Regular Languages of Infinite Trees that are Boolean Combinations of Open Sets

Mikołaj Bojańczyk and Thomas Place*

University of Warsaw

Abstract. In this paper, we study boolean (not necessarily positive) combinations of open sets. In other words, we study positive boolean combinations of safety and reachability conditions. We give an algorithm, which inputs a regular language of infinite trees, and decides if the language is a boolean combination of open sets.

1 Introduction

In this paper, we work with infinite binary trees labeled by a finite alphabet. The set of trees can be interpreted as a compact metric space. The distance between two different trees is 2^{-n}, where n is the smallest depth where the two trees are different. In the topology induced by this distance, a set of trees L is open if for every tree $t \in L$, there is a finite prefix of t such that changing nodes outside the prefix does not affect membership in L. In other words, an open set is a reachability language. We are interested in understanding finite boolean combinations, not necessarily positive, of open sets. The main result of this paper is:

Theorem 1.1. *The following problem is EXPTIME complete. Given a nondeterministic parity automaton on infinite trees, decide if the recognized language is a boolean combination of open sets.*

In other words, this paper provides an effective characterization of boolean (not necessarily positive) combinations of open sets, within the class of regular languages of infinite trees.
A similar version of the problem, where one asks if L is simply an open set, and not a finite boolean combination of open sets, is significantly simpler. Here is the solution to this simpler problem which is folklore to the best of our knowledge. The key observation is that the topological closure of a tree language L is the set

$$closure(L) = \{t : \text{every finite prefix of } t \text{ can be extended to some tree in } L\}.$$

A language L is open if and only if its complement L^c satisfies $L^c = closure(L^c)$. Automata for both L^c and $closure(L^c)$ can be computed based on the automaton for L, and then one can test two regular languages for equality.

* Both authors supported by ERC Starting Grant "Sosna"

The difficulty in Theorem 1.1 is dealing with the boolean combinations.
Our approach to the problem uses forest algebra for infinite trees [4]. We intended to achieve two complementary goals: use the algebra to understand boolean combinations of open sets; and use boolean combinations of open sets to understand the algebra.

Goal 1: understand boolean combinations of open sets. We believe that giving an effective characterization for a class $\mathscr{L}$ of regular languages can be the most mathematically rewarding thing that one can do with $\mathscr{L}$. The ostensible goal of an effective characterization – the algorithm deciding membership in $\mathscr{L}$ – is usually less interesting than the insight into the structure of $\mathscr{L}$ that is needed to get the algorithm. A famous example is the theorem of Schützenberger and McNaughton/Papert, which makes a beautiful connection between logic and algebra: a word language is definable in first-order logic if and only if its syntactic monoid is aperiodic [9, 7].
We believe that our study of boolean combinations of open sets achieves this goal. We discover that this class of languages has a rich structure, which is much more complex in the case of infinite trees than in the case of infinite words. On our way to Theorem 1.1, we provide three conditions which are equivalent to being a boolean combination of open sets, see Theorem 5.3. Two of these conditions are stated in terms of games, and one is stated in terms of algebraic equations. We believe that each of these conditions are interesting in their own right.

Goal 2: understand algebra for infinite trees. The algebraic theory of languages of finite words is well studied, using monoids and semigroups, see the book by Straubing [10]. The algebraic theory of languages of infinite words is also well understood, see the book by Perrin and Pin [8]. There has been quite a lot of recent work on algebra for finite trees [2], but the theory is still not mature. Finally, the algebraic theory of infinite trees is very far from being understood, despite some work [4, 1].
We believe that our study of boolean combinations of open sets has highlighted the kind of tools that might be important in the algebraic theory of infinite trees. An important theme is the use of games. As mentioned previously, we characterize boolean combinations of opens sets in terms of two games, and a set of two identities. Even in the identities, there is a hidden game, which is played in the algebra. We see this as evidence that the algebraic theory of infinite trees will need to take games into account.

Organization of the paper. In Section 2 we give our first characterization of boolean combination of open sets by the means of games. Note that this characterization is not effective and works not only for trees but also for all topological spaces. In Section 3, we provide the basic definitions we need for trees and algebra. In Section 4, we make a finer analysis of the non-effective characterization of Section 2 in the case of trees. Finally, in Section 5 we state

our effective characterization using algebraic identities. Due to space limitations many proofs appear in the appendix.

2 A game characterization

We begin by studying boolean combinations of open sets in arbitrary topological spaces. Fix a topological space. In this paper, we are interested in the topological space of infinite trees, but the discussion in this section works for all spaces.
Let $X_1, \ldots, X_n$ be arbitrary subsets of the topological space. We define a game

$$\mathscr{H}(X_1, \ldots, X_n)$$

which is played by two players, called Alternator and Constrainer. The game is played in n rounds. At the beginning of each round $i \in \{1, \ldots, n\}$, there is an open set U_i. Initially, U_1 is the whole space. Round i of the game is played as follows.

- Alternator chooses a point $x_i \in U_i \cap X_i$. If there is no such point x_i, the game is interrupted and player Constrainer wins immediately. Otherwise,
- Constrainer chooses an open set $U_i \ni x_i$, and the next round is played.

If Alternator manages to survive n rounds, then he wins.
The following lemma shows that the rules of the game could be changed such that Constrainer can only pick base open sets.

Lemma 2.1. *Choose some base for the topology. If Constrainer has a winning strategy, then he has a winning strategy which uses base open sets for $U_1, \ldots, U_n$.*

Suppose that X is a set and $n \in \mathbb{N}$. We write $\mathscr{H}_{\in \notin}(X, n)$ for the game where Alternator needs to alternate n times between X and its complement, that is:

$$\mathscr{H}(X_1, \ldots, X_n) \qquad \text{where } X_i = \begin{cases} X & \text{when } i \text{ is odd} \\ \text{the complement of } X & \text{when } i \text{ is even} \end{cases}.$$

Example 2.2. Consider the space of real numbers, and let X be the rational numbers. Then for every n, Alternator wins the game $\mathscr{H}_{\in \notin}(X, n)$.

Example 2.3. In the real numbers, let X be the complement of $\{1/n : n \in \mathbb{N}\}$. Alternator wins $\mathscr{H}_{\in \notin}(X, 3)$. In the first round, Alternator plays $0 \in X$. In the second round, Alternator plays $1/n \notin X$ for some large n depending on Constrainer's move. In the third round, Alternator plays $1/n + \epsilon \in X$, for some small ϵ depending on Constrainer's move. Constrainer wins $\mathscr{H}_{\in \notin}(X, n)$ for $n \geq 4$.

Proposition 2.4. *The following conditions are equivalent for a set X:*

- *X is a finite boolean combination of open sets*
- *Constrainer wins the game $\mathscr{H}_{\in \notin}(X, n)$ for all but finitely many n.*

Refinement lemma. We now state a lemma, which shows that Alternator's winning sets can be refined in an arbitrary finite way.

Lemma 2.5. *Let $X_1, \dots, X_n$ be sets. For $i \in \{1, \dots, n\}$, let $\mathcal{Y}_i$ a finite family of sets partitioning of X_i. If Alternator wins*

$$\mathscr{H}(X_1, \dots, X_n)$$

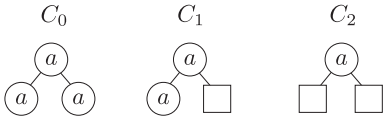
then there exist $Y_1 \in \mathcal{Y}_1, \dots, Y_n \in \mathcal{Y}_n$ such that Alternator wins

$$\mathscr{H}(Y_1, \dots, Y_n).$$

3 Preliminaries on trees

Trees. We use possibly infinite trees where every node has zero or two children. For a finite alphabet A, we denote by H_A the set of infinite binary trees labeled over A. Notions of node, leaf, child, root, descendant are defined as usual. We write '$<$' the descendant relation (the smallest node being the root of the tree). If t is a tree and x a node of t, we write $t(x)$ the label of x in t, and $t|_x$ for the *subtree of t at x*.

Multicontexts. A *multicontext* is a tree with some distinguished unlabeled leaves called its *ports*. The number of ports is called the *arity*, there might be infinitely many ports. Given a multicontext C and a valuation η which maps ports to trees, we write $C[\eta]$ for the tree obtained by replacing each port x by the tree $\eta(x)$. A tree $C[\eta]$ is said to *extend* the multicontext C, conversely the multicontext C is said to be a *prefix* of the tree $C[\eta]$. The set of all trees extending a multicontext C is denoted by $C[*]$. The following picture shows three multicontexts, with arities $0, 1$ and 2, with the ports depicted by squares.



The multicontext C_0 is a tree, and $C_0[*]$ is $\{C_0\}$. The multicontext C_1 is a prefix of every tree where the root label is a, and the left child of the root is a leaf with label a. Finally, C_2 is a prefix of every tree with root label a. We are mostly interested in finite prefixes, which are multicontexts where every path ends in a leaf, which is either a port or a normal leaf.

Contexts. A *context* is a multicontext with exactly one port. We write V_A the set of contexts over A. We write C, D for contexts. Given two contexts C, D, we write $C \cdot D$ for the context obtained by replacing the port of C with D. One can verify that $\cdot$ is associative, therefore, $(V_A, \cdot)$ is a monoid (with the empty context, denoted by $\Box$, as neutral element). V_A also acts on H_A, with $C \cdot t$ defined as the tree obtained by replacing the port of C with t. Finally, we write C^∞ for the infinite tree $C \cdot C \cdot C \cdots$.

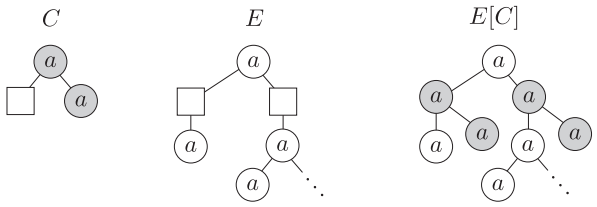
3.1 Tree languages and algebra

We are mainly concerned with regular languages of infinite trees. This is the class of languages of infinite trees that is recognized by nondeterministic parity automata; or equivalently recognized by alternating parity automata; or equivalently can be defined in monadic second-order logic. See [6].
Recall that our goal is to decide if a given regular language L of infinite trees is a boolean combination of open sets. It will be important for us to work with a canonical representation of L. As our canonical representation, we use equivalence classes of trees and contexts with respect to a natural Myhill-Nerode style equivalence, see below.
The equivalence classes form a kind of algebra, which is similar to the forest algebra for infinite trees from [4]. The similarity is that both algebras represent infinite trees. The difference is that the algebra in [4] represents finitely branching unranked trees, while the algebra in this paper represents binary trees. We do not use unranked trees because for unranked finitely branching trees, the topological space is not compact. This is because there is no converging subsequence in a sequence of trees where the n-th tree has n children of the root.

Myhill-Nerode equivalence. Fix L a language of trees over an alphabet A. We define two Myhill-Nerode equivalence relations: one for trees and one for contexts.
Let C be a multicontext, possibly with infinitely many ports. For a tree t, we write $C[t]$ for the tree obtained by putting t in all ports of C. In the Myhill-Nerode equivalence for trees, we say trees t and t' are L-equivalent if

$$C[t] \in L \Leftrightarrow C[t'] \in L \qquad \text{for every multicontext } C.$$

To give a similar definition for contexts, we use a variant of multicontext where the ports can be substituted by contexts and not trees. Such a multicontext is called a *context environment*. Formally speaking, a context environment is defined like a multicontext, except that the ports have exactly one child (instead of being leaves). Given a context environment E and a context C, we write $C[E]$ for the tree obtained by substituting C for every port of E, as in the following picture:



We define two contexts C and C' to be L-equivalent if

$$E[C] \in L \Leftrightarrow E[C'] \in L \qquad \text{for every context environment } E.$$

The algebra. We write H_L for the equivalence classes of trees with respect to L, and V_L for the equivalence classes of contexts with respect to L. We write:

$$\alpha : (H_A, V_A) \to (H_L, V_L)$$

for the two-sorted function which maps trees and contexts to their L-equivalence classes; this function is called the *syntactic morphism of* L. We use the name *tree type* for elements of H_L and *context type* for elements of V_L. It is not difficult to show that both H_L and V_L are finite when L is regular. The syntactic morphism can be computed based on a nondeterministic tree automaton recognizing L, in exponential time [4]. Finiteness of H_L and V_L is necessary but not sufficient for regularity, for instance both H_L and V_L are finite for any language defined in the logic MSO+U [3].

Lemma 3.1. *The following operations respect L-equivalence.*

1. *For every multicontext D, the operation: $t \mapsto D[t]$.*
2. *For every context environment E, the operation: $C \mapsto E[C]$.*
3. *The composition of contexts $(C_1, C_2) \mapsto C_1 \cdot C_2$.*
4. *Substituting a tree in the port of a context: $(C, t) \mapsto C \cdot t$.*
5. *Infinite iteration of a context: $C \mapsto C^\infty$.*
6. *For every letter a, the operations $t \mapsto a(t, \Box)$ and $t \mapsto a(\Box, t)$.*

It follows that the above operations can be applied to elements of the syntactic algebra.

Idempotents. Given any finite monoid V, there is (folklore) a number $\omega(V)$ (denoted by ω when V is understood from the context) such that for each element v of V, v^ω is an idempotent: $v^\omega = v^\omega v^\omega$.

