Separating Regular Languages with Two Quantifier Alternations

Thomas Place
LaBRI, Bordeaux University,
Bordeaux, France
Email: tplace@labri.fr

***Abstract*—We investigate the quantifier alternation hierarchy of first-order logic over finite words. To do so, we rely on the separation problem. For each level in the hierarchy, this problem takes two regular languages as input and asks whether there exists a formula of the level that accepts all words in the first language and no word in the second one. Usually, obtaining an algorithm that solves this problem requires a deep understanding of the level under investigation.**

We present such an algorithm for the level Σ_3 (formulas having at most 2 alternations beginning with an existential block). We also obtain as a corollary that one can decide whether a regular language is definable by a Σ_4 formula (formulas having at most 3 alternations beginning with an existential block).

***Keywords*-First-order logic; Quantifier alternation; Regular languages; Words; Expressive power; Separation; Decidable characterizations;**

I. Introduction

Context. This paper is part of a research effort whose aim is to understand the expressive power of logics over words. In this context, we say that a logic has a *decidable membership problem* (or decidable characterization) when the following problem is decidable: given as input a regular language, decide if it can be defined with a formula of the logic in question. Having a membership algorithm in hand amounts to having an effective description of *all* regular properties that the logic can express. For this reason, obtaining a membership algorithm is often considered as the goal to strive for when trying to get a precise understanding of the expressive power of a logic.

The quest for membership algorithms has been very fruitful. A well-known example of this success is the algorithm obtained for first-order logic (FO). It is based on Schützenberger's Theorem [Sch65], [MP71], which states that a regular language is definable in FO if and only if its *syntactic monoid is aperiodic*. Since the syntactic monoid is a finite computable object and aperiodicity a decidable property, the membership algorithm is immediate from the theorem. However, it is actually from the proof of the theorem than one learns the most about FO. Indeed, this proof, which has often been revisited even as late as [DG08], requires a canonical method for constructing a FO formula for any language having an aperiodic syntactic monoid.

Since this first success a lot of efforts have been made to obtain similar results for natural fragments of FO. Among these, the quantifier alternation hierarchy is prominent. In this hierarchy, FO formulas are classified by counting quantifier alternations in their prenex normal form. A formula is Σ_i if it has at most i blocks of quantifiers and starts with an existential one. Furthermore, since the negation of a Σ_i formula is not Σ_i in general, the hierarchy also considers $\mathcal{B}\Sigma_i$ formulas: boolean combinations of Σ_i formulas. The hierarchy is known to be strict [BK78], [Tho82]: $\Sigma_i \subsetneq \mathcal{B}\Sigma_i \subsetneq \Sigma_{i+1}$. Hence, a natural question is whether one can find a membership algorithm for each level. However, despite having been given a lot of attention, progress on this question has been slow and solutions are only known for the lower levels (see [Pin11], [AK10], [Pin98] for detailed bibliographies).

State of the art. Historically, $\mathcal{B}\Sigma_1$ is the first level to have been given a membership algorithm, a result known as Simon's Theorem [Sim75]. More than a decade later, an algorithm was also found for Σ_2 [Arf87], [PW97]. Following this, it took more than a decade again to obtain membership algorithms for $\mathcal{B}\Sigma_2$ and Σ_3 [PZ14a]. An explanation for this slow progress is each new level has required new conceptual ideas. This is illustrated by the techniques used in [PZ14a]. In order to make progress, the authors had to consider a question that is deeper than membership: the separation problem.

For a given logic, a separation algorithm takes as input *two* regular languages and decides whether there exists a formula of the logic that accepts all words of the first language and no words of the second one. Membership can be reduced to separation: deciding whether a regular language is separable from its complement (also regular) amounts to deciding membership for this language. Hence, separation is often a more difficult question than membership. However, it is also a more rewarding one [CMM13], [PRZ13], [PZ14b], [PZ15]. In particular, membership algorithms for $\mathcal{B}\Sigma_2$ and Σ_3 were obtained in [PZ14a] by relying on two results:

1) A separation algorithm for Σ_2.
2) Results that link membership for $\mathcal{B}\Sigma_i$ and Σ_{i+1} to separation for Σ_i for any i.

The most simple link is with Σ_{i+1}: one can effectively reduce membership for Σ_{i+1} to separation for Σ_i. In view of this result, an immediate question is as follows: "can we define a problem $\mathcal{P}$, such that separation for Σ_{i+1} can be effectively reduced to $\mathcal{P}$ for Σ_i ?" A natural starting point to the investigation of this question is separation for Σ_3.

Contribution. In this paper, we present a separation algorithm for Σ_3 (and therefore a membership algorithm for Σ_4 by the

reduction above [PZ14a]). While we do not provide a definitive answer to the question above, our result seems to indicate that this answer should be "*no*". Indeed, our technique does not work by reduction to an independent problem for Σ_2. Instead, we consider a problem that generalizes simultaneously and links the separation problems for both Σ_2 and Σ_3. We rely on the computation of a new object: a (finite) set of Σ-trees that can be associated to any finite monoid M recognizing regular languages. This set captures information with respect to Σ_2, Σ_3 and M. This includes which pairs of languages recognized by M are separable using Σ_2 or Σ_3. However, a large part of this information goes beyond separation and is only needed for the computation of Σ-trees. We use a least fixpoint algorithm: we saturate a set of trivial Σ-trees with correct operations until a fixpoint is reached.

Our techniques can be seen as a generalization of the techniques of [PZ14a]. In particular, Σ-trees generalize the "Σ_2-chains" used in the separation algorithm of Σ_2. However, Σ-trees are much more involved as they capture and link information about two distinct formalisms (i.e. Σ_2 and Σ_3) in a single object. In fact, it seems that for each new level of the hierarchy getting a separation algorithm requires to consider objects that are both conceptually and technically more involved than for the previous one.

Note that while Σ-trees are based on the monoid definition of regular languages, we do not rely on involved algebraic machinery (Σ-trees could be adapted to work with automata). In particular, our proof is entirely combinatorial. Our main motivation for working with monoids rather than automata is that it slightly simplifies the presentation and enables us to rely on powerful combinatorial theorems such as Simon's factorization forest Theorem [Sim90].

Organization of the paper. We begin with preliminary definitions in Section II. In Section III, we define the mathematical tools we will need. Section IV presents Σ-trees which are the central object of our separation algorithm. The algorithm is presented in Section V. We prove its correctness and completeness in Sections VI and VII respectively. Due to space restrictions, some proofs are omitted and available in the full version of the paper.

II. Preliminaries

In this section, we present the objects that we investigate in the paper. More precisely, we define the quantifier alternation hierarchy within first-order logic and state the membership and separation problems.

A. First-Order Logic and Quantifier Alternation

For the whole paper, we fix a finite alphabet A. We denote by A^* the set of all words over A (including the empty word denoted by ε). If $u, v \in A^*$ are words over A, we denote by $u \cdot v$ or uv the word obtained by concatenation of u and v and by $\mathsf{alph}(u)$ the alphabet of u, *i.e.*, the smallest subset B of A such that $u \in B^*$. Finally, a *factor* of $v \in A^*$ is a word $u \in A^*$ such that $v = u'uu''$ for some $u', u'' \in A^*$.

A *language* is a subset of A^*. In this paper we consider regular languages. These are the languages that can be equivalently defined by monadic second-order logic, finite automata or finite monoids. In the paper we work with the monoid definition. We recall this definition in Section III.

First-Order Logic. We view a word as a logical structure made of a sequence of positions. Each position has a label in the alphabet A and can be quantified. We denote by '$<$' the linear order over the positions. We consider first-order logic, FO, using the following predicates:

- for each $a \in A$, a unary predicate P_a that selects positions labeled with an a.
- a binary predicate '$<$' for the linear order.

To every first-order formula φ, one can associate the language $\{w \in A^* \mid w \models \varphi\}$ of words that satisfy φ. Hence, FO defines a class of languages: the class of all languages that can be defined using a FO formula.

One can classify FO formulas by counting their number of quantifier alternations. Set $i \in \mathbb{N}$, a formula is said to be Σ_i (resp. Π_i) if its prenex normal form has either

- *exactly* $i-1$ quantifier alternations (*i.e.*, exactly i blocks of quantifiers) starting with an $\exists$ (resp. $\forall$), or
- *strictly less* than $i-1$ quantifier alternations (*i.e.*, strictly less than i blocks of quantifiers)

For example, a formula whose prenex normal form is

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

is Π_3. In general, the negation of a Σ_i formula is not a Σ_i formula (it is a Π_i formula). Hence it is relevant to define $\mathcal{B}\Sigma_i$ formulas as FO formulas that are boolean combinations of Σ_i and Π_i formulas. This gives a strict hierarchy of classes of languages as presented in Figure 1.

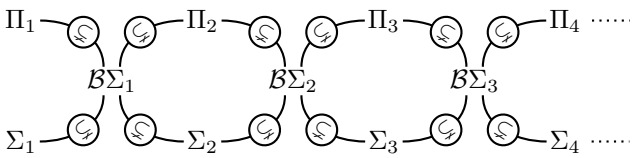


Fig. 1. Quantifier Alternation Hierarchy

B. Decision Problems

We investigate two decisions problems. Each problem is parametrized by a class of languages $\mathcal{C}$. Given such a class $\mathcal{C}$, the *membership problem* for $\mathcal{C}$ is as follows:

INPUT: A regular language L.
OUTPUT: Does $L \in \mathcal{C}$?

The *separation problem* is more general. 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 $\mathcal{C}$ as a class of languages, we say that L_0 is $\mathcal{C}$*-separable* from L_1 if some language in $\mathcal{C}$ separates L_0 from L_1. Observe that when $\mathcal{C}$ is not closed under complement (for example when $\mathcal{C} = \Sigma_i$ or $\mathcal{C} = \Pi_i$), the

definition is not symmetrical: L_0 could be $\mathcal{C}$-separable from L_1 while L_1 is not $\mathcal{C}$-separable from L_0. The separation problem for $\mathcal{C}$ is as follows:

INPUT: Two regular languages L_0 and L_1.
OUTPUT: Is L_0 $\mathcal{C}$-separable from L_1 ?

The separation problem refines the membership problem. Indeed, observe that asking whether a language L is $\mathcal{C}$-separable from its complement is equivalent to asking whether $L \in \mathcal{C}$ (the only potential separator is L itself). Hence, since regular languages are closed under complement, membership can easily be reduced to separation.

Both problems have been extensively studied in the literature for first-order logic and many of its natural fragments. The membership problem was proved to be decidable for FO itself [Sch65], [MP71] and up to the Σ_3,Π_3 level in the hierarchy [Sim75], [Kna83], [Arf87], [PW97], [GS00], [PZ14a]. The separation problem is known to be decidable for FO [PZ14b] and up to the Σ_2,Π_2 level in the hierarchy [CMM13], [PRZ13], [PZ14a], [PZ15]. In this paper, we prove the following theorem.

Theorem 1. *The separation problem is decidable for Σ_3 and Π_3.*

Moreover, it is known [PZ14a] that for any $i \geqslant 1$, the membership problem for Σ_{i+1} can be effectively reduced to the separation problem for Σ_i. Hence, we also obtain the following corollary.

Corollary 2. *The membership problem is decidable for Σ_4 and Π_4.*

The remainder of the paper is devoted to the proof of Theorem 1. In Sections III and IV, we introduce objects that we will need in our separation algorithm. More precisely, in Section III, we recall some well-known combinatorial tools such as monoids and the logical preorders associated to the hierarchy. In Section IV, we introduce a new object that is specific to this paper: Σ-*trees*. Σ-Trees are central to our separation algorithm, which we present in Section V.

III. Tools

In this section we present well-known combinatorial tools that we will need in order to present our results. Namely, we introduce the logical preorders that can be associated to any level in the hierarchy as well as the monoid definition of regular languages. Additionally, we recall a well-known combinatorial result on monoids that will be needed in our proofs: *Simon's Factorization Forests Theorem*.

A. Σ_i preorders

It is usual to associate a preorder over A^* to each level Σ_i in the hierarchy. The definition is based on the notion of quantifier rank. The *quantifier rank* of a first-order formula is the length of the longest sequence of nested quantifiers inside the formula. For example, the following formula,

$$\forall x\ P_a(x) \Rightarrow ((\exists y\ (y < x \wedge P_c(y)) \wedge (\exists y \exists z\ x < y < z \wedge P_b(y)))$$

has quantifier rank 3. In particular, it is well-known that for a fixed k, there is a finite number of non-equivalent first-order formulas of rank less than k.

Set $k, i \geqslant 1$ and $w, w' \in A^*$. We write $w \lesssim_i^k w'$ if *any* Σ_i formula of rank at most k satisfied by w is satisfied by w' as well. One can verify that $\lesssim_i^k$ is a preorder. Moreover, the following facts can be verified from the definitions.

Fact 3. *For all $k, i \geqslant 1$ and all $u, v \in A^*$, we have,*

$$\begin{array}{rcl} u \lesssim_i^{k+1} v & \Rightarrow & u \lesssim_i^k v \\ u \lesssim_{i+1}^k v & \Rightarrow & u \lesssim_i^k v \text{ and } v \lesssim_i^k u \end{array}$$

Proof: The only non-trivial implication is $u \lesssim_{i+1}^k v \Rightarrow v \lesssim_i^k u$. Assume that $u \lesssim_{i+1}^k v$, we prove that $v \lesssim_i^k u$. We need to prove that any Σ_i formula of rank less than k that is satisfied by v is satisfied by u as well. We prove the contrapositive: for any Σ_i formula φ of rank less than k, if u satisfies $\neg\varphi$, then v satisfies $\neg\varphi$. This is immediate from the hypothesis ($u \lesssim_{i+1}^k v$) since the negation of a Σ_i formula is in particular a Σ_{i+1} formula. ∎

Fact 4. *For all $k, i \in \mathbb{N}$, a language $L \subseteq A^*$ can be defined by a Σ_i formula of rank k if and only if L is saturated by $\lesssim_i^k$,* i.e.*, $L = \{w \mid \exists w' \in L \text{ s.t. } w' \lesssim_i^k w\}$.*

We finish with classical properties of the preorders $\lesssim_i^k$. The proofs are based on Ehrenfeucht-Fraïssé arguments. We begin with decomposition and composition lemmas.

Lemma 5 (Decomposition)**.** *Set $k, i \geqslant 1$ and $w_1, w_2, v \in A^*$ such that $w_1 w_2 \lesssim_i^k v$. Then v can be decomposed as $v = v_1 v_2$ with $w_1 \lesssim_i^{k-1} v_1$ and $w_2 \lesssim_i^{k-1} v_2$.*

Lemma 6 (Composition)**.** *Set $k, i \geqslant 1$ and $w_1, w_2, v_1, v_2 \in A^*$ such that $w_1 \lesssim_i^k v_1$ and $w_2 \lesssim_i^k v_2$. Then, we have $w_1 w_2 \lesssim_i^k v_1 v_2$.*

These two first properties are actually inherited from full first-order logic and they do not depend on the level i to which they are applied. This is not the case of the next property which is specific to the quantifier alternation hierarchy and links the preorders $\lesssim_i^k$ and $\lesssim_{i+1}^k$ for all i.

Lemma 7. *Set $k, i \geqslant 1$ and $u, v, u_\ell, u_r \in A^*$ such that $v \lesssim_i^k u$, $u \lesssim_{i+1}^k u_\ell$ and $u \lesssim_{i+1}^k u_r$. Then, for $k' \geqslant 2^k$, we have,*

$$u^{2k'} \lesssim_{i+1}^k (u_\ell)^{k'} v (u_r)^{k'}$$

The property stated in Lemma 7 is generic to all levels in the hierarchy. However, it involves two levels in the same statement. As we will see when defining Σ-trees in Section IV, this makes this property difficult to manipulate. It turns out that when $i = 1$ (and therefore $i + 1 = 2$), it can be replaced by the following lemma which involves only $\lesssim_2^k$ (and whose statement is a consequence of the special case of Lemma 7 when $i = 1$).

