequences occurring at positions $1 \leqslant j \leqslant m-k_n$ are different. Since there are exactly g k_n-sequences occurring in the decomposition, a simple pigeon-hole principle argument yields that $m \leqslant g+k_n$. We use our choice of $\widehat{k}$ to conclude with a similar argument to the one we used in Case 2. By Lemma 20, we have:

$$\mathcal{G}_n^{\widehat{k}}(u) \subseteq \mathcal{G}_n^{\widehat{k}-(m-1))}(u_1) \cdots \mathcal{G}_n^{\widehat{k}-(m-1)}(u_m)$$

Observe that by hypothesis of this case, $\widehat{k}-(m-1) \geqslant \widetilde{k}$. Therefore, by definition of (e,B)-decompositions, for all j, $\mathcal{G}_n^{\widehat{k}-(m-1))}(u_j) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. It is then immediate from Fact 22 that $\mathcal{G}_n^{\widehat{k}-(m-1))}(u_1)\cdots\mathcal{G}_n^{\widehat{k}-(m-1)}(u_m) \in {\downarrow}Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. We conclude that $\mathcal{G}_n^{\widehat{k}}(u) \in {\downarrow}Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ which terminates this case.

Assume now that there exist $j, j' \in \mathbb{N}$ such that $1 \leqslant j < j' \leqslant m-(k_n-1)$, and the k_n-sequences occurring at j and j' are the same. For the remainder of the proof, we set $\mathcal{R}_1,\dots,\mathcal{R}_{k_n+1}$ as this common k_n-sequence. Moreover, we assume that j and j' are chosen minimal and maximal respectively, i.e. there exists no $j'' < j$ or $j'' > j'$ such that $\mathcal{R}_1,\dots,\mathcal{R}_{k_n+1}$ occur at j''. By definition of a k_n-sequence, recall that we have $\mathcal{R}_1,\dots,\mathcal{R}_{k_n+1} \in {\downarrow}Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. Set

$$\begin{aligned} v_1 &= u_1 \cdots u_{j-1}, \\ v_2 &= u_j \cdots u_{j'+k_n} \\ v_3 &= u_{j'+k_n+1} \cdots u_m. \end{aligned}$$

By Lemma 20, we know that

$$\mathcal{G}_n^{\widehat{k}}(u) \subseteq \mathcal{G}_n^{\widehat{k}-2}(v_1) \cdot \mathcal{G}_n^{\widehat{k}-2}(v_2) \cdot \mathcal{G}_n^{\widehat{k}-2}(v_3)$$

We prove that for $i = 1,2,3$, $\mathcal{G}_n^{\widehat{k}-2}(v_i) \in {\downarrow}Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. By Fact 22, it will then be immediate that $\mathcal{G}_n^{\widehat{k}}(u) \in {\downarrow}Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ which terminates the proof. Observe that by choice of j, j', $u_1,\dots,u_{j-1}$ and $u_{j'+k_n+1},\dots,u_m$ are (e,B)-decompositions of index smaller than g (the k_n-sequence $\mathcal{R}_1,\dots,\mathcal{R}_{k_n+1}$ does not occur in these decompositions). Therefore, it is immediate by induction hypothesis on g that $\mathcal{G}_n^{\widehat{k}-2}(v_1), \mathcal{G}_n^{\widehat{k}-2}(v_3) \in {\downarrow}Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$.

It remains to prove that $\mathcal{G}_n^{\widehat{k}-2}(v_2) \in {\downarrow}Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. If $j' \leqslant j + k_n$, then v_2 admits an (e,B)-decomposition of length smaller than $2(k_n+1)$ and we can conclude using Lemma 20 as in the previous case. Therefore, assume that $j' > j + k_n$ and set $v = u_{j+k_n+1} \cdots u_{j'-1}$ and observe that by definition $v_2 = u_j \cdots u_{j+k_n} \cdot v \cdot u_{j'} \cdots u_{j'+k_n}$. Moreover, $\widehat{k} - 2 - 2(k_n+1) \geqslant \widetilde{k}$, using Lemma 20 we get that

$$\mathcal{G}_n^{\widehat{k}-2}(v_2) \subseteq \mathcal{G}_n^{\widetilde{k}}(u_j) \cdots \mathcal{G}_n^{\widetilde{k}}(u_{j+k_n}) \cdot \mathcal{G}_n^{\widetilde{k}}(v) \cdot \mathcal{G}_n^{\widetilde{k}}(u_{j'}) \cdots \mathcal{G}_n^{\widetilde{k}}(u_{j'+k_n})$$

By definition $\mathcal{R}_1,\dots,\mathcal{R}_{k_n+1}$ is the k_n-sequence occurring at both j and j'. Therefore, it follows that

$$\mathcal{G}_n^{\widehat{k}-2}(v_2) \subseteq \mathcal{R}_1 \cdots \mathcal{R}_{k_n+1} \cdot \mathcal{G}_n^{\widetilde{k}}(v) \cdot \mathcal{R}_1 \cdots \mathcal{R}_{k_n+1} \tag{7}$$

Intuitively, we want to find an idempotent in the sequence $\mathcal{R}_1 \cdots \mathcal{R}_{k_n+1}$ in order to apply Operation (2). Observe that since the $\mathcal{R}_j$ are elements of the monoid 2^{M^n} and $k_n = 2^{|M|^n}$, the sequence $\mathcal{R}_1 \cdots \mathcal{R}_{k_n+1}$ must contain a "loop." By this we mean that there exists $j_1 < j_2$ such that $\mathcal{R}_1 \cdots \mathcal{R}_{j_1} = \mathcal{R}_1 \cdots \mathcal{R}_{j_2}$. Set $\mathcal{S}_1 = \mathcal{R}_1 \cdots \mathcal{R}_{j_1}$, $\mathcal{S}_2 = \mathcal{R}_{j_1+1} \cdots \mathcal{R}_{j_2}$ and $\mathcal{S}_3 = \mathcal{R}_{j_2+1} \cdots \mathcal{R}_{k_n+1}$. By definition of

$\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$, we have $\mathcal{R}_1 \cdots \mathcal{R}_{k_n+1} = \mathcal{S}_1 \cdot (\mathcal{S}_2)^\omega \cdot \mathcal{S}_3$. Note that by Fact 22, we have $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3 \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. By replacing this in (7), we get

$$\mathcal{G}_n^{\widehat{k-2}}(u_2) \subseteq \mathcal{S}_1 \cdot (\mathcal{S}_2)^\omega \cdot \mathcal{S}_3 \cdot \mathcal{G}_n^{\tilde{k}}(v) \cdot \mathcal{S}_1 \cdot (\mathcal{S}_2)^\omega \cdot \mathcal{S}_3$$

Moreover, observe that $\tilde{k} \geqslant \ell(n-1)$, therefore, using Fact 24, we get that $\mathcal{S}_3 \cdot \mathcal{G}_n^{\tilde{k}}(v) \cdot \mathcal{S}_1 \subseteq (e, \mathcal{C}_{2,n-1}[B])$. Moreover, since all chains in $\mathcal{S}_2$ have e as first element (see Fact 24), it is immediate that $(\mathcal{S}_2)^\omega \cdot (e, \mathcal{C}_{2,n-1}[B]) \cdot (\mathcal{S}_2)^\omega = (\mathcal{S}_2)^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[B]) \cdot (\mathcal{S}_2)^\omega$. This yields

$$\mathcal{G}_n^{\widehat{k-2}}(u_2) \subseteq \mathcal{S}_1 \cdot (\mathcal{S}_2)^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[B]) \cdot (\mathcal{S}_2)^\omega \cdot \mathcal{S}_3.$$

Since $\mathcal{S}_2 \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$, it is immediate by Operation (2) in the definition of Sat_n that $(\mathcal{S}_2)^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[B]) \cdot (\mathcal{S}_2)^\omega \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$. It then follows from Fact 22 that $\mathcal{S}_1 \cdot (\mathcal{S}_2)^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[B]) \cdot (\mathcal{S}_2)^\omega \cdot \mathcal{S}_3 \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ and therefore that $\mathcal{G}_n^{\widehat{k-2}}(u_2) \in {\downarrow} Sat_n^*[B](C \mapsto \mathfrak{I}_n[C])$ which terminates the proof. ◻

C Proof of Theorem 7: Characterization of $\Sigma_i(<)$

In this appendix, we prove Theorem 7, i.e., our characterization for $\Sigma_i(<)$. For this whole appendix, we assume that the level i in the quantifier alternation hierarchy is fixed.

Theorem 7. *Let L be a regular language and $\alpha : A^* \to M$ be its syntactic morphism. For all $i \geqslant 1$, L is definable in $\Sigma_i(<)$ iff α satisfies:*

$$s^\omega \leqslant s^\omega t s^\omega \quad \text{for all } (t,s) \in \mathcal{C}_{i-1}[\alpha] \tag{3}$$

There are two directions. We give each one its own subsection.

C.1 Equation (3) is necessary

We prove that the syntactic morphism of any $\Sigma_i(<)$-definable language satisfies (3). We state this in the following proposition.

Proposition 26. *Let L be a $\Sigma_i(<)$-definable language and let $\alpha : A^* \to M$ be its syntactic morphism. Then α satisfies (3).*

Proof. By hypothesis, L is defined by some $\Sigma_i(<)$-formula φ. Let k be its quantifier rank. Set $(t,s) \in \mathcal{C}_{i-1}[\alpha]$, we need to prove that $s^\omega \leqslant s^\omega t s^\omega$. Since $(t,s) \in \mathcal{C}_{i-1}[\alpha]$, by definition, there exist v, u such that $\alpha(v) = t$, $\alpha(u) = s$ and $v \lesssim_{i-1}^k u$. By Lemma 15, we immediately obtain

$$u^{2^k\omega} \cdot u^{2^k\omega} \lesssim_i^k u^{2^k\omega} \cdot v \cdot u^{2^k\omega}.$$

It then follows from Lemma 13 that for any $w_1, w_2 \in A^*$ we have:

$$w_1 \cdot u^{2^k\omega} \cdot u^{2^k\omega} \cdot w_2 \quad \lesssim_i^k \quad w_1 \cdot u^{2^k\omega} \cdot v \cdot u^{2^k\omega} \cdot w_2. \tag{8}$$

By definition, this means that $w_1 \cdot u^{2^k\omega} \cdot w_2 \in L$ implies that $w_1 \cdot u^{2^k\omega} v u^{2^k\omega} \cdot w_2 \in L$. Which, by definition of the syntactic preorder, means that $s^\omega \leqslant s^\omega t s^\omega$. ◻

C.2 Equation (3) is sufficient

It remains to prove that whenever α satisfies (3), L is definable in $\Sigma_i(<)$. This is a consequence of the following proposition.

Proposition 27. *Let L be a regular language such that its syntactic morphism $\alpha : A^* \to M$ satisfies* (3)*. Then there exists $k \in \mathbb{N}$ such that for all $u, v \in A^*$:*

$$u \lesssim_i^k v \Rightarrow \alpha(u) \leqslant \alpha(v)$$

Assume for now that Proposition 27 holds and let α satisfy (3). Let then $u, v \in A^*$ with $u \in L$ and $u \lesssim_i^k v$. By Proposition 27, we deduce that $\alpha(u) \leqslant \alpha(v)$ which, by definition of the preorder $\leqslant$, implies that $v \in L$. Therefore, $\lesssim_i^k$ saturates L, so L is definable in $\Sigma_i(<)$.