4 Boolean combinations of open sets of trees

As mentioned in the introduction, we use prefix topology on trees, which yields a topology identical to that of the Cantor space. In this topology, a *base open set* is defined to be any set $C[*]$, where C is a *finite* multicontext. Open sets are defined to be arbitrary unions of base open sets. This topology is the same as the topology generated by a distance, which says that trees are at distance 2^{-n} where n is the smallest depth where the two trees differ. This paper is about finite boolean combination of open sets. Typical boolean combinations of open sets include

- Trees over alphabet $\{a, b, c\}$ which contain at least one a and no b's.
- Trees over alphabet $\{a, b\}$ which contain two or five a's.

Languages, which are not boolean combinations of open sets include

- Trees over alphabet $\{a, b\}$ with finitely many a's.
- Trees over alphabet $\{a, b\}$ with a finite and even number of a's.

Let us revisit the game from Proposition 2.4 in the case of trees. In this special case, points are trees. By Lemma 2.1, we may assume that Constrainer uses base open sets, which are finite multicontexts. The game begins with the whole space, which corresponds to the empty multicontext. In each round, Alternator chooses a tree that extends the current multicontext, and then Constrainer chooses a finite multicontext that is a prefix of the tree chosen by Alternator.

Example 4.1. Consider an alphabet $\{a, b\}$ and the language L="infinitely many a's". It is not difficult to see that Alternator can win the game $\mathscr{H}_{\in \notin}(L, n)$ for every $n \in \mathbb{N}$. This is because every finite multicontext can be extended to a tree with finitely many a's, or to a tree with infinitely many a's. By Proposition 2.4, L is not a boolean combination of open sets.

Proposition 2.4 helps us understand finite boolean combinations of open sets, but it is not an effective characterization. To be effective, we should be able to decide if player Alternator wins $\mathscr{H}_{\in \notin}(L, n)$ for every n. The following simple lemma shows how to decide the winner for a given n.

Lemma 4.2. *Given regular tree languages $L_1, \ldots, L_n$, one can decide who wins $\mathscr{H}(L_1, \ldots, L_n)$. In particular, given L and n, one can decide who wins $\mathscr{H}_{\in \notin}(L, n)$.*

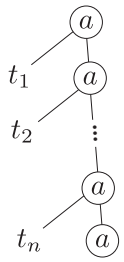
Proof. The statement "Alternator wins the game $\mathscr{H}(L_1, \ldots, L_n)$" can be formalized in monadic second-order logic on the complete binary tree, by a formula which can be computed based on the languages $L_1, \ldots, L_n$. Therefore, the winner can be decided using the Rabin theorem. □

The above lemma gives a semi-algorithm for deciding if a regular language is a finite boolean combination of open sets. For $n = 1, 2, \ldots$, use Lemma 4.2 to compute the winner of $\mathscr{H}_{\in \notin}(L, n)$. If Constrainer wins the game for some n, then he also wins the game for $n + 1, n + 2, \ldots$ and therefore the algorithm can terminate and declare that the L is a finite boolean combination of open sets. If the language is *not* a finite boolean combination of open sets, then the algorithm does not terminate.
Observe that even when the algorithm *does* terminate, it does multiple calls to Rabin's theorem, which has non-elementary complexity.
The main contribution of this paper is a finer analysis of the problem, which yields an algorithm (not a semi-algorithm) deciding if a tree language is a finite boolean combination of open sets.

4.1 The infinite game

Proposition 2.4 can be rephrased as: a language L is *not* a finite boolean combination of open sets if and only if player Alternator can win $\mathscr{H}_{\in \notin}(L, n)$ for arbitrarily large n. One could imagine a variant of the game, more difficult for Alternator, where infinitely many rounds have to be played. Call the infinite variant $\mathscr{H}_{\in \notin}(L, \infty)$. In Example 2.2, which is about rational numbers, player Alternator can win the infinite game.

It is clear that if Alternator wins $\mathscr{H}_{\in\notin}(X,\infty)$, then he also wins $\mathscr{H}_{\in\notin}(X,n)$ for every n. We show a counterexample for the converse implication, which is a regular tree language. This counterexample language necessarily uses trees, because the converse implication holds for regular languages of infinite words. The counterexample language L is the set of trees over $\{a,b\}$ of the form:



such that n is some natural number, and for each $i \in \{1,\ldots,n\}$, the tree t_i is either finite, or contains no b nodes. Observe that in a tree from L, the rightmost branch is necessarily finite.

Fact 1 *Alternator loses $\mathscr{H}_{\in\notin}(L,\infty)$, but wins $\mathscr{H}_{\in\notin}(L,n)$ for every n.*

5 The effective characterization

In this section, we present the main result of the paper.

The context game. So far we have worked with a game $\mathscr{H}(L_1,\ldots,L_n)$, for tree languages $L_1,\ldots,L_n$. We define a similar game for languages of contexts. Recall that contexts are defined as a special case of trees, with an additional port label that appears in exactly one leaf. From the distance on trees, we get a distance on contexts. This yields the definition of a game for a sequence $K_1,\ldots,K_n$ of context languages. To avoid confusion between trees and contexts, we denote the context game by $\mathscr{V}(K_1,\ldots,K_n)$.

Games on types. Consider a language L and its syntactic morphism

$$\alpha_L : (H_A, L_A) \to (H_L, V_L).$$

Recall that by definition of the syntactic morphism, a type $h \in H_L$ is actually equal to the set of trees $\alpha_L^{-1}(h)$. Therefore, it makes sense to talk about the game $\mathscr{H}(h_1,\ldots,h_n)$ for a sequence of tree types. Likewise for context types. Define

$$\mathscr{H}_L \stackrel{\text{def}}{=} \{(h_1,\ldots,h_n) \in (H_L)^n : \ n \in \mathbb{N} \text{ and Alternator wins } \mathscr{H}(h_1,\ldots,h_n)\}$$
$$\mathscr{V}_L \stackrel{\text{def}}{=} \{(v_1,\ldots,v_n) \in (V_L)^n : \ n \in \mathbb{N} \text{ and Alternator wins } \mathscr{V}(v_1,\ldots,v_n)\}$$

A comment on notation is in order here. The sets $\mathscr{H}_L$ and $\mathscr{V}_L$ contain words, over alphabets H_L and V_L, respectively. Usually when dealing with words, one omits

the brackets and commas, and writes abc instead of (a, b, c). When the alphabet is V_L, this leads to ambiguity, since the expression vwu can be interpreted as: 1) a word with a single letter obtained by multiplying v, w, u in the context monoid V_L; or 2) a three-letter word over the alphabet V_L. These two interpretations should not be confused, so we write $(v_1, \ldots, v_n)$ for n-letter words over the alphabet V_L. For the sake of uniformity, we also write $(h_1, \ldots, h_n)$ for n-letter words in over the alphabet H_L, although there is no risk of ambiguity here.

Fact 2 *Both $\mathscr{H}_L$ and $\mathscr{V}_L$ are regular languages of finite words.*

Proof. Both languages are closed under removing letters. Every language closed under removing letters is regular, by Higman's lemma. □

The above fact is amusing, but useless, because it does not say how to compute automa for $\mathscr{H}_L$ and $\mathscr{V}_L$ as a function of the language L.[1] If we are not interested in efficiency, membership in $\mathscr{H}_L$ can be decided with Lemma 4.2. The same kind of algorithm works for $\mathscr{V}_L$. Later on, we give a more efficient algorithm.

$(h_1, \ldots, h_n) \in \mathscr{H}_L$	implies	$(C[h_1], \ldots, C[h_n]) \in \mathscr{H}_L$
$(v_1, \ldots, v_n) \in \mathscr{V}_L$	implies	$(E[v_1], \ldots, E[v_n]) \in \mathscr{H}_L$
$(v_1, \ldots, v_n), (w_1, \ldots, w_n) \in \mathscr{V}_L$	implies	$(v_1 w_1, \ldots, v_n w_n) \in \mathscr{V}_L$
$(v_1, \ldots, v_n) \in \mathscr{V}_L, (h_1, \ldots, h_n) \in \mathscr{H}_L$	implies	$(v_1 h_1, \ldots, v_n h_n) \in \mathscr{H}_L$
$(v_1, \ldots, v_n) \in \mathscr{V}_L$	implies	$(v_1^\infty, \ldots, v_n^\infty) \in \mathscr{H}_L$
$(h_1, \ldots, h_n) \in \mathscr{H}_L$	implies	$(a[\Box, h_1], \ldots, a[\Box, h_n]) \in \mathscr{V}_L$
$(h_1, \ldots, h_n) \in \mathscr{H}_L$	implies	$(a[h_1, \Box], \ldots, a[h_n, \Box]) \in \mathscr{V}_L$

Table 1. Closure properties of $\mathscr{H}_L$ and $\mathscr{V}_L$. C is a multicontext, E is a context environment, and a is a letter.

Lemma 5.1. *The sets $\mathscr{H}_L$ and $\mathscr{V}_L$ satisfy the closure properties in Table 1.*

Notice the similarity of Table 1 with the operations in Lemma 3.1. Another way of stating Lemma 5.1 is that for every $n \in \mathbb{N}$, $(\mathscr{H}_L, \mathscr{V}_L)$ restricted to sequences of length n is a subalgebra of the the n-fold power of the syntactic algebra (H_L, V_L). We define the *alternation* of a word to be its length, after iteratively eliminating letters that are identical to their predecessors. The alternation of *abaabbb* is 4. We say that a set of words has *unbounded alternation* if it contains words with arbitrarily large alternation.

Proposition 5.2. *For a regular language L of infinite trees, the following conditions are equivalent.*

- *Alternator wins the game $\mathscr{H}_{\in \notin}(L, n)$ for infinitely many n.*
- *The set $\mathscr{H}_L$ has unbounded alternation.*

[1] To the best of our knowledge it is possible, although unlikely, that computing $\mathscr{H}_L$ and $\mathscr{V}_L$ is undecidable.

Proof. We begin with the top-down implication. We show that if Alternator wins the game $\mathscr{H}_{\in \notin}(L, n)$, then $\mathscr{H}_L$ contains a word of length n where every two consecutive letters are different. Suppose then that Alternator wins $\mathscr{H}_{\in \notin}(L, n)$, which means that he wins $\mathscr{H}(L_1, \ldots, L_n)$ where L_i is L for odd-numbered rounds and its complement for even-numbered rounds. Both L and its complement can be partitioned into tree types. By Lemma 2.5, Alternator wins $\mathscr{H}(h_1, \ldots, h_n)$ for some sequence of types, such that h_i is included in L or its complement, depending on the parity of i. In particular, the consecutive types are different.
We now do the bottom-up implication. Suppose that $\mathscr{H}_L$ has unbounded alternation. Since $\mathscr{H}_L$ is closed under removing letters, there must be some $g, h \in H_L$ such that $\mathscr{H}_L$ contains all the words

$$(g, h), (g, h, g, h), (g, h, g, h, g, h), \ldots. \tag{1}$$

Since g and h are different elements of the syntactic algebra, it follows that there must be some multicontext C such that the tree type $C[g]$ is contained in L, while the tree type $C[h]$ is disjoint with L. By applying Lemma 5.1 to (1), we conclude $\mathscr{H}_L$ contains all the words

$$\big(C[g], C[h]\big), \big(C[g], C[h], C[g], C[h]\big), \big(C[g], C[h], C[g], C[h], C[g], C[h]\big), \ldots.$$

It follows that Alternator can alternate arbitrarily long between the language L and its complement. □

The main theorem. So far, we have proved Propositions 2.4 and 5.2, which characterize finite boolean combinations of open sets in terms of games. Neither of these game characterizations is effective. We now add a final characterization, which uses identities and is effective.

Theorem 5.3 (Main Theorem). *For a regular language L of infinite trees, the following conditions are equivalent.*

1. *L is a finite boolean combination of open sets.*
2. *Constrainer wins the game $\mathscr{H}_{\in \notin}(L, n)$ for all but finitely many n.*
3. *The set $\mathscr{H}_L$ has bounded alternation.*
4. *The following identities are satisfied.*

$$u^\omega w w^\omega = u^\omega v w^\omega = u^\omega u w^\omega \qquad \text{if } (u, v, w) \in \mathscr{V}_L \text{ or } (w, v, u) \in \mathscr{V}_L \tag{2}$$

$$(u_2 w_2^\omega v)^\omega u_1 w_1^\infty = (u_2 w_2^\omega v)^\infty \qquad \text{if } (u_1, u_2) \in \mathscr{V}_L \text{ and } (w_1, w_2) \in \mathscr{V}_L \tag{3}$$

Theorem 5.3 implies Theorem 1.1 from the introduction, which says that one can decide if the language recognized by a nondeterministic parity automaton on infinite trees, is a boolean combination of open sets. Indeed: there are finitely many sequences of length two and three in $\mathscr{V}_L$. These sequences can be computed using Lemma 4.2. It is then sufficient to test if the identities from item 4 are valid by checking all combinations. A more detailed proof, together with the EXPTIME complexity, is given in the appendix.

In Propositions 2.4 and 5.2, we have shown that conditions 1, 2 and 3 in Theorem 5.3 are equivalent. It remains to show that conditions 3 and 4 are equivalent. The proof of the implication from 4 to 3 is the technical core of this paper, and is in the appendix. Below we prove the much simpler converse implication, which at least can serve to illustrate the identities.

Implication from 3 to 4. We prove the contrapositive: if one of the identities (2) or (3) is violated, then $\mathscr{H}_L$ has unbounded alternation.
Suppose first that (2) is violated. We will show that $\mathscr{V}_L$ has unbounded alternation. This is enough, by the following lemma.