Lemma 8. *Set $k \geqslant 1$ and u, v, u_ℓ, u_r such that $\mathsf{alph}(v) = \mathsf{alph}(u)$, $u \lesssim_2^k u_\ell$ and $u \lesssim_2^k u_r$. Then, for $k' \geqslant 2^{2k}$, we have,*

$$u^{2k'} \lesssim_2^k (u_\ell)^{k'} v (u_r)^{k'}$$

B. Monoid Definition of Regular Languages

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. Observe that A^* is a monoid with concatenation as the multiplication and ε as the neutral element.

Given a 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 $s \in S$, 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*. Such a finite monoid can be computed from any automaton or monadic second-order formula that defines the language.

Morphisms and Separation. Separation takes two regular languages L_0, L_1 as input. It will be convenient to have a *single* monoid recognizing both of them (with different accepting sets), rather than having to deal with two objects. This can always be assumed without loss of generality as such a monoid can easily be constructed. If L_0, L_1 are recognized by two morphisms $\alpha_0 : A^* \to M_0$ and $\alpha_1 : A^* \to M_1$ into finite monoids M_0, M_1. Then, $M_0 \times M_1$ equipped with the componentwise multiplication $(s_0, s_1) \cdot (t_0, t_1) = (s_0 t_0, s_1 t_1)$ 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.

Alphabet compatibility. In the paper, we work with morphisms satisfying an additional property. A morphism $\alpha : A^* \to M$ is said to be *alphabet compatible*, if for all $u, v \in A^*$, $\alpha(u) = \alpha(v) \Rightarrow \mathsf{alph}(u) = \mathsf{alph}(v)$. Note that when α is alphabet compatible for all $s \in M$ having a preimage, $\mathsf{alph}(s)$ is well defined as the unique $B \subseteq A$ such that for all $u \in \alpha^{-1}(s)$, $\mathsf{alph}(u) = B$. The following simple lemma states that we will be able to assume without loss of generality that all our morphisms are alphabet compatible.

Lemma 9. *From any morphism $\alpha : A^* \to M$ into a finite monoid M, one can compute an alphabet compatible morphism β that recognizes all languages recognized by α.*

Proof: Since 2^A is a monoid together with union as the multiplication, it suffices to take β as the morphism:

$$\begin{array}{rccc} \beta : & A^* & \to & M \times 2^A \\ & w & \mapsto & (\alpha(w), \mathsf{alph}(w)) \end{array}$$

■

C. Simon's Factorization Forests Theorem

In order to prove our separation algorithm, we will need a famous combinatorial result on monoids: Simon's Factorization Forests Theorem. We state this theorem here. For more details on factorization forests and a proof of the theorem, we refer the reader to [Boj09], [Kuf08].

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

- *leaves* which are labeled by either a single letter or the empty word.
- *binary inner nodes* which have exactly two children.
- *idempotent inner nodes* which have an arbitrary number of children whose labels $w_1, \ldots, w_n$ satisfy $\alpha(w_1) = \cdots = \alpha(w_n) = e$ 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. The *height* of a factorization forest is the largest $h \in \mathbb{N}$ such that it contains a branch with h inner nodes.

Theorem 10 ([Sim90], [Kuf08])**.** *For all words $w \in A^*$, there exists an α-factorization forest for w of height smaller than $3|M| - 1$.*

In our proof, we will need an additional result on factorization forests that we state below. We define the *idempotent height* of an α-factorization forest as the largest number p such that there exists a branch that contains p idempotent nodes in the forest.

Lemma 11. *Let $w \in A^*$ that admits an α-factorization forest of height h and idempotent height p. Then any factor $u \in A^*$ of w admits an α-factorization forest of height at most $h + 2$ and idempotent height at most p.*

The proof of Lemma 11 is a simple induction on the structure of the factorization forest of w. This forest can be "repaired" into a factorization forest for u by adding additional binary nodes.

IV. Σ-Trees

In this section, we define Σ-trees which are key to our separation algorithm. The notion is based on the following lemma (which is an immediate consequence of Fact 4).

Lemma 12. *Let $i \geqslant 1$ and L_0, L_1 be any two languages. Then L_0 is* **not** *Σ_i-separable from L_1 if and only if for all $k \geqslant 1$ there exists $w_0 \in L_0$ and $w_1 \in L_1$ such that $w_0 \lesssim_i^k w_1$.*

Lemma 12 gives a simple criterion equivalent to Σ_i-separability. However, the quantification is over k which ranges over all naturals and it is not immediate that this criterion can be decided. Indeed, since A^* is infinite, $\lesssim_i^k$ is endlessly refined as k gets larger. The main idea behind our separation algorithm is to exploit the fact that our two inputs are regular languages and proceed as follows:

1) Take some morphism $\alpha : A^* \to M$ into a finite monoid M that recognizes both L_0 and L_1.
2) Abstract the preorder $\lesssim_i^k$ over M with respect to α. This amounts to considering the pairs $(s, t) \in M^2$, such that for all k, one can find witnesses $u, v \in A^*$ such that

$u \lesssim_i^k v$, $\alpha(u) = s$ and $\alpha(v) = t$. We say that (s,t) is a *pair for* Σ_i. The criterion of Lemma 12 can easily be adapted to rely only on the set of such pairs.

3) Compute all pairs $(s,t) \in M^2$ for Σ_i.

Essentially, this is what we do. The difficulty in this approach is finding an algorithm for step 3 above. What we want is a least fixpoint algorithm: the algorithm should start from the set of pairs (s,s) which are trivially pairs for Σ_i (for any i) and saturate this set with operations inspired from Lemmas 6 and 7 until a fixpoint is reached.

However, this is made difficult by the fact that using Lemma 7 requires information that is not captured by the set of pairs (s,t) for Σ_i: information that links Σ_i and Σ_{i+1}. Indeed, using Lemma 7 requires a single word u that simultaneously satisfies the three following properties: $v \lesssim_i^k u$, $u \lesssim_{i+1}^k u_\ell$ and $u \lesssim_{i+1}^k u_r$, in order to conclude that $u^{2k'} \lesssim_{i+1}^k (u_\ell)^{k'} v (u_r)^{k'}$ for a large enough k'. This cannot be adapted to pairs for Σ_i and Σ_{i+1}. Indeed, it may happen that (t,s) is a pair for Σ_i and $(s,s_\ell), (s,s_r)$ are pairs for Σ_{i+1} while the witnesses chosen for s have to be different for all three pairs. This means that a naive adaptation of Lemma 7 as an operation involving Σ_i and Σ_{i+1} pairs only would yield an algorithm that is not correct: pairs that are not pairs for Σ_{i+1} might be computed.

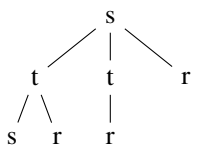
For this reason, we will need to consider objects that are more general than the Σ_i pairs. In particular, in order to use Lemma 7 in the Σ_3 case, we have to consider an object that generalizes both the Σ_3 pairs and the Σ_2 pairs and connects the two objects (note that we avoid having to generalize the Σ_1 pairs as well by relying on Lemma 8). These more general objects are what we call Σ-*trees*.

Note that Σ-trees are tailored to Σ_3-separation. It seems that for higher levels in the hierarchy, objects that are even more general have to be considered.

We separate the presentation in three parts. In the first part, we define a notion of M-*trees* for any finite monoid M. We then define Σ-trees as a special set of M-trees that can be associated to a morphism $\alpha : A^* \to M$. Finally, in the third part, we link Σ-trees to the separation problem.

A. M-Trees for a Monoid M

Fix a finite monoid M. A M-*tree of height* 0 is simply an element of M. For all $h \geqslant 1$, a M-*tree of height* h is a pair $(s, \mathfrak{S})$ where $s \in M$ and $\mathfrak{S}$ is a set of M-trees of height $h-1$. As the name suggests, M-trees can be represented as trees (see Figure 2).



Representation of $(s, \{(t, \{s, r\}), (t, \{r\}), (r, \emptyset)\})$

Fig. 2. Representation of a M-Tree of height 2

Remark 13. *While we define M-trees for all heights, for Σ_3-separation, we only use M-trees of height at most* 2*.*

In the paper, we denote M-trees by $\mathfrak{r}, \mathfrak{s}, \mathfrak{t}, \ldots$ and sets of M-trees by $\mathfrak{R}, \mathfrak{S}, \mathfrak{T}, \ldots$ In particular, for all h, we denote by $\mathfrak{M}(h, M)$ (or $\mathfrak{M}(h)$ when M is clear from the context), the set of all M-trees of height h. For all h, $\mathfrak{M}(h)$ is finite.

Monoid Operation. For all h, we equip the set $\mathfrak{M}(h)$ with a monoid operation. The definition is by induction on h. For $h = 0$, $\mathfrak{M}(0) = M$ is already a monoid by definition. For $h \geqslant 1$, if $(s, \mathfrak{S})$ and $(r, \mathfrak{R})$ are M-trees of height h, we set $(s, \mathfrak{S}) \cdot (r, \mathfrak{R}) = (sr, \mathfrak{S}\mathfrak{R})$ with $\mathfrak{S}\mathfrak{R} = \{\mathfrak{s} \cdot \mathfrak{r} \mid \mathfrak{s} \in \mathfrak{S} \text{ and } \mathfrak{r} \in \mathfrak{R}\}$. One can verify that this multiplication does give a monoid structure to the sets $\mathfrak{M}(h)$ for all $h \geqslant 0$.

Preorder. For all h, we equip the set $\mathfrak{M}(h)$ with a preorder "$\sqsubseteq$" that is compatible with our monoid operation. Intuitively, given a M-tree, a smaller one is obtained by removing subtrees in its representation (see Figure 3). The formal definition is by induction on h. When $h = 0$, the preorder "$\sqsubseteq$" is just equality. Otherwise, given M-trees of height $h \geqslant 1$, $\mathfrak{s} = (s, \mathfrak{S})$ and $\mathfrak{r} = (r, \mathfrak{R})$, we write $\mathfrak{s} \sqsubseteq \mathfrak{r}$ when $s = r$ and for all $\mathfrak{s}' \in \mathfrak{S}$, there exists $\mathfrak{r}' \in \mathfrak{R}$ such that $\mathfrak{s}' \sqsubseteq \mathfrak{r}'$. One can verify that for all $h \in \mathbb{N}$, $\sqsubseteq$ is preorder and the following fact.

Fact 14. *For all $h \geqslant 0$, $\sqsubseteq$ is compatible with the monoid operation of $\mathfrak{M}(h)$: given $\mathfrak{s}, \mathfrak{t}, \mathfrak{s}', \mathfrak{t}' \in \mathfrak{M}(h)$, if $\mathfrak{s} \sqsubseteq \mathfrak{t}$ and $\mathfrak{s}' \sqsubseteq \mathfrak{t}'$, then $\mathfrak{s}\mathfrak{s}' \sqsubseteq \mathfrak{t}\mathfrak{t}'$.*

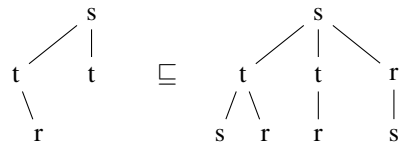


Fig. 3. M-Trees comparable with "$\sqsubseteq$"

Finally, for all h and all $\mathfrak{S} \subseteq \mathfrak{M}(h)$, we denote by $\downarrow\mathfrak{S}$, the *downset of* $\mathfrak{S}$, i.e. $\downarrow\mathfrak{S} = \{\mathfrak{s} \mid \exists \mathfrak{s}' \in \mathfrak{S} \text{ such that } \mathfrak{s} \sqsubseteq \mathfrak{s}'\}$ is the set of all M-trees that are smaller than a tree of $\mathfrak{S}$.

Transforming M-Trees. We define two operations that will allow us to build M-trees of height $h+1$ from M-trees of height h and vice-versa. The first operation, called *root duplication*, is applied to *single M-trees*. It completes a M-tree of height h into a M-tree of type $h+1$ by duplicating its root. Let $h \geqslant 0$ and let $\mathfrak{r} = (r, \mathfrak{R}) \in \mathfrak{M}(h)$, we set $dup(\mathfrak{r})$ as the M-tree $dup(\mathfrak{r}) = (r, \{\mathfrak{r}\}) \in \mathfrak{M}(h+1)$ (see Figure 4).

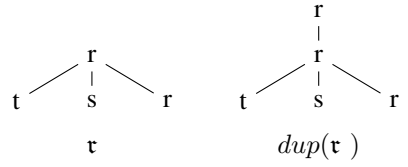


Fig. 4. Root Duplication

Our second operation, called *root removal*, is applied to *sets of M-trees*. It builds a set of M-trees of height $h-1$ from a set of M-trees of height h by removing their roots. Set $h \geqslant 1$ and $\mathfrak{S} \subseteq \mathfrak{M}(h)$, we define $rem(\mathfrak{S}) \subseteq \mathfrak{M}(h-1)$ as the set $rem(\mathfrak{S}) = \bigcup_{(r,\mathfrak{R})\in\mathfrak{S}} \mathfrak{R}$ (see Figure 5).

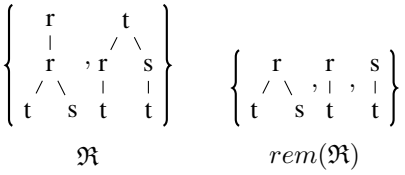


Fig. 5. Root Removal

The following fact states that root duplication and root removal are morphisms and can be verified from the definitions.

Fact 15. *Let $h \geqslant 0$, $\mathfrak{s}, \mathfrak{t} \in \mathfrak{M}(h)$ and $\mathfrak{S}, \mathfrak{T} \subseteq \mathfrak{M}(h+1)$.*

- $rem(\mathfrak{S} \cdot \mathfrak{T}) = rem(\mathfrak{S}) \cdot rem(\mathfrak{T})$
- $dup(\mathfrak{s} \cdot \mathfrak{t}) = dup(\mathfrak{s}) \cdot dup(\mathfrak{t})$

B. Set of Σ-Trees Associated to a Morphism

Given a fixed morphism $\alpha : A^* \to M$ into a finite monoid M, we are not interested in all M-trees, but only in those that carry information with respect to the preorders $\lesssim_i^k$.

Given $h \geqslant 0$ and a tuple $(i_1, \ldots, i_h) \in (\mathbb{N} \setminus \{0\})^h$ of levels in the hierarchy, we associate to α the set $\mathfrak{T}_{i_1,\ldots,i_h}[\alpha] \subseteq \mathfrak{M}(h)$ of $\Sigma_{i_1,\ldots,i_h}$-trees of α. We first define a set of $\Sigma_{i_1,\ldots,i_h}[k]$-trees for each rank k. The $\Sigma_{i_1,\ldots,i_h}$-trees will then be those that are $\Sigma_{i_1,\ldots,i_h}[k]$-trees for all k.