It remains to prove Proposition 27. We begin by choosing k. The choice depends on the following lemma. Recall that $\mathcal{C}_{i,2}^k[\alpha]$ is the set of chains of length 2 belonging to $\mathcal{C}_i^k[\alpha]$.

Fact 28 *For any morphism $\alpha : A^* \to M$ into a finite monoid M, there exists $k_i \in \mathbb{N}$ such that for all $k \geqslant k_i$, $\mathcal{C}_{i,2}^k[\alpha] = \mathcal{C}_{i,2}[\alpha]$.*

Proof. This is because for all $k < k'$, $\mathcal{C}_{i,2}^{k'}[\alpha] \subseteq \mathcal{C}_{i,2}^k[\alpha] \subseteq M^2$. Since M^2 is a finite set, there exists an index k_i such that for all $k \leqslant k_i$, $\mathcal{C}_{i,2}^k[\alpha] = \mathcal{C}_{i,2}^{k_i}[\alpha]$. It is then immediate by definition that $\mathcal{C}_{i,2}^{k_i}[\alpha] = \mathcal{C}_{i,2}[\alpha]$. ☐

Observe that while proving proving the existence k_i is easy, the proof is non-constructive and computing k_i from i, α is a difficult problem. In particular, having k_i allows us to compute all Σ_i-chains of length 2 via a brute-force algorithm. When $i = 2$, we proved in Proposition 5 that it suffices to take $k_2 = 3|M| \cdot 2^{|A|} \cdot 2 \cdot 2^{2^{2|M|^2}}$.

We can now prove Proposition 27. Set k_{i-1} as defined in Fact 28 for $i-1$. This means that (s,t) is a Σ_{i-1}-chain for α iff there exists $u, v \in A^*$ such that $\alpha(u) = s$, $\alpha(v) = t$ and $u \lesssim_{i-1}^{k_{i-1}} v$. We prove that Proposition 27 holds for $k = 6|M| + k_{i-1}$. This follows from the next lemma.

Lemma 29. *Let $h \in \mathbb{N}$ and $u, v \in A^*$, such that u admits an α-factorization forest of height smaller than h. Then*

$$u \lesssim_i^{2h+k_{i-1}} v \Rightarrow \alpha(u) \leqslant \alpha(v)$$

Observe that by Theorem 16 all words admit an α-factorization forest of height less than $3|M|$. Therefore, Proposition 27 is an immediate consequence of Lemma 29. It remains to prove the lemma.

Proof (of Lemma 29). We distinguish three cases depending on the nature of the topmost node in the α-factorization forest of u. If the topmost node is a leaf then u is a single letter word. Moreover, since $2h + k_{i-1} = 2 + k_{i-1} \geqslant 2$, we have $u \lesssim_i^2 v$, therefore, $v = u$ and $\alpha(u) = \alpha(v)$.

If the topmost node is a binary node then $u = u_1 \cdot u_2$ with u_1, u_2 admitting α-factorization forests of height $h_1, h_2 \leqslant h-1$. Using a simple Ehrenfeucht-Fraïssé argument, we get that $v = v_1 \cdot v_2$ with $u_1 \lesssim_i^{2h+k_{i-1}-1} v_1$ and $u_2 \lesssim_i^{2h+k_{i-1}-1} v_2$. Since $2h+k_{i-1}-1 \geqslant 2(h-1)+k_{i-1}$, we can use our induction hypothesis which yields that $\alpha(u_1) \leqslant \alpha(v_1)$ and $\alpha(u_2) \leqslant \alpha(v_2)$. By combining the two we obtain that $\alpha(u) = \alpha(u_1) \cdot \alpha(u_2) \leqslant \alpha(v_1) \cdot \alpha(v_2) = \alpha(v)$.

If the topmost node is an idempotent node for some idempotent e, then $u = u_1 \cdot u' \cdot u_2$ such that $\alpha(u_1) = \alpha(u_2) = \alpha(u') = e$ and u_1, u_2 admit α-factorization forests of height $h_1, h_2 \leqslant h-1$. By using a simple Ehrenfeucht-Fraïssé argument we get that $v = v_1 \cdot v' \cdot v_2$ such that $u_1 \lesssim_i^{2h+k_{i-1}-2} v_1$, $u' \lesssim_i^{2h+k_{i-1}-2} v'$ and $u_2 \lesssim_i^{2h+k_{i-1}-2} v_2$. Applying the induction hypothesis as in the previous case, we get that $e = \alpha(u_1) \leqslant \alpha(v_1)$ and $e = \alpha(u_2) \leqslant \alpha(v_2)$. However, we cannot apply induction on u' since the height of its α-factorization forest has not decreased. We use Equation (3) instead. We know that $u' \lesssim_i^{2h+k_{i-1}-2} v'$, therefore, by choice of k_i, we have $(\alpha(v'), \alpha(u')) \in \mathcal{C}_{i-1}[\alpha]$. Recall that by hypothesis of this case, $\alpha(u') = e$. Therefore, by Equation (3), we get that:

$$\alpha(u) = e \leqslant e \cdot \alpha(v') \cdot e \leqslant \alpha(v_1) \cdot \alpha(v') \cdot \alpha(v_2) = \alpha(v)$$

which terminates the proof. ◻

D Analyzing Σ_2-Chains: Chain Trees

In this appendix, we define chain trees. Chain trees are our main tool in the proof of the difficult 'if' direction of Theorem 10. The main goal of the notion is to analyze how Σ_2-chains are constructed. In particular we are interested in a specific property of the set of Σ_2-chains that we define now.

Alternation. Let M be a finite monoid. We say that a chain $(s_1, \dots, s_n) \in M^*$ has *alternation* ℓ if there are exactly ℓ indices i such that $s_i \neq s_{i+1}$. We say that a set of chains $\mathcal{S}$ has *bounded alternation* if there exists a bound $\ell \in \mathbb{N}$ such that all chains in $\mathcal{S}$ have alternation at most ℓ.

We will see in Appendix E that $\mathcal{C}_2[\alpha]$ having bounded alternation is another characterization of $\mathcal{B}\Sigma_2(<)$. The difficult direction of Theorem 10 will then be reduced to proving that if $\mathcal{C}_2[\alpha]$ has *unbounded alternation* then one of the two equations in the characterization is contradicted. Therefore, we will need a way to analyze how Σ_2-chains with high alternation are built. In particular, we will need to extract a property from the set of Σ_2-chains that decides which equation is contradicted. This is what chain trees are for. Intuitively, a chain tree is associated to a single Σ_2-chain and represents a computation of our algorithm (see Section 4) that yields this Σ_2-chain.

As we explained in the main paper, one can find connections between our proof and that of the characterization of boolean combination of open sets of trees [5]. In [5] as well, the authors consider a notion of "chains" which corresponds to open sets of trees and need to analyze how they are built. This is achieved with an object called "Strategy Tree". Though strategy trees and chain trees share the same purpose, i.e., analyzing how chains are built, there is no connection between the notions themselves since they deal with completely different objects.

We organize the appendix in three subsections. We first define the general notion of chain trees. In the second subsection, we define the main tool we use to analyze chain trees: context values. In particular, we prove that we can use context values to generate B-schemas. Finally, in the last subsection, we define a strict subset of chain trees: the *locally optimal chain trees* and prove that it suffices to consider only such chain trees (i.e., that for any Σ_2-chain there exists a locally optimal chain tree that "computes" it).

D.1 Definition

Set $\alpha : A^* \to M$ a morphism into a finite monoid M. We associate to α a set $\mathbb{T}[\alpha]$ of *chain trees*. As we explained, a chain tree is associated to a single Σ_2-chain for α and represents a way to compute this Σ_2-chain using our algorithm. Note that our algorithm works with sets of compatible sets of Σ_2-chains, while chain trees are for single Σ_2-chains. This difference will be reflected in the definition. For all $n \in \mathbb{N}$ we define $\ell_n = \omega(2^{M^n})$.

Chain Trees. Set $n \in \mathbb{N}$. A *chain tree T of level n* for α is an ordered unranked tree that may have two types of (unlabeled) inner nodes: product nodes and operation nodes, and two types of leaves, labeled with a Σ_2-chain of length n: initial leaves and operation leaves. Moreover, to each node x in the tree, we associate an alphabet $\mathsf{alph}(x) \subseteq A$ and a value $\mathsf{val}(x) \in M^n$ by induction on the structure of the tree.

Intuitively, each type of node corresponds to a part of the algorithm that computes Σ_2-chains. Initial leaves correspond to the initial trivial compatible sets from which the algorithm starts, product nodes correspond to the product (1), finally operation nodes and leaves can only be used together and correspond to the application of (2). We now give a precise definition of each type of node.

Initial Leaves. An initial leaf x is labeled with a constant Σ_2-chain $(s, \cdots, s) \in \mathcal{C}_{2,n}[\alpha, B]$ for some $B \subseteq A$. We set $\mathsf{alph}(x) = B$ and $\mathsf{val}(x) = (s, \cdots, s)$.

Operation Leaves. An operation leaf x is labeled with an arbitrary Σ_2-chain $\bar{s} \in \mathcal{C}_{2,n}[\alpha, B]$ for some $B \subseteq A$. We set $\mathsf{alph}(x) = B$ and $\mathsf{val}(x) = \bar{s}$. Note that we will set constraints on the parents of operation leaves. In particular, these parents are always operation nodes. We will see this in details when defining operation nodes.

Product Nodes. A product node x is unlabeled. It can have an arbitrary number of children $x_1, \dots, x_m$ which are all initial leaves, product nodes or operation nodes. In particular, we set $\mathsf{alph}(x) = \mathsf{alph}(x_1) \cup \cdots \cup \mathsf{alph}(x_m)$ and $\mathsf{val}(x) = \mathsf{val}(x_1) \cdots \mathsf{val}(x_m)$.

Operation Nodes. An operation node x has exactly $2\ell_n + 1$ children sharing the same alphabet B. The $(\ell_n + 1)$-th child, called the *central child* of x, has to be an operation leaf. The other children, called the *context children* of x, are either operation nodes, product nodes or initial leaves and the set of their values must be *compatible for* α, B (i.e. it must belong to $\mathfrak{C}_{2,n}[\alpha, B]$). Finally, we set a restriction on the value of the central child. Since the values of the context children of x form a compatible set of Σ_2-chains, they all share the same first component, that we call t. We require the first component of the value of the central child to be t^{ℓ_n}. This means that the central child is an operation node labeled with $(t^{\ell_n}, s_1, \dots, s_{n-1}) \in \mathcal{C}_{2,n}[\alpha, B]$. Finally, we set $\mathsf{alph}(x) = B$ and $\mathsf{val}(x) = \mathsf{val}(x_1) \cdots \mathsf{val}(x_{2\ell_n+1})$.

This terminates the definition of chain trees. The alphabet and value of a chain tree T, $\mathsf{alph}(T)$ and $\mathsf{val}(T)$, are the alphabet and value of its root. We give an example of a chain tree in Figure 2. Moreover, the following fact is immediate by definition.