Lemma 5.4. *If $\mathscr{V}_L$ has unbounded alternation, then so does $\mathscr{H}_L$.*

The assumption that (2) is violated says that there are $u, v, w \in V_L$ such that

$$(u, v, w) \in \mathscr{V}_L \qquad \text{or} \qquad (w, v, u) \in \mathscr{V}_L,$$

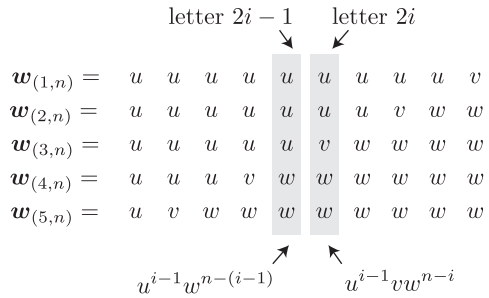
but the three context types $u^\omega w w^\omega$, $u^\omega v w^\omega$ and $u^\omega u w^\omega$ are all equal. If the three context types are not equal, then second one must be different from either the first one or the third one. We only do the proof for the case when $(u, v, w) \in \mathscr{V}$ and when $u^\omega w w^\omega \neq u^\omega v w^\omega$. For nonzero $n \in \mathbb{N}$ and $i \in \{1, \ldots, n\}$, define

$$\boldsymbol{w}_{(i,n)} = (\overbrace{u, \ldots, u}^{2n - 2i + 1 \text{ times}}, v, \overbrace{w, \ldots, w}^{2(i-1) \text{ times}}) \quad \in (H_L)^{2n}.$$

This word is obtained from (u, v, w) by duplicating some letters, and therefore it belongs to $\mathscr{V}_L$. For some n, consider the words

$$\boldsymbol{w}_{(1,n)}, \ldots, \boldsymbol{w}_{(n,n)} \in \mathscr{V}_L.$$

These are n words of length $2n$. Let us multiply all these words coordinate-wise, yielding a word $\boldsymbol{w}$, also of length $2n$, which is depicted in the following picture:



Choose some k, and take $n = k \cdot \omega + 1$, and $i \in \{\omega, 2\omega, \ldots, (k-1) \cdot \omega\}$. Consider letters $2i + 1$ and $2i + 2$ in the word $\boldsymbol{w}$, which are

$$u^i w^{n-i} = u^\omega w w^\omega \qquad u^i v w^{n-i-1} = u^\omega v w^\omega.$$

By assumption, these letters are different, and therefore the word $\boldsymbol{w}$ has alternation at least k. Because k was chosen arbitrarily, it follows that $\mathscr{V}_L$ has unbounded alternation.
Consider now the case when (3) is violated. This means $\mathscr{V}_L$ contains pairs (u_1, u_2) and (w_1, w_2) such that for some $v \in V_L$,

$$e^\infty \neq eu_1 w_1^\infty \qquad \text{for } e \stackrel{\text{def}}{=} (u_2 w_2^\omega v)^\omega.$$

Lemma 5.5. *Let u_1, u_2, w_1, w_2 and e be as above. If $(h_1, \ldots, h_n) \in \mathscr{H}_L$, then*

$$(e^\infty, eh_1, \ldots, eh_n) \in \mathscr{H}_L \tag{4}$$

$$(eu_1 w_1^\infty, eh_1, \ldots, eh_n) \in \mathscr{H}_L \tag{5}$$

Using the lemma, one shows by induction that for every n, the sequence which alternates n times between e^∞ and $e^\infty u_1 w_1^\infty$ belongs to $\mathscr{H}_L$. This completes the proof of the implication from 3 to 4, and also of Theorem 5.3.

6 Conclusion

We have proved an effective characterization of boolean combination of open sets. We hope that the characterization sheds some light on the nature of such languages. Also, we hope that the technical tools we developed, involving both algebra and games, will be useful in future work on regular languages of infinite trees. One class of particular interest is Weak-MSO (i.e. MSO where set quantification is restricted to finite sets). This class can be characterized by wreath products of boolean combinations of open sets.

References

1. Achim Blumensath. An algebraic proof of Rabin's theorem. unpublished manuscript.
2. Mikołaj Bojańczyk. Algebra for trees. In *Handbook of Automata Theory*. European Mathematical Society Publishing House. To appear.
3. Mikołaj Bojańczyk. A bounding quantifier. In *CSL*, pages 41–55, 2004.
4. Mikołaj Bojańczyk and Tomasz Idziaszek. Algebra for infinite forests with an application to the temporal logic ef. In *CONCUR*, pages 131–145, 2009.
5. Hubert Comon, Max Dauchet, Remi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. *Tree Automata Techniques and Applications*. Available on http://tata.gforge.inria.fr, 2007.
6. Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.
7. Robert McNaughton and Seymour Papert. *Counter-Free Automata*. MIT Press, 1971.
8. Dominique Perrin and Jean-Éric Pin. *Infinite Words*. Elsevier, 2004.
9. Marcel Paul Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8, 1965.
10. Howard Straubing. *Finite Automata, Formal Languages, and Circuit Complexity*. Birkhäuser, 1994.

The appendix is divided in three parts. Sections A, B, C and D contain all the proofs for the results in the paper, except for implication from 4 to 3 in Theorem 5.3 and the complexity issues. The technical core of the paper, the implication from 4 to 3 in Theorem 5.3 is presented in Part II. Part III is devoted to the complexity result in Theorem 1.1, note that the notion of strategy forest defined in Part II is reused.

A Appendix to Section 2

In Sections A.1 and A.2, we prove the two implications of Proposition 2.4. In Section A.3, we prove of Lemma 2.5.

Proposition 2.4. *The following conditions are equivalent for a set X:*

- *X is a finite boolean combination of open sets*
- *Constrainer wins the game $\mathscr{H}_{\in\notin}(X, n)$ for all but finitely many n.*

A.1 Top-down

Suppose that X is a finite boolean combination of open sets. It is not difficult to show that every finite boolean combination of open sets can be written as

$$X = (W_1 - V_1) \cup \cdots \cup (W_n - V_n),$$

for $W_1, V_1, \ldots, W_n, V_n$ open sets. We will show that Constrainer has a winning strategy in the game $\mathscr{H}_{\in\notin}(X, 2n+1)$. When playing the first $2n$ rounds, Constrainer's strategy preserves the following invariant, when choosing the open sets $U_1, \ldots, U_{2n}$.

> For $k \in \{0, \ldots, n\}$, suppose that $2k$ rounds have already been played. There exist distinct indexes $i_{k+1}, \ldots, i_k \in \{1, \ldots, n\}$ such that
>
> $$U_{2k+1} = \bigcap_{j \in \{1,\ldots,k\}} W_{i_j} \cap V_{i_j}$$

If Constrainer manages to maintain this invariant up to $n = k$, which means up to round $2n$, then player Alternator cannot make his move in round $2n+1$. This is because the set U_{2n+1} is does not contain any points from X, because it is included in all of the sets $V_1, \ldots, V_n$.
We now show how to satisfy the invariant. When $k = 0$, the invariant is clearly satisfied.
Suppose that the invariant is satisfied for k, which means that $2k$ rounds have already been played. Let $i_1, \ldots, i_k$ be as in the invariant. We now show how Constrainer plays in rounds $2k+1$ and $2k+2$ to satisfy the invariant for $k+1$. For round $2k+1$, assume that Alternator chooses a point $x_{2k+1} \in U_{2k+1}$. By definition of the game, x must belong to X, and therefore it must belong to $W_{i_{k+1}} - V_{i_{k+1}}$ for some $i_{k+1} \in \{1, \ldots, n\}$. Observe that i_{k+1} must be different

from $i_1, \ldots, i_k$, since x_{2k+1} belongs to all of the sets $V_{i_1}, \ldots, V_{i_k}$. Constrainer plays the set $U_{2k+1} \cap W_{i_{k+1}}$.
For round $2k+2$, if Alternator can still play, assume that she chooses a point x_{2k+2}. By definition of the game $x_2 \notin X$. Since $x_{2k+2} \in W_{i_{k+1}}$, this means that $x_2 \in V_{i_{k+1}}$. This open set $U_{2k+1} \cap W_{i_{k+1}} \cap V_{i_{k+1}}$ is Constrainer's answer, which preserves the invariant.

A.2 Bottom-up

In the proof, for every $n \in \mathbb{N}$ we define two equivalence relations: n-point-similarity for points of the space, and n-open-similarity for open sets. The definition is by induction:

- Every two points are 0-point-similar.
- Two open sets U_0 and U_1 are n-set-similar if for every $i \in \{0,1\}$ and $x_i \in U_i$, there is some $x_{i-1} \in U_{1-i}$ such that

$$x_0 \in X \iff x_1 \in X \qquad \text{and} \qquad x_0 \text{ and } x_1 \text{ are } n\text{-point-similar}$$

- Two points x_0 and x_1 are n-point-similar if they are $(n-1)$-point-similar and for every $i \in \{0,1\}$ and open $U_i \ni x_i$, there is some open $U_{i-1} \ni x_{1-i}$ such that

$$U_0 \text{ and } U_1 \text{ are } (n-1)\text{-set-similar}$$

Lemma A.1. *For each n, both n-point-similarity and n-open-similarity have finitely many equivalence classes.*

Proof. Induction. □

From Lemma A.1 it follows that equivalence classes of n-point-similarity are finite boolean combinations of open sets, because an equivalence class of a point is defined in terms of the open sets that contain it.

Lemma A.2. *If $x_0 \in X$ and $x_1 \notin X$ are n-point-similar then Alternator wins $\mathscr{H}(X, n)$.*

Proof. By induction on n, we prove a slightly stronger statement: not Alternator wins the game, but he begins the game by playing point x_0 in the first round.
We describe Alternator's strategy. For $n = 0$, there is nothing to do. Otherwise, player Alternator plays x_0 in the first round. Suppose that Constrainer picks some open set $U_0 \ni x_0$. By assumption on x_0 and x_1 being n-point-similar, there is some open set $U_1 \ni x_1$ such that U_0 and U_1 are $(n-1)$-open-similar.
Because U_1 and U_0 are $(n-1)$-open-similar, and U_1 contains $x_1 \notin X$, it follows that there is some $x_2 \in U_0$ such that

$$x_2 \notin X \qquad \text{and} \qquad x_1 \text{ and } x_2 \text{ are } (n-1)\text{-point-similar.}$$

The point x_2 is Alternators response to U_0. Using the induction assumption for x_1 and x_2 and the complement of X, Alternator continues playing in the remaining $n-1$ rounds. □

We now prove the right-to-left implication in Proposition 2.4. Suppose that Alternator loses the game $\mathscr{H}(X, n)$ for some n. By Lemma A.2 it follows that if two points are n-point-equivalent, then they either both belong to X, or are both outside. It follows that X is a union of equivalence classes of n-point-equivalence. By Lemma A.1, X is a finite boolean combination of open sets.

A.3 Proof of Lemma 2.5

Lemma 2.5. *Let $X_1, \dots, X_n$ be sets. For $i \in \{1, \dots, n\}$, let $\mathcal{Y}_i$ a finite family of sets partitioning of X_i. If Alternator wins*

$$\mathscr{H}(X_1, \dots, X_n)$$

then there exist $Y_1 \in \mathcal{Y}_1, \dots, Y_n \in \mathcal{Y}_n$ such that Alternator wins

$$\mathscr{H}(Y_1, \dots, Y_n).$$

For an open set U, consider a game $\mathscr{H}_U(X_1, \dots, X_n)$ which is played the same way as $\mathscr{H}(X_1, \dots, X_n)$, with the added constraint that Alternator's first move has to belong to U. By induction on n, we prove a slightly more refined version of the lemma, where there is an added parameter – an open set U – and the games are $\mathscr{H}_U(X_1, \dots, X_n)$ and $\mathscr{H}_U(Y_1, \dots, Y_n)$.
The induction base is immediate, because Alternator always wins for $n = 0$.
Consider the first move by Alternator, where he chooses a point $x \in U$. This point necessarily belongs to some $Y_1 \in \mathcal{Y}_1$. For $i \in \mathbb{N}$, let U_i be the open ball around x of radius $1/i$. By definition of the game, we know that Alternator wins

$$\mathscr{H}_{U_i}(X_2, \dots, X_n)$$

for every i. By induction assumption, we know that for every i there exist $Y_2^{(i)} \in \mathcal{Y}_1, \dots, Y_n^{(i)} \in \mathcal{Y}_n$ such that Alternator wins

$$\mathscr{H}_{U_i}(Y_2^{(i)}, \dots, Y_n^{(i)}).$$

By the pigeon-port principle, there must be some $Y_2, \dots, Y_n$ such that

$$(Y_2, \dots, Y_n) = (Y_2^{(i)}, \dots, Y_n^{(i)})$$

holds for infinitely many i. Since the game $\mathscr{H}_V(Y_2, \dots, Y_n)$ grows more difficult for Alternator as the open set V becomes smaller, and since every open set $V \ni x$ contains some U_i, we conclude that Alternator wins

$$\mathscr{H}_V(Y_2, \dots, Y_n)$$

for every $V \ni x$. By viewing V as a response of Constrainer to Alternator's move $x \in Y_1$, we conclude that Alternator wins the game

$$\mathscr{H}_U(Y_1, \dots, Y_n).$$

B Appendix to Section 3

This section contains the proof of Lemma 3.1

Lemma 3.1. *The following operations respect L-equivalence.*

1. *For every multicontext D, the operation: $t \mapsto D[t]$.*
2. *For every context environment E, the operation: $C \mapsto E[C]$.*
3. *The composition of contexts $(C_1, C_2) \mapsto C_1 \cdot C_2$.*
4. *Substituting a tree in the port of a context: $(C, t) \mapsto C \cdot t$.*
5. *Infinite iteration of a context: $C \mapsto C^\infty$.*
6. *For every letter a, the operations $t \mapsto a(t, \Box)$ and $t \mapsto a(\Box, t)$.*

Items 1 and 2 are by definition. We prove Items 3 to 6:

Item 3: Let C_1, C_2, D_1, D_2 be contexts such that both C_1, D_1 and C_2, D_2 are L-equivalent. Let E be some context environment, we prove: $E[C_1 \cdot C_2] \in L \Leftrightarrow E[D_1 \cdot D_2] \in L$. We construct from E, C_1 and E, D_2 respectively two new environments E_C and E_D such that $E_C[C_2] = E[C_1 \cdot C_2]$ and $E_D[D_1] = E[D_1 \cdot D_2]$. Using the L-equivalence, we have:

$$E_C[C_2] \in L \Leftrightarrow E_C[D_2] \in L$$
$$E_D[D_1] \in L \Leftrightarrow E_D[C_1] \in L$$

Note that by definition of E_C, E_D, $E_C[D_2] = E_D[C_1]$. It follows that $E[C_1 \cdot C_2] \in L \Leftrightarrow E[D_1 \cdot D_2] \in L$. Therefore, $C_1 \cdot C_2$ and $D_1 \cdot D_2$ are L-equivalent.