We begin with the general definition. We will then present an example of $\Sigma_{2,3}[k]$-tree. Let $h \geqslant 0$, $(i_1, \ldots, i_h) \in (\mathbb{N} \setminus \{0\})^h$ (if $h = 0$, the tuple is empty) and $k \geqslant 1$. Given $\mathfrak{r} \in \mathfrak{M}(h)$, whether $\mathfrak{r}$ is a $\Sigma_{i_1,\ldots,i_h}[k]$-tree depends on the existence of a *witness* $w \in A^*$. We say that w *is a* $(k, (i_1, \ldots, i_h))$*-witness for* $\mathfrak{r}$ if one of the two following properties hold,

- If $h = 0$, the tuple is empty and we ask that $\alpha(w) = \mathfrak{r}$ (recall that $\mathfrak{M}(0) = M$).
- If $h \geqslant 1$, set $(r, \mathfrak{R}) = \mathfrak{r} \in \mathfrak{M}(h)$. We ask that $\alpha(w) = r$ and for all $\mathfrak{r}' \in \mathfrak{R} \subseteq \mathfrak{M}(h-1)$, there exists $w' \in A^*$ such that $w \lesssim_{i_1}^k w'$ and w' is a $(k, (i_2, \ldots, i_h))$-witness for $\mathfrak{r}'$.

Given $\mathfrak{t} \in \mathfrak{M}(h)$, we let $\mathfrak{t} \in \mathfrak{T}^k_{i_1,\ldots,i_h}[\alpha]$ if and only if there exists a $(k, (i_1, \ldots, i_h))$-witness w for $\mathfrak{t}$. Finally, we define, $\mathfrak{T}_{i_1,\ldots,i_h}[\alpha] = \cap_k \mathfrak{T}^k_{i_1,\ldots,i_h}[\alpha]$, the *set of $\Sigma_{i_1,\ldots,i_h}$-trees for α*. We now illustrate the definition with a $\Sigma_{2,3}[k]$-tree.

Example of $\Sigma_{2,3}[k]$-tree. Let $u, v_1, v_2, v_3, w_1, w_2, w_3 \in A^*$ be words and $r, s_1, s_2, s_3, t_1, t_2, t_3 \in M$ be their images under α. Observe that, when $h = 0$, the definition of witness does not depend on k or the preorders: any word is a $(k, ())$-witness for its image. We will use the fact that w_1, w_2, w_3 are $(k, ())$-witnesses for t_1, t_2, t_3.

In order to build a $\Sigma_{2,3}[k]$-tree out of the words u, v_1, v_2, v_3, w_1, w_2, w_3, we now have to assume some links between these words with respect to the preorders. With respect to $\lesssim_2^k$, we assume that $u \lesssim_2^k v_1$, $u \lesssim_2^k v_2$ and $u \lesssim_2^k v_3$. With respect to $\lesssim_3^k$, we assume that $v_1 \lesssim_3^k w_1$, $v_1 \lesssim_3^k w_2$ and $v_2 \lesssim_3^k w_3$. These links can be represented in a tree-like fashion as depicted in Figure 6.

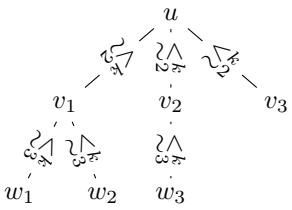


Fig. 6. Links between the words $u, v_1, v_2, v_3, w_1, w_2$ and w_3

Since $\alpha(v_1) = s_1$, $v_1 \lesssim_3^k w_1$ and $v_1 \lesssim_3^k w_2$, we conclude that v_1 is a $(k, 3)$-witness for $(s_1, \{t_1, t_2\})$. Similarly, since $\alpha(v_2) = s_2$ and $v_2 \lesssim_3^k w_3$, v_2 is a $(k, 3)$-witness for $(s_2, \{t_3\})$. Finally, since $\alpha(v_3) = s_3$, v_3 is a $(k, 3)$-witness for $(s_3, \emptyset)$.

Using this knowledge, we can now construct a M-tree of height 2 for which u is a $(k, (2, 3))$-witness. Since $\alpha(u) = r$, $u \lesssim_2^k v_1$, $u \lesssim_2^k v_2$ and $u \lesssim_2^k v_3$ we obtain that u is a $(k, (2, 3))$-witness for the M-tree depicted in Figure 7. Hence this M-tree belongs to $\mathfrak{T}^k_{2,3}[\alpha]$.

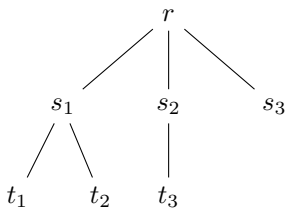


Fig. 7. A $\Sigma_{2,3}[k]$-tree of height 2 and $(k, (2, 3))$-witness u

An important observation is that for any i, the Σ_i pairs we discussed at the beginning of the section are encoded in Σ_i-trees: (s, t) is a pair for Σ_i if and only if $(s, \{t\})$ is a Σ_i-tree.

However $\Sigma_{i_1,\ldots,i_h}$-trees may capture much more information. In particular, $\Sigma_{2,3}$-trees capture exactly the kind of information that pairs for Σ_2 and Σ_3 were lacking. Indeed, assume that the tree $(t, \{s, \{s_\ell, s_r\}\})$ is a $\Sigma_{2,3}$-tree. In particular, this means that (t, s) is a pair for Σ_2 and $(s, s_\ell), (s, s_r)$ are pairs for Σ_3. But we know more: for all k, the same word can serve as the witness for s in all three pairs.

Additional Properties. We finish the definitions with a few lemmas that will be useful when manipulating Σ-trees.

Lemma 16. *Let $h \geqslant 0$ and $(i_1, \ldots, i_h) \in (\mathbb{N} \setminus \{0\})^h$, then, for all $k \in \mathbb{N}$, $\mathfrak{T}_{i_1,\ldots,i_h}[\alpha] \subseteq \mathfrak{T}^{k+1}_{i_1,\ldots,i_h}[\alpha] \subseteq \mathfrak{T}^k_{i_1,\ldots,i_h}[\alpha]$.*

Proof: Immediate from the definition and Fact 3 (for any $k, i \geqslant 1$, $u \lesssim_i^{k+1} v \Rightarrow u \lesssim_i^k v$). ∎

The second lemma is a consequence of Lemma 6 (the preorders $\lesssim_i^k$ are precongruences) and states that $\mathfrak{T}_{i_1,\dots,i_h}[\alpha]$ is a monoid.

Lemma 17. *Let $h \geqslant 0$ and $(i_1,\dots,i_h) \in (\mathbb{N}\setminus\{0\})^h$. For all $k \in \mathbb{N}$, if u,v are $(k,(i_1,\dots,i_h))$-witnesses for $\mathfrak{s},\mathfrak{t} \in \mathfrak{M}(h)$, then uv is a $(k,(i_1,\dots,i_h))$-witness for $\mathfrak{st}$.*

In particular, $\mathfrak{T}^k_{i_1,\dots,i_h}[\alpha]$ and $\mathfrak{T}_{i_1,\dots,i_h}[\alpha]$ are submonoids of $\mathfrak{M}(h)$.

Proof: The proof is immediate using Lemma 6 and an induction on h. ∎

The last lemma states some simple closure properties of the sets of Σ-trees.

Lemma 18. *Let $h \geqslant 0$, and $(i_1,\dots,i_h) \in (\mathbb{N}\setminus\{0\})^h$.*

- $\downarrow\mathfrak{T}_{i_1,\dots,i_h}[\alpha] = \mathfrak{T}_{i_1,\dots,i_h}[\alpha]$.
- *For all $\mathfrak{t} \in \mathfrak{T}_{i_2,\dots,i_h}[\alpha]$, $dup(\mathfrak{t}) \in \mathfrak{T}_{i_1,\dots,i_h}[\alpha]$.*
- $\mathfrak{T}_{i_2,\dots,i_h}[\alpha] = rem(\mathfrak{T}_{i_1,\dots,i_h}[\alpha])$.

Proof: The first item is immediate from the definition. The second item follows from the fact that if w is a $(k,(i_2,\dots,i_h))$-witness for $\mathfrak{t} \in \mathfrak{M}(h-1)$ then it is also a $(k,(i_1,\dots,i_h))$-witness for $dup(\mathfrak{t})$ (since $w \lesssim^k_{i_1} w$). The last item follows from the definition and the second item. ∎

C. Σ-Trees and Separation

Theorem 19. *Let L_1, L_2 be regular languages and let $\alpha : A^* \to M$ be a morphism into a finite monoid M that recognizes them both with accepting sets $F_1, F_2 \subseteq M$.*

Then, for any $i \in \mathbb{N}\setminus\{0\}$, L_1 is Σ_i-separable from L_2 if and only if for all $s_1, s_2 \in F_1, F_2$, $(s_1,\{s_2\}) \notin \mathfrak{T}_i[\alpha]$.

Proof: This is classical and an immediate consequence of Lemma 12. ∎

It follows from Theorem 19 that the separation problem for Σ_i amounts to finding an algorithm that computes $\mathfrak{T}_i[\alpha]$ from a morphism α. Though the terminology used was different, this was proven to be possible for $i = 1$ and $i = 2$. Both algorithms are least fixpoint algorithms. For $i = 1$ the algorithm is simple and only needs to consider the set $\mathfrak{T}_1[\alpha]$ itself (actually, in this case, even considering only pairs for Σ_1 suffices). For $i = 2$, the algorithm slightly generalizes $\mathfrak{T}_2[\alpha]$ by parametrizing the set with subsets of the alphabet (see [PZ14a]). However, by restricting it to alphabet compatible morphisms, this algorithm can be reformulated to work directly with the set $\mathfrak{T}_2[\alpha]$.

In the paper, we present an algorithm for the case $i = 3$. Our technique requires to work with the set $\mathfrak{T}_{2,3}[\alpha]$ (which generalizes $\mathfrak{T}_3[\alpha]$ by Lemma 18). Moreover, it only works for morphisms α that are alphabet compatible (we know from Lemma 9 that this is not restrictive for separation). An interesting observation is that the objects our algorithms consider become more and more complicated as i gets larger. For $i \geqslant 4$, it seems that objects that are more general than Σ-trees need to be considered (in particular the set $\mathfrak{T}_{2,3,4}[\alpha]$ does not seem to capture enough information).

V. A Separation Algorithm for Σ_3

In this section, we present our algorithm which, given an alphabet compatible morphism $\alpha : A^* \to M$ as input, computes the set $\mathfrak{T}_3[\alpha]$. By Lemma 9, we know that from any two regular languages, one can always compute an alphabet compatible morphism that recognizes them both. Hence, by Theorem 19 this will prove Theorem 1: Σ_3-separability is decidable. As we explained, our algorithm actually computes a more general object: the set $\mathfrak{T}_{2,3}[\alpha]$.

Note that intuitively, our algorithm actually computes both $\mathfrak{T}_{2,3}[\alpha]$ and $\mathfrak{T}_3[\alpha]$ simultaneously: by Lemma 18, $\mathfrak{T}_3[\alpha]$ is encoded in $\mathfrak{T}_{2,3}[\alpha]$ ($\mathfrak{T}_3[\alpha] = rem(\mathfrak{T}_{2,3}[\alpha])$) and every M-tree $\mathfrak{t} \in \mathfrak{T}_3[\alpha]$ can be lifted as the M-tree $dup(\mathfrak{t}) \in \mathfrak{T}_{2,3}[\alpha]$.

We now present the algorithm. We assume fixed an alphabet compatible morphism $\alpha : A^* \to M$ into a finite monoid M. Recall that since α is alphabet compatible, $\mathsf{alph}(s)$ is well defined for all $s \in M$ (we may assume without loss of generality that α is surjective). Moreover, we extend the notation to elements of $\mathfrak{M}(h)$ for $h \geqslant 1$, if $\mathfrak{s} = (s,\mathfrak{S}) \in \mathfrak{M}(h)$, we set $\mathsf{alph}(\mathfrak{s}) = \mathsf{alph}(s)$.

As we explained, our algorithm computes a *least fixpoint*. Initially, we define a set $\mathfrak{I}$, that contains only trivial $\Sigma_{2,3}$-trees. Then, we saturate this map with four operations until a fixpoint is reached.

Let us first define the set $\mathfrak{I} \subseteq \mathfrak{T}_{2,3}[\alpha] \subseteq \mathfrak{M}(2)$. We define $\mathfrak{I} = \{(\alpha(w),\{(\alpha(w),\{\alpha(w)\})\}) \mid w \in A^*\}$. Observe that it is immediate that $\mathfrak{I} \subseteq \mathfrak{T}_{2,3}[\alpha]$ and that $\mathfrak{I}$ can be computed.

We now define our least fixpoint procedure. To any $\mathfrak{R} \subseteq \mathfrak{M}(2)$, we associate a set $Sat(\mathfrak{R}) \subseteq \mathfrak{M}(2)$ that we define as the smallest set (with respect to inclusion) such that $\mathfrak{R} \subseteq Sat(\mathfrak{R})$ and which is closed under the following four operations.

(1) Downset Closure. This reflects the fact that $\mathfrak{T}_{2,3}[\alpha]$ is closed under downset (see Lemma 18). We ask that,

$$\downarrow Sat(\mathfrak{R}) \subseteq Sat(\mathfrak{R}) \tag{1}$$

(2) Multiplication Closure. This reflects Lemma 17, i.e. the fact that $\mathfrak{T}_{2,3}[\alpha]$ is a monoid. We ask that,

$$Sat(\mathfrak{R}) \cdot Sat(\mathfrak{R}) \subseteq Sat(\mathfrak{R}) \tag{2}$$

(3) Σ_2 Closure. For all $\mathfrak{r} \in Sat(\mathfrak{R})$, consider $\mathfrak{S} = \{\mathfrak{s} \in rem(Sat(\mathfrak{R})) \mid \mathsf{alph}(\mathfrak{s}) = \mathsf{alph}(\mathfrak{r})\} \subseteq \mathfrak{M}(1)$, we ask that,

$$\mathfrak{r}^\omega \cdot (1_M,\mathfrak{S}) \cdot \mathfrak{r}^\omega \in Sat(\mathfrak{R}) \tag{3}$$

with ω as $\omega(\mathfrak{M}(2))$. This operation is based on Lemma 8. When $Sat(\mathfrak{R}) = \mathfrak{T}_{2,3}[\alpha]$, $\mathfrak{S}$ is the set that contains all M-trees of $\mathfrak{T}_3[\alpha]$ having alphabet $\mathsf{alph}(\mathfrak{r})$. We use this set and $\mathfrak{r}$ to construct a new M-tree which, by Lemma 8, has to belong to $\mathfrak{T}_{2,3}[\alpha]$ as well.

(4) Σ_3 Closure. For all $r \in M$ and all $\mathfrak{s} \in \mathfrak{M}(1)$ such that $(r,\{\mathfrak{s}\}) \in Sat(\mathfrak{R})$, we ask that,

$$dup(\mathfrak{s}^\omega \cdot (1_M,\{1_M,r\}) \cdot \mathfrak{s}^\omega) \in Sat(\mathfrak{R}) \tag{4}$$

with ω as $\omega(\mathfrak{M}(1))$. This operation is based on Lemma 7. Given a M-tree of the form $(r,\{\mathfrak{s}\})$ in $\mathfrak{T}_{2,3}[\alpha]$, one can prove

using Lemma 7 that the M-tree $\mathfrak{s}^\omega \cdot (1_M, \{1_M, r\}) \cdot \mathfrak{s}^\omega$ has to belong to $\mathfrak{T}_3[\alpha]$. We then lift this Σ_3-tree as a $\Sigma_{2,3}$-tree with root duplication.

This finishes the definition of $Sat(\mathfrak{R})$. It is immediate from the definition that given $\mathfrak{R}$ as input, one can compute $Sat(\mathfrak{R})$ with a least fixpoint algorithm. It now remains to prove that this algorithm can be used to compute $\mathfrak{T}_{2,3}[\alpha]$ from $\mathfrak{I}$. We state this result in the following proposition.