Fact 30 *Let T be a chain tree and let $x_1, \dots, x_m$ be its leaves listed from left to right. Then $\mathsf{val}(T) = \mathsf{val}(x_1) \cdots \mathsf{val}(x_m)$.*

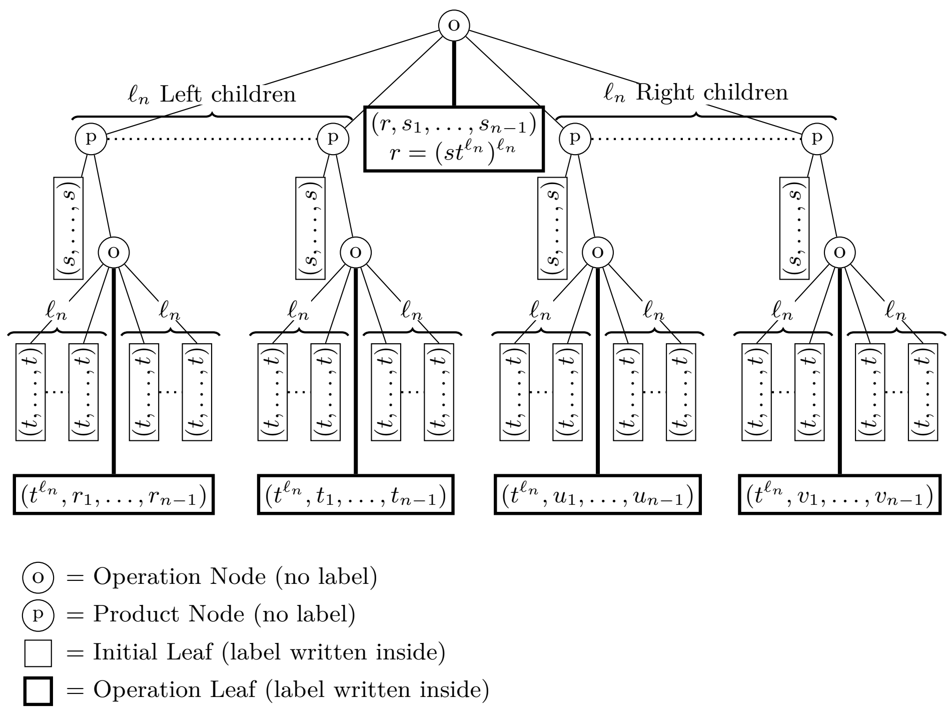

Fig. 2. An example of chain tree of level n

We denote by $\mathbb{T}_n[\alpha, B]$ the set of all chain trees of level n and alphabet B associated to α and by $\mathbb{T}[\alpha]$ the set of all chain trees associated to α. If $\mathbb{S}$ is a set of chain trees, we define $\mathsf{val}(\mathbb{S}) = \{\mathsf{val}(T) \mid T \in \mathbb{S}\}$. We now state "correctness" and "completeness" of chain trees, i.e., a chain is a Σ_2-chain iff it is the value of some chain tree. We prove this as a consequence of the validity of our algorithm for computing Σ_2-chains, stated in Proposition 5.

Proposition 31. $\mathcal{C}_{2,n}[\alpha, B] = \mathsf{val}(\mathbb{T}_n[\alpha, B])$.

Proof. That $\mathsf{val}(\mathbb{T}_n[\alpha, B]) \subseteq \mathcal{C}_{2,n}[\alpha, B]$ is immediate by definition and Fact 2. We concentrate on the other inclusion. Since Proposition 5 deals with sets of compatible Σ_2-chains rather than just Σ_2-chains, we prove a slightly stronger result. Two chain trees are said *compatible* if they have the same structure, the same alphabet and differ only by the labels of their operation leaves. For all $T \in \mathbb{T}[\alpha]$, we set $id(T) \subseteq \mathbb{T}[\alpha]$ as the set of all chain trees that are compatible with T.

Lemma 32. *Let $B \subseteq A$. Then*

$$Sat^*_n[B](C \mapsto \mathfrak{I}_n[C]) \quad \subseteq \quad \downarrow\{\mathsf{val}(id(T)) \mid \mathsf{alph}(T) = B\}$$

By Proposition 5, if $\bar{s}$ is a Σ_2-chain of length n for α, B, there exists $\mathcal{S} \in Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$ such that $\bar{s} \in \mathcal{S}$. Therefore the inclusion $\mathcal{C}_{2,n}[\alpha, B] \subseteq \mathsf{val}(\mathbb{T}_n[\alpha, B])$ is an immediate consequence of Lemma 32. It remains to prove Lemma 32.

Let $\mathcal{T} \in Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$. We need to construct $T \in \mathbb{T}[\alpha]$ such that $\mathcal{T} \subseteq \mathsf{val}(id(T))$. By definition, $\mathcal{T} \in Sat^j_n[B](C \mapsto \mathfrak{I}_n[C])$ for some $j \in \mathbb{N}$. We proceed by induction on j. Assume first that $j = 0$. Then $\mathcal{T} = \{(s, \dots, s)\} \in \mathfrak{I}_n[B]$. By definition this means that $\mathcal{T} = \{\mathsf{val}(T)\}$ where T is the chain tree composed of a single initial leaf with label $(s, \dots, s)$ and alphabet B. Assume now that $j \geqslant 1$. For all $D \subseteq A$, we set $\mathfrak{T}_D = Sat^{j-1}_n[D](C \mapsto \mathfrak{I}_n[C])$. By definition, we have $\mathcal{T} \in \mathfrak{T}_B \cup \mathfrak{M}_B \cup \mathfrak{O}_B$ with

$$\begin{aligned}
\mathfrak{M}_B &= \cup_{C \cup D = B}(\mathfrak{T}_C \cdot \mathfrak{T}_D) \\
\mathfrak{O}_B &= \{\mathcal{S}^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[\alpha, B]) \cdot \mathcal{S}^\omega \mid \mathcal{S} \in \mathfrak{T}_B\}
\end{aligned}$$

If $\mathcal{T} \in \mathfrak{T}_B$, the result is immediate by induction hypothesis. Assume now that $\mathcal{T} \in \mathfrak{M}_B$. By definition, this means that there exist C, D such that $C \cup D = B$ and $\mathcal{T}_C, \mathcal{T}_D$ in $\mathfrak{T}_C, \mathfrak{T}_D$ such that $\mathcal{T} = \mathcal{T}_C \cdot \mathcal{T}_B$. Using our induction hypothesis, we get T_C, T_D such that $\mathcal{T}_C \subseteq \mathsf{val}(id(T_C))$ and $\mathcal{T}_D \subseteq \mathsf{val}(id(T_D))$. Consider T the chain tree whose topmost node is a product with T_C, T_D as children. It is immediate by definition that $\mathcal{T} \subseteq \mathsf{val}(id(T_C)) \cdot \mathsf{val}(id(T_D)) = \mathsf{val}(id(T))$.

It remains to treat the case when $\mathcal{T} \in \mathfrak{O}_B$. By definition, we get $\mathcal{S} \in \mathfrak{T}_B$ such that $\mathcal{T} = \mathcal{S}^\omega \cdot (1_M, \mathcal{C}_{2,n-1}[\alpha, B]) \cdot \mathcal{S}^\omega$. Note that since $\mathcal{S}, \mathcal{T} \in Sat^*_n[B](C \mapsto \mathfrak{I}_n[C])$, by Proposition 5, $\mathcal{S}, \mathcal{T} \in \mathfrak{C}_{2,n}[\alpha, B]$. We denote by s the first element common to all chains in $\mathcal{S}$. Note that since $\ell_n = \omega(2^{M^n})$, the first element common to all chains in $\mathcal{T}$ is s^{ℓ_n}. Set $\mathcal{R}_{s^{\ell_n}}$ as the set of all Σ_2-chains of length n for α, B that have s^{ℓ_n} as first element. By definition $\mathcal{T} \subseteq \mathcal{R}_{s^{\ell_n}}$. Moreover,

$$\mathcal{T} = \mathcal{S}^{\ell_n} \cdot \mathcal{T} \cdot \mathcal{S}^{\ell_n} \subseteq \mathcal{S}^{\ell_n} \cdot \mathcal{R}_{s^{\ell_n}} \cdot \mathcal{S}^{\ell_n}$$

By induction hypothesis there exists a chain tree $T_{\mathcal{S}}$ of alphabet B such that $\mathcal{S} \subseteq \mathsf{val}(id(T_{\mathcal{S}}))$. Let T be the chain tree whose topmost node is an operation node whose context children are all copies of $T_{\mathcal{S}}$ and whose central child is the operation leaf labeled with some arbitrary chosen Σ_2-chain in $\mathcal{R}_{s^{\ell_n}}$. Observe that by definition, $\mathsf{val}(id(T)) = (\mathsf{val}(id(T_{\mathcal{S}})))^{\ell_n} \cdot \mathcal{R}_{s^{\ell_n}} \cdot (\mathsf{val}(id(T_{\mathcal{S}})))^{\ell_n}$. Therefore, since $\mathcal{S} \subseteq \mathsf{val}(id(T_{\mathcal{S}}))$, we have $\mathcal{T} \subseteq \mathsf{val}(id(T))$ which terminates the proof. Note that the tree we obtained is particular: all subtrees rooted at context children of an operation node are identical. $\square$

Alternation and Recursive Alternation of a Chain Tree. The *alternation* of a chain tree is the alternation of its value. We say that $\mathbb{T}[\alpha]$ has *unbounded alternation* if the set $\mathsf{val}(\mathbb{T}[\alpha])$ has unbounded alternation. Note that by Proposition 31, $\mathcal{C}_2[\alpha]$ has unbounded alternation iff $\mathbb{T}[\alpha]$ has unbounded alternation.

In the proof we will be interested in another property of chain trees: *recursive alternation*. Recursive alternation corresponds to the maximal alternation of labels of operation leaves in the tree. More precisely, if T is a chain tree, its *recursive alternation* is the largest natural j such that there exists an *operation leaf* in T whose label has alternation j. An important idea in the proof will be to separate the case when we can find a set of chain trees with unbounded alternation but bounded recursive alternation from the converse one. However, in order to make this work, we will need to add one more condition to our trees. Intuitively, we need to know that if a tree has high recursive alternation, this is necessary, i.e, the tree cannot be modified into a tree that has low alternation while keeping the same value. This is what we do with local optimality.

D.2 Locally Optimal Chain Trees

For all n, B, we define a strict subset of $\mathbb{T}_n[\alpha, B]$ called the set of *Locally Optimal Chain Trees*. We then prove that we can assume without loss of generality that all chain trees we consider are locally optimal.

We first define local optimality as a property of a single node x in a chain tree T. We will generalize the notion to a whole tree by saying that it is locally optimal if and only if all its *operation leaves* are locally optimal. Given a node x, local optimality of x depends on two parameters of x: its value $\mathsf{val}(x)$ and a new parameter called its *context value*, $\mathsf{cval}(x)$, that we define now.