Item 4: Let C, C' be L-equivalent contexts and t, t' be $L - equivalent$ trees. Let D be a multicontext, we prove: $D[C[t]] \in L \Leftrightarrow D[C'[t']] \in L$. Let D' be the multicontext such that $D'[t] = D[C[t]]$ and E the context environment such that $E[C'] = D[C'[t]]$. Using the L-equivalence we have:

$$D'[t] \in L \Leftrightarrow D'[t'] \in L$$
$$E[C'] \in L \Leftrightarrow E[C] \in L$$

By definition of E, D' we have $D'[t'] = E[C]$. It follows that $D[C[t]] \in L \Leftrightarrow D[C'[t']] \in L$. Therefore, $C[t]$ and $C'[t']$ are L-equivalent.

Item 5: Let C, C' be L-equivalent contexts. Let D be some multicontext, we prove that $D[C^\infty] \in L \Leftrightarrow D[C'^\infty] \in L$. Consider the context environment E constructed from D by replacing each port of D with an infinite chain of ports. Using L-equivalence of C and C' we get:

$$E[C] \in L \Leftrightarrow E[C'] \in L$$

By definition of E we have $E[C] = D[C^\infty]$ and $E[C'] = D[C'^\infty]$. it follows that $D[C^\infty] \in L \Leftrightarrow D[C'^\infty] \in L$. Therefore, C^∞ and C'^∞ are L-equivalent.

Item 6: We only do the proof for $t \mapsto a(t, \Box)$, the other operations is handled symmetrically. Let t, t' be L-equivalent trees. Let E be some context environment, we prove that $E[a(t, \Box)] \in L \Leftrightarrow E[a(t', \Box)] \in L$. By inserting a

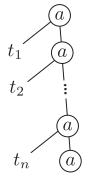
into the ports of E we construct a multicontext C such that for all trees s, $C[s] = E[a(s, \Box)]$. Using L-equivalence of t and t' we get:

$$C[t] \in L \Leftrightarrow C[t'] \in L$$

Therefore, $a(t, \Box)$ and $a(t', \Box)$ are L-equivalent.

C Appendix to Section 4

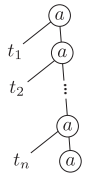
This appendix is devoted to the proof of Fact 1. Recall that L is the set of trees over $\{a, b\}$ of the form:



such that n is some natural number, and for each $i \in \{1, \ldots, n\}$, the tree t_i is either finite, or contains no b nodes.

Fact 1. *Alternator loses $\mathscr{H}_{\in\notin}(L, \infty)$, but wins $\mathscr{H}_{\in\notin}(L, n)$ for every n.*

We first prove that Alternator wins $\mathscr{H}_{\in\notin}(L, n)$ for every n by giving a winning strategy. Fix n, we provide a winning strategy for $\mathscr{H}_{\in\notin}(L, 2n)$.
If $t_1, \ldots, t_n$ are trees (finite or infinite), we write $[t_1, \ldots, t_n]$ for the following tree



- For $i \in \{1, \ldots, n\}$ define L_i to be the set of trees $[t_1, \ldots, t_n]$ such that the trees $t_1, \ldots, t_{i-1}$ are finite and the trees $t_i, \ldots, t_n$ are infinite with only a nodes. Of course, L_i is included in L.
- We define $\bar{L}_i$ like L_i, except that the tree t_i is required to be an infinite tree containing both a and b nodes. Of course, $\bar{L}_i$ is disjoint from L.

Every prefix of a tree in L_i can be completed into a tree in $\bar{L}_i$. Every prefix of a tree in $\bar{L}_i$ can be completed into a tree in L_{i+1}. It follows that Alternator wins the game

$$\mathscr{H}(L_1, \bar{L}_1, L_2, \bar{L}_2, \ldots, L_n, \bar{L}_n)$$

and therefore also Alternator wins $\mathscr{H}_{\in\notin}(L, n)$.

We now show that Constrainer wins $\mathscr{H}_{\in \notin}(L, \infty)$. Consider the tree played by Alternator in the first round. Since this tree belongs to L, it must be of the form $[t_1, \ldots, t_n]$. Let C_1 be a finite prefix of this tree which contains the leaf at depth $n+1$ on the right-most path. Constrainer uses a strategy, which preserves preserves the following invariant.

1. All multicontexts played by Constrainer extend the multicontext C_1. Consequently, all trees played by Alternator are of the form $[s_1, \ldots, s_n]$. Therefore, for $k \in \{1, \ldots, n\}$, it is meaningful to talk about the k-th coordinate of the tree played by Alternator in a round; which refers to the tree s_k.
2. Suppose that Alternator plays a tree $[s_1, \ldots, s_n]$ in some round i. Let C_i be a finite prefix of this tree such that for every coordinate $k \in \{1, \ldots, n\}$:
   - If s_k is finite, then C_i contains the whole subtree s_k;
   - If s_k contains some b, then C_i contains some b in the subtree s_k.

   In the next round Constrainer chooses C_i. Consequently, if $i < j$ are rounds and $k \in \{1, \ldots, n\}$, then
   - If the k-th coordinate of Alternator's tree in round i is finite, then also the k-th coordinate of Alternator's tree in round j is finite.
   - If the k-th coordinate of Alternator's tree in round i contains a b, then also the k-th coordinate of Alternator's tree in round j contains a b.

   the index of trees played by Alternator decreases or stays the same as the rounds progress.

In an odd-numbered round, Alternator's tree belongs to the language, so all coordinates with a b are finite trees. In an even-numbered round, Alternator's tree is outside the language, so at least one coordinate with a b is infinite. Therefore, when going from an odd-numbered round to the next, even-numbered, round, Alternator must change some coordinate from an infinite tree without b's to an infinite tree with b's. It follows, that the number coordinate's with b's increases in each even-numbered round. Since this can happen at most n times, Alternator must lose after at most $2n$ rounds.

D Appendix to Section 5

This section contains proofs of the results from Section 5, with the exception of the implication from 4 to 3 in Theorem 5.3. The implication from 4 to 3 in Theorem 5.3 is proved in Part II of the appendix.

D.1 Proof of Lemma 5.1

Lemma 5.1. *The sets $\mathscr{H}_L$ and $\mathscr{V}_L$ satisfy the closure properties in Table 1.*

All properties are proved by composing strategies, we prove the first one. All other properties are proved similarly. Assume that $(h_1, \ldots, h_n) \in \mathscr{H}_L$, and consider some multicontext C (possibly with infinitely many ports). For all i let L_i be the set of trees that are Alternator's first move in some winning strategy for $\mathscr{H}(h_i, \ldots, h_n)$. Note that since $(h_1, \ldots, h_n) \in \mathscr{H}_L$, L_i is non-empty for all i.

Lemma D.1. *For all $i \leq n$, for all tree t obtained by plugging trees of L_i in the ports of C. Alternator has a winning strategy in $\mathscr{H}(C[h_i], \ldots, C[h_n])$ such the tree chosen in round 1 is t.*

Proof. We proceed by induction on i. For $i = n$ this is obvious. Assume the result holds for i we prove it for $i - 1$.
Let p be a prefix of t, for all subtree s plugged into a port of C, p yields some (possibly empty) prefix p_s of s. By $s \in L_{i-1}$, therefore p_s can be completed into a tree $s' \in L_i$. It follows that p can be completed into t' obtained by plugging trees of L_i in the ports of C. By induction hypothesis, Alternator has winning strategy in $\mathscr{H}(C[h_i], \ldots, C[h_n])$ such the tree chosen in round 1 is t'. Finally we conclude that Alternator has winning strategy in $\mathscr{H}(C[h_{i-1}], \ldots, C[h_n])$ such the tree chosen in round 1 is t. □

The result then follows from Lemma D.1 for $i = 1$.

D.2 Proof of Lemma 5.5

Recall that $\mathscr{V}_L$ contains pairs (u_1, u_2) and (w_1, w_2) such that for some $v \in V_L$,

$$e^\infty \neq e u_1 w_1^\infty \qquad \text{for } e \stackrel{\text{def}}{=} (u_2 w_2^\omega v)^\omega.$$

Lemma 5.5. *Let u_1, u_2, w_1, w_2 and e be as above. If $(h_1, \ldots, h_n) \in \mathscr{H}_L$, then*

$$(e^\infty, eh_1, \ldots, eh_n) \in \mathscr{H}_L \tag{6}$$

$$(eu_1 w_1^\infty, eh_1, \ldots, eh_n) \in \mathscr{H}_L \tag{7}$$

Let us first prove (6). Let E be some context of type e. To win the game

$$\mathscr{H}(e^\infty, eh_1, \ldots, eh_n),$$

player Alternator uses the following strategy. In the first round, he plays the tree E^∞. Suppose that Constrainer picks a finite prefix of E^∞. This finite prefix must be a prefix of E^n for some n. Observe that the type of E^n is also e, because e is idempotent. Alternator removes the whole subtree in the port of E^n, and replaces it by the tree he plays in the first move of the strategy which yields $(h_1, \ldots, h_n) \in \mathscr{H}_L$. He then continues playing this strategy, only substituted in the port of E^n, and never changes a label that in the prefix E^n. It is not difficult to see that the sequence of types seen as a result is $e^\infty, eh_1, \ldots, eh_n$.
Let us prove (7). Let E be some context of type E. By assumption $(u_1, u_2) \in \mathscr{V}_L$, Alternator has a winning strategy in the game $\mathscr{V}(u_1, u_2)$. Let U_1 be the context that Alternator chooses in the first round of this game. This context is of type u_1. Likewise, define a context W_1 for $(w_1, w_2) \in \mathscr{V}_L$. To win the game

$$\mathscr{H}(eu_1w_1^\infty, eh_1, \ldots, eh_n),$$

player Alternator uses the following strategy. He begins by playing the tree $EU_1W_1^\infty$. Suppose that player Constrainer chooses a prefix of this tree. This prefix is included in $EU_1W_1^n$ for some n. Player Alternator uses his strategies in the games $\mathscr{V}(u_1, u_2)$ and $\mathscr{V}(w_1, w_2)$ to change the context $EU_1W_1^n$ into a context C of type $eu_2w_2^n$ which has the same port, and which has the same labels on nodes chosen by Constrainer. Observe that context C depends on Constrainer's move. Let $k \in \mathbb{N}$ be such that $k + n$ is a multiple of ω. To the context C, player appends a context D of type $w_2^k v$, so that the resulting context CD has type e. We then proceed as in the proof of (6): in the port of the context CD, player Alternator plays the winning strategy in the game $\mathscr{H}(h_1, \ldots, h_n)$.

Part II of the Appendix.

Implication from 4 to 3 in Theorem 5.3

This part is devoted to the proof of the implication from 4 to 3 in Theorem 5.3: if both Identities (2) and (3) are satisfied then the set $\mathscr{H}_L$ has bounded alternation. This part is divided in four sections. In order to prove this property, we will need to analyze the set $\mathscr{H}_L$. By definition, $\mathscr{H}_L$ yields a set of strategies for the type alternation game. The first section, Section E, we do a sharp analysis of these strategies and prove that they can always be assumed to be in some normal form that we call locally optimal strategy trees.
In Section F, we use the results of Section E to present the main structure of the proof. Finally, Sections G and H are devoted to the proof of the two main propositions in Section E.

E Strategy trees

Assume fixed a regular language L together with it syntactic morphism $\alpha : (H_A, V_a) \to (H_L, V_L)$.

Type trees. A *type tree* is a tree over the alphabet H_L. For a tree t over the alphabet A, the *type tree induced by* t is a tree with the same nodes as t, where the label of node x is the type of the subtree $t|_x$. Of course if σ is the type tree induced by t, then root label of σ is the type of t.

Global and local consistency. Let σ be a type tree, and let t be a tree over A, both with the same nodes. We say that σ is *locally consistent* with t if for every node x, whose label in t is a:

- If x is a leaf, then the $\sigma(x)$ is the type of the tree a.
- If x has two children, x_l and x_r, then $\sigma(x)$ is the type obtained by applying the letter a to the pair of types $\sigma(x_l)$ and $\sigma(x_r)$.

It is easy to see that the type tree induced by t is locally consistent with t.

Example. Consider an alphabet $A = \{0, 1\}$ and a morphism where $H = V = \{0, 1\}$ and

$$\text{eval}(t) = \begin{cases} 1 & \text{if } t \text{ has infinitely many 1's} \\ 0 & \text{otherwise} \end{cases}.$$

Consider any tree t. Regardless of the choice of t, a tree where all nodes are labeled by 1 is locally consistent with t. Also, a tree where all nodes are labeled by 0 is locally consistent with t, again regardless of t. More generally, a type tree σ is locally consistent with t if and only if the set of nodes in σ that have label 1 is a union of infinite paths.
The following lemma shows that local consistency is preserved under limits.