Proposition 20. $\mathfrak{T}_{2,3}[\alpha] = Sat(\mathfrak{I})$

It follows from Proposition 20 that from any alphabet compatible morphism α, one can compute the set $\mathfrak{T}_{2,3}[\alpha]$. Therefore, one can compute $\mathfrak{T}_3[\alpha]$. Finally, we obtain from Theorem 19 that one can decide the Σ_3-separation problem which proves our main theorem: Theorem 1.

It now remains to prove Proposition 20. We prove that there exists $\ell \in \mathbb{N}$ (depending only on α) such that

$$\mathfrak{T}_{2,3}[\alpha] \subseteq \mathfrak{T}^\ell_{2,3}[\alpha] \subseteq Sat(\mathfrak{I}) \subseteq \mathfrak{T}_{2,3}[\alpha]$$

That $\mathfrak{T}_{2,3}[\alpha] \subseteq \mathfrak{T}^\ell_{2,3}[\alpha]$ is immediate from Lemma 16 for any $\ell \in \mathbb{N}$. The inclusion $Sat(\mathfrak{I}) \subseteq \mathfrak{T}_{2,3}[\alpha]$ corresponds to *correctness* of the algorithm: all computed M-trees are $\Sigma_{2,3}$-trees. The proof amounts to showing that $\mathfrak{T}_{2,3}[\alpha]$ is closed under the four operations and is based on Lemmas 7 and 8. We present this proof in Section VI.

Finally, the most difficult inclusion is that there exists ℓ such that $\mathfrak{T}^\ell_{2,3}[\alpha] \subseteq Sat(\mathfrak{I})$. This inclusion corresponds to *completeness* of the algorithm: all $\Sigma_{2,3}$-trees are computed. We present this proof in Section VII.

VI. Correctness in Proposition 20

Recall that an alphabet compatible morphism $\alpha : A^* \to M$ is fixed. We prove that $Sat(\mathfrak{I}) \subseteq \mathfrak{T}_{2,3}[\alpha]$ in Proposition 20. By definition $Sat(\mathfrak{I})$ is the smallest set containing $\mathfrak{I}$ and closed under the four operations of Section V. It is immediate from the definition that $\mathfrak{I} \subseteq \mathfrak{T}_{2,3}[\alpha]$. We need to prove that $\mathfrak{T}_{2,3}[\alpha]$ is closed under downset closure, multiplication closure, Σ_2 closure and Σ_3 closure.

Closure under downset and multiplication is immediate from Lemmas 18 and 17. We now prove Σ_2 closure and Σ_3 closure. We devote a subsection to each proof.

A. Σ_2 Closure

This is a consequence of Lemma 8. Set $\mathfrak{r} \in \mathfrak{T}_{2,3}[\alpha]$ and $\mathfrak{S} = \{\mathfrak{s} \in rem(\mathfrak{T}_{2,3}[\alpha]) \mid \mathsf{alph}(\mathfrak{s}) = \mathsf{alph}(\mathfrak{r})\}$. We have to prove that

$$\mathfrak{t} = \mathfrak{r}^\omega \cdot (1_M, \mathfrak{S}) \cdot \mathfrak{r}^\omega \in \mathfrak{T}_{2,3}[\alpha]$$

By definition, $\mathfrak{T}_{2,3}[\alpha] = \cap_k \mathfrak{T}^k_{2,3}[\alpha]$, hence, it suffices to prove that for all k, $\mathfrak{t} \in \mathfrak{T}^k_{2,3}[\alpha]$: we need to find a $(k,(2,3))$-witness $w \in A^*$ for $\mathfrak{t}$.

Since $\mathfrak{r} \in \mathfrak{T}_{2,3}[\alpha]$, by definition, there exists a $(k,(2,3))$-witness u for $\mathfrak{r}$. Set $k' = 2^{2k}\omega$ (with ω as $\omega(\mathfrak{M}(2))$). We prove that $w = u^{2k'}$ is a $(k,(2,3))$-witness for $\mathfrak{t}$.

Set $(r, \mathfrak{R}) = \mathfrak{r}$, it follows that $\mathfrak{t} = (r^\omega, \mathfrak{R}^\omega \mathfrak{S} \mathfrak{R}^\omega)$. By definition of u, $\alpha(u) = r$, hence, $\alpha(w) = r^\omega$. Set $\mathfrak{t}' \in \mathfrak{R}^\omega \mathfrak{S} \mathfrak{R}^\omega$, we have to find a $(k,3)$-witness w' for t' such that $w \lesssim^k_2 w'$.

By definition $\mathfrak{t}' = \mathfrak{r}_1 \cdot \mathfrak{s} \cdot \mathfrak{r}_2$ with $\mathfrak{r}_1, \mathfrak{r}_2 \in \mathfrak{R}^\omega$ and $\mathfrak{s} \in \mathfrak{S}$. By Lemma 17, we know that $u^{k'}$ is a $(k,(2,3))$-witness for $(r^\omega, \mathfrak{R}^\omega)$, hence there exists $(k,3)$-witnesses u_1, u_2 for $\mathfrak{r}_1, \mathfrak{r}_2$ such that $u^{k'} \lesssim^k_2 u_1$ and $u^{k'} \lesssim^k_2 u_2$. Moreover, we know from Lemma 18 that $rem(\mathfrak{T}_{2,3}[\alpha]) = \mathfrak{T}_3[\alpha]$, hence $\mathfrak{S} = \{\mathfrak{s} \in \mathfrak{T}_3[\alpha] \mid \mathsf{alph}(\mathfrak{s}) = \mathsf{alph}(\mathfrak{r})\}$. This means that there exists a $(k,3)$-witness v for $\mathfrak{s}$. We define $w' = u_1 v u_2$ which is by definition and Lemma 17 a $(k,3)$-witness for $\mathfrak{t}'$. It remains to prove that $w \lesssim^k_2 w'$.

We prove that $w = u^{2k'} \lesssim^k_2 u^{k'} v u^{k'} \lesssim^k_2 u_1 v u_2 = w'$. We have $\mathsf{alph}(\mathfrak{r}) = \mathsf{alph}(\mathfrak{s})$, hence $\mathsf{alph}(u) = \mathsf{alph}(v)$. Moreover, $k' = 2^{2k}\omega$, hence we can apply Lemma 8 to conclude that $u^{2k'} \lesssim^k_2 u^{k'} v u^{k'}$. Finally, since $u^{k'} \lesssim^k_2 u_1$ and $u^{k'} \lesssim^k_2 u_2$, we obtain $u^{k'} v u^{k'} \lesssim^k_2 u_1 v u_2$ from Lemma 6.

B. Σ_3 Closure

This is a consequence of Lemma 7. Set $r \in M$ and $\mathfrak{s} \in \mathfrak{M}(1)$ such that $(r, \{\mathfrak{s}\}) \in \mathfrak{T}_{2,3}[\alpha]$. We have to prove that

$$dup(\mathfrak{s}^\omega \cdot (1_M, \{1_M, r\}) \cdot \mathfrak{s}^\omega) \in \mathfrak{T}_{2,3}[\alpha]$$

By Lemma 18, we know that $\{dup(\mathfrak{t}) \mid \mathfrak{t} \in \mathfrak{T}_3[\alpha]\} \subseteq \mathfrak{T}_{2,3}[\alpha]$. Hence it suffices to prove that,

$$\mathfrak{t} = \mathfrak{s}^\omega \cdot (1_M, \{1_M, r\}) \cdot \mathfrak{s}^\omega \in \mathfrak{T}_3[\alpha]$$

By definition, $\mathfrak{T}_3[\alpha] = \cap_k \mathfrak{T}^k_3[\alpha]$, hence, it suffices to prove that for all k, $\mathfrak{t} \in \mathfrak{T}^k_3[\alpha]$: we need to find a $(k,3)$-witness $w \in A^*$ for $\mathfrak{t}$.

By hypothesis, there is a $(k,(2,3))$-witness v for $(r, \{\mathfrak{s}\})$. Hence there is a $(k,3)$-witness u for $\mathfrak{s}$ such that $v \lesssim^k_2 u$. Set $k' = 2^k\omega$ and $w = u^{2k'\omega}$ (with $\omega = \omega(\mathfrak{M}(1))$), we prove that w is $(k,3)$-witness for $\mathfrak{t}$.

Set $(s, S) = \mathfrak{s}$, by definition $\mathfrak{t} = (s^\omega, S^\omega \cdot \{1_M, r\} \cdot S^\omega)$. By choice of u, $\alpha(u) = s$ and by choice of k', $\alpha(w) = s^\omega$. Set $t' \in S^\omega \cdot \{1_M, r\} \cdot S^\omega$, we have to find $w' \in A^*$ such that $\alpha(w') = t'$ and $w \lesssim^k_3 w'$.

By definition, there exists $s_1, s_2 \in S^{k'}$ such that either $t' = s_1 \cdot s_2$ or $t' = s_1 \cdot r \cdot s_2$. By Lemma 17, $u^{k'}$ is a $(k,3)$-witness for $\mathfrak{s}^{k'} = (s^{k'}, S^{k'})$. Hence, there exist u_1, u_2 such that $\alpha(u_1) = s_1$, $\alpha(v_2) = s_2$, $u^{k'} \lesssim^k_3 u_1$ and $u^{k'} \lesssim^k_3 u_2$.

If $t' = s_1 \cdot s_2$, we simply set $w' = u_1 u_2$, it is then immediate from Lemma 6, that $w = u^{k'} u^{k'} \lesssim^k_3 u_1 u_2 = w'$. Otherwise, $t' = s_1 \cdot r \cdot s_2$ and we set $w' = u_1 v u_2$. Observe that since $v \lesssim^k_2 u$ and $k' = 2^k\omega$, we can apply Lemma 7 and obtain that $w = u^{2k'} \lesssim^k_3 u^{k'} v u^{k'}$. It then follows from Lemma 6 that $u^{k'} v u^{k'} \lesssim^k_3 u_1 v u_2 = w'$. Hence $w \lesssim^k_3 w'$ and we are finished.

VII. Completeness in Proposition 20

Recall that an alphabet compatible morphism $\alpha : A^* \to M$ into a finite monoid M is fixed. In this section, we prove that there exists $\ell \in \mathbb{N}$ such that the inclusion $\mathfrak{T}^\ell_{2,3}[\alpha] \subseteq Sat(\mathfrak{I})$ holds in Proposition 20. Note that we do not state the constant ℓ explicitly. However, it can be computed from our proof

argument. We begin with additional terminology that we require in order to present our argument.

Generated Σ-Trees. Set $k \in \mathbb{N}$, to any $w \in A^*$, we associate a $\Sigma_3[k]$-tree $\mathfrak{g}_k(w) \in \mathfrak{M}(1)$ as follows,

$$\mathfrak{g}_k(w) = (\alpha(w), \{\alpha(w') \mid w \lesssim_3^k w'\}) \in \mathfrak{M}(1)$$

By definition, $\mathfrak{g}_k(w)$ is the maximal M-tree for which w is a $(k,3)$-witness.

Fact 21. *For all $k \geqslant 1$, $\mathfrak{T}_3^k[\alpha] = \downarrow\{\mathfrak{g}_k(w) \mid w \in A^*\}$.*

We define a similar object for $\Sigma_{2,3}[k]$-trees. For technical reasons, we need this object to have more parameters. Set $k_2, k_3 \in \mathbb{N}$, $w \in A^*$ and $L \subseteq A^*$. We define,

$$\mathfrak{g}^L_{k_2,k_3}(w) = (\alpha(w), \{\mathfrak{g}_{k_3}(w') \mid w \lesssim_2^{k_2} w' \wedge w' \in L\}) \in \mathfrak{M}(2)$$

Again, one can verify the following fact from the definitions.

Fact 22. *For all $k \geqslant 1$, $\mathfrak{T}_{2,3}^k[\alpha] = \downarrow\{\mathfrak{g}^{A^*}_{k,k}(w) \mid w \in A^*\}$.*

We will also use the following fact which follows from Fact 3 (i.e. the preorders $\lesssim_i^k$ are refined when k gets larger):

Fact 23. *For all $L \subseteq A^*$, $w \in A^*$ and $k_2, k_3, k_2', k_3' \geqslant 1$ such that $k_2 \leqslant k_2'$ and $k_3 \leqslant k_3'$,*

$$\begin{array}{rcl} \mathfrak{g}_{k_3'}(w) & \sqsubseteq & \mathfrak{g}_{k_3}(w) \\ \mathfrak{g}^L_{k_2',k_3'}(w) & \sqsubseteq & \mathfrak{g}^L_{k_2,k_3}(w) \end{array}$$

This finishes the definition of generated Σ-trees. We can now prove the inclusion $\mathfrak{T}_{2,3}^\ell[\alpha] \subseteq Sat(\mathfrak{I})$ for some ℓ. We rely on two propositions that we state below. We say that a language is *closed under factors* if any factor of a word in the language is in the language as well.

Proposition 24 (Σ_2 Level)**.** *There exists $k_2 \in \mathbb{N}$ such that for all $k \in \mathbb{N}$, there exists $k_3 \in \mathbb{N}$ such that for all $L \subseteq A^*$, if L is closed under factors and $\mathfrak{g}_k(u) \in rem(Sat(\mathfrak{I}))$ for any $u \in L$, then for all $w \in A^*$, $\mathfrak{g}^L_{k_2,k_3}(w) \in Sat(\mathfrak{I})$.*

Proposition 25 (Σ_3 Level)**.** *There exists $k \in \mathbb{N}$ such that for all $w \in A^*$, $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$.*

Let us first use the two propositions to conclude the proof of completeness in Proposition 20. Set k_2 as defined in Proposition 24 and k as defined in Proposition 25. From our choice of k, we obtain a third natural k_3 from Proposition 24. Set $\ell = max(k_2, k_3)$, we prove that $\mathfrak{T}_{2,3}^\ell[\alpha] \subseteq Sat(\mathfrak{I})$. Let $\mathfrak{t} \in \mathfrak{T}_{2,3}^\ell[\alpha]$, by Fact 22 and Fact 23 we get $w \in A^*$ such that $\mathfrak{t} \sqsubseteq \mathfrak{g}^{A^*}_{k_2,k_3}(w)$. Moreover, we know from our choice of k in Proposition 25 that for all $u \in A^*$, $\mathfrak{g}_k(u) \in rem(Sat(\mathfrak{I}))$. Hence, by choice of k_2, k_3, setting $L = A^*$ in Proposition 24 yields $\mathfrak{g}^{A^*}_{k_2,k_3}(w) \in Sat(\mathfrak{I})$. By downset closure, we finally obtain that $\mathfrak{t} \in Sat(\mathfrak{I})$.

It now remains to prove Propositions 24 and 25. Both proofs are inductions on the structure of a factorization forest of w. They are both generalizations of the proof of the Σ_2-separation algorithm of [PZ14a]. The proof of Proposition 24 relies on Σ_2 closure and is independent from Proposition 25. It is available in the full version of the paper.

We concentrate on Proposition 25. The proof relies on Σ_3 closure and uses Proposition 24 as a subresult (this is where we use the parameter L in the statement of Proposition 24). This interaction between the two results is not surprising. This reflects the fact that in order to compute some $\Sigma_{2,3}$-trees the least fixpoint algorithm might have to alternate several times between Σ_2 and Σ_3 closure.

We devote the remainder of the section to proving Proposition 25. We define a new morphism γ that generalizes α. We will then use γ-factorization forests.

