The fraction of large random trees representing a given Boolean function in implicational logic*

Hervé Fournier† Danièle Gardy† Antoine Genitrini‡

Bernhard Gittenberger§

March 26, 2011

Abstract

We consider the logical system of Boolean expressions built on the single connector of implication and on positive literals. Assuming all expressions of a given size to be equally likely, we prove that we can define a probability distribution on the set of Boolean functions expressible in this system. Then we show how to approximate the probability of a function f when the number of variables grows to infinity, and that this asymptotic probability has a simple expression in terms of the complexity of f. We also prove that most expressions computing any given function in this system are "simple", in a sense that we make precise. The probability of all read-once functions of a given complexity is also evaluated in this model. At last, using the same techniques, the relation between the probability of a function and its complexity is also obtained when random expressions are drawn according to a critical branching process.

Keywords: Boolean functions; Implicational formulas; Complexity; Limiting ratio; Probability distribution; Analytic combinatorics; Read-once functions; Branching processes.

1 Introduction

Write at random a Boolean expression on given sets of Boolean variables and of connectors: we obtain a Boolean function. How random is this Boolean function? E.g., what is the probability that we obtain a tautology? A literal? Any specified function? Is the probability of obtaining a given function related to the complexity of the function? Does the Shannon effect, i.e. the fact that "almost all" functions have maximal complexity, still hold for this probability distribution? These and some others are questions that we would like to investigate for general logical systems.

We present here a first step in this direction, with an in-depth study of the simple system obtained from the single connector of implication and positive literals. A former result was obtained in [12], where the probability of tautologies was precisely quantified. One motivation for considering this system came from its relation to intuitionistic logic, which was explored in [35, 21]. It was shown in [12] that asymptotically almost every tautology of classical logic is intuitionistic

*A preliminary version of this work appeared in MFCS'08 [11]. This research was partially supported by the A.N.R. projects *SADA* and *BOOLE* and by the P.H.C. Amadeus project *Probabilities and tree representations for Boolean functions*. The last author's work has been supported by *FWF (Austrian Science Foundation), National Research Area S9600, grant S9604* and *ÖAD, grant F03/2010*.

†Laboratoire PRiSM, CNRS UMR 8144 and Université de Versailles Saint-Quentin-en-Yvelines, 45 avenue des États-Unis, 78035 Versailles, France. Email: {herve.fournier, daniele.gardy}@prism.uvsq.fr. Part of this work was done while the first author was on sabbatical in l'Équipe de Logique Mathématique, CNRS FRE 3233, Université Paris Diderot.

‡Laboratoire LIP6, CNRS UMR 7606 and Université Pierre et Marie Curie, 4 place Jussieu, 75252 Paris cedex 05, France. Email: antoine.genitrini@lip6.fr.

§Technische Universität Wien, Wiedner Hauptstrasse 8-10/104, A-1040 Wien, Austria. Email: gittenberger@dmg.tuwien.ac.at.

as well. This comparison between cassical and intuitionistic logic was later extended to the full propositional system [15].

Note that not every Boolean function is expressible in the system of implication. Indeed, it is easily seen that a function can be obtained if and only if it can be written as $x \vee g$ for some Boolean variable x and function g – this set of functions corresponds to the Post class S_0 [29]. We remark that the set of expressible Boolean functions is only a small fragment of the set of all Boolean functions. (The quotient of the number of expressible functions and the number of all functions actually tends to zero when the number of variables tends to infinity.) Note, however, that this ostensible restriction of the implicational system is not really relevant here. In fact, both from a qualitative and a quantitative point of view, the implicational system captures all Boolean functions since there is a canonical inclusion from the set of all Boolean functions on k variables into the set of Boolean functions over implication on $k + 1$ variables, namely $f \mapsto (f \to z)$.

Furthermore, the implicational fragment plays an important rôle in logics in general. In fact, with respect to validity, almost every logical system relies on the *modus ponens* and therefore the implicational fragment is the minimal axiomatizable fragment. From the satisfiability point of view the implicational fragment is closely related to Horn formulas (first defined in [17], see [31] for an introduction). In fact, the Boolean formulas of the implicational system are precisely the definite Horn clauses. Horn formulas are important in the theory of programming since their satisfiability can be decided algorithmically in linear time (see [5]). Moreover, in the programming language Prolog programs are stated as Horn clauses.

Consider the ratio of the number of formulas of size ℓ that compute a fixed Boolean function f, among all formulas of size ℓ, and let the size grow to infinity. It is possible to show that the limit of this ratio exists for a wide variety of logical systems [14], and that we can thus define a probability distribution on the set of Boolean functions.

It was shown that the tautologies in the implication system have the simple shape $(..., a, ...) \to a$ with high probability, and that, if the number k of Boolean variables grows large enough, the probability of a tautology is asymptotically $1/k$ [12]. The next natural step is then to try and compute the probability that a random expression computes a literal, a function $x_i \vee x_j$, etc., and to check if the "average" expression computing, e.g., a literal, has a simple form. When studying the random expressions that compute a given Boolean function f, one major parameter is the complexity $L(f)$, i.e. the size of the smallest expressions that represent f. We shall prove in the present paper that the probability of any given function f depends exponentially on its complexity; in passing we are also able to characterize the shape of a random expression computing f, and to show that these expressions are obtained quite simply from minimal trees.

The efforts to define non uniform probability distributions, induced by random Boolean expressions, or formulas, on the set of Boolean functions, date back several years. The starting point is generally the description of formulas as *trees* of a suitable shape and suitably labelled. The first efforts in this direction were by Paris et al. [27] on *And/Or* trees (i.e. expressions built on the two connectors $\wedge$ and $\vee$); the underlying model was that of binary Catalan trees, suitably labelled. The study of these trees was further pursued by Lefman and Savický [24], who proved by a pruning argument the existence of a probability distribution induced by random expressions, and established important lower and upper bounds for the probability of any Boolean function in terms of its complexity. At the same time, Woods [34] proved independently the existence of a limiting distribution for general formulas. Some of the authors of the present paper then gave an alternative construction of the probability distribution for And/Or trees, together with an improvement on the upper bound [4]. The survey paper [14] presents an overview of the probability distributions induced by random Boolean expressions on Boolean functions, and of the way we can obtain them using the tools of analytic combinatorics: enumeration of formulas/trees by generating functions, the Drmota-Lalley-Woods theorem for solving an algebraic system of equations, and asymptotics.

We should also mention that several researchers have concentrated on the probability of tautologies, i.e. on the probability of the single constant function *True*. Let us mention the Polish school around Zaionc, who began a systematic investigation of the probability of a tautology in various logical frameworks [26, 18, 36, 19]. The case of implication with negative literals was considered in [13]. See also [25, 20] for the expressions built on the single equivalence connector. For And/Or trees, we refer the reader to Woods's result that the tautologies have asymptotic probability $3/4k$, and that almost all of them have the simple form $\ell \vee \cdots \vee \bar{\ell} \vee \ldots$ [33], and to Kozik [22] for a different, later proof. This last work also considers the probability of all Boolean functions over And/Or trees; as in the present paper, it is shown that this system exhibits an exponential dependence between the probability of a function and its complexity.

Significant results have also been established for a different family of formulas/trees, namely balanced trees obtained by iteration of a single connector. The first result in this area is due to Valiant [32], whose aim was to compute a Boolean expression for the function *Majority* with high enough probability. Then Boppana [2] and Gupta and Mahajan [16] improved Valiant's result for majority; Boppana went on to prove that iteration by a single, well-chosen connector gives a distribution concentrated on one of the threshold functions. Savický [30] showed that iterating a nonlinear and monotone connector leads to the uniform distribution on the set of all Boolean functions. Brodsky and Pippenger [3] presented a systematic study of different classes of connectors and of the distributions induced on Boolean functions; these distributions are either uniform on subsets of Boolean functions, or concentrated on a single function. Finally some of us studied balanced And/Or trees [10] and proved that a limiting distribution exists and is concentrated on linear threshold functions. Our trees are obtained by iteration too, but with two distinct connectives *and* and *or*, chosen at random.

The present paper is organized as follows. We show in Section 2 how all the trees computing a specific Boolean function can be derived from a finite set of minimal trees by a few simple operations. Our main results are also given in this section, namely the asymptotic expression of the probability of the Boolean function in terms of its complexity, and the (relatively) simple form of a random expression computing a Boolean function. The four following sections are devoted to the proof of these results. We first recall in Section 3 basic facts and former results on tautologies, i.e. on the trees that compute the simplest Boolean function in our system: the constant *True*. Next we give technical results on expansions and on the inverse operation of pruning in Section 4, before considering irreducible trees and their expansions in Section 5. Section 6 is devoted to analyze the specific class of read-once functions, because we can give a more detailed result for such functions. Next we introduce a second probability distribution, based on branching processes, and we give the new version of the theorem depending on this distribution in Section 7. Finally we present possible extensions in Section 8.

2 Limiting ratio of trees computing a given function

We begin by a brief presentation of the formulas we consider, then give a couple of definitions in order to state our main result concerning the limiting ratio of trees computing a given function.