Lemma E.1. *Let $(t_n)_{n\in\mathbb{N}}$ be a sequence of trees that converges to t_* and let $(\sigma_n)_{n\in\mathbb{N}}$ be a sequence of type trees that converges to σ_*. If σ_n is locally consistent with t_n for every n, then σ_* is locally consistent with t_*.*

Strategy tree. We now introduce the key concept of this paper, which is called a strategy tree. A *strategy tree* is a tuple

$$\sigma = (t, \sigma_1, \ldots, \sigma_n)$$

where

1. t is a tree over A, called the *support* of t;
2. σ_1 is the type tree induced by t;
3. the type trees $\sigma_2, \ldots, \sigma_n$ are locally consistent with t;
4. for each node x, the sequence $(\sigma_2(x), \ldots, \sigma_n(x))$ belongs to $\mathscr{H}_L$.

It follows that the trees $t, \sigma_1, \ldots, \sigma_n$ all have the same nodes, and therefore σ can be interpreted as a single tree over the alphabet $A \times H^n$. The number n is called the *duration* of the strategy tree. We define the *root sequence* of a strategy tree to be the sequence of root labels of $\sigma_1, \ldots, \sigma_n$. If the duration is n, the root sequence is in H^n.
Intuitively speaking, a strategy tree is a special kind of strategy for Alternator. In the first round, Alternator plays the support t. However, Alternator also declares all the types that will appear in nodes of t as the game progresses. More specifically, he declares that for every node $x \in \text{nodes}(t)$ and round $k \in \{2, \ldots, n\}$, he has a strategy so that for the tree in round k, the subtree in node x has type $\sigma_k(x)$.

Proposition E.2. *A sequence belongs to $\mathscr{H}_L$ if and only if it is the root value of some strategy tree.*

The rest of Section E is devoted to proving the proposition.
In the proof, we use a more detailed variant of the game. For a multicontext C, we define the game

$$\mathscr{H}_C(h_1, \ldots, h_n)$$

the same way as $\mathscr{H}(h_1, \ldots, h_n)$, with the added restriction that the first tree played by alternator must extend the multicontext C. This is essentially the same definition as in the proof of Lemma 2.5, but specialized to the case of the topology on trees.

Lemma E.3. *Let C be a finite multicontext. Consider valuations*

$$\eta_1, \ldots, \eta_n : \text{ports}(C) \to H.$$

If $(\eta_1(x), \ldots, \eta_n(x)) \in \mathscr{H}_L$ for every port x, then Alternator wins the game

$$\mathscr{H}_C(C[\eta_1], \ldots, C[\eta_n]).$$

Proof. This is a refinement of the first property in Lemma 5.1. For every port x of C, Alternator has a winning strategy for the game $\mathscr{H}(\eta_1(x), \dots, \eta_n(x))$.
We describe a winning strategy for Alternator in $\mathscr{H}_C(C[\eta_1], \dots, C[\eta_n])$. Alternator starts with a tree obtained by plugging the initial tree for his winning strategy on $\mathscr{H}(\eta_1(x), \dots, \eta_n(x))$ in every port x of C. Then every prefix of this tree, yields a prefix for each subtree plugged into a port of C, Alternator can then use his strategies for the games $\mathscr{H}(\eta_1(x), \dots, \eta_n(x))$ to answer. □

We now prove Proposition E.2. We state a more refined version of the proposition which can be proved by induction.

Lemma E.4. *For a sequence $(h_1, \dots, h_n) \in H^*$ and a finite multicontext C the following conditions are equivalent:*

1. *Alternator wins $\mathscr{H}_C(h_1, \dots, h_n)$;*
2. *There is a strategy tree whose support extends C, and whose root sequence is $(h_1, \dots, h_n)$.*

Proof. Induction on n. The base case of the induction, when $n = 0$, is trivial. We do the induction step.
Let us begin with the easier implication from 2 to 1. Suppose that

$$(t, \sigma_1, \dots, \sigma_n)$$

is a strategy tree as in item 2. Alternator's strategy is as follows. In the first round, she plays the tree t, which has type h_1. Suppose Constrainer chooses a prefix D of t. Consider the valuations

$$\eta_2, \dots, \eta_n : \mathrm{ports}(D) \to H \qquad \eta_i(x) = \sigma_i(x).$$

For every $i \in \{2, \dots, n\}$, the root label of σ_i is the same as $D[\eta_i]$, because σ_i is locally consistent with t and D is a prefix of t. By Lemma E.3, Alternator wins the game

$$\mathscr{H}_D(D[\eta_2], \dots, D[\eta_n]).$$

This shows that for every choice of D, Alternator has a winning strategy in the remaining part of the game.
The rest of this proof concerns the more difficult implication from 1 to 2.
Suppose that player Alternator wins $\mathscr{H}_C(h_1, \dots, h_n)$. Let t be the tree of type h_1 that Alternator plays in the first round. This tree has prefix C. Also, for every finite prefix D of t, player Alternator wins $\mathscr{H}(D, h_2, \dots, h_n)$. By the induction assumption, for every finite prefix D of t, there is a strategy tree

$$\sigma_D = (t_D, \sigma_{D2}, \dots, \sigma_{Dn})$$

such that t_D has prefix D, and the root sequence of σ_D is $(h_2, \dots, h_n)$.
A sequence of finite multicontexts $(D_i)_{i \in \mathbb{N}}$ is said to converge to t if all of the multicontexts are prefixes of t, and for every $j \in \mathbb{N}$, only finitely many multicontexts have some port at depth at most j. By compactness, there is an infinite

sequence of finite multicontexts $(D_i)_{i\in\mathbb{N}}$ which converge to the tree t and such that all of the sequences

$$(t_{D_i})_{i\in\mathbb{N}} \qquad (\sigma_{D_i 2})_{i\in\mathbb{N}} \qquad \dots \qquad (\sigma_{D_i n})_{i\in\mathbb{N}}$$

are convergent. Let the limits of these sequences be

$$t_* \qquad \sigma_{*2} \qquad \dots \qquad \sigma_{*n}.$$

Because the sequence $(D_i)_{i\in\mathbb{N}}$ converges to t, it follows that $t_* = t$. For each D, the type trees

$$(\sigma_{D2}, \dots, \sigma_{Dn})$$

are locally consistent with t_D. Therefore, by Lemma E.1 it follows that the limits $\sigma_{*2}, \dots, \sigma_{*n}$ are locally consistent with t. Finally, define σ_{*1} to be the unique type tree that is globally consistent with t. We have just proved that

$$(\sigma_{*1}, \dots, \sigma_{*n})$$

is a strategy tree. Because root values are preserved under limits, the root value of this strategy tree is the desired $(h_1, \dots, h_n)$. □

E.1 Locally optimal strategy trees

In the proof, we will be using a special kind of strategy trees, called locally optimal strategy trees.

Distance. So far we have used the distance

$$\mathit{distance}(s,t) = \sup\{\frac{1}{2^n} : t \text{ and } s \text{ differ on depth } n\},$$

which we call the *prefix distance*. In the definition of locally optimal strategy trees, it will be more convenient to use a different distance, called the *discounted distance*, which is defined below. Fix some enumeration $x_1, x_2, \dots$ of all the nodes of the infinite complete binary tree. For two trees s and t and a node x, we define

$$\mathit{distance}(t(x), s(x)) = \begin{cases} 0 & \text{if } t \text{ and } s \text{ are undefined in } x \\ 0 & \text{if } t \text{ and } s \text{ are defined in } x, \text{ and have the same label} \\ 1 & \text{otherwise} \end{cases}$$

The discounted distance on trees is defined by

$$\mathit{distance}_\lambda(s,t) = \sum_{n\geq 1} \frac{1}{2^n} \cdot \mathit{distance}(s(x_n), t(x_n)),$$

The following lemma is a well-known result about the Cantor space.

Lemma E.5. *Regardless of the enumeration $x_1, x_2, \ldots$, the prefix and discounted distances yield the same topology.*

Proof. We prove that the discounted distance yields the prefix topology. Fix some arbitrary enumeration $x_1, \ldots, x_n, \ldots$. There are two directions.

First, we prove that open sets for the prefix topology are also open sets for the discounted distance. Let C be a finite multicontext and $C[*]$ be the corresponding base open set. We show that $C[*]$ can be defined as an infinite union of open balls for the discounted distance.
We set X as the set of nodes x of the infinite binary tree such that for every two trees s, t with prefix C, $distance(t(x), s(x)) = 0$. Since C is finite, X must also be finite. Let n be the largest integer such that $x_n \in X$. Finally, consider the set:

$$L_\lambda = \bigcup_{t \in C[*]} \{s \mid distance_\lambda(s,t) < \frac{1}{2^n}\}$$

We show that $L_\lambda = C[*]$. By definition $C[*] \subseteq L_\lambda$. Let $s \in L_\lambda$, by definition $s \in \{s \mid distance_\lambda(s,t) < \frac{1}{2^n}\}$ for some $t \in C[*]$. For all $x \in X$, $distance(t(x), s(x)) = 0$ (recall that all $x \in X$ are among the n first positions in the enumeration). Therefore, $s\ in C[*]$. We get L_λ.

For the other direction, we prove that open sets for the discounted distance are also open sets for the prefix topology. Let $L_\lambda = \{s \mid distance_\lambda(s,t) < k\}$ be an open ball with t a tree and k some real number. We prove that L_λ is an infinite union of open sets for the prefix topology. For all trees $s \in L_\lambda$ we select one of its prefixes. Then we prove that the open set obtained from this set of prefixes is L_λ.
Let $s \in L_\lambda$ and let $d = distance_\lambda(s,t)$, by definition of L_λ, $d < k$. Let n be the smallest integer such that:

$$\sum_{i>n} \frac{1}{2^n} < k - d$$

Let p be some prefix of s that contains all nodes of the enumeration up to x_n. Note for all trees s' with prefix p, by definition of n $distance_\lambda(s,s') < k - d$ and therefore $distance_\lambda(s',t) < k$. Finally consider P, the set of all such prefixes for all $s \in L_\lambda$. Let L be the open set obtained from P. By definition, $L_\lambda \subseteq L$. Conversely, if $s' \in L$, by definition, s' has a prefix $p \in P$ and $distance_\lambda(s',t) < k$, i.e. $s' \in L_\lambda$. We conclude $L = L_\lambda$. □

Locally optimal strategy trees. We now define locally optimal strategy trees. Consider a strategy tree $(t, \sigma_1, \ldots, \sigma_n)$ and let $v \in V$. The strategy tree is called *locally optimal for* v if for every $i \in \{2, \ldots, n\}$, and for every type tree σ'_i, if σ'_i is locally consistent with t and $v \cdot val(\sigma'_i) = v \cdot val(\sigma_i)$, then

$$distance(\sigma_{i-1}, \sigma_i) \leq distance(\sigma_{i-1}, \sigma'_i)$$

Lemma E.6. *For every strategy tree σ, for every $v \in V_L$ there exists a locally optimal one for v such that*

$$(v \cdot val(\sigma_1), \dots, v \cdot val(\sigma_n)) = (v \cdot val(\sigma'_1), \dots, v \cdot val(\sigma'_n))$$

Proof. Suppose that $(\sigma_1, \dots, \sigma_n)$ is a strategy tree with support t. Consider the set Σ_2, of strategy trees that are locally consistent with t and which have the same root value as σ_2. Thanks to Lemma E.1, this is a closed set. As a closed subset of the compact space of all trees, the set Σ_2 is compact. It follows some element of Σ_2 minimizes the distance with respect to σ_1. We choose this element as the new σ_2, and then proceed likewise for the remaining coordinates $3, \dots, n$. □

When v is the neutral element, we just say locally optimal. In that case Lemma E.6 can be rephrased: for every strategy tree, there exists a locally optimal one with the same root sequence.
Moreover notice that if a strategy tree is locally optimal for some v then it also locally optimal for the neutral element. We state one last lemma, which represents a property of locally optimal strategy trees that will be used repeatedly.

F Proof plan

Now that locally optimal strategy trees are defined we can prove the implication from 4 to 3 in Theorem 5.3. Note that this section contains only the outline of the proof which is based on two very involved propositions which are proved in the last two remaining sections.
Assume fixed L a regular language and $\alpha : (H_A, V_A) \to (H_L, V_L)$ the associated syntactic morphism. Assume that L satisfies Identities (2) and (3). We need to prove that $\mathscr{H}_L$ has bounded alternation.
The proof involves two parameters of strategy trees that we define now:

- We define the *root alternation* of a strategy tree σ to be the alternation of its root sequence.
- We define the *limit alternation* of σ to be the maximal number k such that infinitely many subtrees of σ have root alternation k.

Let Σ_L be the set of all locally optimal strategy trees for the language L. Observe that by Proposition E.2 and Lemma E.6, $\mathscr{H}_L$ has bounded alternation iff Σ_L has bounded root alternation.
This last property is a consequence of the two following lemmas that are respectively consequences of Identity (2) and Identity (3):

Proposition F.1. *Let Σ be a set of locally optimal strategy trees with bounded limit alternation. Then Σ has bounded root alternation.*

Proposition F.2. *Assume there exists a set Σ of locally optimal strategy trees with unbounded root alternation. Then there exists a set of strategy trees Σ' with both unbounded root alternation and bounded limit alternation.*

The proofs can be found respectively in Sections H and G. We finish this section by using them to prove that Σ_L has bounded root alternation. We proceed by contradiction, assume that Σ_L has unbouned root alternation. By Proposition F.2, this means that there exists a set of locally optimal strategy trees Σ' with both unbounded root alternation and bounded limit alternation. But his contradicts Proposition F.1. Therefore, the root alternation of Σ_L is bounded and so is the alternation of $\mathscr{H}_L$.