Context Value of a Node. Let T be a chain tree of level n and let $x_1, \dots, x_m$ be the leaves of T sorted in prefix order. Recall that by Fact 30, $\mathsf{val}(T) = \mathsf{val}(x_1) \cdots \mathsf{val}(x_m)$. To every node x of T, we associate a pair $\mathsf{cval}(x) \in (\mathcal{C}_{2,n}[\alpha])^2$ called the *context value* of x. Set $x_i, \dots, x_j$ as the leaves of the subtree rooted at x (in prefix order). We set $\mathsf{cval}(x) = (\mathsf{val}(x_1) \cdots \mathsf{val}(x_{i-1}), \mathsf{val}(x_{j+1}) \cdots \mathsf{val}(x_m))$. Note that since for all i, $\mathsf{val}(x_i) \in \mathcal{C}_{2,n}[\alpha]$, $\mathsf{cval}(x)$ is indeed a pair in $(\mathcal{C}_{2,n}[\alpha])^2$. By definition and Fact 30 one can verify the two following facts:

Fact 33 *Let x be a node in a chain tree T and set $\mathsf{cval}(x) = (\bar{s}, \bar{s}')$. Then $\mathsf{val}(T) = \bar{s} \cdot \mathsf{val}(x) \cdot \bar{s}'$.*

Fact 34 *Let x be an inner node in a chain tree T and set $\mathsf{cval}(x) = (\bar{s}, \bar{s}')$. Set $z_1, \dots, z_k$ as the children of x with context values $\mathsf{cval}(z_i) = (\bar{q}_i, \bar{q}_i{}')$. Then, for all i, $\bar{q}_i = \bar{s} \cdot \mathsf{val}(z_1) \cdots \mathsf{val}(z_{i-1})$ and $\bar{q}_i{}' = \mathsf{val}(z_{i+1}) \cdots \mathsf{val}(x_k) \cdot \bar{s}'$.*

In many cases, we will work with context values that are constant, i.e. $\mathsf{cval}(x) = ((s, \dots, s), (s', \dots, s'))$. In these cases, if $(t_1, \dots, t_n)$ is a chain, it will be convenient to simply write $s \cdot (t_1, \dots, t_n) \cdot s'$ for $(s, \dots, s) \cdot (t_1, \dots, t_n) \cdot (s', \dots, s')$.

Local Optimality. Set $(s, s') \in M^2$ and T a chain tree. Let x be any node in T, $(t_1, \cdots, t_n) = \mathsf{val}(x)$ and $((s_1, \cdots, s_n), (s'_1, \cdots, s'_n)) = \mathsf{cval}(x)$. We say that x is *locally optimal for* (s, s') if for all $i < n$ such that $t_i \neq t_{i+1}$ the following condition holds:

$$s \cdot s_{i+1} \cdot t_i \cdot s'_{i+1} \cdot s' \neq s \cdot s_{i+1} \cdot t_{i+1} \cdot s'_{i+1} \cdot s'$$

Intuitively this means that for all i, changing t_i to t_{i+1} in the value of x is necessary to get alternation at position i in the value of the tree (see Fact 33). We say that a chain tree T is *locally optimal for* (s, s') if all its **operation leaves** are locally optimal for (s, s'). We say that T is locally optimal iff it is locally optimal for $(1_M, 1_M)$. This means that locally optimality of a chain tree only depends on the context values and labels of operation leaves in the tree. The following fact is immediate from the definitions:

Fact 35 *Let $(s, s') \in M^2$. Assume that T is locally optimal for (s, s'). Then T is locally optimal (i.e. locally optimal for $(1_M, 1_M)$).*

We finish with our main proposition, which states that for any chain tree, there exists a locally optimal one with the same value. In particular, this means that we will always be able to assume that our chain trees are locally optimal.

Proposition 36. *Let $T \in \mathbb{T}_n[\alpha, B]$ and $(s, s') \in M^2$. There exists $T' \in \mathbb{T}_n[\alpha, B]$ which is locally optimal for (s, s') and such that $s \cdot \mathsf{val}(T) \cdot s' = s \cdot \mathsf{val}(T') \cdot s'$.*

Proof. Set $T \in \mathbb{T}_n[\alpha, B]$, we explain how to construct T'. For all $i < n$, we define the i-alternation of T as the number of operation leaves x in T such that $\mathsf{val}(x) = (t_1, \cdots, t_n)$ with $t_i \neq t_{i+1}$. Finally, we define the *index of* T as the sequence of its i-alternations ordered with increasing i.

We can now describe the construction. Assume that T is not locally optimal for (s, s'). We explain how to construct a second chain tree T' such that

1. $s \cdot \mathsf{val}(T) \cdot s' = s \cdot \mathsf{val}(T') \cdot s'$.
2. T' has strictly smaller index than T.

It then suffices to apply this operation recursively to T until we get the desired tree. We now explain the construction. Since T is not locally optimal for (s, s'), there exists an operation leaf x of T that is not locally optimal for (s, s'). Let

$(t_1,\dots,t_n) = \mathsf{val}(x)$ and $((s_1,\dots,s_n),(s'_1,\dots,s'_n)) = \mathsf{cval}(x)$. By choice of x, there exists $i < n$ such that $t_i \neq t_{i+1}$ and $ss_{i+1}t_is'_{i+1}s' = ss_{i+1}t_{i+1}s'_{i+1}s'$. We set T' as the chain tree obtained from T by replacing the label of x with $(t_1,\dots,t_i,t_i,t_{i+2},\dots,t_n)$. By choice of i and Fact 33, it is immediate that $s \cdot \mathsf{val}(T) \cdot s' = s \cdot \mathsf{val}(T') \cdot s'$. Moreover, for any $j < i$, T, T' have the same j-alternation and T' has by definition strictly smaller i-alternation than T. It follows that T' has strictly smaller index than T which terminates the proof. ☐

E Proof of Theorem 10: Characterization of $\mathcal{B}\Sigma_2(<)$

This appendix is devoted to the proof of Theorem 10, i.e., the decidable characterization of $\mathcal{B}\Sigma_2(<)$. We actually prove a more general theorem that includes a second characterization in terms of alternation of $\mathcal{C}_2[\alpha]$, which will be needed as an intermediary step when proving the difficult 'if' direction of Theorem 10.

Theorem 37. *Let L be a regular language and let $\alpha : A^* \to M$ be its syntactic morphism. The three following properties are equivalent:*

1. *L is definable in $\mathcal{B}\Sigma_2(<)$.*
2. *$\mathcal{C}_2[\alpha]$ has bounded alternation.*
3. *M satisfies the following equations:*

$$\begin{array}{c} s_1^\omega s_3^\omega = s_1^\omega s_2 s_3^\omega \\ s_3^\omega s_1^\omega = s_3^\omega s_2 s_1^\omega \end{array} \quad \text{for } (s_1,s_2,s_3) \in \mathcal{C}_2[\alpha] \tag{4}$$

$$\begin{array}{c} (s_2t_2)^\omega s_1 (t'_2s'_2)^\omega = (s_2t_2)^\omega s_2 t_1 s'_2 (t'_2 s'_2)^\omega \\ \text{for } (s_1,s_2,s'_2) \text{ and } (t_1,t_2,t'_2)\ B\text{-schemas for some } B \subseteq A \end{array} \tag{5}$$

Observe that Theorem 10 is exactly the equivalence between Items 1 and 3 in Theorem 37. Therefore it suffices to prove Theorem 37. Intuitively, Item 2 seems harder to decide than Item 3, since it requires computing a description of the whole set $\mathcal{C}_2[\alpha]$ rather than just the Σ_2-chains and sets of compatible Σ_2-chains of length 2 and 3. However, it will serve as a convenient intermediary for proving Item 3.

We now turn to the proof of Theorem 37. We prove that $1 \Rightarrow 3 \Rightarrow 2 \Rightarrow 1$. In this appendix, we give full proofs for the two "easy" directions: $1 \Rightarrow 3$ and $2 \Rightarrow 1$. For the direction $3 \Rightarrow 2$, we use chain trees to reduce the proof to two propositions. We then give each proposition its own Appendix: Appendix F and Appendix G.

E.1 $1 \Rightarrow 3$

We prove the direction $1 \Rightarrow 3$ in Theorem 37 which is stated in the following lemma.

Lemma 38. *Let L be a regular language and α be its syntactic morphism. Assume that L is definable in $\mathcal{B}\Sigma_2(<)$, then α satisfies* (4) *and* (5) .

The remainder of this subsection is devoted to proving Lemma 38. The proof is an Ehrenfeucht-Fraïssé argument. We begin by defining the equivalence associated to $\mathcal{B}\Sigma_2(<)$. For any $k \in \mathbb{N}$, we write $w \cong_2^k w'$ iff w and w' satisfy the same $\mathcal{B}\Sigma_2(<)$ formulas of quantifier rank k. Therefore, a language if definable by a $\mathcal{B}\Sigma_2(<)$ formula of rank k iff it is saturated by $\cong_2^k$. One can verify that $\cong_2^k$ is an equivalence and that $w \cong_2^k w'$ iff $w \lesssim_2^k w'$ and $w' \lesssim_2^k w$.

We can now prove the lemma. By hypothesis there exists some $\mathcal{B}\Sigma_2(<)$ formula φ that defines L, we set k as the quantifier rank of this formula.

Proving Equation (4)**.** Set $(s_1, s_2, s_3) \in \mathcal{C}_2[\alpha]$, we prove that $s_1^\omega s_3^\omega = s_1^\omega s_2 s_3^\omega$ (the dual case is proved in the same way). We prove that there exist $w_1, w_2, w_3 \in A^*$ such that $\alpha(w_1) = s_1$, $\alpha(w_2) = s_2$, $\alpha(w_3) = s_3$ and for all pair of words $u, v \in A^*$:

$$uw_1^{2^k\omega} w_3^{2^k\omega} v \quad \cong_2^k \quad uw_1^{2^k\omega} w_2 w_3^{2^k\omega} v \tag{9}$$

Set $N = 2^k\omega$. By definition of $\cong_2^k$, (9) means that $u(w_1^N w_3^N)v$ and $u(w_1^N w_2 w_3^N)v$ cannot be distinguished by a $\mathcal{B}\Sigma_2(<)$ formula of quantifier rank k. Hence, by definition of k, we get

$$u(w_1^N w_3^N)v \in L \quad \text{iff} \quad u(w_1^N w_2 w_3^N)v \in L$$

Therefore, by definition of w_1, w_2, w_3, of N, and of the syntactic monoid this will prove that $s_1^\omega s_3^\omega = s_1^\omega s_2 s_3^\omega$.

Since $(s_1, s_2, s_3) \in \mathcal{C}_2[\alpha]$ by assumption, there exist w_1, w_2, w_3 such that $w_1 \lesssim_2^k w_2 \lesssim_2^k w_3$ and $\alpha(w_1) = s_1$, $\alpha(w_2) = s_2$, $\alpha(w_3) = s_3$. Set $u, v \in A^*$. We need to prove that

$$u(w_1^N w_3^N)v \quad \lesssim_2^k \quad u(w_1^N w_2 w_3^N)v \tag{10}$$

$$u(w_1^N w_2 w_3^N)v \quad \lesssim_2^k \quad u(w_1^N w_3^N)v \tag{11}$$

By definition of w_1, w_2, we have $w_1 \lesssim_2^k w_2$. By Lemma 14, we obtain $w_1^{N-1} \lesssim_2^k w_1^N$. Therefore, using Lemma 13 we first get $w_1^N \lesssim_2^k w_1^N w_2$, and then that (10) holds.