Trees over implication

In this paper we consider Boolean formulas and their representations as trees which are built with the single connector of implication (denoted by $\to$) and positive literals.

Definition 1 *A tree over implication is a full binary tree whose internal nodes are all labelled by $\to$ and leaves by literals chosen from the set $\{x_1, \ldots, x_k\}$. Since each such tree can be interpreted as a Boolean formula, we use the terms formula and tree synonymously. We denote the set of all formulas by $\mathcal{F}_k$.*

Each formula, or tree, is associated to a specific Boolean function: We say that a tree is computing *a specific Boolean function f if the Boolean formula given by the tree is a representation of f. The Boolean function computed by a tree A is denoted by $[A]$. A tautology is a tree A such that $[A] = True$.*

Definition 2 *Every tree $A \in \mathcal{F}_k$ can be written in a unique way as*

$$A = A_1 \to (A_2 \to (\ldots \to (A_p \to r(A)) \ldots))$$

where $A_i \in \mathcal{F}_k$ and $r(A) \in \{x_1, \ldots, x_k\}$ – see Figure 1. Such a tree will also be denoted by

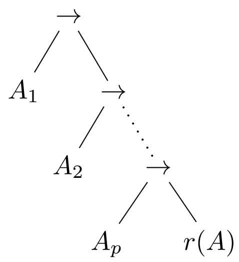

Figure 1: Decomposition of a tree along its right branch.

$A_1, \ldots, A_p \to r(A)$. *The subtrees $A_1, \ldots, A_p$ are called the* premises *of A, and the rightmost leaf $r(A)$ is called the* goal *of A. Analogously, premises and goal of any subtree of A is defined. The goals of the premises of A are called the* subgoals *of A (the subgoals of A are $r(A_1), \ldots, r(A_p)$).*

Limiting ratio

We define the size $|A|$ of a tree A as the number of its *leaves*. Our goal is to quantify the fraction of trees which compute a given function f. This leads to the following definition.

Definition 3 *The* limiting ratio *of a subset $\mathcal{A} \subseteq \mathcal{F}_k$ of trees is defined as*

$$\mu_k(\mathcal{A}) = \lim_{n\to\infty} \frac{|\{A \in \mathcal{A} \ : \ |A| = n\}|}{|\{A \in \mathcal{F}_k \ : \ |A| = n\}|}$$

if this limit exists.

We denote by $\mathcal{F}_k(f)$ the set of all trees from $\mathcal{F}_k$ computing f so $\mathcal{F}_k(f) = \{A \in \mathcal{F}_k, [A] = f\}$ and now we define the limiting ratio of a function f *as the limiting ratio of $\mathcal{F}_k(f)$.*

Introducing the generating functions $\sum_n |\{A \in \mathcal{F}_k \ : \ |A| = n, [A] = f\}| z^n$, the results of Drmota [6], Lalley [23] and Woods [34] give us an easy way to prove that the limiting ratio of each Boolean function is defined in the system $\mathcal{F}_k$ – i.e. for all Boolean functions f, the limit defining $\mu_k(\mathcal{F}_k(f))$ exists. These theorems are nicely described in Flajolet and Sedgewick [8, 9]. For the sake of self-containedness we will state the theorem in Section 3. The proof which appears in Section 3 justifies that the Drmota-Lalley-Woods conditions are satisfied in the system of implication. In the following, $\mu_k(\mathcal{F}_k(f))$ is abbreviated with $\mu_k(f)$.

Valid expansions of a tree

Now we define three rules, called *expansion rules*, that allow, starting from a tree A, to obtain larger trees computing the same function as A.

Definition 4 *Let A be a tree and B one of its subtrees and let the root of B be denoted by ν.*

Valid expansion by a "tautology"*: A tree A' is called a valid expansion of A by a tautology at node ν, if A' is obtained by replacing the subtree B with the subtree $C \to B$ in A, where C is a tautology. Of course A' computes the same function as A since $[C \to B] = [B]$.*

Valid expansion by "goal α"*: Let $\alpha \in \{x_1, \ldots, x_k\}$. If substituting B with $C \to B$ yields a tree A' computing the same function as A* for any tree C with goal α, *we say that any of these trees A' is obtained from A by a valid expansion of type "goal α" at node ν.*

The third expansion of A is called

Valid expansion by "premise α"*: Let again $\alpha \in \{x_1, \ldots, x_k\}$. If substituting B with $C \to B$ yields a tree A' computing the same function as A* for any tree C with a premise equal to α, *we say that any of these trees A' is obtained from A by a valid expansion of type "premise α" at node ν.*

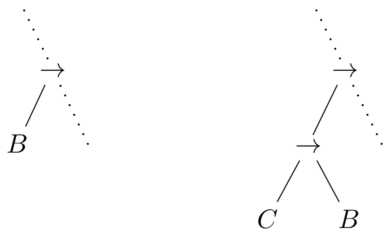

Figure 2: Valid expansion with the subtree C in the root of B.

Figure 2 represents the shape of the tree obtained after a valid expansion at the root of B.

Definition 5 *Given a tree A, we define $E(A)$ to be the set of all trees obtained from A by a single valid expansion of any of the three types defined above. We naturally extend E to any set of trees $\mathcal{A} \subseteq \mathcal{F}_k$ by letting $E(\mathcal{A}) = \bigcup_{A \in \mathcal{A}} E(A)$. In the same way we define $E^0(\mathcal{A}) = \mathcal{A}$, $E^i(\mathcal{A}) = E(E^{i-1}(\mathcal{A}))$ and $E^*(\mathcal{A}) = \bigcup_{i \in \mathbb{N}} E^i(\mathcal{A})$. As shortcuts, we define $E^+(\mathcal{A}) = \bigcup_{i>0} E^i(\mathcal{A})$ and for any integer p, $E^{\geqslant p}(\mathcal{A}) = \bigcup_{i \geqslant p} E^i(\mathcal{A})$.*

Note that all trees in $E(A)$ (and thus in $E^*(\{A\})$ as well) compute the same function as A.

The number of valid expansions of a tree

Given a tree A, we define $\lambda(A)$ as the number of types of valid expansions of A; more precisely, this is the number of pairs (ν, α), where ν is a node of A (either an internal node or a leaf) and $\alpha \in \{x_1, \ldots, x_k\}$, such that an expansion of type "goal α" is valid in the node ν, plus the number of pairs (ν, α) such that an expansion of type "premise α" is valid in the node ν, plus $2|A| - 1$ (this is counting the tautology expansions in every of the $2|A| - 1$ nodes of A).

Definition 6 *For a Boolean function f depending on a finite number of variables of $\{x_i \mid i > 0\}$ and that can be computed with implication, we define its* complexity *$L(f)$ to be the size of a smallest tree (over implication) computing f.*

Note that in general there is no unique smallest tree but rather several smallest trees computing a given function. Trees of size $L(f)$ computing f are called *minimal trees* of f; their set is denoted by $\mathcal{M}_f$. Given a Boolean function f, we define

$$\lambda(f) = \sum_{M \in \mathcal{M}_f} \lambda(M).$$

It will be proved that $\lambda(f)$ does not depend on the number k of ambient variables. We can state the main result of this paper now.

Theorem 7 *Let f be a Boolean function different from True. Almost all trees computing f are obtained by a single expansion of a minimal tree of f:*

$$\mu_k(f) \sim \mu_k(E(\mathcal{M}_f)).$$

As a consequence, the limiting ratio of f is asymptotically (as $k \to \infty$) equal to:

$$\mu_k(f) = \frac{\lambda(f)}{4^{L(f)}\ k^{L(f)+1}} + O\left(\frac{1}{k^{L(f)+2}}\right).$$

A proof of this theorem is given at the end of Section 5, where bounds on $\lambda(f)$ are also provided – see Proposition 41. A rough outline is as follows: When manipulating trees by means of expansions and its inverse, pruning (defined in Section 4), we will encounter trees which are irreducible (see Section 4) but not minimal trees. Therefore we will study the set of irreducible trees, decompose it into the set of minimal trees and its complement and eventually show that the complement as well as the set of trees obtained by iterated expansion do not contribute to the limiting ratio.

Remark. Note that so far, we proved neither that the family of all trees computing f nor that the family $E(\mathcal{M}_f)$ has a limiting ratio. In order to do this, some work has to be done. The existence of $\mu_k(f)$ is addressed in the next section in Proposition 10, that of $\mu_k(E(\mathcal{M}_f))$ in Lemma 32.

3 Drmota-Lalley-Woods conditions and tautologies

In this section, we first introduce the generating function $f_k(z) = \sum_{n=1}^{\infty} |\{A \in \mathcal{F}_k \ : \ |A| = n\}| z^n$ that enumerates all trees $\mathcal{F}_k$ by their size – we recall that the size of a tree is defined as its number of leaves. The function $f_k(z)$ satisfies the following equation: $f_k(z) = kz + f_k(z)^2$, obtained by using the recursive building of the trees, and as $f_k(0) = 0$, we get

$$f_k(z) = \frac{1 - \sqrt{1 - 4kz}}{2}.$$

Moreover its nth coefficient is related to the $(n-1)$th Catalan number: $[z^n]f_k(z) = C_{n-1}k^n = \frac{1}{n}\binom{2n-2}{n-1}k^n$.