G Limit alternation is unbounded

This section is devoted to showing Proposition F.2.

(TODO rewrite this whole section)

We proceed in two steps, using a new object called the *strategy graph* G_L of a language L. The strategy graph represents a subset of the strategies available to Alternator in the game on tree types. We then show that if Proposition F.2 does not hold, then the strategy graph of L verify some special property on its strongly connected components that violates identity (2).

Strategy Graph. We define a graph G_L depending on the language L, as follows. The nodes are $V_L \times H_L$. From a node (v, h) there is an edge to a node (v', h') if there exists:

$$(u_1, u_2), (w_1, w_2) \in \mathscr{V}_L$$

such that

$$h = vu_1w_1^\infty \qquad \text{and} \qquad v' = vu_2w_2^\omega.$$

Intuitively, there is an edge from (v, h) to (v', h') iff there exists a tree t of type h such that:

- t can be decomposed as the concatenation of a context of type v and another tree.
- Every prefix of t can be completed into a context of type v'.

Observe that if there exists nodes $(v_1, h_1), \dots, (v_n, h_n)$ such that for all i, there is an edge between (v_i, h_i) and (v_{i+1}, h_{i+1}), then $(h_1, \dots, h_n) \in \mathscr{H}_L$. In that case, we say that $(h_1, \dots, h_n)$ is *represented* in G_L. (note that not all sequences in $\mathscr{H}_L$ are necessarily represented in G_L).
The strategy graph is said to be *recursive* iff there exists a strongly connected component that contains two nodes (v, h) and (v', h') with $h \neq h'$.

Lemma G.1. *If G_L is recursive, then identity* (3) *is violated.*

Proof. We proceed by contradiction. Assume that identity (3) is verified and fix two nodes (v, h) and (v', h') that are in the same strongly component of G_L. We prove that $h = h'$ and therefore that G_L is not recursive.

It is not difficult to see that if there is a path from (v, h) to (v', h'), then there is also a single edge. This means that there are

$$(u_1, u_2), (w_1, w_2) \in \mathscr{V}_L$$

such that

$$h = vu_1 w_1^\infty \qquad \text{and} \qquad v' = vu_2 w_2^\omega.$$

Likewise, there must be

$$(u_1', u_2'), (w_1', w_2') \in \mathscr{V}_L$$

such that

$$h' = v'u_1'(w_1')^\infty \qquad \text{and} \qquad v = v'u_2'(w_2')^\omega.$$

From identity (3) it follows that

$$\begin{aligned}(u_2(w_2)^\omega u_2'(w_2')^\omega)^\omega u_1(w_1)^\infty &= (u_2(w_2)^\omega u_2'(w_2')^\omega)^\infty \\ (u_2'(w_2')^\omega u_2(w_2)^\omega)^\omega u_1'(w_1')^\infty &= (u_2'(w_2')^\omega u_2(w_2)^\omega)^\infty\end{aligned}$$

If we apply v to the first equality and v' to the second, we get:

$$\begin{aligned}h &= v(u_2(w_2)^\omega u_2'(w_2')^\omega)^\infty \\ h' &= v'(u_2'(w_2')^\omega u_2(w_2)^\omega)^\infty = v(u_2(w_2)^\omega u_2'(w_2')^\omega)^\infty\end{aligned}$$

We conclude $h = h'$. □

Given Lemma G.1, Proposition F.2 follows from the following lemma:

Lemma G.2. *If Proposition F.2 does not hold, then the graph G_L is recursive.*

The rest of this section is devoted to the proof of Lemma G.2. We do this in two steps, in G.1, we define the notion of context zone which allows to extract contexts from strategy trees as well as sequences in $\mathscr{V}_L$. Then in G.2 we provide the formal proof of Lemma G.2.

G.1 Context Zones

A *context zone* is a set X of nodes in a tree such that for some node x (called the root of X) and some node y (called the port of X), the set X contains the nodes that are in the subtree of x, but not in the subtree of y. It is not difficult to see that when X is nonempty, the root and port of X can be uniquely defined in terms of X. We say that context zones $X_1, \ldots, X_n$ are consecutive if for each $i \in \{1, \ldots, n-1\}$, the port of X_i is the root of X_{i+1}. The union of consecutive context zones is a context zone.
Let $\sigma = (t, \sigma_1, \ldots, \sigma_n)$ be a strategy tree, and let X be a context zone. For a context zone X and a round $i \in \{1, \ldots, n\}$, we define

$$val(\sigma, X, i) \in V_L$$

as follows by induction on the number of ancestors of the port that are contained in X. If there is only one such ancestor, the port of X is a child of its root. Let a be the label of x and $y \in X$ be the other child of x. We set $val(\sigma, X, i)$ as $a(\Box, \sigma_i(y))$ if y is a right child and $a(\sigma_i(y), \Box))$ if y is a left child. Otherwise, let x be the root of X, y its port and z the child of x which is an ancestor of y. Let X_1 be the context node of root x and port z and X_2 the context zone of root z and port y. We set $val(\sigma, X, i)$ as $val(\sigma, X_1, i) \cdot val(\sigma, X_2, i)$.

Fact 3 *Let $\sigma = (t, \sigma_1, \ldots, \sigma_n)$ be a strategy tree and X be a context zone in σ. Then $(val(\sigma, X, 1), \ldots, val(\sigma, X, n)) \in winvl$.*

Proof. This is by construction and Lemma 5.1. □

G.2 Proof of Lemma G.2

Assume Proposition F.2 does not hold, then there exists a set of locally optimal strategy trees Σ with unbounded root alternation and all sets of locally optimal strategy trees Σ' with unbounded root alternation also have unbounded limit alternation. Note that it follows from the existence of Σ and Proposition E.2 that $\mathscr{H}_L$ has unbounded alternation.

We proceed has follows, we define a special kind of node for the graph G_L and prove that: a) by hypothesis, G_L must contain such a node, b) if G_L contains such a node, then it is recursive for a strongly connected component reachable from this node.
A node (v, h) in the graph G_L is called *alternating* if $v \cdot \mathscr{H}_L$ contains words that begin with h and have arbitrarily high alternation.

Lemma G.3. *G_L contains at least one alternating node.*

Proof. This is because $\mathscr{H}_L$ has unbounded alternation. Therefore, by pigeonhole principle there exists some $h \in H_L$ such that there $\mathscr{H}_L$ contains words that begin with h and with arbitrarily high alternation. By definition, this means that $(1_{V_L}, h)$ is alternating. □

Lemma G.4. *Every alternating node (v, h) in G_L has an outgoing edge to some alternating node (v', h') such that $h \neq h'$.*

Before we prove the lemma, we use it together with Lemma G.3 to prove that G_L is recursive and end the proof of Lemma G.2. Combining Lemmas G.3 and G.4, we obtain that there exists an infinite path in the graph G_L

$$(v_1, h_1), (v_2, h_2), \dots \qquad \text{such that } h_i \neq h_{i+1} \text{ for all } i.$$

By pigeon-hole principle, there exists at least one node (v_i, h_i) which is repeated infinitely many times in this path. By definition, this means that (v_i, h_i) and (v_{i+1}, h_{i+1}) are in the same strongly connected component. Moreover, since $h_i \neq h_{i+1}$, we conclude that G_L is recursive.
The rest of this section is now devoted to the proof of Lemma G.4.

Proof. Let (v, h) be an alternating node in the graph G_L. We need to construct an alternating node (v', h') reachable from (v, h) and such that $h \neq h'$. This is done in two steps, first we use our hypothesis to construct a special set of strategy trees, Σ_v with unbounded limit alternation. In the second step, we select a strategy tree with large enough limit alternation in this set and use it to construct (v', h').

Construction of Σ_v. Since (v, h) is alternating, for every $n \in \mathbb{N}$ the set $\mathscr{H}_L$ contains a sequence $h_1, \dots, h_n$ such that

- $h = vh_1$
- the sequence $vh_1, \dots, vh_n$ is alternating (for all i, $vh_i \neq vh_{i+1}$).

Let $\mathscr{H}_{(v,h)} \subseteq \mathscr{H}_L$ be the set of these sequences.

Lemma G.5. *There exists a set of strategy trees Σ_v such that:*

1. *All strategy trees in Σ_v are locally optimal for v.*
2. *For all $(h_1, \dots, h_n) \in \mathscr{H}_{(v,h)}$, there exists a root sequence $(h'_1, \dots, h'_n)$ of a strategy tree in Σ_v such that $(vh_1, \dots, vh_n) = (vh'_1, \dots, vh'_n)$.*

Proof. Σ_v can be constructed in two steps. First, we know from Proposition E.2 that there exists a set of strategy trees Σ such that: $(h_1, \dots, h_n) \in \mathscr{H}_{(v,h)}$ iff $(h_1, \dots, h_n)$ is the root sequence of some strategy tree in Σ. Applying Lemma E.6 to Σ then yields Σ_v. □

Fact 4 *Σ_v has unbounded limit alternation.*

Proof. By construction, Σ_v has unbounded root alternation and contains only locally optimal trees. By hypothesis, this means that it must have unbounded limit alternation.

Construction of (v', h'). We now choose a strategy tree in Σ_v with large enough limit alternation and use it to construct the desired alternating node. Let K be the set of numbers k such that

$$(g,h)^k \in \mathscr{H}_L \quad \text{and} \quad (g,h)^{k+1} \notin \mathscr{H}_L \qquad \text{for some } g \neq h \in H_L.$$

The set K is finite, because for every choice of $g \neq h$, the above condition can hold for at most one k. Set:

$$l = |H_L|^2 \cdot (\max(K) + 1)$$

Finally, let $\sigma = (t, \sigma_1, \ldots, \sigma_n)$ be a strategy tree in Σ_v with limit alternation at least l (such a strategy tree exists because of Fact 4). We construct (v', h') from σ in several steps. We begin by proving a result that we will reuse several times in the proof in order to prove that special nodes of σ have alternation l. Then we show that there exists an infinite path in σ such that all nodes in the path are labeled with sequences of alternation l. We extract an infinite subset of nodes of the path that all share the same sequence. Finally we use this last sequence to construct (v', h').

Lemma G.6. *Let $\sigma = (t, \sigma_1, \ldots, \sigma_n)$ be a locally optimal strategy tree and x, y two nodes of σ such that x is a descendant of y. Then for all $i \leq n$:*

$$\sigma_i(x) \neq \sigma_{i+1}(x) \Rightarrow \sigma_i(y) \neq \sigma_{i+1}(y)$$

Proof. Assume that $\sigma_i(x) \neq \sigma_{i+1}(x)$ and $\sigma_i(y) = \sigma_{i+1}(y)$. We show that this contradicts local optimality. Consider the type tree σ'_{i+1} defined as follows:

- For all descendants z of y, $sigma'_{i+1}(z) = \sigma_i(z)$.
- For all other nodes z, $sigma'_{i+1}(z) = \sigma_{i+1}(z)$

By construction, σ'_{i+1} is locally consistent with t (this is by definition for all nodes $z \neq y$, and because $\sigma_i(y) = \sigma_{i+1}(y)$ for y). Note that by definition, for all nodes z:

$$distance(\sigma_i(z), \sigma'_{i+1}(z)) \leq distance(\sigma_i(z), \sigma_{i+1}(z))$$

Moreover, $distance(\sigma_i(x), \sigma'_{i+1}(x)) = 0$ and $distance(\sigma_i(x), \sigma_{i+1}(x)) = 1$. Combining all this we obtain:

- $distance_\lambda(\sigma_i, \sigma'_{i+1}) < distance_\lambda(\sigma_i, \sigma_{i+1})$.
- $val(\sigma'_{i+1}) = val(\sigma_{i+1})$.

This contradicts local optimality of σ. □

We can now extract the path π of σ:

Lemma G.7. *There exists an infinite path π of σ such that for all $x \in \pi$, $(\sigma_1(x), \ldots, \sigma_n(x))$ has alternation at least l.*

Proof. Consider the set Z of nodes z that satisfy:

$$limitalternation(\sigma|_z) \geq l$$

By definition, the root of σ must be in Z and every node in Z must have a child in Z. Therefore there exists an infinite path π in Z. We prove that for all nodes $x \in \pi$, $(\sigma_1(x), \ldots, \sigma_n(x))$ has alternation l. This is by local optimality. By definition, all $x \in \pi$ have a descendant y that has alternation at least l. Therefore, by Lemma G.6 x must have alternation at least l. □

Now we extract $X \subseteq \pi$ such all nodes in X are labeled with the same sequence in $\mathscr{H}_L$.