The proof of (11) is similar: by definition, we have $w_2 \lesssim_2^k w_3$, and by Lemma 14 we get $w_3^N \lesssim_2^k w_3^{N-1}$. Using Lemma 13 again, we conclude that $w_2 w_3^N \lesssim_2^k w_3^N$, and then that (11) holds.

Proving Equation (5)**.** It remains to prove that α satisfies Equation (5). We begin with a lemma on B-schemas.

Lemma 39. *Assume that (s_1, s_2, s_2') is a B-schema. Then for all $k \in \mathbb{N}$ there exist $w_1, w_2, w_2' \in A^*$ such that:*

- $\mathsf{alph}(w_1) = \mathsf{alph}(w_2) = \mathsf{alph}(w_2') = B$.
- $\alpha(w_1) = s_1, \alpha(w_2) = s_2$ *and* $\alpha(w_2') = s_2'$.
- *for all* $u \in B^*$, $w_1 \lesssim_2^k w_2 u w_2'$.

Proof. This is proved using Lemma 15. Fix a B-schema (s_1, s_2, s'_2) and $k \in \mathbb{N}$. By definition, there exist $\mathcal{T} \in \mathfrak{C}_2[\alpha, B]$ and $r_1, r'_1 \in M$ satisfying $s_1 = r_1 r'_1$, $(r_1, s_2) = (t_1, t_2) \cdot (q, q_2)$ and $(r_1, s'_2) = (q, q'_2) \cdot (t'_1, t'_2)$ with $(t_1, t_2), (t'_1, t'_2) \in \mathcal{C}_2[\alpha, B]$ and $(q, q_2), (q, q'_2) \in \mathcal{T}^\omega = \mathcal{T}^{2^{2k}\omega}$. By definition of Σ_2-chains, we obtain words $v_1, v, v'_1, w_2, w'_2 \in A^*$ satisfying the following properties:

a) $\mathsf{alph}(v_1) = \mathsf{alph}(v) = \mathsf{alph}(v'_1) = \mathsf{alph}(w_2) = \mathsf{alph}(w'_2) = B$
b) $\alpha(v_1) = t_1$, $\alpha(v'_1) = t'_1$, $\alpha(w_2) = t_2 q_2$, $\alpha(w'_2) = q'_2 t'_2$ and $\alpha(v^{2^{2k}\omega}) = q$.
c) $v_1 v^{2^{2k}\omega} \lesssim_2^k w_2$ and $v^{2^{2k}\omega} v'_1 \lesssim_2^k w'_2$.

Set $w_1 = v_1 v^{2^{2k}\omega} v^{2^{2k}\omega} v'_1$ and observe that by item *a)*, $\mathsf{alph}(w_1) = \mathsf{alph}(w_2) = \mathsf{alph}(w_2) = B$. Moreover, by item *b)*, $\alpha(w_1) = t_1 q q t'_1 = r_1 r'_1 = s_1$, $\alpha(w_2) = t_2 q_2 = s_2$ and $\alpha(w'_2) = q'_2 t'_2 = s'_2$. Finally, it is immediate using Ehrenfeucht-Fraïssé games that for any word $u \in B^*$, $u \lesssim_1^k v^{2^k\omega}$. Therefore it follows from Lemma 15 that $w_1 \lesssim_2^k v_1 v^{2^{2k}\omega} u v^{2^{2k}\omega} v'_1$. Using item *c)*, we then conclude that $w_1 \lesssim_2^k w_2 u w'_2$. ☐

We can now use Lemma 39 to prove that α satisfies Equation (5). Let (s_1, s_2, s'_2) and (t_1, t_2, t'_2) be B-schemas. Let $w_1, w_2, w'_2 \in A^*$ of images s_1, s_2, s'_2 and $v_1, v_2, v'_2 \in A^*$ of images t_1, t_2, t'_2 satisfying the conditions of Lemma 39. We prove that for any $u, v \in A^*$:

$$u[(v_2 w_2)^N v_1 (w'_2 v'_2)^N] v \quad \cong_2^k \quad u[(v_2 w_2)^N v_2 w_1 v'_2 (w'_2 v'_2)^N] v \qquad (12)$$

where again $N = 2^k\omega$. By definition of the syntactic monoid and since L is defined by a $\mathcal{B}\Sigma_2(<)$ formula of rank k, Equation (5) will follow. Observe that the words v_1, v_2, v'_2 and w_1, w_2, w'_2 given by Lemma 39 satisfy

$$v_1 \lesssim_2^k v_2 w_1 v'_2, \qquad (13)$$

$$w_1 \lesssim_2^k w_2 v_1 w'_2. \qquad (14)$$

Using Lemma 13, we may multiply (13) by $u(v_2 w_2)^N$ on the left and by $(w'_2 v'_2)^N v$ on the right:

$$u(v_2 w_2)^N v_1 (w'_2 v'_2)^N v \quad \lesssim_2^k \quad u(v_2 w_2)^N v_2 w_1 v'_2 (w'_2 v'_2)^N v.$$

For the converse direction, from Lemma 14, we have $(v_2 w_2)^N \lesssim_2^k (v_2 w_2)^{N-1}$ and $(w'_2 v'_2)^N \lesssim_2^k (w'_2 v'_2)^{N-1}$. Using (14) and Lemma 13 again, we conclude that:

$$u(v_2 w_2)^N v_2 w_1 v'_2 (w'_2 v'_2)^N v \quad \lesssim_2^k \quad u(v_2 w_2)^{N-1} v_2 (w_2 v_1 w'_2) v'_2 (w'_2 v'_2)^{N-1} v$$

i.e.,

$$u(v_2 w_2)^N v_2 w_1 v'_2 (w'_2 v'_2)^N v \quad \lesssim_2^k \quad u(v_2 w_2)^N v_1 (w'_2 v'_2)^N v.$$

E.2 2 ⇒ 1

We prove the direction 2 ⇒ 1 in Theorem 37 which is stated in the following lemma.

Lemma 40. *Let L be a regular language and α its syntactic morphism. Assume that $\mathcal{C}_2[\alpha]$ has bounded alternation, then L is definable in $\mathcal{B}\Sigma_2(<)$.*

Proof. Assume that $\mathcal{C}_2[\alpha]$ has bounded alternation. We prove that there exists $k \in \mathbb{N}$ such that for all $w, w' \in A^*$, $w \cong_2^k w' \Rightarrow \alpha(w) = \alpha(w')$. This proves that L is saturated with $\cong_2^k$ and hence definable by a $\mathcal{B}\Sigma_2(<)$ formula of quantifier rank k.

We proceed by contradiction. Assume that for all $k \in \mathbb{N}$ there exists $w_k, w'_k \in A^*$ such that $w_k \cong_2^k w'_k$ and $\alpha(w_k) \neq \alpha(w'_k)$. Notice that since there are only finitely many pairs in M^2, there must exist a pair $(s, s') \in M^2$ such that $s \neq s'$ and there exists arbitrarily large naturals k such that $\alpha(w_k) = s$ and $\alpha(w'_k) = s'$. We prove that $(s, s')^* \subseteq \mathcal{C}_2[\alpha]$ which contradicts that $\mathcal{C}_2[\alpha]$ has unbounded alternation (recall that $s \neq s'$). By definition for all $k \in \mathbb{N}$ there exists $\ell \geqslant k$ such that $\alpha(w_\ell) = s$ and $\alpha(w'_\ell) = s'$, since $\ell \geqslant k$ and by definition of $\cong_2^k$ this means that :

$$w_\ell \lesssim_2^k w'_\ell \lesssim_2^k w_\ell \lesssim_2^k w'_\ell \lesssim_2^k w_\ell \lesssim_2^k w'_\ell \lesssim_2^k \cdots$$

Hence for all k, j, $(s, s')^j \in \mathcal{C}_2^k[\alpha]$ and therefore, for all j $(s, s')^j \in \mathcal{C}_2[\alpha]$ which terminates the proof. ◻

E.3 3 ⇒ 2

This is the most difficult direction of Theorem 37. We state it in the following proposition.

Proposition 41. *Let L be a regular language, $\alpha : A^* \to M$ be its syntactic morphism. Assume that α satisfies* (4) *and* (5)*, then $\mathcal{C}_2[\alpha]$ has bounded alternation.*

For the remaining of the section, we assume that L, M and α are fixed as in the statement of the proposition. We prove the contrapositive of Proposition 41: if $\mathcal{C}_2[\alpha]$ has unbounded alternation, then either Equation (4) or Equation (5) must be contradicted. We use chain trees to separate this property into two properties that we will prove in Appendix F and Appendix G. Consider the two following propositions

Proposition 42. *Assume that there exists a set of locally optimal chain trees $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ with unbounded alternation but bounded recursive alternation. Then α does not satisfy Equation* (4)*.*

Proposition 43. *Assume that there exists a set of locally optimal chain trees $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ with unbounded alternation and that all such sets have unbounded recursive alternation. Then α does not satisfy Equation* (5)*.*

Proposition 42 and Proposition 43 are proven in Appendix G and Appendix F. We finish this appendix by using them to conclude the proof of Proposition 41.

If $\mathcal{C}[\alpha]$ has unbounded alternation. By Proposition 36, we know that there exists a set of locally optimal chain trees $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ with unbounded alternation. If $\mathbb{S}$ can be chosen with bounded recursive alternation, there is a contradiction to Equation (4) by Proposition 42. Otherwise there is a contradiction to Equation (5) by Proposition 43 which terminates the proof of Proposition 41.

F Proof of Proposition 43

Recall that we fixed a morphism $\alpha : A^* \to M$ into a finite monoid M. We prove Proposition 43.

Proposition 43. *Assume that there exists a set of locally optimal chain trees* $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ *with unbounded alternation and that all such sets have unbounded recursive alternation. Then α does not satisfy Equation* (5)*.*

We define a new object that is specific to this case: the *Chain Graph*. The chain graph describes a construction process for a subset of the set of Σ_2-chains for α. While this subset is potentially strict, we will prove that under the hypothesis of Proposition 43, it is sufficient to derive a contradiction to Equation (5).

Chain Graph. We define a graph $G[\alpha] = (V, E)$ whose edges are labeled by subsets of the alphabet A. We call $G[\alpha]$ the *chain graph* of α. The set V of nodes of $G[\alpha]$ is the set $V = M^2 \times M$. Let $((s, s'), u)$ and $((t, t'), v)$ be nodes of $G[\alpha]$ and $B \subseteq A$, then E contains an edge labeled by B from $((s, s'), u)$ to $((t, t'), v)$ iff there exists a B-schema $(s_1, s_2, s'_2) \in M^3$ such that:

- $s \cdot s_1 \cdot s' = u$.
- $s \cdot s_2 = t$ and $s'_2 \cdot s' = t'$.

Observe that the definition does not depend on v. We say that $G[\alpha]$ is *recursive* if it contains a cycle such that

a) all edges in the cycle are labeled by the same alphabet $B \subseteq A$,
b) the cycle contains two nodes $((s, s'), u)$, $((t, t'), v)$ such that $u \neq v$.

We now prove Proposition 43 as a consequence of the two following propositions.