Notice that the set of subsets $\mathcal{A} \subseteq \mathcal{F}_k$ having a limiting ratio $\mu_k(\mathcal{A})$ is closed under (relative) complement and finite disjoint union. Moreover, if $\mathcal{A}, \mathcal{B}$ are disjoint and have a limiting ratio, then $\mu_k(\mathcal{A} \cup \mathcal{B}) = \mu_k(\mathcal{A}) + \mu_k(\mathcal{B})$.

In order to show that every Boolean function has a limiting ratio, we will employ the Drmota-Lalley-Woods Theorem. This theorem deals with systems of the form

$$y_j = \Phi_j(z, y_1, \ldots, y_m), \qquad j = 1, \ldots, m, \tag{1}$$

of functional equations for the functions $y_j(z)$. For our purpose it is enough to assume that the functions Φ_j are real polynomials. To be able to state the theorem for polynomial systems, we have to introduce some concepts where we follow the presentation given in [9, Chapter VII]:

Definition 8 *The valuation of a power series $y(z) = \sum y_n z^n$ is defined by $\mathrm{val}(y) = \min\{j \mid y_j \neq 0\}$; and for a vector $\mathbf{y}$ of power series as the minimal valuation of all its components. The distance of two power series $y_1(z)$ and $y_2(z)$ is defined by $d(y_1, y_2) = 2^{-\mathrm{val}(y_1 - y_2)}$; and for vectors analogously.*

For a power series $y(z)$ let $S_y = \{n \mid y_n \neq 0\}$. The power series $y(z)$ is called algebraicly aperiodic if $\max\{d \mid \exists r \in \mathbb{N} : S_y \subseteq r + d\mathbb{N}\} = 1$.

A (polynomial) system of functional equations of the form (1), or written in vector notation as $\mathbf{y} = \Phi(z, \mathbf{y})$, is called

- algebraicly positive *if all the component polynomials Φ_j have non-negative coefficients;*
- algebraicly proper *if it satisfies a Lipschitz condition $d(\Phi(z, \mathbf{y}), \Phi(z, \tilde{\mathbf{y}})) < Kd(\mathbf{y}, \tilde{\mathbf{y}})$ for some positive constant $K < 1$;*
- algebraicly irreducible *if its dependency graph is strongly connected;*
- algebraicly aperiodic *if all of its component solutions $y_j(z)$ are algebraicly aperiodic.*

Theorem 9 (Drmota-Lalley-Woods, [6, 23, 34]) *Let $\mathbf{y} = \Phi(z, \mathbf{y})$ be a nonlinear polynomial system which is algebraicly positive, proper, irreducible, and aperiodic. Then all component solutions $y_j(z)$ have the same radius of convergence $R < \infty$ and there exist functions $h_j(z)$ which are analytic at the origin such that in a neighbourhood of R we have $y_j(z) = h_j(\sqrt{1 - z/R})$.*

Proposition 10 *If f is a Boolean function on the variables $\{x_1, \ldots, x_k\}$, its limiting ratio $\mu_k(f)$ exists.*

Proof: The aim of this proof is to show the existence of a limiting ratio $\mu_k(f)$ for the set of trees $\mathcal{F}_k(f) = \{A \in \mathcal{F}_k \mid [A] = f\}$ computing a given function f. For technical reasons, the size of a tree in this section is defined as the number of internal nodes (since it differs by one from the number of leaves considered in the rest of the paper, the existence and value of $\mu_k(f)$ are not affected by this change). Let $\{f_1, \ldots, f_p\}$ be the set of Boolean functions which can be computed by formulas of $\mathcal{F}_k$. We denote by ϕ_i the generating function of trees computing f_i; that is, $\phi_i(z) = \sum_{n=0}^{\infty} \alpha_{i,n} z^i$ where $\alpha_{i,n}$ is the number of trees of size n computing f_i. Let Φ be the polynomial system obtained by considering the inductive structure of expressions. Consider one variable $y_i \in \mathbb{C}[[z]]$ for each $1 \leqslant i \leqslant p$. For $1 \leqslant \ell \leqslant p$, the ith equation of the system Φ is $y_i = 1_{\{f_i \text{ literal}\}} + z \sum_{f_j, f_k;\ f_j \to f_k = f_i} y_j y_k$, where $1_{\{f_i \text{ literal}\}} = 1$ if f_i is a literal and 0 otherwise. Obviously $(\phi_1, \ldots, \phi_p)$ is a solution of the polynomial system Φ (the unique one).

To show the existence of $\mu_k(f_i)$ for all i, it is enough to check that the polynomial system Φ satisfies the conditions of the Drmota-Lalley-Woods Theorem. Obviously the system Φ is nonlinear (it is quadratic) and algebraicly positive. Let us check it is algebraicly proper. For this, we have to show that for $a, b \in \mathbb{C}[[z]]^p$, $\mathrm{val}(\Phi(a) - \Phi(b)) > \mathrm{val}(a - b)$. For $\ell \in \{1, \ldots, p\}$, we have $\Phi_\ell(a) - \Phi_\ell(b) = z \sum_{i,j} (a_i a_j - b_i b_j)$ where the sum is over all i, j such that $f_i \to f_j = f_\ell$. Notice that $a_i a_j - b_i b_j = \frac{1}{2}((a_i + b_i)(a_j - b_j) + (a_j + b_j)(a_i - b_i))$. It follows that $\mathrm{val}(\Phi_\ell(a) - \Phi_\ell(b)) \geqslant 1 + \mathrm{val}(a - b)$. This gives $\mathrm{val}(\Phi(a) - \Phi(b)) > \mathrm{val}(a - b)$.

Now we show that the system Φ is algebraicly irreducible. For this, it is enough to check that there is a "path" from any f_i to *True* and a path from *True* to any f_j in the system. The first part comes from the fact that $f_i \to f_i$ computes *True*, and the second from the fact that *True* $\to f_j$ computes f_j.

At last, let us check that Φ is algebraicly aperiodic. Let E be an expression computing a function f_i. Both expressions $(x_1 \to x_1) \to E$ and $(x_1 \to (x_1 \to x_1)) \to E$ compute f_i. Since these expressions are of size $|E| + 2$ and $|E| + 3$ and $(2, 3) = 1$, this shows there is at least one expression of size ℓ computing f_i when ℓ is large enough. This ensures aperiodicity. $\square$

As we know that all Boolean functions admit a limiting ratio, we can refer to the limiting ratio of tautologies now.

Definition 11 *A* simple tautology *is a formula such that one premise is equal to the goal of the formula.*

It is obvious that a simple tautology is indeed a tautology.

We recall some results from [12] on trees computing the constant function *True*. It was proved there that the limiting ratio of all tautologies is equivalent to the limiting ratio of the family of simple tautologies. The limiting ratio of the set $\mathcal{S}_k$ of simple tautologies is equal to:

$$\mu_k(\mathcal{S}_k) = \frac{4k+1}{(2k+1)^2} = \frac{1}{k} + O\left(\frac{1}{k^2}\right).$$

Moreover, the limiting ratio of all tautologies (denoted by $\mathcal{T}_k$) was given for k tending to infinity:

$$\mu_k(\mathcal{T}_k) = \frac{1}{k} + O\left(\frac{1}{k^2}\right).$$

Finally we recall two facts on the structure of tautologies.

Definition 12 *For a node ν of a tree A, we define the* left depth *of the node ν as the number of edges going left which are needed to reach ν from the root of A. We define in the same way the left depth of a subtree B of A as the left depth of its root.*

Let A be a tree and B one of its subtrees; B is called a left subtree *of A if the root of B is the left son of its first ancestor. A left subtree of size one is called a* left leaf.

In the following, $\bar{f}$ denotes the negation of the Boolean function f. As a special case, $\bar{\alpha}$ denotes the Boolean function which is true if and only if α is false.

Lemma 13 *Let A be a tree which is a tautology. Then there is a premise A' of A which has the same goal as A.*

Proof: Assume that A has the premises $A_1, \ldots, A_p$ and goal α. Let $f_i = [A_i]$ and $\beta_i = r(A_i)$ for $i = 1, \ldots, p$. Then $[A] = \alpha \vee \bar{f}_1 \vee \cdots \vee \bar{f}_p$ is a tautology and therefore $\alpha \vee \bar{\beta}_1 \vee \cdots \vee \bar{\beta}_p$ as well. Thus one of the β_i must be equal to α. $\square$

The next fact is [12, Lemma 1]. For the sake of self-containedness we also present the proof.