For the remainder of the section, we fix k_2 as the natural defined in Proposition 24. By definition, k_2 only depends on the morphism $\alpha : A^* \to M$. Consider the preorder $\lesssim_2^{k_2}$. We denote by $\cong_2^{k_2}$ the equivalence generated by $\lesssim_2^{k_2}$. Recall that by Lemma 6, $\cong_2^{k_2}$ is a congruence for concatenation, hence the quotient $A^*/\cong_2^{k_2}$ is a finite monoid (finiteness comes from the fact that there are only finitely many non equivalent Σ_2 formulas of quantifier rank k_2). We write

$$\begin{array}{rccl} \gamma: & A^* & \to & M \times (A^*/\cong_2^{k_2}) \\ & w & \mapsto & (\alpha(w), [w]_{\cong_2^{k_2}}) \end{array}$$

with $[w]_{\cong_2^{k_2}}$ as the equivalence class of w. Proposition 25 is now a consequence of the following proposition.

Proposition 26. *For all $h, p \in \mathbb{N}$, there exists $k \in \mathbb{N}$ such that for all $w \in A^*$ admitting a γ-factorization forest of height at most h and idempotent height at most p, $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$.*

By Theorem 10 any word $w \in A^*$ admits a γ-factorization forest of height (and hence idempotent height) at most $3|M \times (A^*/\cong_2^{k_2})| - 1$. Hence Proposition 25 is an immediate consequence of Proposition 26.

To simplify notations, for all $h, p \in \mathbb{N}$, we write that $w \in A^*$ satisfies $\mathcal{H}(p,h)$ when w admits a γ-factorization forest of height at most h and idempotent height at most p. We prove that for all h, p there exists k such that for any w satisfying $\mathcal{H}(p,h)$, $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$. We use induction on two parameters that we list by order of importance:

1) the idempotent height p.
2) the height h.

In the base case $h = 0$ (and therefore $p = 0$). We set $k = 2$. A word w admitting a γ-factorization forest of height $h = 0$ is either a single letter $a \in A$ or the empty word. One can verify that since $k = 2$, the only word $w' \in A^*$ such that $w \lesssim_3^k w'$ is w itself. Hence,

$$\mathfrak{g}_k(w) = (\alpha(w), \{\alpha(w)\}) \in rem(\mathfrak{I}) \subseteq rem(Sat(\mathfrak{I}))$$

We now set $h, p \in \mathbb{N}$ such that $h \geqslant 1$. We begin by using induction to define the natural k that we need. For the remainder of the proof, we will denote by n the size of the set of M-trees of height 1: $n = |\mathfrak{M}(1)|$. Using induction and Proposition 24, we can define the three following naturals:

1) By induction on h, there exists $k_h \in \mathbb{N}$ such that for any word w satisfying $\mathcal{H}(p, h-1)$, $\mathfrak{g}_{k_h}(w) \in rem(Sat(\mathfrak{I}))$.

2) By induction on p, there exists $k_p \in \mathbb{N}$ such that for any word w satisfying $\mathcal{H}(p-1, h+n+1)$, $\mathfrak{g}_{k_p}(w) \in rem(Sat(\mathfrak{I}))$.
3) We can use Proposition 24 for k_p and we obtain $k_3 \in \mathbb{N}$ such that for all $L \subseteq A^*$, if L is closed under factors and $\mathfrak{g}_{k_p}(u) \in rem(Sat(\mathfrak{I}))$ for any $u \in L$, then for all $v \in A^*$, $\mathfrak{g}^L_{k_2,k_3}(v) \in Sat(\mathfrak{I})$.

For the remainder of the proof, we will use k_h, k_p and k_3 to denote these naturals. Finally, we define:

$$\begin{aligned} k' &= max(k_h, k_p, k_2, k_3) \\ k &= 3n + k' \end{aligned}$$

Note that in order to define k_p and k_3, we implicitly assumed that $p > 0$. When $p = 0$, all factorization forests we consider have binary nodes only and setting $k' = k_h$ suffices. We now prove that for any $w \in A^*$ that satisfies $\mathcal{H}(p, h)$, we have $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$. The proof works by decomposing w into factors according to the γ-factorization forest given by $\mathcal{H}(p, h)$. We will then be able to treat these factors using our choice of k. The decomposition is made with the following decomposition lemma.

Lemma 27 (Decomposition Lemma)**.** *Let $w_1, w_2 \in A^*$ and $k \geqslant 1$, then $\mathfrak{g}_k(w_1 w_2) \sqsubseteq \mathfrak{g}_{k-1}(w_1) \cdot \mathfrak{g}_{k-1}(w_2)$.*

Proof: Set $(s, S) = \mathfrak{g}_k(w_1 w_2)$, $(s_1, S_1) = \mathfrak{g}_{k-1}(w_1)$ and $(s_2, S_2) = \mathfrak{g}_{k-1}(w_2)$. By definition, we have $s = \alpha(w_1 w_2) = s_1 s_2$. We need to prove that $S \subseteq S_1 S_2$. Set $t \in S$, by definition there exists $v \in A^*$ such that $\alpha(v) = t$ and $w_1 w_2 \lesssim^k_3 v$. Using Lemma 5, we obtain that $v = v_1 v_2$ with $w_1 \lesssim^{k-1}_3 v_1$ and $w_2 \lesssim^{k-1}_3 v_2$. It follows that $\alpha(v_1) \in S_1$ and $\alpha(v_2) \in S_2$. Hence, $t = \alpha(v_1 v_2) \in S_1 S_2$ which terminates the proof. ∎

After having decomposed $\mathfrak{g}_k(w)$ with Lemma 27, we will need to prove two things:

1) That some individual factors are in $rem(Sat(\mathfrak{I}))$.
2) That the composition of the factors is in $rem(Sat(\mathfrak{I}))$.

The first item will always be obtained by choice of k. The second item will be obtained from the first one by using downset, multiplication and Σ_3 closure. In particular, we will often use the following fact which is a simple combination of multiplication and downset closure.

Fact 28. *Set $\mathfrak{t}, \mathfrak{t}' \in rem(Sat(\mathfrak{I}))$ and $\mathfrak{s} \sqsubseteq \mathfrak{t} \cdot \mathfrak{t}'$. Then $\mathfrak{s} \in rem(Sat(\mathfrak{I}))$.*

Proof: From Fact 15, we know that $rem(Sat(\mathfrak{I})) \cdot rem(Sat(\mathfrak{I})) = rem(Sat(\mathfrak{I}) \cdot Sat(\mathfrak{I}))$. Hence, we have $\mathfrak{t} \cdot \mathfrak{t}' \in rem(Sat(\mathfrak{I}) \cdot Sat(\mathfrak{I}))$ and it follows that $\mathfrak{t} \cdot \mathfrak{t}' \in rem(Sat(\mathfrak{I}))$ by multiplication closure. Finally, that $\mathfrak{s} \in rem(Sat(\mathfrak{I}))$ follows by downset closure. ∎

We will also need the following composition lemma.

Lemma 29 (Composition Lemma)**.** *Let $w_1, w_2 \in A^*$ and $k \geqslant 0$, then $\mathfrak{g}_k(w_1) \cdot \mathfrak{g}_k(w_2) \sqsubseteq \mathfrak{g}_k(w_1 w_2)$.*

Proof: Set $(s, S) = \mathfrak{g}_k(w_1 w_2)$, $(s_1, S_1) = \mathfrak{g}_k(w_1)$ and $(s_2, S_2) = \mathfrak{g}_{k-1}(w)$. By definition, we have $s = s_1 s_2$, we now prove that $S_1 S_2 \subseteq S$. Let $s_1 \in S_1$ and $s_2 \in S_2$, we obtain $v_1, v_2 \in A^*$ such that $\alpha(v_1) = s_1$, $\alpha(v_2) = s_2$, $w_1 \lesssim^k_3 v_1$ and $w_2 \lesssim^k_3 v_2$. Hence, it follows from Lemma 6 that $w_1 w_2 \lesssim^k_3 v_1 v_2$ and $s_1 s_2 \in S$. ∎

We can now start the proof. Recall that we want to prove that $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$. If the γ-factorization forest of w given by $\mathcal{H}(p, h)$ is a leaf, we conclude using the same argument as in the case $h = 0$ above. Otherwise, we consider two cases depending on whether the topmost node in the forest is binary or idempotent.

A. First Case : Binary Node

By hypothesis, w can be decomposed as $w = w_1 w_2$ with w_1, w_2 satisfying $\mathcal{H}(p, h-1)$. By choice of k_h, we know that $\mathfrak{g}_{k_h}(w_1) \in rem(Sat(\mathfrak{I}))$ and $\mathfrak{g}_{k_h}(w_2) \in rem(Sat(\mathfrak{I}))$.

We know from Lemma 27 that $\mathfrak{g}_k(w) \sqsubseteq \mathfrak{g}_{k-1}(w_1) \cdot \mathfrak{g}_{k-1}(w_2)$. Moreover, since by definition $k-1 \geqslant k_h$, it follows from Fact 23 that $\mathfrak{g}_k(w) \sqsubseteq \mathfrak{g}_{k_h}(w_1) \cdot \mathfrak{g}_{k_h}(w_2)$. It then follows from Fact 28 that $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$.

B. Second Case: Idempotent Node

Set $e = \alpha(w)$. Recall that $\gamma(w)$ is by hypothesis an idempotent, hence e and $[w]_{\cong^{k_2}_2}$ are idempotents as well. We first detail our hypothesis: w admits a $\gamma(w)$*-decomposition*.

$\gamma(w)$-Decompositions. Given any $u \in A^*$, we say that u admits a $\gamma(w)$*-decomposition* $u_1, \ldots, u_m$ if

(a) $u = u_1 \cdots u_m$,
(b) $\gamma(u_1) = \cdots = \gamma(u_m) = \gamma(w)$.
(c) for all i, u_i satisfies $\mathcal{H}(p-1, h-1)$.

Note that since $\gamma(w)$ is idempotent, Item b means that for all $i \leqslant j$, $\alpha(u_i \cdots u_j) = e$ and $u_i \cdots u_j \cong^{k_2}_2 w$. By hypothesis of this case, we have the following fact.

Fact 30. *w admits a $\gamma(w)$-decomposition $w_1, \ldots, w_m$.*

Our objective is to prove that $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$. Note that, in general, the number of factors m in the $\gamma(w)$-decomposition of w can be arbitrarily large. This means that we cannot use an argument similar to the previous case. Indeed, using Lemma 27, we could obtain that $\mathfrak{g}_k(w) \sqsubseteq \mathfrak{g}_{k-m}(w_1) \cdots \mathfrak{g}_{k-m}(w_m)$. However, since m can be arbitrarily large, we have no guarantee that $k - m$ is large enough to obtain $\mathfrak{g}_{k-m}(w_i) \in rem(Sat(\mathfrak{I}))$ (or even that $k - m \geqslant 0$).

Instead, we partition $w_1 \cdots w_m$ as a bounded number of subdecompositions that we can treat using the Σ_3 closure. The partition is defined by induction on a parameter of the $\gamma(w)$-decomposition that we define now.

Index of a $\gamma(w)$-decomposition. Recall that we defined $k' = max(k_h, k_p, k_2, k_3)$ (i.e. $k' = k - 3n$). Let $u \in A^*$ that admits a $\gamma(w)$-decomposition $u_1, \ldots, u_m$. Let $(f, F) \in \mathfrak{M}(1)$ be an idempotent and $i \leqslant m$, we say that (f, F) can be *inserted* at position i if there exists $1 \leqslant j \leqslant i$ such that $i - (j-1) \leqslant n$ and,

$$\mathfrak{g}_{k'}(u_j) \cdots \mathfrak{g}_{k'}(u_i) = \mathfrak{g}_{k'}(u_j) \cdots \mathfrak{g}_{k'}(u_i) \cdot (f, F)$$

The index of a $\gamma(w)$-decomposition $u_1, \ldots, u_m$ is the number of distinct idempotents $(f, F) \in \mathfrak{M}(1)$ that can be inserted at

some position $i \leqslant m$. Observe that by definition, the index of any $\gamma(w)$-decomposition is bounded by $n = |\mathfrak{M}(1)|$.

Lemma 31. *Let $u \in A^*$ admitting a $\gamma(w)$-decomposition $u_1, \ldots, u_m$ of index q and set $\hat{k} \geqslant q + 2n + k'$. Then $\mathfrak{g}_{\hat{k}}(u) \in rem(Sat(\mathfrak{I}))$.*

Before proving this lemma, we use it to conclude the idempotent case. We know that our $\gamma(w)$-decomposition $w_1, \ldots, w_m$ of w has an index $q \leqslant n$. Moreover, we know that $k = 3n + k' \geqslant q + 2n + k'$, hence, it is immediate from Lemma 31 that $\mathfrak{g}_k(w) \in rem(Sat(\mathfrak{I}))$. It now remains to prove Lemma 31.

Proof: We begin by treating the case when m is "small" in front of $\hat{k}$. We use an argument that is similar to the binary node case. In the main proof (which uses induction on q), we will use this special case twice.

Lemma 32. *Let $l \geqslant m + k_p$, the $\mathfrak{g}_l(u) \in rem(Sat(\mathfrak{I}))$.*

Proof: By hypothesis, $l - (m-1) > k_p$. Hence, using Lemma 27 $m-1$ times and Fact 23, we obtain that,

$$\mathfrak{g}_l(u) \sqsubseteq \mathfrak{g}_{k_p}(u_1) \cdots \mathfrak{g}_{k_p}(u_m)$$

Since by hypothesis all factors u_i satisfy $\mathcal{H}(p-1, h-1)$ (and therefore $\mathcal{H}(p-1, h+n+1)$), we obtain by choice of k_p that for all i, $\mathfrak{g}_{k_p}(u_i) \in rem(Sat(\mathfrak{I}))$. It then follows from Fact 28 that $\mathfrak{g}_l(u) \in rem(Sat(\mathfrak{I}))$. ∎

We can now start the main proof. We proceed by induction on q. The induction base and the induction step are both based on the following lemma which states that if m is large enough, we know that an idempotent can be inserted at position $i \leqslant n$.

Lemma 33. *Assume that $m > n$. There exists $i < i' \leqslant n+1$ such that,*

$$\mathfrak{g}_{k'}(u_1) \cdots \mathfrak{g}_{k'}(u_i) = \mathfrak{g}_{k'}(u_1) \cdots \mathfrak{g}_{k'}(u_i) \cdot (e, E)$$

with $(e, E) = (\mathfrak{g}_{k'}(u_{i+1}) \cdots \mathfrak{g}_{k'}(u_{i'}))^\omega$.

Proof: We use the pigeon-hole principle. By definition of n, there exists $i < i' \leqslant n+1$ such that $\mathfrak{g}_{k'}(u_1) \cdots \mathfrak{g}_{k'}(u_i) = \mathfrak{g}_{k'}(u_1) \cdots \mathfrak{g}_{k'}(u_{i'})$. The result then follows. ∎

We can now start the induction. In the base case, $q = 0$ and no idempotent can be inserted. It follows from Lemma 33 that $n \geqslant m$ and therefore that $\hat{k} \geqslant m + k_p$. We conclude that $\mathfrak{g}_{\hat{k}}(u) \in rem(Sat(\mathfrak{I}))$ using Lemma 32.