Lemma G.8. *There exists an infinite set of nodes $X \subseteq \pi$ and $(h_1, \ldots, h_n) \in \mathscr{H}_L$,$(v_1, \ldots, v_n) \in \mathscr{V}_L$ such that:*

- *For all $x \in X$, $(\sigma_1(x), \ldots, \sigma_n(x)) = (h_1, \ldots, h_n)$.*
- *For all $x \in X$, $(val(\sigma, Y, 1), \ldots, val(\sigma, Y, n)) = (v_1, \ldots, v_n)$ (where Y is the context zone with the root of σ as root and x as port).*

Proof. There are finitely many sequences within H_L^n and V_L^n and π is infinite, therefore, we can then extract an infinite subset X from π such that for all $x \in X$ the sequences

$$(\sigma_1(x), \ldots, \sigma_n(x))$$
$$(val(\sigma, x, 1), \ldots, val(\sigma, x, n))$$

are independent from the choice of x. Choose, $(h_1, \ldots, h_n)$ and $(v_1, \ldots, v_n)$ as these sequences. By definition $(h_1, \ldots, h_n) \in \mathscr{H}_L$ and it follows from Fact 3 that $(v_1, \ldots, v_n) \in \mathscr{V}_L$. □

We finish by constructing (v', h') and proving that it satisfies the required properties. Recall that $l = |H_L|^2 \cdot (\max K + 1)$, therefore, since $alternation(h_1, \ldots, h_n) \geq l$, it follows from a pigeon-hole principle argument that there exists $g \neq g' \in H_L$ and a set $I \subseteq \{1, dots, n-1\}$ of size at least $\max(K) + 1$ such that:

$$h_i = g \qquad \text{and} \quad h_{i+1} = g' \qquad \text{for every } i \in I.$$

It follows that the word $(h_1, \ldots, h_n) \in \mathscr{H}_L$ contains a subsequence which alternates between g and g' at least $\max(K) + 1$ times. By choice of K, it follows that:

$$(g, g')^k \in \mathscr{H}_L \qquad \text{for all } k \in \mathbb{N}.$$

Choose some arbitrary $i \in I$, say the first element in I and set:

$$u = v_i$$
$$u' = v_{i+1}$$

Consider the two nodes $(vu', vu'g')$ and $(vu', vu'g)$ in G_L:

Lemma G.9. *The three following Properties hold:*

1. $vu'g \neq vu'g'$.
2. $(vu', vu'g')$ *and* $(vu', vu'g)$ *are both alternating.*
3. G *contains edges from* (v, h) *to both nodes.*

Before proving Lemma G.9 we use it to conclude the proof of Lemma G.4. Indeed, Item 1 means that either $vu'g \neq h$ or $vu'g' \neq h$. It follows that there exists (v', h') alternating, reachable from (v, h) and such that $h \neq h'$.
We finish with the proof of Lemma G.9.

Proof. **Item 1.** Recall that by construction in Lemma G.5, σ is locally optimal for v. Let x be any node in X, say the topmost $x \in X$. We construct a type tree i, σ'_i with root label $u'g$ and such that:

$$distance_\lambda(\sigma_i, \sigma'_i) < distance_\lambda(\sigma_i, \sigma_{i+1}).$$

Since, the root label of σ_{i+1} is $v_{i+1}h_{i+1} = u'g'$ it then follows by local optimality for v of σ that $u \neq u'g'$.
Consider the type tree σ'_i locally consistent with t and such that:

$$\sigma'_i(z) = \sigma_i(z) \quad \text{when } z \text{ is in the subtree of } x$$
$$\sigma'_i(z) = \sigma_{i+1}(z) \text{ when } z \text{ and } x \text{ are incomparable}$$

The labels of strict ancestors of x are then obtained inductively by local consistency with t. Note that by definition the root label of σ'_i is $val(\sigma, x, i+1)\cdot\sigma_i(x) = u'g$. We now prove that:

$$distance_\lambda(\sigma_i, \sigma'_i) < distance_\lambda(\sigma_i, \sigma_{i+1}).$$

This is because for every node z of σ:

$$distance(\sigma_i(z), \sigma'_i(z)) \leq distance(\sigma_i(z), \sigma_{i+1}(z))$$

This is true by definition for all nodes that are not ancestors of y. Moreover, $distance(\sigma_i(x), \sigma_{i+1}(x)) = 1$ therefore it follows from Lemma G.6 that this is also the case for all ancestors of x. Finally, by construction $distance(\sigma_i(x), \sigma'_i(x)) = 0$. By definition of $distance_\lambda$, it follows that:

$$distance_\lambda(\sigma_i, \sigma_i') < distance_\lambda(\sigma_i, \sigma_{i+1}).$$

Item 2. This is actually a consequence of Item 1. Recall that:

$$(g, g')^k \in \mathscr{H}_L \qquad \text{for all } k \in \mathbb{N}.$$

Therefore, it follows from Item 1 and Lemma 5.1 that:

$$(vu'g, vu'g')^k \in \mathscr{H}_L \qquad \text{for all } k \in \mathbb{N}.$$

And we are finished for Item 2.

Item 3. Note that by definition, the existence of an edge between (v, h) and (v', h') does not depend on h'. Therefore it is sufficient to prove that there is an edge between (v, h) and $(vu', vu'g')$.
Recall from Lemma G.8 that $(v_1, \ldots, v_n) \in \mathscr{V}_L$, therefore $(v_1, u') = (v_1, v_{i+1}) \in \mathscr{V}_L$.
Apply the Ramsey theorem to the function

$$f : \binom{X}{2} \to V_L^2$$

which maps a pair of nodes $x < y$ in the chain X to the pair

$$(val(\sigma, Y, 1), val(\sigma, Y, i+1)) \in V_L^2.$$

where Y is the context zone with root x and port y. We know from Fact 3 that the range of the function f is included in $\mathscr{V}_L$. From the Ramsey theorem we get an infinite subset $Z \subseteq X$ such that all pairs from Z are mapped to the same value, call it

$$(w_1, w') \in \mathscr{V}_L$$

Let $x < y$ be two nodes in Z and Y the context zone containing nodes that are in the subtree of x but not in the subtree of y. By Lemma G.8 we have:

$$\begin{array}{lll} v_1 = val(\sigma, y, 1) & = val(\sigma, x, 1) \cdot val(\sigma, Y, 1) & = v_1 \cdot w_1 \\ u' = val(\sigma, y, i+1) & = val(\sigma, x, i+1) \cdot val(\sigma, Y, i+1) & = u' \cdot w' \end{array}$$

Because σ_1 is the type tree of t, its root label is $v_1 w_1^\infty$. Therefore by definition of $\sigma \in \Sigma_v$:

$$h = vv_1 w_1^\infty.$$

Finally, because $u'w' = u'$, we have

$$vu' = vu'(w')^\omega.$$

By definition, there is an edge between (v, h) and $(vu', vu'g')$.

□

This completes the proof of Lemma G.4. □

H Limit alternation is bounded

This section is devoted to showing Proposition F.1.
We do this in two steps, using a new object called *strategy matrices* as an intermediary. Strategy matrices represent special strategies for Alternator in the game on tree types. In our first step, we show that if Proposition F.1 does not hold, then there exists special strategy matrices of arbitrarily large size. Then we show that the existence of a sufficiently large special strategy matrix violates identity (2).

Strategy matrices. A *strategy matrix* is a rectangular matrix with entries from V_L such that where row belongs to $\mathscr{V}_L$.
Consider a strategy matrix with n rows and k columns. The idea is that the matrix represents the evolution of k contexts in a strategy for alternator with n rounds. The entry in the i-th row and j-th column represents the value of the i-th context in the j-th round. Typically such a strategy matrix is produced from a strategy tree of duration n, and k consecutive context zones.
A strategy matrix M is called *parity alternating* if for some $n \in \mathbb{N}$, if it has it has $2n$ columns and n rows, and

a) For every $i \in \{1, \ldots, n\}$,
   - Columns $2i-1$ and $2i$ have the same entries in all rows except for row i.
   - The values of columns $2i-1$ and $2i$ are different.

b) Condition a) holds when the order of columns is reversed.

In case *a*) holds the matrix is called *top-down* and in case *b*) holds, it is called *bottom-up*. The set of parity alternating matrices with n rows and $2n$ columns is denoted by $\mathbb{P}_n$.
Proposition F.1 follows from the following two lemmas.

Lemma H.1. *If Proposition F.1 does not hold, then $\mathbb{P}_n$ is nonempty for all n.*

Lemma H.2. *If $\mathbb{P}_n$ is nonempty for all n, then identity* (2) *is violated.*

The lemmas are proved in the two following sections.

H.1 Proof of Lemma H.1

If Proposition F.1 does not hold, then there exists an infinite set Σ of locally optimal strategy trees such that:

$$\begin{aligned} \sup_{\sigma\in\Sigma} limitalternation(\sigma) &< \infty \\ \sup_{\sigma\in\Sigma} rootalternation(\sigma) &= \infty \end{aligned}$$

Fix some arbitrary $n \in \mathbb{N}$, we need to construct a strategy matrix $M \in \mathbb{P}_n$. We proceed as follows: first we choose a strategy tree $\sigma \in \Sigma$ with big enough root alternation. Then we split the support of σ into n consecutive context zones; which yields a strategy matrix with n rows. By modifying that matrix a bit, and using local optimality of σ, we get a matrix in $\mathbb{P}_n$.
Let l be the maximal limit alternation among strategy trees in σ. Choose a strategy tree $\sigma \in \Sigma$ with root alternation at least $l \cdot 2^{n^2}$ and let m be the duration of σ.

Lemma H.3. *There are consecutive contexts zones $X_1, \ldots, X_n$ in σ and a sequence of rounds $i_1, \ldots, i_n \in \{1, \ldots, m-1\}$ such that the sequence of rounds is either strictly increasing or strictly decreasing, and*

$$val(\sigma, i_j, X_j) \neq val(\sigma, i_j + 1, X_j) \qquad \text{for every } j \in \{1, \ldots, n\}$$

Proof. For two infinite paths π and π' in σ, define $diverge(\pi, \pi')$ to be the deepest common node in both the paths. For a round $i \in \{1, \ldots, m-1\}$, define:

$$change_i(\sigma) = \{x \in \text{nodes}(\sigma) : \sigma_i(x) \neq \sigma_{i+1}(x)\}.$$

Lemma H.4. *Let X be a context zone, and let π be an infinite path that passes through the root of X, but not through the port. Then*

$$\pi \subseteq change_i(\sigma) \quad \Rightarrow \quad val(\sigma, i, X) \neq val(\sigma, i+1, X)$$

holds for every round $i \in \{1, \ldots, m-1\}$.

Proof. This is by local optimality of σ. Assume that $\pi \subseteq change_i(\sigma)$ and $val(\sigma, i, X) = val(\sigma, i+1, X)$ for some round i. We construct a type tree that is closer to σ_i than σ_{i+1} regarding the discounted distance and with the same root value as σ_{i+1}, contradicting local optimality.
Consider the type tree σ'_{i+1} defined as follows:

- For nodes $x \in X$ $\sigma'_{i+1}(x) = \sigma_i(x)$. By definition:

$$distance(\sigma_i(x), \sigma'_{i+1}(x)) = 0 \leq distance(\sigma_i(x), \sigma_{i+1}(x))$$

- For other nodes y $\sigma'_{i+1}(y) = \sigma_{i+1}(y)$. By definition:

$$distance(\sigma_i(y), \sigma'_{i+1}(y)) = distance(\sigma_i(y), \sigma_{i+1}(y))$$

Because $\pi \in change_i(\sigma)$, there exists at least one node $x \in X$ such that $distance(\sigma_i(x), \sigma_{i+1}(x)) = 1$. It follows that:

$$distance_\lambda(\sigma_i, \sigma'_{i+1}) < distance_\lambda(\sigma_i, \sigma_{i+1})$$

Moreover since $val(\sigma, i, X) = val(\sigma, i+1, X)$, we have $val(\sigma'_{i+1}) = val(\sigma_{i+1})$. This contradicts local optimality of σ. □

Lemma H.5. *There is a set Π of at least 2^{n^2} infinite paths, and a function*

$$change : \Pi \to \{1, \ldots, m-1\}$$

such that

$$\pi \subseteq change_{change(\pi)}(\sigma) \qquad \text{for every } \pi \in \Pi.$$

Proof. By definition of a tree strategy, every node in $change_j(\sigma)$ has at least one child in $change_j(\sigma)$. Therefore, each of the nonempty sets $change_j(\sigma)$ contains at least one infinite path. By the assumption on the root alternation there are at least $l \cdot 2^{n^2}$ rounds j where $change_j(\sigma)$ is nonempty. By the assumption on limit alternation, an infinite path can be contained in sets $change_j(\sigma)$ for at most l different values of j. This proves the statement of the lemma. □

Lemma H.6. *Let Π be a set of at 2^k infinite paths in a binary tree. There exist consecutive context zones $X_1, \ldots, X_k$ and paths $\pi_1, \ldots, \pi_k \in \Pi$ such that for every $i \in \{1, \ldots, k\}$, path π_i passes through the ports of contexts $X_1, \ldots, X_{i-1}$, but not through the port of X_i.*

Proof. The proof is by induction on k. The induction base of $k = 1$ is obvious.
Now assume that this holds for k and consider a set Π of 2^{k+1} paths. Let x be the deepest node in the tree that belongs to all paths of Π (x exists since the root belongs to all paths of Π). Let Π_r (respectively, Π_l) be those paths in Π that pass through the right child of x (respectively, the left child of x). One of the sets Π_r or Π_l must have at least half of the paths, i.e. at least 2^k paths. By symmetry, assume that Π_l has at least 2^k paths and let y be the left child of x. We apply the induction hypothesis to Π_l and obtain paths $\pi_2, \ldots, \pi_{k+1} \in \Pi_l$ and consecutive context zones $X_2, \ldots, X_{k+1}$ such that for every $i \in \{2, \ldots, k+1\}$, path π_i passes through the ports of contexts $X_2, \ldots, X_{i-1}$, but not through the port of X_i.
We slightly modify X_2 be setting y as its root. Note that this does not affect the properties of the paths $\pi_2, \ldots, \pi_{k+1} \in \Pi_l$. Now, we define X_1 as the context zone with the root x and the root of X_2 as port. Let π_1 be some arbitrary path in Pi_r. By definition, $X_1, \ldots, X_{k+1}$ are consecutive context zones. Moreover, the paths $\pi_2, \ldots, \pi_{k+1}$ are paths or Π_l and therefore pass through y, i.e. the port of X_1. Finally be definition π_1 passes through the right child of x and therefore not through the port of X_1. □