Lemma 14 *If A is a non-simple tautology, then among the leaves of left depth at most 3 there are either three occurrences of the same variable or there are two distinct variables each of which appears twice.*

Proof: Assume that A has the premises $A_1, \ldots, A_p$ and goal α. If two premises have goal α, then α appears three times and we are done. So, by the previous lemma we have to consider only the case where exactly one premise has goal α. Without lost of generality assume that $r(A_i) = \alpha_i \neq \alpha$ for $i = 1, \ldots, p-1$ and $r(A_p) = \alpha$. Furthermore set $f_i = [A_i]$ for $i = 1, \ldots, p$. Certainly, A_p cannot be reduced to a literal, otherwise A would be a simple tautology. Thus there are functions $g_1, \ldots, g_m$ such that $[A_p] = \alpha \vee \bar{g}_1 \vee \cdots \vee \bar{g}_m$. Since $\alpha \vee \bar{f}_1 \vee \cdots \vee \bar{f}_p$ is a tautology, we must have

$$\alpha \vee \bar{\alpha}_1 \vee \cdots \vee \bar{\alpha}_{p-1} \vee \bar{g}_j = \text{True} \tag{2}$$

for any $j = 1, \ldots, m$. Therefore, let us consider functions of the form $\alpha \vee \bar{\alpha}_1 \vee \cdots \vee \bar{\alpha}_{p-1} \vee g$ which are tautologies. If g is a literal x, then $x \in \{\alpha_1, \ldots, \alpha_{p-1}, \alpha\}$. Otherwise g is of the form $\gamma \vee \bar{h}_1 \vee \cdots \vee \bar{h}_s$ and assume that h_i is represented by a tree with goal γ_i. Then

$$\bar{\gamma}_1 \vee \cdots \vee \bar{\gamma}_s \vee \gamma \vee \bar{\alpha}_1 \vee \cdots \vee \bar{\alpha}_{p-1} \vee \alpha = \text{True}$$

and hence we must have either $\alpha \in \{\gamma_1, \ldots, \gamma_s\}$ and thus three occurrences of α or $\gamma \in \{\gamma_1, \ldots, \gamma_s, \alpha_1, \ldots, \alpha_{p-1}\}$, which means that α and γ appear twice. Since γ has left depth 2 and the γ_i have left depth 3, the proof is complete. $\square$

4 Expansion, extension, and pruning

Now we study some of the properties of the expansion rules defined in Section 2. First, we define a cutting process which will eventually be used to define the inversion of expansions which we refer to as *pruning*.

Definition 15 (Cutting subtrees) *Given a tree A and a left subtree B of A, we denote by $A \setminus B$ the tree obtained by removing B from A. More precisely, since B is a left subtree of A, it is the left son of a subtree of the form $B \to C$ in A; the tree $A \setminus B$ is obtained by substituting the subtree $B \to C$ by C in A – see Figure 3.*

Consider trees $B_1, B_2, \ldots, B_\ell$ which are disjoint subtrees of A, i.e., no node of A is in more than one tree B_i. Then we write $A \setminus \{B_1, B_2, \ldots, B_\ell\}$ for $(\cdots((A \setminus B_1) \setminus B_2) \setminus \ldots B_{\ell-1}) \setminus B_\ell$. Note that the actual order of $B_1, B_2, \ldots, B_\ell$ does not affect the resulting tree.

The following three lemmas give (necessary and) sufficient conditions for a tree to be a single expansion of a certain type of a smaller tree.

Lemma 16 *Let A be a tree and B be a left subtree of A. If B is a tautology, then A is obtained by a single valid expansion of type "tautology" of $A \setminus B$.*

Proof: This is obvious from the definition of expansions by tautologies. $\square$

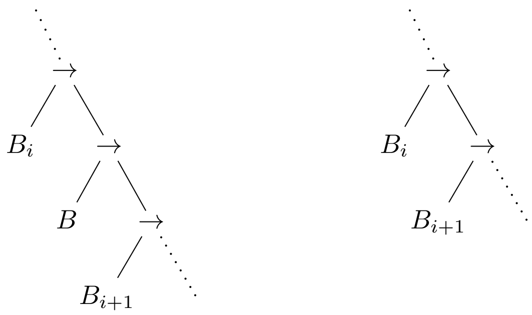

Figure 3: Cutting a left subtree B.

Remark. In the following, substituting B by a Boolean function f in the tree A means that we replace the subtree B of A by a leaf labelled with f. Strictly speaking, the resulting tree does no more belong to the set of trees over implication according to Definition 1 (except if f is a positive literal), but it still represents a Boolean function and so the notion of the function computed by this tree still makes sense.

Lemma 17 *Let A be a tree and B be a left subtree of A. Let β be the goal of B. If substituting B by 1 or β in A yields a tree computing $[A]$ in both cases, then A is obtained by a single valid expansion of type "goal β" of $A \setminus B$.*

Proof: Let A_1 be the tree A where B is replaced with β, and A_2 be the tree A where B is replaced with 1. Let B' be any tree with "goal β", and A' be the tree obtained from A by replacing B with B'. Of course $\beta \leqslant [B'] \leqslant 1$. Then by induction on the size of the formula, we obtain $[A] = [A_1] \leqslant [A'] \leqslant [A_2] = [A]$ or $[A] = [A_1] \geqslant [A'] \geqslant [A_2] = [A]$, depending whether the left depth of the root of B is even or odd. In any case, $[A'] = [A]$. Moreover, $[A \setminus B] = [A]$ since $[A \setminus B] = [A_2]$. $\square$

Recall that $\bar{\beta}$ denotes the negation of the variable β.

Lemma 18 *Let A be a tree and B be a left subtree of A. Suppose that B has a premise of size one β. If substituting B with 1 or $\bar{\beta}$ in A gives a tree computing $[A]$ in both cases, then A is obtained by a single valid expansion of type "premise β" of $A \setminus B$.*

Proof: The proof is similar to the previous one. $\square$

Definition 19 (Pruning) *When going from A to $A \setminus B$ such that we are in the situation of one of the three lemmas above, we shall say $A \setminus B$ is obtained by pruning the left subtree B in A.*

Note the difference between pruning a subtree and cutting a subtree: if A is a tree and B one of its left subtrees, then the term "cutting the subtree B" means that we remove B from A without any condition on B, whereas the term "pruning the subtree B" means that we remove B from A in such a way that A is a valid expansion of $A \setminus B$. However, both final trees are denoted by $A \setminus B$.

Definition 20 *A tree which cannot be pruned is called an* irreducible tree.

Of course all minimal trees computing a function f are irreducible. However, the converse is not true; indeed consider the function $f = x_1 \vee (\bar{x}_2 \wedge \bar{x}_3) \vee (\bar{x}_2 \wedge x_4)$. It can be checked that $(x_4 \to x_2) \to (((x_2 \to x_3) \to x_3) \to x_1)$ computes f, is irreducible, but not minimal since $((x_3 \to x_4) \to x_2) \to x_1$ is smaller and also computes f. We also remark that the system of pruning rules is not confluent.

Now we define a new way of getting large trees from a smaller one. But this time, it does not preserve the function computed by the initial tree; its purpose is to establish some upper bounds on the limiting ratio of expansions. This new mapping X is called *extension* (it is different from expansion). The relation between extensions and expansions is given below.

Definition 21 *An* extension *X of a tree T is defined recursively as follows: if T consists of a single leaf α, $X(\alpha)$ is the set of all trees whose goal is labelled by α. If $T = L \to R$, we let*

$$X(L \to R) = \{A_1 \to (\ldots \to (A_p \to (\tilde{L} \to \tilde{R}))\ldots) \mid p \geqslant 0,\ A_1, \ldots, A_p \in \mathcal{F}_k,\ \tilde{L} \in X(L),\ \tilde{R} \in X(R)\}.$$

We naturally extend X to a set of trees $\mathcal{A} \subseteq \mathcal{F}_k$ by letting $X(\mathcal{A}) = \bigcup_{A \in \mathcal{A}} X(A)$.

Notice that $X(X(\mathcal{A})) = X(\mathcal{A})$ for any $\mathcal{A} \subseteq \mathcal{F}_k$. Figure 4 shows a graphical representation of the

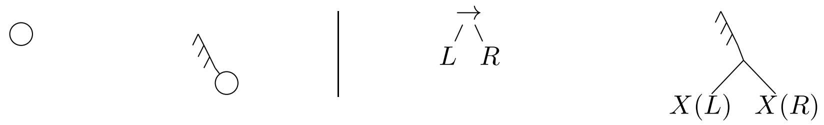

Figure 4: The recursive definition of the *extension* mapping.

recursive definition of this mapping. A single leaf α is extended to a tree having the original leaf α as its goal. According to the canonical representation of the tree (cf. Figure 1) it can be seen as a decomposition of a comb-like structure and the goal (omitting the left subtrees of the main branch). Figure 5 shows the general shape of extensions of a given tree: again we omit the left subtrees of the comb-like substructures for the sake of clearness. The omitted trees are arbitrary trees.