Assume now that $q > 0$, if $n \geqslant m$, we can again conclude using Lemma 32. Otherwise, we get $i < i' \leqslant n+1$ and an idempotent $(e, E) \in \mathfrak{M}(1)$ as defined in Lemma 33. Set $j \leqslant m$ as the largest number such that (e, E) can be inserted at position j in the $\gamma(w)$-decomposition of u. Observe that j has to exist (and $i \leqslant j$) since (e, E) can be inserted at position i. Using Lemma 27, we obtain that,

$$\mathfrak{g}_{\hat{k}}(u) \sqsubseteq \mathfrak{g}_{\hat{k}-1}(u_1 \cdots u_j) \cdot \mathfrak{g}_{\hat{k}-1}(u_{j+1} \cdots u_m)$$

Observe that $u_{j+1}, \ldots, u_m$ is a $\gamma(w)$-decomposition whose index is smaller than that of $u_1, \ldots, u_m$ (all idempotents that can be inserted in $u_{j+1}, \ldots, u_m$ can be inserted in $u_1, \ldots, u_m$ and by choice of j, (e, E) cannot be inserted in $u_{j+1}, \ldots, u_m$). Hence, using induction in Lemma 31, we conclude that $\mathfrak{g}_{\hat{k}-1}(u_{j+1} \cdots u_m) \in rem(Sat(\mathfrak{I}))$.

We now prove that $\mathfrak{g}_{\hat{k}-1}(u_1 \cdots u_j) \in rem(Sat(\mathfrak{I}))$. In view of Fact 28, this will suffice to terminate the proof. We define $j' = j - (n-1)$. Observe that when $j' \leqslant i$, since $i \leqslant n$, we have $j \leqslant 2n-1$ and $\hat{k} - 1 \geqslant j + k_p$. Hence, that $\mathfrak{g}_{\hat{k}-1}(u_1 \cdots u_j) \in rem(Sat(\mathfrak{I}))$ is a consequence of Lemma 32. Assume now that $j' > i$. We consider the following objects:

$$\begin{array}{rcl} (e, T) & = & \mathfrak{g}_{k'}(u_1) \cdots \mathfrak{g}_{k'}(u_i) \\ (e, S) & = & \mathfrak{g}_{k'}(u_{i+1} \cdots u_{j'-1}) \cdot \mathfrak{g}_{k'}(u_{j'}) \cdots \mathfrak{g}_{k'}(u_j) \end{array}$$

Recall that $\hat{k} \geqslant q + 2n + k'$, in particular $\hat{k} - 1 \geqslant 2n + k'$. Therefore, using Lemma 27 several times (at most $2n$ times since $i \leqslant n$ and $j - j' = n - 1$) and Fact 23 one can verify that:

$$\mathfrak{g}_{\hat{k}-1}(u_1 \cdots u_j) \sqsubseteq (e, T) \cdot (e, S)$$

Moreover, by definition of the positions i, j and of the idempotent (e, E) , we have $(e, T) \cdot (e, E) = (e, T)$ and $(e, S) \cdot (e, E) = (e, S)$. Hence we obtain,

$$\mathfrak{g}_{\hat{k}-1}(u_1 \cdots u_j) \sqsubseteq (e, T) \cdot (e, E) \cdot (e, S) \cdot (e, E)$$

Observe that since $k' \geqslant k_p$, it follows from Fact 23 that $(e, T) \sqsubseteq \mathfrak{g}_{k_p}(u_1) \cdots \mathfrak{g}_{k_p}(u_i)$. Hence, it follows from our choice of k_p and Fact 28 that $(e, T) \in rem(Sat(\mathfrak{I}))$. By Fact 28, it now remains to prove that $(e, E) \cdot (e, S) \cdot (e, E) \in rem(Sat(\mathfrak{I}))$. We define,

$$\begin{array}{rcl} (e, S') & = & \mathfrak{g}_{k'}(u_{i+1} \cdots u_j) \\ (e, R) & = & \mathfrak{g}_{k'}(u_{i+1} \cdots u_{i'}) \end{array}$$

It is immediate from Lemma 29 that $(e, S) \sqsubseteq (e, S')$, and $(e, E) = (\mathfrak{g}_{k'}(u_{i+1}) \cdots \mathfrak{g}_{k'}(u_{i'}))^\omega \sqsubseteq (e, R)^\omega$. Hence, we have

$$(e, E) \cdot (e, S) \cdot (e, E) \sqsubseteq (e, R)^\omega \cdot (e, S') \cdot (e, R)^\omega$$

By downset closure, it then suffices to prove that $(e, R)^\omega \cdot (e, S') \cdot (e, R)^\omega \in rem(Sat(\mathfrak{I}))$. Set an (arbitrary) order on the elements of S' and observe that

$$(e, R)^\omega \cdot (e, S') \cdot (e, R)^\omega \sqsubseteq \prod_{s \in S'} (e, R)^\omega \cdot (1_M, \{1_M, s\}) \cdot (e, R)^\omega$$

where the product indexed by S' is made in the order that we chose. Therefore, that $(e, R)^\omega \cdot (e, S') \cdot (e, R)^\omega \in rem(Sat(\mathfrak{I}))$ follows from Fact 28 and the next lemma.

Lemma 34. *$(e, R)^\omega \cdot (1_M, \{1_M, s\}) \cdot (e, R)^\omega \in rem(Sat(\mathfrak{I}))$ for any $s \in S'$.*

We finish with the proof of Lemma 34. We prove that for all $s \in S'$, we have $(s, \{(e, R)\}) \in Sat(\mathfrak{I})$. Using Σ_3 closure, we will then be able to conclude that:

$$dup((e, R)^\omega \cdot (1_M, \{1_M, s\}) \cdot (e, R)^\omega) \in Sat(\mathfrak{I})$$

By definition of root duplication and removal, it will then follow that $(e, R)^\omega \cdot (1_M, \{1_M, s\}) \cdot (e, R)^\omega \in rem(Sat(\mathfrak{I}))$, finishing the proof.

We now prove that for any $s \in S'$, $(s, \{(e, R)\}) \in Sat(\mathfrak{I})$. This is a consequence of our choice of k_2 and k_3 as given by Proposition 24. We prove that $(s, \{(e, R)\}) \sqsubseteq \mathfrak{g}^L_{k_2,k_3}(v)$ for some $v \in A^*$ and a L satisfying the appropriate conditions. Recall that (e, R) and (e, S') are defined as follows,

$$\begin{array}{rcl}(e, S') & = & \mathfrak{g}_{k'}(u_{i+1} \cdots u_j) \\ (e, R) & = & \mathfrak{g}_{k'}(u_{i+1} \cdots u_{i'})\end{array}$$

Hence, since $s \in S'$, there exists $v \in A^*$ such that $\alpha(v) = s$ and $u_{i+1} \cdots u_j \lesssim_3^{k'} v$. Since $k_2 \leqslant k'$, we can use the first item in Fact 3 to obtain that $u_{i+1} \cdots u_j \lesssim_3^{k_2} v$. Moreover, using the second item in Fact 3, we get that $v \lesssim_2^{k_2} u_{i+1} \cdots u_j$.

Furthermore, since $u_1, \ldots, u_m$ is a $\gamma(w)$-decomposition, we have $\gamma(u_{i+1} \cdots u_j) = \gamma(u_{i+1} \cdots u_{i'})$. Therefore, we have $u_{i+1} \cdots u_j \cong_2^{k_2} u_{i+1} \cdots u_{i'}$. By combining this with $v \lesssim_2^{k_2} u_{i+1} \cdots u_j$, we obtain that $v \lesssim_2^{k_2} u_{i+1} \cdots u_{i'}$.

Define L as the language of all factors of $u_{i+1} \cdots u_{i'}$. Since $u_{i+1} \cdots u_{i'} \in L$ and $v \lesssim_2^{k_2} u_{i+1} \cdots u_{i'}$, we get,

$$(s, \{(e, R)\}) = (s, \{\mathfrak{g}_{k'}(u_{i+1} \cdots u_{i'})\}) \sqsubseteq \mathfrak{g}^L_{k_2,k'}(v)$$

Finally, since $k_3 \leqslant k'$, it follows from Fact 23 that $\mathfrak{g}^L_{k_2,k'}(v) \sqsubseteq \mathfrak{g}^L_{k_2,k_3}(v)$ and therefore that $(s, \{(e, R)\}) \sqsubseteq \mathfrak{g}^L_{k_2,k_3}(v)$. By downset closure it now suffices to prove that $\mathfrak{g}^L_{k_2,k_3}(v) \in Sat(\mathfrak{I})$ to conclude that $(s, \{(e, R)\}) \in Sat(\mathfrak{I})$ and terminate the proof.

Recall that we used Proposition 24 to choose k_2, k_3 so that if L is closed under factors and $\mathfrak{g}_{k_p}(u) \in rem(Sat(\mathfrak{I}))$ for any $u \in L$, then for any $v \in A^*$, $\mathfrak{g}^L_{k_2,k_3}(v) \in Sat(\mathfrak{I})$. Since our language L of factors of $u_{i+1} \cdots u_{i'}$ is closed under factors by definition, we only have to prove that for any $u \in L$, $\mathfrak{g}_{k_p}(u) \in rem(Sat(\mathfrak{I}))$. By choice of k_p, this amounts to proving that any $u \in L$ satisfies $\mathcal{H}(p-1, h+n+1)$. This is what we do now.

By hypothesis on $\gamma(w)$-decompositions all words among $u_{i+1}, \ldots, u_{i'}$ satisfy $\mathcal{H}(p-1, h-1)$: they admit a γ-factorization forest of height at most $h-1$ and idempotent height at most $p-1$. Since $i' - i \leqslant n$, one can combine these forests into a single forest for $u_{i+1} \cdots u_{i'}$ by adding at most n binary nodes. The resulting forest has height at most $h-1+n$ and idempotent height at most $p-1$. By Lemma 11, this means that any factor of $u_{i+1} \cdots u_{i'}$ (i.e. any word of L) admits a γ-factorization forest of height at most $h+n+1$ and idempotent height at most $p-1$. Hence any word of L satisfies $\mathcal{H}(p-1, h+n+1)$ which terminates the proof. ∎

VIII. Conclusion

We proved that separation is decidable for Σ_3 and Π_3. It is known [PZ14a] that this also yields decidability of the membership problem for Σ_4 and Π_4. Another interesting consequence is that these results can be lifted [PZ15] to the variants of these logics whose signature has been enriched with the successor relation: separation is decidable for $\Sigma_3(<, +1)$ and $\Pi_3(<, +1)$ and membership is decidable for $\Sigma_4(<, +1)$ and $\Pi_4(<, +1)$.

In order to obtain these results, we defined new objects (Σ-trees) that are much more involved than the machinery necessary for solving Σ_2-separability. The most immediate question is whether this new machinery can be reused for higher levels. Unfortunately, the answer appears to be negative: while Σ-trees remain relevant, Σ_4-separability seems to require new ideas and more general objects.

Another question is $\mathcal{B}\Sigma_3$. As explained in the introduction, it was proven in [PZ14a] that for all i, membership for $\mathcal{B}\Sigma_i$ as well is linked to separation for Σ_i. However, the link is not as strong as the one with Σ_{i+1} and no formal reduction is known. In particular, while this link is used in [PZ14a] to obtain a $\mathcal{B}\Sigma_2$-membership algorithm, the proof technique is dependent on the inner workings of the Σ_2-separation algorithm. Again, whether this can be replicated for $\mathcal{B}\Sigma_3$ is not clear. However, the link with Σ_3-separability remains and it would be interesting to know what can be obtained from our algorithm.

References

[AK10] J. Almeida and O. Klíma. New decidable upper bound of the 2nd level in the Straubing-Thérien concatenation hierarchy of star-free languages. *Disc. Math. & Theo. Comp. Sci.*, 2010.

[Arf87] M. Arfi. Polynomial operations on rational languages. In *STACS'87*, 1987.

[BK78] J. Brzozowski and R. Knast. The dot-depth hierarchy of star-free languages is infinite. *J. Comp. Syst. Sci.*, 1978.

[Boj09] M. Bojańczyk. Factorization forests. In *DLT'09*, 2009.

[CMM13] W. Czerwiński, W. Martens, and T. Masopust. Efficient separability of regular languages by subsequences and suffixes. In *ICALP'13*, 2013.

[DG08] V. Diekert and P. Gastin. First-order definable languages. In *Logic and Automata: History and Perspectives*, volume 2. Amsterdam Univ. Press, 2008.

[GS00] C. Glaßer and H. Schmitz. Languages of dot-depth 3/2. In *(STACS'00)*, 2000.

[Kna83] R. Knast. A semigroup characterization of dot-depth one languages. *RAIRO*, 1983.

[Kuf08] M. Kufleitner. The height of factorization forests. In *MFCS'08*, 2008.

[MP71] R. McNaughton and S. Papert. *Counter-Free Automata*. 1971.

[Pin98] J.E. Pin. Bridges for concatenation hierarchies. In *ICALP'98*, 1998.

[Pin11] J.E. Pin. Theme and variations on the concatenation product. In *CAI'11*, 2011.

[PRZ13] T. Place, L. Rooijen, and M. Zeitoun. Separating regular languages by piecewise testable and unambiguous languages. In *MFCS'13*, 2013.

[PW97] J.E. Pin and P. Weil. Polynomial closure and unambiguous product. *Theory Comp. Syst.*, 1997.

[PZ14a] T. Place and M. Zeitoun. Going higher in the first-order quantifier alternation hierarchy on words. In *ICALP'14*, 2014.

[PZ14b] T. Place and M. Zeitoun. Separating regular languages with first-order logic. In *CSL-LICS'14*, 2014.

[PZ15] T. Place and M. Zeitoun. Separation and the successor relation. In *STACS'15*, 2015.

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

[Sim75] I. Simon. Piecewise testable events. In *2nd GI Conference on Automata Theory and Formal Languages*, 1975.

[Sim90] I. Simon. Factorization forests of finite height. *TCS*, 1990.

[Str94] Howard Straubing. *Finite Automata, Formal Logic and Circuit Complexity*. 1994.

[Tho82] W. Thomas. Classifying regular events in symbolic logic. *J. Comp. Syst. Sci.*, 1982.

APPENDIX A
APPENDIX TO SECTION III: EHRENFEUCHT-FRAÏSSÉ GAMES

In this appendix, we prove Lemma 5, Lemma 6, Lemma 7 and Lemma 8. The proofs are all based on Ehrenfeucht-Fraïssé arguments.

We divide the appendix in two parts. First we recall the definition of Ehrenfeucht-Fraïssé games. Then we use them to prove the lemmas.

A. Ehrenfeucht-Fraïssé games

It is well known that the preorders $\lesssim_i^k$ can be given a game definition. Fix $i \in \mathbb{N}$, we define the game associated to $\lesssim_i^k$. We call this game the Σ_i *game*.

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*. The counter c is initially set to 0 and remains smaller or equal to i as the game progresses.

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 $j_1, j_2 \leqslant j$, the labels at positions x_{j_1}, x'_{j_1} are the same, and $x_{j_1} < x_{j_2}$ iff $x'_{j_1} < x'_{j_2}$.

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

Lemma 35 (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 Σ_i game played on w, w' with w as the initial active word.*

B. Proof of the Lemmas

We begin with the Decomposition and Composition Lemmas that we restate below.

Lemma 5 (Decomposition Lemma)**.** *Set $k, i \geqslant 1$ and $u_1, u_2, v \in A^*$ such that $u_1 u_2 \lesssim_i^k v$. Then v can be decomposed as $v = v_1 v_2$ with $u_1 \lesssim_i^{k-1} v_1$ and $u_2 \lesssim_i^{k-1} v_2$.*

Proof: By hypothesis Duplicator has a winning strategy in the Σ_i game played on $u_1 u_2, v$ with $u_1 u_2$ as initial active word. Assume Spoiler begins by placing a pebble on the leftmost position of u_2 and let x be the position in v given as answer by Duplicator's winning strategy. We decompose v as $v = v_1 v_2$ such that x is the leftmost position in v_2. Since Duplicator has a strategy for playing $k-1$ more rounds, it follows that $u_1 \lesssim_i^{k-1} v_1$ and $u_2 \lesssim_i^{k-1} v_2$. ∎

Lemma 6 (Composition Lemma)**.** *Set $k, i \geqslant 1$ and $u_1, u_2, v_1, v_2 \in A^*$ such that $u_1 \lesssim_i^k v_1$ and $u_2 \lesssim_i^k v_2$. Then, we have $u_1 v_1 \lesssim_i^k u_2 v_2$.*

Proof: By Lemma 35, Duplicator has a winning strategy for playing k rounds in the Σ_i game played on u_1, v_1 with u_1 as the initial active word and in the Σ_i game played on u_2, v_2 with u_2 as the initial active word. These strategies can easily be combined in a single strategy for playing k rounds in the Σ_i game played on $u_1 u_2, v_1 v_2$ with $u_1 u_2$ as the initial active word which terminates the proof. ∎

We now turn to Lemma 7. We will need the following intermediate lemma which states a well-known property of full first-order logic.