Let Π and the function *change* be as defined in Lemma H.5. Apply Lemma H.6 to Π, yielding a sequence of paths, call it $\tau_1, \ldots, \tau_{n^2}$, and a sequence of context zones, call it $Y_1, \ldots, Y_{n^2}$. Consider the sequence

$$change(\tau_1), \ldots, change(\tau_{n^2}) \in \{1, \ldots, m-1\}$$

Like in any sequence of n^2 different numbers, we can find a sequence of indexes

$$j_1 < \ldots < j_n \in \{1, \ldots, n^2\}$$

such that the sequence

$$change(\tau_{j_1}), \ldots, change(\tau_{j_n})$$

is either increasing or decreasing. For $i \in \{1, \ldots, n\}$, define π_i to be τ_{j_i} and X_i to be the union of the context zones $Y_{j_i} \cup \cdots \cup Y_{j_{i+1}-1}$. By construction, we know that the path π_i passes through the ports of the context zones $X_1, \ldots, X_{i-1}$, but not through the port of the zone X_i. By Lemma H.4, we know that

$$val(\sigma, choice(\pi_i), X_i) \neq val(\sigma, choice(\pi_i) + 1, X_i)$$

Therefore, the statement of Lemma H.3 holds if we define $i_1, \ldots, i_n$ to be

$$choice(\pi_1), \ldots, choice(\pi_n).$$

This ends the proof of Lemma H.3. □

Consider a matrix M, a row j, and a column i which is not the first column. Define a new column, denoted by

$$\text{almostcopy}_i^{\neq j}(M),$$

as follows. The new column is equal to column i of M in all rows, except for row j, where it is equal to column $i-1$ of M.
Let σ a strategy tree of duration m an $X_1, \ldots, X_n$ consecutive context zones in σ. We define a strategy matrix with n rows and m columns

$$M = matrix(\sigma, X_1, \ldots, X_n),$$

by setting $M[j, i]$ as $val(\sigma, i, X_j)$. Note that it follows from Fact 3 that every row of M belongs to $\mathcal{V}_L$. Therefore, M is indeed a strategy matrix.

Lemma H.7. *Let N be a strategy matrix defined by*

$$N = matrix(\sigma, X_1, \ldots, X_n),$$

for some locally optimal strategy tree σ, and consecutive contexts $X_1, \ldots, X_n$. Let j be a row and i a column, which is not the first column. If columns i and $i-1$ in N have different entries in row j, then the value of the column

$$\text{almostcopy}_i^{\neq j}(N),$$

is different than the value of column i in N.

Proof. This is a consequence of local optimality. Assume there exists a row i and a column j such that $\text{almostcopy}_i^{\neq j}(N)$ is the value of column i in N. We construct a type tree that is closer to σ_{i-1} than σ_i regarding the discounted distance and with the same root value as σ_i, contradicting local optimality.
Consider the type tree σ'_i defined as follows:

- For nodes $x \in X_j$ $\sigma'_i(x) = \sigma_{i-1}(x)$. By definition:

$$distance(\sigma_i(x), \sigma'_{i+1}(x)) = 0 \leq distance(\sigma_i(x), \sigma_{i+1}(x))$$

- For other nodes y $\sigma'_i(y) = \sigma_i(y)$. By definition:

$$distance(\sigma_{i-1}(y), \sigma'_i(y)) = distance(\sigma_{i-1}(y), \sigma_i(y))$$

Because i and $i-1$ have different entries in row j, there exists at least one node $x \in X_j$ such that $distance(\sigma_{i-1}(x), \sigma_i(x)) = 1$. It follows that:

$$distance_\lambda(\sigma_{i-1}, \sigma'_i) < distance_\lambda(\sigma_{i-1}, \sigma_i)$$

Moreover, $val(\sigma_i)$ is the value of column i and $val(\sigma'_i)$ is the value of column $\text{almostcopy}_i^{\neq j}(N)$. Therefore by hypothesis, both values are equal and this contradicts local optimality of σ. □

We are now ready to prove Lemma H.1. Let $X_1, \ldots, X_n$ and $i_1, \ldots, i_n$ be as in Lemma H.3. Consider the strategy matrix

$$N = matrix(\sigma, X_1, \ldots, X_n).$$

By Lemma H.3, we know that for every $j \in \{1, \ldots, n\}$ the entries in row j are different in columns i_j and $i_j + 1$.
Suppose first that the sequence $i_1, \ldots, i_n$ is strictly increasing. Define a new matrix M, which has n rows and $2n$ columns as follows.

- For $j \in \{1, \ldots, n\}$, column $2j-1$ of M is $\text{almostcopy}_{i_j+1}^{\neq j}(N)$.
- For $j \in \{1, \ldots, n\}$, column $2j$ of M is column $i_j + 1$ of N.

To complete the proof of Lemma H.1, we will show that M belongs to $\mathbb{P}_n$. The dimensions of the matrix M are correct: it has n rows and $2n$ columns.
We now show that M is a strategy matrix, which means that each row belongs to $\mathscr{V}_L$. For $j \in \{1, \ldots, n\}$, let us see how the j-th row of M, call it

$$M[j, 1], \ldots, M[j, 2n]$$

depends on the j-th row of N, call it

$$N[j, 1], \ldots, N[j, m]$$

By reading the definition of M, we see that the dependency is

– When $k \neq j$, then $M[j, 2k-1] = M[j, 2k] = N[j, i_k + 1]$.
– When $k = j$, then $M[j, 2k-1] = N[j, i_k]$ and $M[j, 2k] = N[j, i_k + 1]$.

It follows that the j-th row of M is obtained from the j-th row of N by eliminating some letters and duplicating some other letters. Since $\mathscr{V}_L$ is closed under eliminating and duplicating letters, and since N was a strategy matrix, it follows that also M is a strategy matrix.
By Lemma H.7, for every $j \in \{1, \ldots, n\}$, the values of columns $2j-1$ and $2j$ are different. By construction, columns $2j-1$ and $2j$ have the same entries, except for row j.
When the sequence $i_1, \ldots, i_n$ is strictly decreasing, the matrix M is defined like for a strictly increasing sequence, except that the columns of M are filled in not from left to right, but from right to left. Formally speaking,

– For $j \in \{1, \ldots, n\}$, column $2(n-j+1)-1$ of M is $\text{almostcopy}^{\neq j}_{i_j+1}(N)$.
– For $j \in \{1, \ldots, n\}$, column $2(n-j+1)$ of M is column $i_j + 1$ of N.

The proof that M belongs to $\mathbb{P}_n$ is the same as above.

H.2 Proof of Lemma H.2

Assume that $\mathbb{P}_n$ is non empty for every n. We assume that there are top-down parity alternating matrices of size for every integer n; the other case is handled symmetrically.

Lemma H.8. *There are $v, w, x, y \in V_L$ such that v and w are idempotents and $\mathbb{P}_4$ contains the matrix*

$$\begin{array}{cccccccc} x & y & v & v & v & v & v & v \\ w & w & x & y & v & v & v & v \\ w & w & w & w & x & y & v & v \\ w & w & w & w & w & w & x & y \end{array}$$

Before proving Lemma H.8 we use it to conclude the proof of Lemma H.2. Let M be the matrix described in Lemma H.8. Let $\{u_1, \ldots, u_8\}$ be the values of all the columns in M. The matrix is in $\mathbb{P}_4$, so $u_3 \neq u_4$. Because v and w are idempotent,

$$u_3 = vxww = vxw = vvxw = u_5$$

For the same reason, $u_4 = u_6$ This is depicted in the picture below

$$\begin{array}{cccccccc} & & u_3 & u_4 & u_3 & u_4 & & \\ & & \uparrow & \uparrow & \uparrow & \uparrow & & \\ x & y & v & v & v & v & v & v \\ w & w & x & y & v & v & v & v \\ w & w & w & w & x & y & v & v \\ w & w & w & w & w & w & x & y \end{array}$$

Since u_3 and u_4 are different, then at least one of them is different than vw. Without loss of generality suppose that $u_3 \neq vw$. Because each row of the matrix belongs to $\mathscr{V}_L$, and $\mathscr{V}_L$ is closed under removing letters, it follows that

$$(w, x, v) \in \mathscr{V}_L.$$

This means that we have a violation of the identity (2), which requires that

$$vw = v^\omega \cdot v \cdot w^\omega = v^\omega \cdot x \cdot w^\omega = u_3$$

We are left with the proof of Lemma H.8, which fills the rest of the section.

We define below two rewriting rules for strategy matrices, which we call *safe rules*.

1. Remove a column i. The result is a matrix with one less column.
2. Merge row i with row $i+1$. The result is a matrix with one less row. We describe this operation in more detail. Let

   $$(v_{11}, \ldots, v_{1n}), \ldots, (v_{k1}, \ldots, v_{kn}) \in V_L^n.$$

   be the rows in a matrix with k rows and n columns. The merge operation removes the rows

   $$(v_{i1}, \ldots, v_{in}) \quad \text{and} \quad (v_{(i+1)1}, \ldots, v_{(i+1)n})$$

   and replaces them by the row

   $$(v_{i1} \cdot v_{(i+1)1}, \ldots, v_{in} \cdot v_{(i+1)n}),$$

   which is the product of the two removed rows in the monoid V_L^n.

An important observation is that safe rules preserve parity alternating matrices.

Lemma H.9. *For every $M \in \mathbb{P}_n$ and $i \in \{2, \ldots, n-1\}$ applying the rules*

– *remove column i; and then*

– *merge row i with $i+1$ or $i-1$*

yields a matrix in $\mathbb{P}_{n-1}$.

Proof. Immediate by definition of the rules. □

The following lemma is the key part of the proof it says that by applying safe rules to a large enough parity alternating matrix we can obtain a matrix that is essentially the matrix M describe in Lemma H.8.

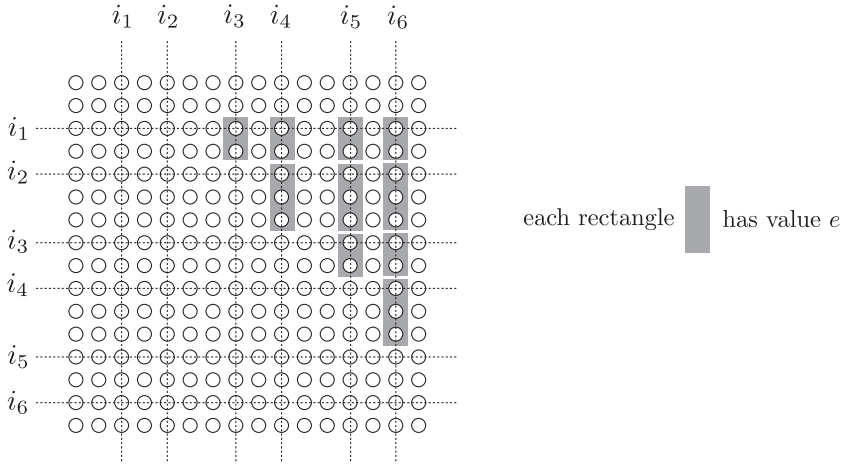
Lemma H.10. *For each $m \in \mathbb{N}$ there is some $n \in \mathbb{N}$ such that for any matrix in $\mathbb{P}_n$ there is a sequence of safe rules that yields a matrix in $N \in \mathbb{P}_m$ and two idempotents $e, f \in V_L$ such that:*

– *All elements above the diagonal, but not in the first row, are equal to e.*
– *All elements below the diagonal, but not in the last row, are equal to f.*

Proof. The proof uses the Ramsey Theorem for hypergraphs with edges of size 3. This theorem says that for every $m \in \mathbb{N}$ there exists a number $f(m)$ such that for any complete hypergraph with edges colored over V_L, there exists a complete sub-hypergraph of size m in which all edges share the same color. We choose $n = f(f(m))$.
Fix a matrix $M \in \mathbb{P}_n$. Consider a hypergraph where the nodes are $\{1, \ldots, n\}$ and an edge $\{i < j < k\}$ is colored by by the value obtained by multiplying, in the monoid V_L, the cells that appear in rows $i, \ldots, j-1$ of column k. By choice of n, we can apply the Ramsey Theorem to this coloring, and get a subset of size $k = f(m)$:

$$I = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$$

such that all hyperedges on I have the same color, say $e \in V_L$. This color must be an idempotent, i.e. it must satisfy $e = ee$. This situation is illustrated below:



Merge the following groups of rows together:

$$\{1, \ldots, i_1 - 1\} \quad \{i_1, \ldots, i_2 - 1\} \quad \ldots \quad \{i_{k-2}, \ldots, i_{k-1} - 1\} \quad \{i_{k-1}, \ldots, n\}.$$

Then, delete all columns except for those from $I - \{1, i_1, \ldots, i_n, n\}$. The resulting matrix has the property that all elements above the diagonal, except for the first row, are equal to e.
We repeat the same operation below the diagonal, obtaining the desired matrix N which is in $\mathbb{P}_m$ because of Lemma H.9. □

We finish the proof of Lemma H.8. By hypothesis, we can apply Lemma H.10 for $m = 6$. Let N be the resulting matrix. Now consider the matrix obtained by:

- Deleting the first, second, last and before-last columns;
- Deleting the first last and last rows.

One can verify that this is the matrix described in Lemma H.8.

Part III of the Appendix.

Complexity