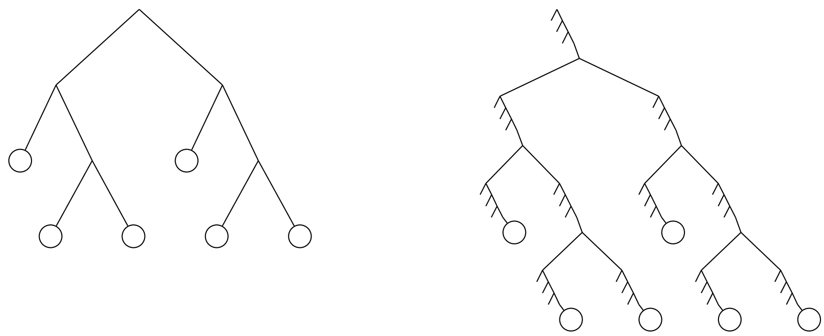

Figure 5: A tree A on the left and the set $X(A)$ it defines, on the right.

Lemma 22 *For any tree A, $E^*(A) \subseteq X(A)$.*

Proof: Let A be a tree. Since $X(X(A)) = X(A)$, all is needed is to prove that $E(A) \subseteq X(A)$. Recall that any tree $A' \in E(A)$ is obtained by substituting a subtree B of A with a tree of the form $C \to B$. It is clear from the definition of extensions that $C \to B \in X(B)$, and it follows that $A' \in X(A)$. $\square$

We would like to prove now that a set of trees that contains "too many" repetitions of some variables before a fixed left depth has a small limiting ratio. However, since we are not sure about the existence of the limiting ratio of the considered family, we shall use the limit superior

$$\bar{\mu}_k(\mathcal{A}) = \limsup_{n \to \infty} \frac{|\{A \in \mathcal{A}\ :\ |A| = n\}|}{|\{A \in \mathcal{F}_k\ :\ |A| = n\}|}$$

to state an upper bound.

Let $\mathcal{V}$ be a fixed finite subset of the variables $\{x_i \mid i > 0\}$, independent of the number of variables k we consider. Let p and q be two integers. Let $\mathcal{A}_q^p(\mathcal{V})$ be the set of trees of $\mathcal{F}_k$ which contain at least p leaves labelled in $\mathcal{V}$, all of them being of left depth at most q. The rest of this section is dedicated to an upper bound on $\bar{\mu}_k((\mathcal{A}_q^p(\mathcal{V})))$.

To this aim, we introduce a second family of trees. Let $\mathcal{B}_q^p(\mathcal{V})$ the set of trees $B \in \mathcal{F}_k$ such that $p \leqslant |B| \leqslant pq+1$ and which contain at least p leaves labelled in $\mathcal{V}$. Notice that $\mathcal{B}_q^p(\mathcal{V})$ is finite – as opposed to $\mathcal{A}_q^p(\mathcal{V})$. The sets of trees $\mathcal{A}_q^p(\mathcal{V})$ and $\mathcal{B}_q^p(\mathcal{V})$ are linked by the following lemma.

Lemma 23 *For a fixed set of variables $\mathcal{V}$ and two integers p and q we have $\mathcal{A}_q^p(\mathcal{V}) \subseteq X(\mathcal{B}_q^p(\mathcal{V}))$ and consequently $E^*(\mathcal{A}_q^p(\mathcal{V})) \subseteq X(\mathcal{B}_q^p(\mathcal{V}))$.*

Proof: Let $A \in \mathcal{A}_q^p(\mathcal{V})$. Let $\nu_1, \ldots, \nu_p$ be p leaves of A labelled with variables from $\mathcal{V}$, all with left depth at most q. Let $C_1, \ldots, C_r$ be the set of maximal (with respect to the subtree relation) left subtrees of A not containing any of the nodes ν_i. Let B be the tree obtained from A by removing all C_i, i.e. $B = A \setminus \{C_1, \ldots, C_r\}$. Of course $A \in X(B)$, and it can be checked that $B \in \mathcal{B}_q^p(\mathcal{V})$: indeed the largest tree B that can be obtained is when all nodes ν_i have a left depth q and belong to distinct premises of A, and $|B| = pq+1$ in this case. Thus $A \in X(\mathcal{B}_q^p(\mathcal{V}))$. The second part of the lemma follows from Lemma 22 and the fact that $X(X(B_q^p(\mathcal{V}))) = X(\mathcal{B}_q^p(\mathcal{V}))$. $\square$

Lemma 24 *For a fixed set of variables $\mathcal{V}$ and two integers p and q we have $\bar{\mu}_k(X(\mathcal{B}_q^p(\mathcal{V}))) = O(1/k^p)$.*

Proof: For two generating functions $f, g \in \mathbb{R}[[z_1, z_2, \ldots, z_\ell]]$, we write $f \prec g$ if $[z_1^{n_1} \cdots z_\ell^{n_\ell}]f \leqslant [z_1^{n_1} \cdots z_\ell^{n_\ell}]g$ for all $n_1, \ldots, n_\ell \in \mathbb{N}$. Let $\gamma = |\mathcal{V}|$. Let $\phi_{\mathcal{B}_q^p(\mathcal{V})}(z,t)$ be the bivariate generating function enumerating trees of $\mathcal{B}_q^p(\mathcal{V})$, z is marking the leaves and t all the nodes. It satisfies:

$$\phi_{\mathcal{B}_q^p(\mathcal{V})}(z,t) \prec \sum_{\ell=p}^{pq+1} C_{\ell-1} \binom{\ell}{p} \gamma^p k^{\ell-p} z^\ell t^{2\ell-1}.$$

Then we get:

$$\phi_{X(\mathcal{B}_q^p(\mathcal{V}))}(z) \prec \sum_{\ell=p}^{pq+1} C_{\ell-1} \binom{\ell}{p} \gamma^p k^{1-\ell-p} z^{1-\ell} f_k(z)^{2\ell-1}.$$

Using Lagrange inversion [7, Chapter 3] we obtain:

$$\begin{aligned} [z^n]\phi_{X(\mathcal{B}_q^p(\mathcal{V}))}(z) &\leqslant \sum_{\ell=p}^{pq+1} C_{\ell-1} \binom{\ell}{p} \gamma^p k^{1-\ell-p} [z^{n+\ell-1}] f_k(z)^{2\ell-1} \\ &\leqslant \sum_{\ell=p}^{pq+1} C_{\ell-1} \binom{\ell}{p} \gamma^p k^{n-p} \frac{2\ell-1}{n+\ell-1} \binom{2n-2}{n+\ell-1}. \end{aligned}$$

Since $[z^n]f_k(z) = k^n C_{n-1}$, we have

$$\frac{[z^n]\phi_{X(\mathcal{B}_q^p(\mathcal{V}))}(z)}{[z^n]f_k(z)} \leqslant \sum_{\ell=p}^{pq+1} C_{\ell-1} \binom{\ell}{p} \gamma^p k^{-p} (2\ell-1) \frac{1}{C_{n-1}} \frac{1}{n+\ell-1} \binom{2n-2}{n+\ell-1}.$$

Now notice that for any $\ell \geqslant 0$,

$$\frac{1}{C_{n-1}} \frac{1}{n+\ell-1} \binom{2n-2}{n+\ell-1} = \frac{n}{n+\ell-1} \frac{\binom{2n-2}{n+\ell-1}}{\binom{2n-2}{n-1}} \leqslant 2.$$

Thus

$$\bar{\mu}_k(X(\mathcal{B}_q^p(\mathcal{V}))) \leqslant \frac{2\gamma^p}{k^p} \sum_{\ell=p}^{pq+1} C_{\ell-1} \binom{\ell}{p} (2\ell-1).$$

This gives $\bar{\mu}_k(X(\mathcal{B}_q^p(\mathcal{V}))) = O(1/k^p)$. $\square$

Corollary 25 *For a fixed set of variables $\mathcal{V}$ and two integers p and q, $\bar{\mu}_k(E^*(\mathcal{A}_q^p(\mathcal{V}))) = O(1/k^p)$.*

Proof: Immediate from Lemmas 23 and 24. $\square$

5 Irreducible trees and their expansions

In this section we study the limiting ratios of expansions of minimal trees of a function f. We will identify those classes of trees which will be relevant for the asymptotical value of the limiting ratio of f and show that the other are of negligible size.

Definition 26 *Let f be a Boolean function different from True. The variable x is called an* essential variable *of f if the two functions obtained by evaluating x to 0 and to 1 are two distinct functions. Otherwise, x is called an* inessential variable *of f.*

Definition 27 *Let A be a tree and ν one of its nodes (either an internal node or a leaf). We define $\Delta(\nu)$ to be the smallest left subtree of A (cf. Definition 12) containing ν, if there exists one, and to be the whole tree A itself, if there is no such left subtree. Hence, $\Delta(\nu)$ is the whole tree if ν is at left depth 0, whereas it is a left subtree if ν has a positive left depth.*

In the same way, for a node ν of positive left depth, we define $\Delta^2(\nu)$ to be the smallest left subtree strictly containing $\Delta(\nu)$ as a subtree, if such a tree exists, to be the whole tree if $\Delta(\nu)$ is a proper subtree of A. Otherwise (if ν is at left depth 0), $\Delta^2(\nu)$ is undefined. See Figure 6.