Lemma 36. *Let $k, k_1, k_2 \in \mathbb{N}$ be such that $k_1, k_2 \geqslant 2^k - 1$ and $u \in A^*$. Then, for all $i \geqslant 1$:*

$$u^{k_1} \lesssim_i^k u^{k_2}$$

Proof: This is well known for full first-order logic: it can be proven that any FO formula of rank k that is satisfied by u^{k_1} is satisfied by u^{k_2} as well (see [Str94] for details). ∎

We can now prove Lemma 7. We restate the lemma below.

Lemma 7. *Set $k, i \in \mathbb{N}$ and $u, v, u_\ell, u_r \in A^*$ such that $u \lesssim_{i+1}^k u_\ell$, $u \lesssim_{i+1}^k u_r$ and $v \lesssim_i^k u$. Then, for $k' \geqslant 2^k$, we have,*

$$u^{2k'} \lesssim_{i+1}^k (u_\ell)^{k'} v (u_r)^{k'}$$

Proof: We prove a slightly more general result as stated in the following claim:

Claim. *Set $k, i \in \mathbb{N}$, $u, v \in A^*$ and $\ell, \ell', r, r' \geqslant 2^k$ such that $v \lesssim_i^k u$. Then,*

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

It is immediate from the claim that under the conditions of Lemma 7, we have $u^{2k'} \lesssim_{i+1}^k (u)^{k'} v (u)^{k'}$. Moreover, by Lemma 6, we have $(u)^{k'} v (u)^{k'} \lesssim_{i+1}^k (u_\ell)^{k'} v (u_r)^{k'}$. Hence it follows by transitivity that $u^{2k'} \lesssim_{i+1}^k (u_\ell)^{k'} v (u_r)^{k'}$ which proves Lemma 7.

It now remains to prove the claim. Set $k, i \in \mathbb{N}$, $u, v \in A^*$ and $\ell, \ell', r, r' \geqslant 2^k$ such that $v \lesssim_i^k u$. We prove that Duplicator has a winning strategy in the k-rounds Σ_{i+1} game played on $u^{\ell+r}$ and $(u)^{\ell'} v (u)^{r'}$ with $u^{\ell+r}$ as initial active word. The proof is by induction on k. We describe how Duplicator can play the first round, the strategy for the following rounds is then obtained by induction or from Lemma 36. We distinguish two cases depending on Spoiler's move.

Case 1: Spoiler plays in $(u)^{\ell'} v (u)^{r'}$. This means that the alternation counter c is increased to 1 and that the active word becomes $(u)^{\ell'} v (u)^{r'}$. We prove that $(u)^{\ell'} v (u)^{r'} \lesssim_i^k u^{\ell+r}$. By Lemma 35 this gives Duplicator a winning strategy for all remaining rounds.

Observe that by Lemma 36, we know that $(u)^{r'} \lesssim_i^k (u)^{r-1}$ and $(u)^{\ell'} \lesssim_i^k (u)^{\ell}$. Hence, since $v \lesssim_i^k u$ by hypothesis, it follows from Lemma 6 that $(u)^{\ell'} v (u)^{r'} \lesssim_i^k u^{\ell+r}$.

Case 2: Spoiler plays in $u^{\ell+r}$. Let x be the position of $u^{\ell+r}$ on which Spoiler puts his pebble. By definition x is is inside a copy of the word u. Since $u^{\ell+r}$ contains more than 2^{k+1} copies of u, by symmetry we can assume that there are at least 2^k copies of u 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 u inside $(u)^{\ell'} v (u)^{r'}$ and is at the same relative position inside this copy as x is in its own copy of u. Therefore, to fully define x', it only remains to define the copy of u in which we choose x'. Let n be the number of copies of u to the *left* of x in $u^{\ell+r}$, that is, x belongs to the $(n+1)$-th copy of u starting from the left of $u^{\ell+r}$. If $n < 2^{k-1} - 1$, then x' is chosen inside the $(n+1)$-th copy of u starting from the left of $(u)^{\ell'} v (u)^{r'}$. Otherwise, x' is chosen inside the 2^{k-1}-th copy of u starting from the left of $(u)^{\ell'} v (u)^{r'}$. Observe that these copies always exist and are inside $(u)^{\ell'}$, since $\ell' \geqslant 2^k$.

Set $u^{\ell+r} = w_p u w_q$ and $(u)^{\ell'} v (u)^{r'} = w'_p u w'_q$, with the two distinguished u factors being the copies containing the positions x, x'. By Lemma 35, 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 u. Therefore $w_p \lesssim_{i+1}^{k-1} w'_p$ follows Lemma 36. 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 $w_q \lesssim_{i+1}^{k-1} w'_q$. ∎

We finish with the proof of Lemma 8 that we restate below.

Lemma 8. *Set $k \in \mathbb{N}$ and u, v, u_ℓ, u_r such that $u \lesssim_2^k u_\ell$, $u \lesssim_2^k u_r$ and $\mathsf{alph}(v) = \mathsf{alph}(u)$. Then, for $k' \geqslant 2^{2k}$, we have,*

$$u^{2k'} \lesssim_2^k (u_\ell)^{k'} v (u_r)^{k'}$$

Proof: Using a simple Ehrenfeucht-Fraïssé argument, it can be verified that when $\mathsf{alph}(v) = \mathsf{alph}(u)$, we have $v \lesssim_1^k u^{2^k - 1}$. The lemma is then a simple consequence of Lemma 7. ∎

Appendix B
Appendix to Section III: Factorization Forests

In this appendix, we prove Lemma 11. Recall the statement of Lemma 11.

Lemma 11. *Let $w \in A^*$ that admits an α-factorization forest of height h and idempotent height p. Then any factor $u \in A^*$ of w admits an α-factorization forest of height at most $h+2$ and idempotent height at most p.*

Proof: The result is a consequence of the following claim which treats the special case when u is either a prefix or a suffix of w.

Claim. *Let $w \in A^*$ that admits an α-factorization forest of height h and idempotent height p. Then any prefix or suffix $u \in A^*$ of w admits an α-factorization forest of height at most $h+1$ and idempotent height at most p.*

Observe that a factor u of w is by definition the prefix of a suffix of w. Hence, using the claim twice, we obtain that any factor u of w admits an α-factorization forest of height at most $(h+1)+1 = h+2$ and idempotent height at most p which terminates the proof of Lemma 11

It now remains to prove the claim. We prove the prefix case (the suffix case can be proved using a symmetrical argument). Let w that admits an α-factorization forest of height h and idempotent height p and let u be a prefix of w. We construct an α-factorization forest for u by induction on the structure of the α-factorization forest of w.

If the topmost node in the forest of w is a leaf node, $w = a$ or $w = \varepsilon$. Hence $p = 0$ and $h = 1$. Since u is a prefix of w, we have $u = a$ or $u = \varepsilon$. Hence u admits an α-factorization forest of height $1 < 2 = h+1$ and idempotent height $0 = p$.

If the topmost node in the forest of w is a binary node, it has two children labeled with w_1, w_2 (admitting α-factorization forests of heights at most $h-1$ and idempotent heights at most p) and $w = w_1 w_2$. By definition, u is either a prefix of w_1 (in which case the result is immediate by induction) or there exists a prefix u' of w_2 such that $u = w_1 u'$. In the latter case, we know by induction hypothesis that u' admits an α-factorization forest of height at most $(h-1)+1 = h$ and idempotent height at most p. Using one binary node, one can then combine the forest of w_1 and the forest of u' into a forest for u. By definition, this new forest has height at most $h+1$ and idempotent height at most p.

Finally, if the topmost node is an idempotent node, its children are labeled with $w_1, \ldots, w_n$ (admitting α-factorization forests of heights at most $h-1$ and idempotent heights at most $p-1$), $w = w_1 \cdots w_n$ and $\alpha(w_1) = \cdots \alpha(w_n)$ is an idempotent $e \in M$. Set i as the smallest natural such that u is a prefix of $w_1 \cdots w_i$. If $i = 1$, u is a prefix of w_1 and the result is immediate by induction. Otherwise there exists a prefix u' of w_i such that $u = w_1 \cdots w_{i-1} u'$. We know that,

- u' admits an α-factorization forest of height at most $(h-1)+1 = h$ and idempotent height at most $p-1$ (this is by induction).
- $w_1 \cdots w_{i-1}$ admits an α-factorization forest of height at most h and idempotent height at most p (whose topmost node is an idempotent node with children labeled with $w_1, \ldots, w_{i-1}$).

These two forests can be combined into a single forest for u with one binary node. By definition, this forest has height at most $h+1$ and idempotent height at most p. ∎

Appendix C
Appendix to Section VII: Proof of Proposition 24

In this appendix, we terminate the completeness proof of Proposition 20 that we began in Section VII. It remained to prove Proposition 24.

Recall that an alphabet compatible morphism $\alpha : A^* \to M$ into a finite monoid M is fixed. Let us restate Proposition 24.

Proposition 24 (Σ_2 Level)**.** *There exists $k_2 \in \mathbb{N}$ such that for all $k \in \mathbb{N}$, there exists $k_3 \in \mathbb{N}$ such that for all $L \subseteq A^*$, if L is closed under factors and $\mathfrak{g}_k(u) \in rem(Sat(\mathfrak{I}))$ for any $u \in L$, then for all $w \in A^*$, $\mathfrak{g}^L_{k_2,k_3}(w) \in Sat(\mathfrak{I})$.*

As for Proposition 25 we prove Proposition 24 by relying on a generalized proposition. Note that in this case it suffices to work directly with the morphism α.

Proposition 37. *For all $h \in \mathbb{N}$, there exists $k_2 \in \mathbb{N}$ such that for all $k \in \mathbb{N}$, there exists $k_3 \in \mathbb{N}$ such that for all $L \subseteq A^*$, if L is closed under factors and $\mathfrak{g}_k(u) \in rem(Sat(\mathfrak{I}))$ for any $u \in L$, then for all $w \in A^*$ admitting an α-factorization forest of height at most h, $\mathfrak{g}^L_{k_2,k_3}(w) \in Sat(\mathfrak{I})$.*

By Theorem 10 any word $w \in A^*$ admits an α-factorization forest of height at most $3|M| - 1$. Hence Proposition 24 is an immediate consequence of Proposition 37. It now remains to prove Proposition 37. We proceed by induction on h. The proof structure is similar to that of Proposition 26. However, because this result is tied to the Σ_2 level, we are able to simplify the argument in many places (in particular, observe that the idempotent height is no longer used as an induction parameter). The main difference occurs in the idempotent case when proving the result corresponding to Lemma 31.

In the base case $h = 0$, we choose $k_2 = 2$. For any $k \in \mathbb{N}$, we choose $k_3 = 2$ as well. Any word w admitting an α-factorization forest of height at most $h = 0$ is either single letter $a \in A$ or the empty word. One can verify that since $k_2 = k_3 = 2$, for any $u, v \in A^*$, $w \lesssim_2^{k_2} u \lesssim_3^{k_3} v \Rightarrow w = u = v$. Hence, for any $L \in A^*$ either $w \notin L$ and $\mathfrak{g}^L_{k_2,k_3}(w) = (\alpha(w), \emptyset) \in \downarrow\mathfrak{I}$ or $w \in L$ and $\mathfrak{g}^L_{k_2,k_3}(w) = (\alpha(w), \{(\alpha(w), \{\alpha(w)\})\}) \in \mathfrak{I}$.

This terminates the induction base. Assume now that $h \geqslant 1$. For the remainder of the proof, we will denote by n, the size of the set of M-trees of height 2: $n = |\mathfrak{M}(2)|$. We first need to choose $k_2 \in \mathbb{N}$. For this, we use our induction hypothesis which we state in the following fact.

Fact 38 (Induction Hypothesis)**.** *There exists k'_2 such that for all $k \in \mathbb{N}$ there exists $k'_3 \in \mathbb{N}$ such that for all $L \subseteq A^*$, if L is closed under factors and $\mathfrak{g}_k(u) \in rem(Sat(\mathfrak{I}))$ for any $u \in L$, then for all $w \in A^*$ admitting an α-factorization forest of height at most $h - 1$, $\mathfrak{g}^L_{k'_2,k'_3}(w) \in Sat(\mathfrak{I})$.*

We choose $k_2 = k'_2 + 3n$ where k'_2 is as defined in Fact 38. Set $k \in \mathbb{N}$, we now need to choose $k_3 \in \mathbb{N}$. Let k'_3 be as defined in Fact 38 for the natural k we just defined. We choose $k_3 = max(k, k'_3) + 3n$.

Finally, set $L \subseteq A^*$ closed under factors and such that $\mathfrak{g}_k(u) \in rem(Sat(\mathfrak{I}))$ for all $u \in L$. We fix $w \in A^*$ that admits an α-factorization forest of height at most h. We need to prove that

$$\mathfrak{g}^L_{k_2,k_3}(w) \in Sat(\mathfrak{I})$$

As we did in the proof of Proposition 26, the proof decomposes w according to its α-factorization forest. To do so we use the following decomposition lemma (note that the proof of this lemma is where we need L to be closed under factors).

Lemma 39 (Decomposition Lemma)**.** *Let $L \subseteq A^*$ that is closed under factors, $w_1, w_2 \in A^*$ and $k_2, k_3 \geqslant 1$, then*

$$\mathfrak{g}^L_{k_2,k_3}(w_1 w_2) \sqsubseteq \mathfrak{g}^L_{k_2-1,k_3-1}(w_1) \cdot \mathfrak{g}^L_{k_2-1,k_3-1}(w_2)$$

Proof: We define $(s, \mathfrak{S}) = \mathfrak{g}^L_{k_2,k_3}(w_1 w_2)$, $(s_1, \mathfrak{S}_1) = \mathfrak{g}^L_{k_2-1,k_3-1}(w_1)$, $(s_2, \mathfrak{S}_2) = \mathfrak{g}^L_{k_2-1,k_3-1}(w_2)$. By definition, $s = \alpha(w_1 w_2) = s_1 s_2$. Set $\mathfrak{s} \in \mathfrak{S}$, we need to find $\mathfrak{s}' \in \mathfrak{S}_1 \cdot \mathfrak{S}_2$ such that $\mathfrak{s} \sqsubseteq \mathfrak{s}'$. By definition, there exists $u \in L$ such that $\mathfrak{s} = \mathfrak{g}_{k_3}(u)$ and $w_1 w_2 \lesssim_3^{k_2} u$. Using Lemma 5, we obtain that $u = u_1 u_2$ with $w_1 \lesssim_3^{k_2-1} u_1$ and $w_2 \lesssim_3^{k_2-1} u_2$. Set $\mathfrak{s}' = \mathfrak{g}_{k_3-1}(u_1) \cdot \mathfrak{g}_{k_3-1}(u_2)$.