Proposition 44. *Assume that $G[\alpha]$ is recursive. Then α does not satisfy* (5)*.*

Proposition 45. *Assume that there exists a set of locally optimal chain trees* $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ *with unbounded alternation and that all such sets have unbounded recursive alternation. Then $G[\alpha]$ is recursive.*

Observe that Proposition 43 is an immediate consequence of Propositions 44 and 45. Before proving them, note that the notion of chain graph is inspired from the notion of strategy graph in [5]. This is because both notions are designed to derive contradiction to similar equations. However, our proof remains fairly different from the one of [5]. The reason for this is that the main difficulty here is proving Proposition 45, i.e., going from chain trees (which are unique to our setting) to a recursive chain graph. On the contrary, the much simpler proof of Proposition 44 is similar to the corresponding one in [5].

F.1 Proof of Proposition 44

Proposition 44. *Assume that $G[\alpha]$ is recursive then α does not satisfy* (5)*.*

Assume that $G[\alpha]$ is recursive. By definition, we get $B \subseteq A$, a cycle whose edges are all labeled with B and two consecutive nodes $((s, s'), u)$ and $((t, t'), v)$ in this cycle such that $u \neq v$. Since there exists an edge $((s, s'), u) \xrightarrow{B} ((t, t'), v)$, we obtain a B-schema (s_1, s_2, s'_2) such that

$$\begin{aligned} u &= s \cdot s_1 \cdot s', \\ t &= s \cdot s_2, \\ t' &= s'_2 \cdot s'. \end{aligned}$$

Moreover, one can verify that since $((s, s'), u)$ and $((t, t'), v)$ are in the same cycle with all edges labeled by B, there exists another B-schema (t_1, t_2, t'_2) and $w, w' \in B^*$ such that

$$\begin{aligned} v &= t \cdot t_1 \cdot t', \\ s &= t \cdot t_2 \cdot \alpha(w), \\ s' &= \alpha(w') \cdot t'_2 \cdot t'. \end{aligned}$$

By combining all these definitions we get:

$$\begin{aligned} u &= s(s_2 t_2 \alpha(w))^{\omega+1} s_1 (\alpha(w') t'_2 s'_2)^{\omega+1} s' \\ v &= s(s_2 t_2 \alpha(w))^{\omega+1} s_2 t_1 s'_2 (\alpha(w') t'_2 s'_2)^{\omega+1} s' \end{aligned}$$

Set $r_1 = \alpha(w) s_1 \alpha(w')$, $r_2 = \alpha(w) s_2$ and $r'_2 = s'_2 \alpha(w')$. One can verify that since $w, w' \in B^*$ and (s_1, s_2, s'_2) is a B-schema, (r_1, r_2, r'_2) is a B-schema as well. Moreover, by reformulating the equalities above we get:

$$\begin{aligned} u &= s s_2 t_2 (r_2 t_2)^{\omega} r_1 (t'_2 r'_2)^{\omega} t'_2 s'_2 s' \\ v &= s s_2 t_2 (r_2 t_2)^{\omega} r_2 t_1 r'_2 (t'_2 r'_2)^{\omega} t'_2 s'_2 s' \end{aligned}$$

Therefore, Equation (5) would require $u = v$. Since $u \neq v$ by hypothesis, α does not satisfy (5) and we are finished.

F.2 Proof of Proposition 45

Proposition 45. *Assume that there exists a set of locally optimal chain trees $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ with unbounded alternation and that all such sets have unbounded recursive alternation. Then $G[\alpha]$ is recursive.*

In the remainder of the section, we assume that α satisfies the hypothesis of Proposition 45. Set $B \subseteq A$ and let $((s, s'), u)$ be a node of $G[\alpha]$, we say that $((s, s'), u)$ is *B-alternating* if for all n, there exists $(s_1, \dots, s_n) \in \mathcal{C}_{2,n}[\alpha, B]$ such that the chain $(s s_1 s', \dots, s s_n s')$ has alternation $n - 1$ and $s s_1 s' = u$.

Lemma 46. *$G[\alpha]$ contains at least one B-alternating node for some B.*

Proof. This is because $\mathbb{S}$ has unbounded alternation. It follows that there exists a least one $u \in M$ such that there are Σ_2-chains with arbitrary high alternation and u as first element. By definition, the node $((1_M, 1_M), u)$ is then B-alternating for some B. ☐

For the remainder of the proof we define B as a minimal alphabet such that there exists a B-alternating node in $G[\alpha]$. By this we mean that for any $C \subsetneq B$, there exists no C-alternating node in $G[\alpha]$.

Lemma 47. *Let $((s, s'), u)$ be any B-alternating node of $G[\alpha]$. Then there exists a node $((t, t'), v)$ such that*

1. *$((t, t'), v)$ is B-alternating.*
2. *$((s, s'), u) \xrightarrow{B} ((t, t'), v)$.*
3. *$u \neq v$.*

By definition $G[\alpha]$ has finitely many nodes. Therefore, since by definition, there exists at least one B-alternating node, it is immediate from Lemma 47 that $G[\alpha]$ must contain a cycle whose edges are all labeled by B. Moreover, by Item 3 in Lemma 47, this cycle contains two nodes $((s, s'), u)$ and $((t, t'), v)$ such that $u \neq v$. We conclude that $G[\alpha]$ is recursive which terminates the proof of Proposition 45. It remains to prove Lemma 47.

Proof. We proceed in three steps. We first use our hypothesis to construct a special set of chain trees $\mathbb{U}$ of alphabet B. Then, we choose a chain tree T in $\mathbb{U}$ with large enough recursive alternation. Finally, we use T to construct the desired node $((t, t'), v)$. We begin with the construction of $\mathbb{U}$.

Construction of $\mathbb{U}$. We construct a set $\mathbb{U}$ of chain trees that satisfies the following properties:

1. For all $T \in \mathbb{U}$, $\mathsf{alph}(T) = B$.
2. All chains in $s \cdot val(\mathbb{U}) \cdot s'$ have u as first element.
3. All trees in $\mathbb{U}$ are locally optimal for (s, s').
4. $\mathbb{U}$ has unbounded recursive alternation.

We use the fact that $((s, s'), u)$ is B-alternating and the hypothesis in Proposition 45. Since $((s, s'), u)$ is B-alternating, we know that for any $n \in \mathbb{N}$, there exists $(s_1, \dots, s_n) \in \mathcal{C}_2[\alpha, B]$ such that the chain $(ss_1s', \dots, ss_ns')$ has alternation $n - 1$ and $ss_1s' = u$. We denote by $\mathcal{R} \subseteq \mathcal{C}_2[\alpha, B]$ the set of all these Σ_2-chains. Observe that by definition, $\mathcal{R}$ has unbounded alternation. It follows from Proposition 31 that one can construct a set of chain trees $\mathbb{U}'$ whose set of values is exactly $\mathcal{R}$. By definition, $\mathbb{U}'$ satisfies Items 1 and 2 and $s \cdot val(\mathbb{U}') \cdot s'$ has unbounded alternation.

We now use Proposition 36 to construct $\mathbb{U}$ from $\mathbb{U}'$ which is locally optimal for (s, s') and satisfies $s \cdot val(\mathbb{U}') \cdot s' = s \cdot val(\mathbb{U}) \cdot s'$. We now know that $\mathbb{U}$ satisfies properties 1 to 3. Observe that by definition $\mathbb{U}$ has unbounded alternation. By hypothesis of Proposition 45, it follows that $\mathbb{U}$ has also unbounded recursive alternation and all items are satisfied.

Choosing a chain tree $T \in \mathbb{U}$. We now select a special chain tree T in $\mathbb{U}$. We want T to have large enough recursive alternation in order to use it to construct the node $((t, t'), v)$. We define the needed recursive alternation in the following lemma.

Lemma 48. *There exists $K \in \mathbb{N}$ such that for all $t_1, t_2 \in M$ and all $C \subseteq A$, $(t_1, t_2)^K \in \mathcal{C}_2[\alpha, C] \Rightarrow (t_1, t_2)^* \subseteq \mathcal{C}_2[\alpha, C]$.*

Proof. It suffices to take K as the largest k such that there exists $t_1, t_2 \in M$ and $C \subseteq A$ with $(t_1, t_2)^{k-1} \in \mathcal{C}_2[\alpha, C]$ but $(t_1, t_2)^k \notin \mathcal{C}_2[\alpha, C]$. ◻

Set $m = |M|^2 \cdot K$ with K as defined in Lemma 48. By hypothesis on $\mathbb{U}$ (see property 4) there exists a tree $T \in \mathbb{U}$ with recursive alternation m. We set n as the level of T.

Construction of the node $((t, t'), v)$. Set r as the first element in $\mathsf{val}(T)$. Recall that by choice of T in $\mathbb{U}$, $srs' = u$. By definition of recursive alternation, T must contain an operation leaf x whose label $\mathsf{val}(x) = (t_1, \dots, t_n)$ has alternation m. Set $((s_1, \dots, s_n), (s'_1, \dots, s'_n)) = \mathsf{cval}(x)$ and $C = \mathsf{alph}(x)$. Note that since $\mathsf{alph}(T) = B$, $C \subseteq B$. Recall that by Fact 33, we have

$$s \cdot \mathsf{val}(T) \cdot s' = s \cdot (s_1, \dots, s_n) \cdot (t_1, \dots, t_n) \cdot (s'_1, \dots, s'_n) \cdot s'$$

Note that $(t_1, \dots, t_n) \in \mathcal{C}_2[\alpha, C]$, $(s_1, \dots, s_n) \in \mathcal{C}_2[\alpha]$ and $(s'_1, \dots, s'_n) \in \mathcal{C}_2[\alpha]$. We know that $(t_1, \dots, t_n)$ has alternation $m = |M|^2 \cdot K$. It follows from a pigeonhole principle argument that there exists $q_1 \neq q_2 \in M$ and a set $I \subseteq \{1, \dots, n-1\}$ of size at least K such that for all $i \in I$, $t_i = q_1$ and $t_{i+1} = q_2$. Observe that by definition, the chain $(q_1, q_2)^K$ is a subword of $(t_1, \dots, t_n)$ and therefore a Σ_2-chain for α, C. By choice of K it follows that $(q_1, q_2)^* \subseteq \mathcal{C}_2[\alpha, C]$. Note that this means that the node $((1_M, 1_M), q_1)$ is C-alternating. Therefore, by minimality of B, we have $C = B$. Choose some arbitrary $i \in I$, say the first element in I. Recall that $T \in \mathbb{U}$ and therefore locally optimal for (s, s'). The following fact is immediate by definition of local optimality:

Fact 49 $ss_{i+1}q_1s'_{i+1}s' \neq ss_{i+1}q_2s'_{i+1}s'$.

We now define the node $((t, t'), v)$. It is immediate from Fact 49 that either $ss_{i+1}q_1s'_{i+1}s' \neq u$ or $ss_{i+1}q_2s'_{i+1}s' \neq u$, we set $v \neq u$ as this element. Finally, we set $t = ss_{i+1}$ and $t' = s'_{i+1}s'$. Observe that by Fact 49 $tq_1t' \neq tq_2t'$, therefore since $(q_1, q_2)^* \subseteq \mathcal{C}_2[\alpha, B]$ and by choice of v, we know that $((t, t'), v)$ is B-alternating. It remains to prove that $((s, s'), u) \xrightarrow{B} ((t, t'), v)$. We already know that $u = srs'$, $t = ss_{i+1}$ and $t' = s'_{i+1}s'$. We need to prove that (r, s_{i+1}, s'_{i+1}) is a B-schema.