We shall also write $\Delta(B)$ for a subtree B as a shortcut for $\Delta(\nu)$, where ν is the root of B (and in the same way $\Delta^2(B)$ for $\Delta^2(\nu)$).

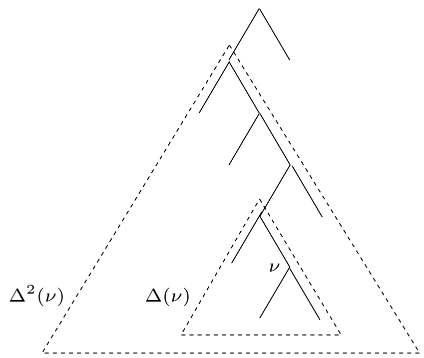

Figure 6: The left subtrees $\Delta(\nu)$ and $\Delta^2(\nu)$ associated to a node ν of a tree.

Lemma 28 *Any tree A computing a function $f \neq$ True contains at least $L(f)$ occurrences of essential variables of f.*

Proof: Let A be a tree computing $f \neq$ *True*. First remark that the goal of A is an essential variable; otherwise A would compute the constant *True* function.

Now consider the set $D = \{\Delta(\nu) \mid \nu \text{ is labelled with an inessential variable}\}$. Note that for two inessential variables ν_1 and ν_2 the two trees $\Delta(\nu_1)$ and $\Delta(\nu_2)$ have no common vertices unless one of these trees is a subtree of the other. Therefore there exist maximal trees with respect to the subtree relation. Let $\{\Delta_1, \ldots, \Delta_p\}$ be the set of maximal trees in D. Of course all Δ_i are disjoint and all of them are left subtrees, since the goal of A is an essential variable.

We claim that the tree $A' = A \setminus \{\Delta_1, \ldots, \Delta_p\}$ (cf. Definition 15) computes f. Assign the value 1 to all inessential variables. Then all Δ_i evaluate to 1, since their goals are inessential variables. Since the Δ_i are left subtrees, each Δ_i has a right brother T_i and $\Delta_i \to T_i$ is a subtree of A. Clearly, $[\Delta_i \to T_i] = [T_i]$ and therefore A' computes indeed f.

Since A' contains only essential variables, we have $|A| \geqslant |A'| \geqslant L(f)$ and it follows that there are at least $L(f)$ essential variables. $\square$

The size of a tree A computing $f \neq True$ can be written as $|A| = L(f) + \eta + \theta$, where $L(f) + \eta$ is the number of leaves labelled with essential variables and θ is the number of leaves labelled with inessential variables; notice that $\eta \geqslant 0$ by Lemma 28. Given a function f different from *True*, we decompose the set of irreducible trees computing f into the following disjoint sets:

- $\mathcal{M}_f$ is the set of all minimal trees, i.e. trees of size $L(f)$ (case $\eta = \theta = 0$);
- $\mathcal{P}_{f,1}$ is the set of irreducible trees of size greater than $L(f)$, with exactly $L(f)$ occurrences of essential variables and at least one occurrence of an inessential variable (case $\eta = 0$, $\theta > 0$);
- $\mathcal{P}_{f,2}$ is the set of irreducible trees of size $L(f) + 1$, without any occurrence of inessential variables (case $\eta = 1$, $\theta = 0$);
- $\mathcal{P}_{f,3}$ is the set of irreducible trees of size greater than $L(f) + 1$, with exactly $L(f) + 1$ occurrences of essential variables and $\theta > 0$ occurrences of *all distinct* inessential variables (case $\eta = 1$, $\theta > 0$, first part);
- $\mathcal{P}_{f,4}$ is the set of irreducible trees of size greater than $L(f) + 2$, with exactly $L(f) + 1$ occurrences of essential variables and $\theta > 0$ occurrences of inessential variables such that at least one inessential variable is repeated (case $\eta = 1$, $\theta > 0$, second part);
- $\mathcal{P}_{f,5}$ is the set of irreducible trees containing at least $L(f)+2$ occurrences of essential variables (case $\eta \geqslant 2$, $\theta \geqslant 0$).

Of course any tree computing f falls in an iterated expansion of an irreducible tree computing f (obtained by repeated pruning). Theorem 7 relies on evaluating the limiting ratios of $E^*(\mathcal{C})$ for each of the classes $\mathcal{C}$ defined above.

Let f be a Boolean function different from *True*, and Γ the set of its essential variables. Let us define

$$\mathcal{N} \;=\; \mathcal{A}_{L(f)+2}^{L(f)+2}(\Gamma) \;\cup \bigcup_{\substack{\alpha \in \{x_1,\ldots,x_k\} \\ \alpha \notin \Gamma}} \mathcal{A}_{L(f)+2}^{L(f)+3}(\Gamma \cup \{\alpha\}) \;\cup \bigcup_{\substack{\alpha,\beta \in \{x_1,\ldots,x_k\} \\ \alpha,\beta \notin \Gamma,\ \alpha \neq \beta}} \mathcal{A}_{L(f)+2}^{L(f)+4}(\Gamma \cup \{\alpha, \beta\}).$$

Note that $\mathcal{N}$ implicitly depends on the considered function f. We shall make constant use of the following:

Proposition 29 *For any function f different from True, $\bar{\mu}_k(E^*(\mathcal{N})) = O(1/k^{L(f)+2})$.*

Remark. Note that in view of Theorem 7 this means that all trees which are either elements of $\mathcal{N}$ or obtained by (iterated) expansion of a tree in $\mathcal{N}$ form a set of asymptotically negligible size.

Proof: By Corollary 25, it holds that $\bar{\mu}_k(\mathcal{A}_{L(f)+2}^{L(f)+2}(\Gamma)) = O(1/k^{L(f)+2})$. Using the same corollary, we obtain that for any variable $\alpha \notin \Gamma$:

$$\bar{\mu}_k(E^*(\mathcal{A}_{L(f)+2}^{L(f)+3}(\Gamma \cup \{\alpha\}))) = O\left(\frac{1}{k^{L(f)+3}}\right).$$

In the same way, for any distinct variables $\alpha \notin \Gamma$ and $\beta \notin \Gamma$:

$$\bar{\mu}_k(E^*(\mathcal{A}_{L(f)+2}^{L(f)+4}(\Gamma \cup \{\alpha, \beta\}))) = O\left(\frac{1}{k^{L(f)+4}}\right).$$

The result follows. $\square$

5.1 Expansions of minimal trees

We first need to establish some restrictions on the possible types of expansions in $\mathcal{M}_f$ (these restrictions also hold for the family $\mathcal{P}_{f,2}$: this will be used later).

Definition 30 *An assignment (partial evaluation) is the substitution of some (or all) of the variables $x_1, \ldots, x_k$ by values True or False (denoted by 1 and 0). Given an assignment b and a tree T, the expression $[T_{|b}]$ denotes the function computed by the tree T under the assignment b, i.e., the function we obtain, if we replace the variables which are affected by b by their values as given in b.*

For example, the tree $T = (x \to y) \to ((z \to z) \to w)$ computes the function $[T] = (x \to y) \to w$. Under various assignments we may get different functions: $[T_{|z=1}] = [T]$, $[T_{|x=1}] = y \to w$, $[T_{|y=1}] = w$, $[T_{|x=1,y=0}] = 1$.

Likewise, we use the notation $f_{|b}$ if we partially evaluate a function instead of a tree.

Lemma 31 *Let f be a Boolean function different from True, and $A \in \mathcal{M}_f \cup \mathcal{P}_{f,2}$. No valid expansion of type "goal α" or "premise α" with respect to an inessential variable α is possible in A.*

Proof: Assume by contradiction that it is possible to perform a valid expansion of type "goal α" or "premise α" with respect to an inessential variable α in some node ν of $A \in \mathcal{M}_f \cup \mathcal{P}_{f,2}$. Notice that the left depth of ν is at least 1; otherwise f would be equal to *True*. We shall first prove that $|\Delta(\nu)| = 1$.

Suppose the valid expansion in ν is of type "goal α". Let A' be the tree obtained by expanding A in ν with the left subtree equal to a leaf α. Then we have $[A'_{|\alpha=0}] = [A \setminus \Delta(\nu)] = f$. We conclude that $|\Delta(\nu)| = 1$ and $|A| = L(f) + 1$ (otherwise we would have a tree smaller than $L(f)$ computing f).

Suppose now the valid expansion in ν is of type "premise α". Let x be the goal of A, and let A' be the tree obtained by expanding A in ν with the left subtree $\alpha \to x$. Then we have $[A'_{|\alpha=1,\ x=0}] = [A \setminus \Delta(\nu)_{|x=0}] = f_{|x=0}$ and of course $[A \setminus \Delta(\nu)_{|x=1}] = 1 = f_{|x=1}$. Again we conclude that $[A \setminus \Delta(\nu)] = f$; it follows that $|\Delta(\nu)| = 1$ and $|A| = L(f) + 1$ in this case too.