Observe that since L is assumed to be closed under factors, $u_1, u_2 \in L$. Hence, we have $\mathfrak{g}_{k_3-1}(u_1) \in \mathfrak{S}_1$, $\mathfrak{g}_{k_3-1}(u_2) \in \mathfrak{S}_2$ and $\mathfrak{s}' \in \mathfrak{S}_1 \cdot \mathfrak{S}_2$. Finally that $\mathfrak{s} = \mathfrak{g}_{k_3}(u) \sqsubseteq \mathfrak{g}_{k_3-1}(u_1) \cdot \mathfrak{g}_{k_3-1}(u_2) = \mathfrak{s}'$ is an immediate consequence of Lemma 27. ∎

We can now start the proof. By hypothesis, w admits an α-factorization forest of height at most h. If this height is 0, we conclude as in the base case above. Otherwise, we distinguish two cases depending on the nature of the topmost node in this forest.

A. Case 1: Binary Node

By hypothesis $w = w_1 w_2$ with w_1, w_2 admitting α-factorization forests of heights at most $h - 1$. Hence, by choice of k'_2, k'_3 in Fact 38, $\mathfrak{g}^L_{k'_2,k'_3}(w_1) \in Sat(\mathfrak{I})$ and $\mathfrak{g}^L_{k'_2,k'_3}(w_2) \in Sat(\mathfrak{I})$.

Using Lemma 39 we obtain that $\mathfrak{g}^L_{k_2,k_3}(w) \sqsubseteq \mathfrak{g}^L_{k_2-1,k_3-1}(w_1) \cdot \mathfrak{g}^L_{k_2-1,k_3-1}(w_2)$. Moreover, since by definition $k_3 - 1 \geqslant k'_3$ and $k_2 - 1 \geqslant k'_2$, it follows from Fact 23 that $\mathfrak{g}^L_{k_2,k_3}(w) \sqsubseteq \mathfrak{g}^L_{k'_2,k'_3}(w_1) \cdot \mathfrak{g}^L_{k'_2,k'_3}(w_2)$. It is then immediate from multiplication and downset closure that $\mathfrak{g}^L_{k_2,k_3}(w) \in Sat(\mathfrak{I})$.

B. Case 2: Idempotent Node

Set $e = \alpha(w)$ and $B = \mathsf{alph}(w)$. Recall that since α is alphabet compatible, any word of image e has alphabet B. We also define $\mathfrak{S} = \{\mathfrak{s} \in rem(Sat(\mathfrak{I})) \mid \mathsf{alph}(\mathfrak{s}) = B\}$. Notice that by Σ_2 closure, for any $\mathfrak{r} \in Sat(\mathfrak{I})$ such that $\mathsf{alph}(\mathfrak{r}) = B$, we have,

$$\mathfrak{r}^\omega \cdot (1_M, \mathfrak{S}) \cdot \mathfrak{r}^\omega \in Sat(\mathfrak{R})$$

We begin by adapting the notion of decomposition to our hypothesis.

e-Decompositions. Given any $u \in A^*$, we say that u admits a *e-decomposition* $u_1, \ldots, u_m$ if

(a) $u = u_1 \cdots u_m$,
(b) $\alpha(u_1) = \cdots = \alpha(u_m) = e$.
(c) for all i, $\mathfrak{g}^L_{k'_2,k'_3}(u_i) \in Sat(\mathfrak{I})$.

In particular, since $\mathsf{alph}(e) = B$ observe that for any e-decomposition $u_1, \ldots, u_m$, $\mathsf{alph}(u_i) = B$, for all i.

Lemma 40. *w admits a e-decomposition $w_1, \ldots, w_m$.*

Proof: By hypothesis of this case, we know that w admits a decomposition $w = w_1 \cdots w_m$ that satisfies Item a, Item b and such that all factors w_i admit a α-factorization forest of height at most $h-1$. Item c is by choice of k_2', k_3' in Fact 38. ∎

We will also need the following lemma which is where we use our hypothesis on L. Recall that we defined $\mathfrak{S} = \{\mathfrak{s} \in rem(Sat(\mathfrak{I})) \mid \mathsf{alph}(\mathfrak{s}) = B\}$.

Lemma 41. *Let $u \in A^*$ that admits an e-decomposition $u = u_1 \cdots u_m$. Then for all $i \leqslant j$, $\mathfrak{g}^L_{k_2',k}(u_i \cdots u_j) \sqsubseteq (e, \mathfrak{S})$.*

Proof: By definition, $\alpha(u_i \cdots u_j) = e$ and, in particular, $\mathsf{alph}(u_i \cdots u_j) = B$. Let $v \in L$ such that $u_i \cdots u_j \lesssim_2^{k_2'} v$. We need to prove that $\mathfrak{g}_k(v) \in \mathfrak{S}$, i.e. that $\mathfrak{g}_k(v) \in rem(Sat(\mathfrak{I}))$ and that $\mathsf{alph}(\mathfrak{g}_k(v)) = B$.

That $\mathfrak{g}_k(v) \in rem(Sat(\mathfrak{I}))$ is by hypothesis on the language L. Furthermore, we have $\mathsf{alph}(\mathfrak{g}_k(v)) = \mathsf{alph}(\alpha(v)) = \mathsf{alph}(v)$. We can now use the fact that $u_i \cdots u_j \lesssim_2^{k_2'} v$ to conclude that $\mathsf{alph}(v) = \mathsf{alph}(u_i \cdots u_j) = B$ (the alphabet of a word can be tested in Σ_2). ∎

As we did in the proof of Proposition 25, we will decompose the e-decomposition of w into a bounded number of subdecompositions. Let us first adapt the notion of index to e-decompositions.

Index of an e-decomposition. Set $\ell = k_3 - 3n$ (hence, by definition, $\ell = max(k, k_3')$). Let $u \in A^*$ that admits an e-decomposition $u_1, \ldots, u_m$. Let $(f, \mathfrak{F}) \in \mathfrak{M}(2)$ be an idempotent and $i \leqslant m$, we say that $(f, \mathfrak{F})$ can be *inserted* at position i if there exists $1 \leqslant j \leqslant i$ such that $i - (j-1) \leqslant n$ and,

$$\mathfrak{g}^L_{k_2',\ell}(w_j) \cdots \mathfrak{g}^L_{k_2',\ell}(w_i) = \mathfrak{g}^L_{k_2',\ell}(w_j) \cdots \mathfrak{g}^L_{k_2',\ell}(w_i) \cdot (f, \mathfrak{F})$$

The index of an e-decomposition $u_1, \ldots, u_m$ is the number of distinct idempotents $(f, \mathfrak{F}) \in \mathfrak{M}(2)$ that can be inserted at some position $i \leqslant m$. Observe that by definition, the index of any e-decomposition is bounded by $n = |\mathfrak{M}(2)|$.

Lemma 42. *Let $u \in A^*$ that admits a e-decomposition $u_1, \ldots, u_m$ of index q and set $\hat{k_2} \geqslant q + 2n + k_2'$ and $\hat{k_3} \geqslant q + 2n + \ell$. Then $\mathfrak{g}^L_{\hat{k_2},\hat{k_3}}(u) \in Sat(\mathfrak{I})$.*

Before proving this lemma, we use it to conclude the idempotent case. We know that our e-decomposition $w_1, \ldots, w_m$ of w has an index $q \leqslant n$. Moreover, by definition $k_2 \geqslant 3n + k_2'$ and $k_3 \geqslant 3n + \ell$, hence, it is immediate from Lemma 42 that $\mathfrak{g}^L_{k_2,k_3}(w) \in Sat(\mathfrak{I})$. It now remains to prove Lemma 42.

Proof: As we did when proving the Σ_3 result, we begin by considering the case when m is "small".

Lemma 43. *Assume that $p_2 \geqslant m + k_2'$ and $p_3 \geqslant m + k_3'$, then $\mathfrak{g}^L_{p_2,p_3}(u) \in Sat(\mathfrak{I})$.*

Proof: We have $p_2 - (m-1) \geqslant k_2'$ and $p_3 - (m-1) \geqslant k_3'$, hence, we can apply Lemma 39 and Fact 23 to obtain that,

$$\mathfrak{g}^L_{p_2,p_3} \sqsubseteq \mathfrak{g}^L_{k_2',\ell}(u_1) \cdots \mathfrak{g}^L_{k_2',\ell}(u_m) \sqsubseteq \mathfrak{g}^L_{k_2',k_3'}(u_1) \cdots \mathfrak{g}^L_{k_2',k_3'}(u_m)$$

Moreover, it follows from Item c in the definition of e-decompositions that for all i, $\mathfrak{g}^L_{k_2',k_3'}(u_i) \in Sat(\mathfrak{I})$. It then follows from the multiplication and downset closures that $\mathfrak{g}^L_{p_2,p_3}(u) \in Sat(\mathfrak{I})$. ∎

We can now start the main proof which works by induction on q. The proof is based on the following lemma.

Lemma 44. *Assume that $n > m$. There exists $i \leqslant n$ such that,*

$$\mathfrak{g}^L_{k_2',\ell}(w_1) \cdots \mathfrak{g}^L_{k_2',\ell}(w_i) = \mathfrak{g}^L_{k_2',\ell}(w_1) \cdots \mathfrak{g}^L_{k_2',\ell}(w_i) \cdot (e, \mathfrak{E})$$

with $(e, \mathfrak{E}) \in Sat(\mathfrak{I})$ an idempotent.

Proof: It is immediate from the pigeon-hole principle that there exists $i < i' \leqslant n+1$ such that,

$$\mathfrak{g}^L_{k_2',\ell}(w_1) \cdots \mathfrak{g}^L_{k_2',\ell}(w_i) = \mathfrak{g}^L_{k_2',\ell}(w_1) \cdots \mathfrak{g}^L_{k_2',\ell}(w_{i'})$$

Hence, we set $(e, \mathfrak{E}) = (\mathfrak{g}^L_{k_2',\ell}(w_{i+1}) \cdots \mathfrak{g}^L_{k_2',\ell}(w_{i'}))^\omega$. That $(e, \mathfrak{E}) \in Sat(\mathfrak{I})$ is immediate from Item c in the definition of e-decompositions and the multiplication closure. ∎

It follows from Lemma 44 that when $q = 0$ (i.e. when no idempotent can be inserted), we have $m \leqslant n$. Hence, $\hat{k_2} \geqslant m + k_2'$ and $\hat{k_3} \geqslant m + k_3'$ and the induction base follows from Lemma 43. It remains to treat the case when $q \geqslant 1$. For the remainder of the proof, we assume that we are in this case.

If $m \leqslant n$, we conclude using Lemma 43 as in the base case. Otherwise, let $i \leqslant n$ and $(e, \mathfrak{E}) \in Sat(\mathfrak{I})$ be as defined in Lemma 44. Set $j \leqslant m$ as the largest number such that $(e, \mathfrak{E})$ can be inserted at position j in the e-decomposition of u. Observe that j has to exist since $(e, \mathfrak{E})$ can be inserted at position i. In particular, we have $i \leqslant j$. Using Lemma 39, we obtain that,

$$\mathfrak{g}^L_{\hat{k_2},\hat{k_3}}(u) \sqsubseteq \mathfrak{g}^L_{\hat{k_2}-1,\hat{k_3}-1}(u_1 \cdots u_j) \cdot \mathfrak{g}^L_{\hat{k_2}-1,\hat{k_3}-1}(u_{j+1} \cdots u_m)$$

Observe that $u_{j+1}, \ldots, u_m$ is a e-decomposition whose index is smaller than that of $u_1, \ldots, u_m$ (all idempotents that can be inserted in $u_{j+1}, \ldots, u_m$ can be inserted in $u_1, \ldots, u_m$ and by choice of j, $(e, \mathfrak{E})$ cannot be inserted in $u_{j+1}, \ldots, u_m$). Hence, we can apply our induction hypothesis in Lemma 42 and conclude that $\mathfrak{g}^L_{\hat{k_2}-1,\hat{k_3}-1}(u_{j+1} \cdots u_m) \in Sat(\mathfrak{I})$.

We now prove that $\mathfrak{g}^L_{\hat{k_2}-1,\hat{k_3}-1}(u_1 \cdots u_j) \in Sat(\mathfrak{I})$. It will then follow from multiplication and downset closure that $\mathfrak{g}^L_{\hat{k_2},\hat{k_3}}(u) \in Sat(\mathfrak{I})$. Set $j' = j - (n-1)$ and observe that when $j' \leqslant i$, since $i \leqslant n$, we have $j \leqslant 2n - 1$. Hence, that $\mathfrak{g}^L_{\hat{k_2}-1,\hat{k_3}-1}(u_1 \cdots u_j) \in Sat(\mathfrak{I})$ follows from Lemma 43.

We finish with the case when $i < j'$. We consider the following objects:

$$(e, \mathfrak{S}') = \mathfrak{g}^L_{k_2',\ell}(u_{i+1} \cdots u_{j'-1}) \cdot \mathfrak{g}^L_{k_2',\ell}(u_{j'}) \cdots \mathfrak{g}^L_{k_2',\ell}(u_j)$$
$$(e, \mathfrak{R}) = \mathfrak{g}^L_{k_2',\ell}(u_1) \cdots \mathfrak{g}^L_{k_2',\ell}(u_i)$$

Recall that $\hat{k_2} \geqslant q + 2n + k_2'$ and $\hat{k_3} \geqslant q + 2n + \ell$. Therefore, using Lemma 39 several times (at most $2n$ times), one can verify that:

$$\mathfrak{g}^L_{\hat{k_2}-1,\hat{k_3}-1}(u_1 \cdots u_j) \sqsubseteq (e, \mathfrak{R}) \cdot (e, \mathfrak{S}')$$

Moreover, by definition of the positions i, j and of the idempotent $(e, \mathfrak{E})$, we have $(e, \mathfrak{R}) \cdot (e, \mathfrak{E}) = (e, \mathfrak{R})$ and $(e, \mathfrak{S}') \cdot (e, \mathfrak{E}) = (e, \mathfrak{S})$. Hence we obtain,

$$\mathfrak{g}^L_{\hat{k_2}-1,\hat{k_3}-1}(u_1 \cdots u_j) \sqsubseteq (e, \mathfrak{R}) \cdot (e, \mathfrak{E}) \cdot (e, \mathfrak{S}') \cdot (e, \mathfrak{E})$$

By multiplication closure and Item c in the definition of e-decompositions $(e, \mathfrak{R}) \in Sat(\mathfrak{I})$. Therefore, by multiplication closure and downset closure, it suffices to prove that $(e, \mathfrak{E}) \cdot (e, \mathfrak{S}') \cdot (e, \mathfrak{E}) \in Sat(\mathfrak{I})$ to conclude. We use Σ_2 closure.

Observe that by definition and Lemma 41, $(e, \mathfrak{S}') \sqsubseteq (e, \mathfrak{S})^{j-j'+2}$. It is also immediate by multiplication closure that $\mathfrak{S}\mathfrak{S} \subseteq \mathfrak{S}$ and therefore that $(e, \mathfrak{S})^{j-j'+2} \sqsubseteq (e, \mathfrak{S})$. Moreover, since $\mathsf{alph}((e, \mathfrak{E})) = B$, it is immediate by Σ_2-closure that

$$(e, \mathfrak{E}) \cdot (e, \mathfrak{S}) \cdot (e, \mathfrak{E}) = (e, \mathfrak{E}) \cdot (1_M, \mathfrak{S}) \cdot (e, \mathfrak{E}) \in Sat(\mathfrak{I})$$

Hence, by downset closure, we obtain that $(e, \mathfrak{E}) \cdot (e, \mathfrak{S}') \cdot (e, \mathfrak{E}) \in Sat(\mathfrak{I})$ which terminates the proof.

■