Using the definition of operation nodes, we prove that $r = s_1s'_1$ and define $\mathcal{T} \in \mathfrak{C}_{2,2}[\alpha, B]$ such that $(s_1, s_{i+1}) \in \mathcal{C}_{2,2}[\alpha, B] \cdot \mathcal{T}^\omega$ and $(s'_1, s'_{i+1}) \in \mathcal{T}^\omega \cdot \mathcal{C}_{2,2}[\alpha, B]$

which terminates the proof. Set y as the parent of x. By definition, y is an operation node, set $x_1, \dots, x_{2\ell_n+1}$ as the children of y ($x = x_{\ell_n+1}$). By definition,

$$\mathcal{R} = \{\mathsf{val}(x_1), \dots, \mathsf{val}(x_{\ell_n}), \mathsf{val}(x_{\ell_n+2}), \dots, \mathsf{val}(x_{2\ell_n+1})\} \in \mathfrak{C}_{2,n}[\alpha, B]$$

Set t has the common first value of all chains in $\mathcal{R}$ and $(\bar{q}, \bar{q}') = \mathsf{cval}(y)$. By Fact 34, we have

$$\bar{s} = \bar{q} \cdot \mathsf{val}(x_1) \cdots \mathsf{val}(x_{\ell_n}) \text{ and } \mathsf{val}(x_{\ell_n+2}) \cdots \mathsf{val}(x_{2\ell_n+1}) \cdot \bar{q}' = \bar{s}' \qquad (15)$$

By Fact 33, and definition of operation nodes, $r = s_1 t^{\ell_n} s'_1$. It follows that $r = s_1 s'_1$.

Since T has alphabet B, we have $\bar{q} \in \mathcal{C}_{2,n}[\alpha, C]$ for some $C \subseteq B$. Using (15) and the definition of ℓ_n as $\omega(2^{M^n})$, we get that $\bar{s} \in \mathcal{C}_{2,n}[\alpha, C] \cdot \mathcal{R}^\omega$. Moreover, since $Rs^\omega \subseteq \mathcal{C}_{2,n}[\alpha, B]$, $\bar{s} \in \mathcal{C}_{2,n}[\alpha, B]$. Using a symetrical argument, we get that $\bar{s}' \in \mathcal{R}^\omega \cdot \mathcal{C}_{2,n}[\alpha, B]$.

Finally, set $\mathcal{T}$ as the set of chains of length 2 obtained from chains in $\mathcal{R}$ by keeping only the values at component 1 and $i+1$. Since Σ_2-chains are closed under subwords, it is immediate from $\mathcal{R} \in \mathfrak{C}_{2,n}[\alpha, B]$ that $\mathcal{T} \in \mathfrak{C}_{2,2}[\alpha, B]$. Moreover, by definition, we have $(s_1, s_{i+1}) \in \mathcal{C}_{2,2}[\alpha, B] \cdot \mathcal{T}^\omega$ and $(s'_1, s'_{i+1}) \in \mathcal{T}^\omega \cdot \mathcal{C}_{2,2}[\alpha, B]$. We conclude that (r, s_{i+1}, s'_{i+1}) is a B-schema which terminates the proof. ◻

G Proof of Proposition 42

Recall that we fixed the morphism $\alpha : A^* \to M$. We prove Proposition 42.

Proposition 42. *Assume that there exists a set of locally optimal chain trees $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ with unbounded alternation but bounded recursive alternation. Then α does not satisfy Equation* (4)*.*

As for the previous section, we will use a new object that is specific to this case: *chain matrices.*

Chain Matrices. Let $n \in \mathbb{N}$. A *chain matrix of length n* is a rectangular matrix with n columns and such that rows belong to $\mathcal{C}_{2,n}[\alpha]$. If $\mathcal{M}$ is a chain matrix, we will denote by $\mathcal{M}_{i,j}$ the entry at row i (starting from the top) and column j (starting from the left) in $\mathcal{M}$. If $\mathcal{M}$ is a chain matrix of length n and with m rows, we call the chain $\big((\mathcal{M}_{1,1} \cdots \mathcal{M}_{m,1}), \dots, (\mathcal{M}_{1,n} \cdots \mathcal{M}_{m,n})\big)$, the *value* of $\mathcal{M}$. By Fact 2, the value of a chain matrix is a Σ_2-chain. We give an example with 3 rows in Figure 3.

Given a chain matrix, $\mathcal{M}$, the *alternation* of $\mathcal{M}$ is the alternation of its value. Finally, the *local alternation* of a chain matrix, $\mathcal{M}$, is the largest natural m such that $\mathcal{M}$ has a row with alternation m. We now prove the two following propositions.

Proposition 50. *Assume that there exists a set of locally optimal chain trees $\mathbb{S} \subseteq \mathbb{T}[\alpha]$ with unbounded alternation and recursive alternation bounded by $K \in \mathbb{N}$. Then there exist chain matrices with arbitrarily large alternation and local alternation bounded by K.*

s_1	s_2	s_3	s_4	$\cdots$	s_n
t_1	t_2	t_3	t_4	$\cdots$	t_n
r_1	r_2	r_3	r_4	$\cdots$	r_n

$$\text{Value} \Longrightarrow (s_1t_1r_1, s_2t_2r_2, s_3t_3r_3, s_4t_4r_4, \quad \dots \quad , s_nt_nr_n \;)$$

Fig. 3. Value of chain matrix with 3 rows

Proposition 51. *Assume that there exist chain matrices with arbitrarily large alternation and local alternation bounded by $K \in \mathbb{N}$. Then α does not satisfy* (5).

Proposition 42 is an immediate consequence of Proposition 50 and 51. Note that chain matrices are reused from [5] (where they are called "strategy matrices"). Moreover, in this case going from chain trees to chains matrices (i.e. proving Proposition 50) is simple and the main difficulty is proving Proposition 51. This means that while our presentation is slightly different from that of [5], the arguments themselves are essentially the same. We give a full proof for the sake of completeness. We begin by proving Proposition 50.

Proof (of Proposition 50). We prove that for all $n \in \mathbb{N}$, there exists a chain matrix $\mathcal{M}$ of alternation n and local alternation bounded by K. By definition of $\mathbb{S}$ there exists a tree $T \in \mathbb{S}$ whose value has alternation n and has recursive alternation bounded by K. Set $x_1, \dots, x_m$ as leaves of T listed from left to right. By Fact 30, $\mathsf{val}(T) = \mathsf{val}(x_1) \cdots \mathsf{val}(x_m)$. Observe that by definition, for all i, $\mathsf{val}(x_i)$ has alternation bounded by K. Therefore it suffices to set $\mathcal{M}$ as the m rows matrix where row i is filled with $\mathsf{val}(x_i)$. ☐

It now remains to prove Proposition 51. We proceed as follows. Assuming there exists a chain matrix $\mathcal{M}$ with local alternation bounded by K and very large alternation, we refine $\mathcal{M}$ in several steps to ultimately obtain what we call a *contradiction matrix*. There are two types of contradiction matrices, *increasing* and *decreasing*, both are chain matrices of length 6 and with the following entries:

u_1	v_1	f	f	f	f
e	e	u_2	v_2	f	f
e	e	e	e	u_3	v_3

Increasing Contradiction Matrix

f	f	f	f	u_3	v_3
f	f	u_2	v_2	e	e
u_1	v_1	e	e	e	e

Decreasing Contradiction Matrix

such that e, f are idempotents and $fu_2e \neq fv_2e$. As the name suggests, the existence of a contradiction matrix contradicts Equation (4). This is what we state in the following lemma.

Lemma 52. *If there exists a contradiction matrix, α does not satisfy* (4).

Proof. Assume that we have an increasing contradiction matrix (the other case is treated in a symmetrical way). Since $fu_2e \neq fv_2e$, either $fu_2e \neq fe$ or $fv_2e \neq fe$. By symmetry assume it is the former. Since e, f are idempotents, this means that $f^\omega u_2 e^\omega \neq f^\omega e^\omega$. However by definition of chain matrices $(e, u_2, v_2, f) \in \mathcal{C}_2[\alpha]$ and therefore $(e, u_2, f) \in \mathcal{C}_2[\alpha]$ which contradicts Equation (4). Note that we only used one half of Equation (4), the other half is used in the decreasing case. ☐

By Lemma 52, it suffices to prove the existence of a contradiction matrix to conclude the proof of Proposition 51. This is what we do in the remainder of this Appendix. By hypothesis, we know that there exist chain matrices with arbitrarily large alternation and local alternation bounded by $K \in \mathbb{N}$. For the remainder of the section, we assume that this hypothesis holds. We use several steps to prove that we can choose our chain matrices with increasingly strong properties until we get a contradiction matrix. We use two intermediaries that we call *Tame Chain Matrices* and *Monotonous Chain Matrices*. We divide the proof in three subsections, one for each step.

G.1 Tame Chain Matrices

Let $\mathcal{M}$ be a chain matrix of *even length* 2ℓ and let $j \leqslant \ell$. The *set of alternating rows for* j, denoted by $\mathsf{alt}(\mathcal{M}, j)$, is the set $\{i \mid \mathcal{M}_{i,2j-1} \neq \mathcal{M}_{i,2j}\}$. Let $(s_1, \dots, s_{2\ell})$ be the value of $\mathcal{M}$. We say that $\mathcal{M}$ is *tame* if

a) for all $j \leqslant \ell$, $s_{2j-1} \neq s_{2j}$,
b) for all $j \leqslant \ell$, $\mathsf{alt}(\mathcal{M}, j)$ is a singleton and
c) if $j \neq j'$ then $\mathsf{alt}(\mathcal{M}, j) \neq \mathsf{alt}(\mathcal{M}, j')$.

We represent a tame chain matrix of length 6 in Figure 4. Observe that the definition only considers the relationship between odd columns and the next even column. Moreover, observe that a tame chain matrix of length 2ℓ has by definition alternation at least ℓ.

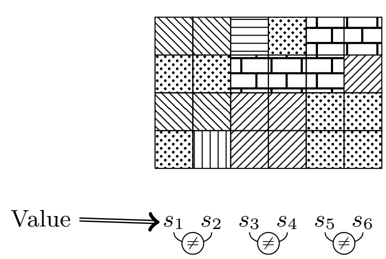

Fig. 4. A tame chain matrix of length 6

Lemma 53. *There exists tame chain matrices of arbitrarily large length.*

Proof. Set $n \in \mathbb{N}$, we explain how to construct a tame chain matrix of length $2n$. By hypothesis, there exists a chain matrix $\mathcal{M}$ with local alternation at most K and alternation greater than $2nK$. Set m as the number of rows of $\mathcal{M}$. We explain how to modify $\mathcal{M}$ to obtain a matrix satisfying a), b) and c). Recall that Σ_2-chains are closed under subwords, therefore removing columns from $\mathcal{M}$ yields a chain matrix. Since $\mathcal{M}$ has alternation $2nK$, it is simple to see that by removing columns one can obtain a chain matrix of length $2nK$ that satisfies a). We denote by $\mathcal{N}$ this matrix. We now proceed in two steps: first, we modify the entries in $\mathcal{N}$ to get a matrix $\mathcal{P}$ of length $2nK$ satisfying both a) and b). Then we use our bound on local alternation to remove columns and enforce c) in the resulting matrix.