So, no matter what type of valid expansion we apply, we know that $\Delta(\nu)$ is a leaf, say, with label y. In the tree A, the left subtree $\Delta(\nu)$ computes y. Moreover, for both types of expansion, we have shown that $A \setminus \Delta(\nu)$ still computes f; in other words: substituting $\Delta(\nu)$ by 1 or its goal y in A does not change the computed function. It follows by Lemma 17 that A is reducible, which is absurd. $\Box$

Notice that Lemma 31 shows that $\lambda(f)$ defined in Section 2 does not depend on the number of variables k we consider.

Lemma 32 *Let f be a Boolean function different from True. Then the tree class obtained by applying a single expansion to any minimal tree of f, $E(\mathcal{M}_f)$, has a limiting ratio. Moreover, we get:*

$$\mu_k(E(\mathcal{M}_f)) = \frac{\lambda(f)}{4^{L(f)}\ k^{L(f)+1}} + O\left(\frac{1}{k^{L(f)+2}}\right).$$

Proof: Let f be a Boolean function different from *True*. We will compute the formula given in the assertion which in particular implies the existence of the limiting ratio. Given $A \in \mathcal{M}_f$ and ν a node of A, let $\lambda_\nu(A)$ the number of valid expansion types in the node ν of A, and $E_\nu(A)$ the set of trees obtained by a single valid expansion in ν.

Notice that, given a tree B obtained from $A \in \mathcal{M}_f$ by adding one left subtree of size at least $L(f)$, it is possible to determine A; this is because B has only one left subtree of size $|B| - L(f)$. Hence for $n \geqslant 2L(f)$, it holds that

$$|\{T \in E(\mathcal{M}_f)\ :\ |T| = n\}| = \sum_{A \in \mathcal{M}_f, \nu \in A} |\{T \in E_\nu(A)\ :\ |T| = n\}|.$$

It follows that:

$$\mu_k(E(\mathcal{M}_f)) = \sum_{A \in \mathcal{M}_f, \nu \in A} \mu_k(E_\nu(A)). \tag{3}$$

For $A \in \mathcal{M}_f$ and ν a node of A, we want to estimate $\mu_k(E_\nu(A))$ now. Let us first consider the limiting ratio of trees obtained by an expansion of a given type in a node ν of a tree $A \in \mathcal{M}_f$. For t a type of expansion ("tautology", "goal α", or "premise α"), we denote by $E^t_\nu(A)$ the set of all trees obtained from A by expanding it in ν with respect to the valid expansion of type t. The generating function of $E^{taut}_\nu(A)$ is equal to $\phi_{\mathcal{T}_k}(z) \cdot z^{L(f)}$ and it can be computed that its limiting ratio is $\mu_k(E^{taut}_\nu(A)) = 1/4^{L(f)}k^{L(f)+1} + O(1/k^{L(f)+2})$. The generating function of $E^{goal\ \alpha}_\nu(A)$ is equal to $(1/k)f_k(z)z^{L(f)}$ and has limiting ratio $\mu_k(E^{goal\ \alpha}_\nu(A)) = 1/4^{L(f)}k^{L(f)+1} + O(1/k^{L(f)+2})$. In the same way, the generating function of $E^{prem\ \alpha}_\nu(A)$ is equal to $(z/(1-f_k(z)+z))f_k(z)z^{L(f)}$ and has limiting ratio $\mu_k(E^{prem\ \alpha}_\nu(A)) = 1/4^{L(f)}k^{L(f)+1} + O(1/k^{L(f)+2})$.

Let $\mathcal{I} = E^{t_1}_\nu(A) \cap E^{t_2}_\nu(A)$ where $t_1 \neq t_2$ are two distinct valid types of expansions in the node ν of A. From the structure of tautologies recalled in Section 3, it follows that $\mathcal{I} \subseteq \mathcal{N}$. Proposition 29 gives $\bar{\mu}_k(\mathcal{I}) = O(1/k^{L(f)+2})$ and by inclusion-exclusion principle, we obtain:

$$\mu_k(E_\nu(A)) = \frac{\lambda_\nu(A)}{4^{L(f)}\ k^{L(f)+1}} + O\left(\frac{1}{k^{L(f)+2}}\right). \tag{4}$$

Together Equations (3) and (4) yield the result. $\square$

The last step towards the proof of Theorem 7 is to study the limiting ratio of expansions of the minimal trees computing a given function. Recall that $\mathcal{N}$ is a set of negligible size (Proposition 29). The next lemma shows that by applying several consecutive expansions we do not gain significantly more trees than by applying just one expansion on minimal trees.

Lemma 33 *For a Boolean function f different from True, $E^{\geqslant 2}(\mathcal{M}_f) \setminus E(\mathcal{M}_f) \subseteq \mathcal{N}$.*

Proof: Let $A_2 \in E^2(\mathcal{M}_f) \setminus E(\mathcal{M}_f)$. There exist $A_0 \in \mathcal{M}_f$ and $A_1 \in E(\mathcal{M}_f)$ such that $A_1 \in E(A_0)$ and $A_2 \in E(A_1)$. Let B_1 be the left subtree which has been added to A_0 to get A_1, and B_2 be the left subtree which has been added to A_1 to get A_2.

By Lemma 31 we know that A_1 is not obtained after an expansion of type "goal α" or "premise α" with respect to an inessential variable α; hence $A_1 \in \mathcal{A}^{L(f)+1}_{L(f)+1}(\Gamma) \cup \bigcup_{\alpha \notin \Gamma} \mathcal{A}^{L(f)+2}_{L(f)+1}(\Gamma \cup \{\alpha\})$ – A_1 belongs to the second part if it is obtained after an expansion of type *tautology* whose goal is an inessential variable. Observe that $A_1 \in \mathcal{N}$ if B_1 is a tautology which is not a *simple tautology.* Indeed, by Lemma 14 there exist either three occurrences of the same variable or two distinct variables appearing twice among the leaves of the four first left levels of the tautology. Thus, if B_1 is a tautology which is not simple, $A_1 \in \mathcal{N}$ and consequently A_2 belongs to the same set.

Now, we assume that if B_1 is a tautology, then it is a simple tautology. Let S_1 be the set of the leaves of B_1, whose left depth (with respect to B_1) is smaller or equal to 2. If there exist two nodes in S_1 such that their labels are essential variables, or three nodes such that their labels are one essential variable and two occurrences of an inessential variable, or three nodes labelled with the same inessential variable, or four nodes such that their labels are two repetitions of two distinct inessential variables, then $A_1 \in \mathcal{N}$ and consequently A_2 belongs to the same set. Now suppose this is not the case. We consider two cases, depending on the location of the second expansion with respect to the first one. We recall that in both cases, $A_1 \in \mathcal{A}^{L(f)+1}_{L(f)+1}(\Gamma) \cup \bigcup_{\alpha \notin \Gamma} \mathcal{A}^{L(f)+2}_{L(f)+1}(\Gamma \cup \{\alpha\})$.

First case: The node in which the second expansion B_2 has been done does not belong to B_1. If B_2 is a tautology then we conclude that $A_2 \in \mathcal{N}$.

If B_2 is added after an expansion of type "goal α" or "premise α" with respect to an essential variable α then $A_2 \in \mathcal{N}$. Now, we assume that B_2 is added after an expansion of type "goal α" or "premise α" with respect to an inessential variable.

Let S_2 be the set of the leaves of B_2 whose left depth (with respect to B_2) is smaller or equal to 1. If S_2 contains an essential variable, then $A_2 \in \mathcal{N}$ and we are done. For the same reason, if there is a repetition of an inessential variable in $S_1 \cup S_2$, then $A_2 \in \mathcal{N}$. In the following we assume this is not the case. Thus there exists a partial assignment of the inessential variables such that B_2 computes 0 and no variable of S_1 is valuated. Then, by evaluating the inessential variables of B_1, it is not possible to find an assignment such that B_1 computes 1 – otherwise $A \setminus \{B_1, \Delta^2(B_2)\}$ would compute f with size at most $L(f) - 1$. Since it is not possible to evaluate B_1 to 1, its goal must be an essential variable. But no other essential variable does label a node of S_1 and there exists no repetition in the labels of S_1. If B_1 was not reduced to a leaf, we could evaluate B_1 to 1. But it is not the case, so B_1 is reduced to a leaf, necessarily labelled by an essential variable. We conclude that the tree $A' := A_2 \setminus \Delta^2(B_2)$ computes f and its size is $L(f)$, so it is a minimal tree. There exists an assignment of the inessential variables such that B_2 computes 0 and another one such that it computes 1 (because $r(B_2)$ is inessential). So $\Delta^2(B_2)$ computes respectively 1 and $r(\Delta^2(B_2))$ under these assignments. Using Lemma 17, we conclude that A_2 is obtained after one valid expansion of the minimal tree A', this is absurd since we assumed $A_2 \notin E(\mathcal{M}_f)$.