Construction of $\mathcal{P}$. Let $j \leqslant nK$ such that $\mathsf{alt}(\mathcal{N}, j)$ is of size at least 2. We modify the matrix to reduce the size of $\mathsf{alt}(\mathcal{N}, j)$ while preserving a). One can then repeat the operation to get the desired matrix. Let $i \in \mathsf{alt}(\mathcal{N}, j)$. Set $s_1 = \mathcal{N}_{1,2j-1} \cdots \mathcal{N}_{i-1,2j-1}$ and $s_2 = \mathcal{N}_{i+1,2j-1} \cdots \mathcal{N}_{m,2j-1}$. We distinguish two cases.

First, if $s_1\mathcal{N}_{i,2j-1}s_2 \neq s_1\mathcal{N}_{i,2j}s_2$, then for all $i' \neq i$, we replace entry $\mathcal{N}_{i',2j}$ with entry $\mathcal{N}_{i',2j-1}$. One can verify that this yields a chain matrix of length $2nK$, local alternation bounded by K. Moreover, it still satisfies a), since $s_1\mathcal{N}_{i,2j-1}s_2 \neq s_1\mathcal{N}_{i,2j}s_2$. Finally, $\mathsf{alt}(\mathcal{N}, j)$ is now a singleton, namely $\{i\}$.

In the second case, we have $s_1\mathcal{N}_{i,2j-1}s_2 = s_1\mathcal{N}_{i,2j}s_2$. In that case, we replace $\mathcal{N}_{i,2j-1}$ with $\mathcal{N}_{i,2j}$. One can verify that this yields a chain matrix of length $2nK$, local alternation bounded by K. Moreover, it still satisfies a) since we did not change the value on the whole. Finally, the size of $\mathsf{alt}(\mathcal{N}, j)$ has decreased by 1.

Construction of the tame matrix. We now have a chain matrix $\mathcal{P}$ of length $2nK$, with local alternation bounded by K and satisfying both a) and b). Since a) and b) are satisfied, for all $j \leqslant nK$ there exists exactly one row i such that $\mathcal{N}_{i,2j-1} \neq \mathcal{N}_{i,2j}$. Moreover, since each row has alternation at most K, a single row i has this property for at most K indices j. Therefore, it suffices to remove at most $n(K-1)$ pairs of odd-even columns to get a matrix that satisfies c). Since the original matrix had length $2nK$, this leaves a matrix of length at least $2n$ and we are finished. ◻

G.2 Monotonous Chain Matrices

Let $\mathcal{M}$ be a tame chain matrix of length $2n$ and let $x_1, \dots, x_n$ be naturals such that for all j, $\mathsf{alt}(\mathcal{M}, j) = \{x_j\}$. We say that $\mathcal{M}$ is a *monotonous chain matrix* if it has exactly n rows and $1 = x_1 < x_2 < \cdots < x_n = n$ (in which case the matrix is said *increasing*) or $n = x_1 > x_2 > \cdots > x_n = 1$ (in which case we say the matrix is *decreasing*). We give a representation of the increasing case in Figure 5.

Lemma 54. *There exists monotonous chain matrices of arbitrarily large length.*

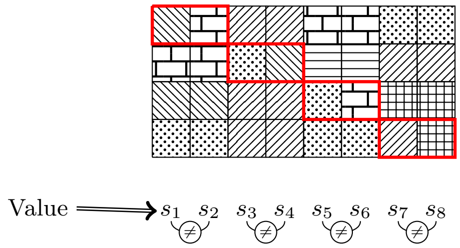

Fig. 5. A monotonous chain matrix (increasing)

Proof. Set $n \in \mathbb{N}$, we explain how to construct a tame chain matrix of length $2n$. By Lemma 53, there exists a tame chain matrix $\mathcal{M}$ of length $2n^2$. Set $x_1, \ldots, x_{n^2}$ the indices such that for all j, $\mathsf{alt}(\mathcal{M}, j) = \{x_j\}$. Note that by tameness, $x_j \neq x_{j'}$ for $j \neq j'$. Since the sequence $x_1, \ldots, x_{n^2}$ is of length n^2, we can extract, using Erdös-Szekeres theorem, a monotonous sequence of length n, $x_{j_1} < \cdots < x_{j_n}$ or $x_{j_1} > \cdots > x_{j_n}$ with $j_1 < \cdots < j_n$. By symmetry we assume it is the former and construct an increasing chain matrix of length n.

Let $\mathcal{P}$ be the matrix of length $2n$ obtained from $\mathcal{M}$, by keeping only the pairs of columns $2j-1, 2j$ for $j \in \{j_1, \ldots, j_n\}$. Set $x'_1, \ldots, x'_n$ the indices such that for all j, $\mathsf{alt}(\mathcal{P}, j) = \{x'_j\}$. By definition, $x'_1 < \cdots < x'_n$. We now want $\mathcal{P}$ to have exactly n rows. Note that the rows that do not belong to $x'_1 < \cdots < x'_n$ are constant chains. We simply merge these rows with others. For example, if row i is labeled with the constant chain $(s, \ldots, s)$, let $(s_1, \ldots, s_{2n})$ be the label of row $i+1$. We remove row i and replace row $i+1$ by the Σ_2-chain $(ss_1, \ldots, ss_{2n})$. Repeating the operation yields the desired increasing monotonous chain matrix. □

G.3 Construction of the Contradiction Matrix

We can now use Lemma 54 to construct a contradiction matrix and end the proof of Proposition 42. We state this in the following proposition.

Proposition 55. *There exists a contradiction matrix.*

The remainder of this appendix is devoted to the proof of Proposition 55. The result follows from a Ramsey argument. We use Lemma 54 to choose a monotonous matrix of sufficiently large length. Then, we use Ramsey's Theorem (for hypergraphs with edges of size 3) to extract the desired contradiction matrix.

We first define the length of the monotonous chain matrix that we need to pick. By Ramsey's Theorem, for every $m \in \mathbb{N}$ there exists a number $\varphi(m)$ such that for any complete 3-hypergraph with hyperedges colored over the monoid M, there exists a complete sub-hypergraph of size m in which all edges share the same color. We choose $n = \varphi(\varphi(4)+1)$. By Lemma 54, there exists a monotonous chain matrix $\mathcal{M}$ of length $2n$. Since it is monotonous, $\mathcal{M}$ has n rows.

By symmetry, we assume that $\mathcal{M}$ is increasing and use it to construct an increasing contradiction matrix. We use our choice of n to extract a contradiction matrix from $\mathcal{M}$. We proceed in two steps using Ramsey's Theorem each time. In the first step we treat all entries above the diagonal in $\mathcal{M}$ and in the second step all entries below the diagonal. We state the first step in the next lemma.

Lemma 56. *There exists an increasing monotonous matrix $\mathcal{N}$ of length $2 \cdot \varphi(4)$ such that all cells above the diagonal contain the same idempotent $f \in M$.*

Proof. This is proved by applying Ramsey's Theorem to $\mathcal{M}$. Consider the complete 3-hypergraph whose nodes are $\{0, \ldots, n\}$. We label the hyperedge $\{i_1, i_2, i_3\}$ where $i_1 < i_2 < i_3$ by the value obtained by multiplying in the monoid M, the cells that appear in rows $i_1 + 1, \ldots, i_2$ in column $2i_3 - 1$. Observe that since $i_1 < i_2 < i_3$, by monotonicity, these entries are the same as in column $2i_3$. More formally, the label of the hyperedge $\{i_1, i_2, i_3\}$ is therefore

$$\mathcal{M}_{i_1+1,2i_3-1} \cdots \mathcal{M}_{i_2,2i_3-1} = \mathcal{M}_{i_1+1,2i_3} \cdots \mathcal{M}_{i_2,2i_3}.$$

By choice of n, we can apply Ramsey's Theorem to this coloring. We get a subset of $\varphi(4) + 1$ vertices, say $K = \{k_1, \ldots, k_{\varphi(4)+1}\} \subseteq \{0, \ldots, n\}$, such that all hyperedges connecting nodes in K have the same color, say $f \in M$. For $i_1 < i_2 < i_3 < i_4$ in K, note that the color of the hyperedge $\{i_1, i_3, i_4\}$ is by definition the product of the colors of the hyperedges $\{i_1, i_2, i_4\}$ and $\{i_2, i_3, i_4\}$. Therefore, the common color f needs to be an idempotent (i.e. $f = ff$). We now extract the desired matrix $\mathcal{N}$ from $\mathcal{M}$ according to the subset K. The main idea is that the new row i in $\mathcal{N}$ will be the merging of rows $k_i + 1$ to k_{i+1} in $\mathcal{M}$ and the new pair of columns $2j - 1, 2j$ will correspond to the pair $2k_{j+1} - 1, 2k_{j+1}$ in $\mathcal{M}$.

We first merge rows. For all $i \geqslant 1$, we "merge" all rows from $k_i + 1$ to k_{i+1} into a single row. More precisely, this means that we replace the rows $k_i + 1$ to k_{i+1} by a single row containing the Σ_2-chain

$$(\mathcal{M}_{k_i+1,1} \cdots \mathcal{M}_{k_{i+1},1}, \ldots, \mathcal{M}_{k_i+1,2n} \cdots \mathcal{M}_{k_{i+1},2n})$$

Moreover, we remove the top and bottom rows, i.e. row 1 to k_1 and rows $k_{\varphi(4)+1}$ to $\varphi(4) + 1$. Then we remove all columns from 1 to $2k_2 - 2$, all columns from $2k_{\varphi(4)+1} + 1$ to $2n$, and for all $i \geqslant 2$, all columns from $2k_i + 1$ to $2k_{i+1} - 2$. One can verify that these two operations applied together preserve monotonicity. Observe that the resulting matrix $\mathcal{N}$ has exactly $2 \cdot \varphi(4)$ columns. Moreover, the cell $i, 2j$ in the new matrix contains entry $\mathcal{M}_{k_i+1,2k_{j+1}} \cdots \mathcal{M}_{k_{i+1},2k_{j+1}}$. In particular if $j > i$, by definition of the set K, this entry is f, which means $\mathcal{N}$ satisfies the conditions of the lemma. ◻

It remains to apply Ramsey's Theorem a second time to the matrix $\mathcal{N}$ obtained from Lemma 56 to treat the cells below the diagonal and get the contradiction matrix. We state this in the following last lemma.

Lemma 57. *There exists an increasing monotonous matrix $\mathcal{P}$ of length 6 such that all cells above the diagonal contain the same idempotent $f \in M$ and all cells below the diagonal contain the same idempotent $e \in M$ (i.e. $\mathcal{P}$ is an increasing contradiction matrix).*

Proof. The argument is identical to the one of Lemma 56. This time we apply it to the matrix $\mathcal{N}$ of length $2 \cdot \varphi(4)$ for the cells below the diagonal. $\square$