Second case: The node in which the second expansion B_2 has been done belongs to B_1. If the first expansion of A_0 is of type "goal α", then A_2 can be obtained by a single expansion of A_0: this is absurd because $A_2 \notin E(\mathcal{M}_f)$. If the first expansion of A_0 is of type "premise x", then the only case such that $A_2 \notin E(\mathcal{M}_f)$, is that B_2 is an expansion of the premise reduced to x in B_1. Furthermore, we can assume no element of S_2 is labelled in the same way as those from S_1 (otherwise $A_2 \in \mathcal{N}$ and we are done). So there exists an assignment of the inessential variables such that B_1 (which contains B_2) computes 0. Then $A_2 \setminus \Delta^2(B_1)$ computes f with size at most $L(f) - 1$; this is absurd. Finally it remains to consider the case where the first expansion is a tautology. If it is a simple tautology (the other case has been studied before), then the only case such that $A_2 \notin E(\mathcal{M}_f)$ is that B_2 is an expansion of the special premise which is equal to the goal of tautology. For the same reason as in the previous case, this is not possible. $\square$

5.2 Expansions of other irreducible trees

We first prove that the two sets $\mathcal{P}_{f,1}$ and $\mathcal{P}_{f,3}$ are empty.

Lemma 34 *For any Boolean function f different from True, the set $\mathcal{P}_{f,1}$ is empty.*

Proof: Suppose that $\mathcal{P}_{f,1}$ is not empty, and let $A \in \mathcal{P}_{f,1}$. The size of A is $L(f) + \theta$, with exactly $L(f)$ occurrences of essential variables and $\theta > 0$ occurrences of inessential ones. Let $\{\Delta_1, \ldots, \Delta_p\}$ be the set of maximal $\Delta(\nu)$ (with respect to the subtree relation) when ν runs over all the leaves of A labelled by an inessential variable. If we assign the value 1 to all inessential variables, all Δ_i attain the value 1, because their goals are inessential variables. Moreover, since they are left subtrees, the tree $A' := A \setminus \{\Delta_1, \ldots, \Delta_p\}$ computes f. Since A contains $L(f)$ occurrences of essential variables, no Δ_i contains an essential variable.

Suppose now that there is no assignment of the inessential variables such that Δ_1 attain the value 0: then Δ_1 is a tautology and A is reducible, this is absurd. Hence there exists an assignment a of all inessential variables such that Δ_1 attains the value 0 under the assignment a. Notice that Δ_1 cannot be a premise of A because $f \neq$ *True*, so $\Delta_1^2 := \Delta^2(\Delta_1)$ is really a left subtree (not the whole tree itself) and attains the value 1 under the assignment a (because its premise Δ_1 computes 0). Let $S = \{\Delta_i \mid [\Delta_{i|a}] = 1\} \cup \{\Delta_i^2 \mid [\Delta_{i|a}] = 0\}$. The set S is composed of left subtrees, all evaluating to 1 under a. Moreover S contains Δ_1^2 which contains at least one essential variable (its goal) – otherwise Δ_1 would not be maximal. Thus $A'' := A \setminus S$ is of size at most $L(f) - 1$ and computes f, this is absurd. $\square$

Lemma 35 *For any Boolean function f different from True, the set $\mathcal{P}_{f,3}$ is empty.*

Proof: Let us suppose that $\mathcal{P}_{f,3}$ is not empty, and let $A \in \mathcal{P}_{f,3}$. Let $|A| = L(f) + 1 + \theta$, A containing θ occurrences of all distinct inessential variables. We define $\{\Delta_1, \ldots, \Delta_p\}$ as in proof of Lemma 34. Since $[A] = [A \setminus \{\Delta_1, \ldots, \Delta_p\}]$, the set $\bigcup_i \Delta_i$ contains at most one occurrence of an essential variable.

Suppose first that $\bigcup_i \Delta_i$ contains only inessential variables. Notice that in this case, there exists an assignment of the inessential variables such that all Δ_i evaluate to 0 (such an assignment exists since there is no repetition of inessential variables). It follows that $A \setminus \{\Delta_1^2, \ldots, \Delta_p^2\}$ is a tree computing f. Hence all Δ_i^2 are equal to the same left subtree of A. It follows that the Δ_i are all (and the only) premises of a single left subtree of A whose goal is an essential variable α. We claim that in this case, A is obtained from $A \setminus \Delta_1^2$ by a single valid expansion of the type "goal α". Indeed, put all inessential variables to 1. Then Δ_1^2 attains the value α and A computes f under this assignment. Now choose a (complete) assignment of the inessential variables such that Δ_1 evaluates to 0 (it exists). Then Δ_1^2 evaluates to 1 and the tree A computes f under this assignment. By Lemma 17, A is reducible. This is absurd.

Now, let us suppose that $\bigcup_i \Delta_i$ contains exactly one essential variable – say, it lies in Δ_1. There exists an assignment of the inessential variables such that Δ_1 attains the value 1 and all other Δ_i attain 0. Thus $A \setminus \{\Delta_1, \Delta_2^2, \ldots, \Delta_p^2\}$ computes f. But its size is strictly smaller than $L(f)$ if $p > 1$. Thus necessarily $p = 1$. By definition of Δ_1, α cannot be its goal. So α lies in a premise of Δ_1.

Case 1: this premise is of size 1 (i.e. it is reduced to α). We claim that in this case, A is obtained from $A \setminus \Delta_1$ by a single valid expansion of the type "premise α". Indeed, put all inessential variables to 1. Then Δ_1 evaluates to 1 and the obtained tree computes f. Now choose an assignment of the inessential variables such that Δ_1 evaluates to $\bar{\alpha}$ (it exists). The obtained tree computes f. By Lemma 18, A is reducible. This is a contradiction.

Case 2: this premise is of size at least 2. In this case, we can find an assignment of the inessential variables such that Δ_1 evaluates to 0 (write Δ_1 as an and/or tree and notice that α lies in the scope of a conjunction). Hence, cutting Δ_1^2 from A yields a tree strictly smaller than $L(f)$ computing f. This is absurd. □

We shall study expansions of $\mathcal{P}_{f,2}$ now. Note that $\mathcal{P}_{f,2}$ may be empty or not, depending on f: $\mathcal{P}_{f,2}$ is empty for $f = x_1$ while $\mathcal{P}_2(g)$ is not empty for $g = x_1 \vee (\bar{x}_2 \wedge x_3 \wedge x_4) \vee (\bar{x}_2 \wedge \bar{x}_5)$. Indeed, it can be checked that $L(g) = 6$ and $(x_3 \to (x_4 \to x_2)) \to (((x_5 \to x_2) \to x_2) \to x_1)$ belongs to $\mathcal{P}_2(g)$.

Lemma 36 *For any Boolean function f different from True, $E^+(\mathcal{P}_{f,2}) \subseteq \mathcal{N}$.*

Proof: This is trivial if $\mathcal{P}_{f,2}$ is empty. Assume $\mathcal{P}_{f,2} \neq \emptyset$, and let $A \in E(\mathcal{P}_{f,2})$. We have $A \in E(I)$ for some irreducible tree $I \in \mathcal{P}_{f,2}$. We will prove that A satisfies one of the following conditions:

- A contains at least $L(f)+2$ occurrences of essential variables with left depth at most $L(f)+2$;
- A contains $L(f) + 1$ occurrences of essential variables and two occurrences of the same inessential variable, all with a left depth at most $L(f) + 2$.

This will prove that $A \in \mathcal{N}$.

If A is obtained from I by an expansion of type *goal* or *premise*, then this expansion must be done with respect to an essential variable by Lemma 31. Now remark that I is of size $L(f) + 1$, so all its nodes are of left depth at most $L(f)$ (there is at least a node per left depth). Since expansions preserve left depth of nodes present in the initial tree, we conclude that A satisfies the first condition above.

Suppose now that A is obtained from I by an expansion of type *tautology*. From the results recalled in Section 3, we know that this tautology contains two occurrences of some variable x in its nodes of left depth at most 1. If x is an essential variable of f, then A satisfies the first condition above. Otherwise, if x is an inessential variable of f, then A satisfies the second condition above. □

We turn our attention towards $\mathcal{P}_{f,4}$ and $\mathcal{P}_{f,5}$ now – it is easily checked that both $\mathcal{P}_{f,4}$ and $\mathcal{P}_{f,5}$ are non empty for any function $f \neq$ *True*.

Proposition 37 *For any Boolean function f different from True, $\mathcal{P}_{f,4} \cup \mathcal{P}_{f,5} \subseteq \mathcal{N}$.*

Proof: Let $A \in \mathcal{P}_{f,4} \cup \mathcal{P}_{f,5}$. Let $|A| = L(f) + \eta + \theta$ be the size of A, where $L(f) + \eta$ corresponds to the number of occurrences of essential variables, and θ is the number of occurrences of inessential variables. Either $\eta \geqslant 2$ and $\theta \geqslant 0$ or $\eta = 1$ and $\theta \geqslant 2$ with at least two occurrences of the same inessential variable. In the same way as before, let

$$\{\Delta_1, \ldots, \Delta_p\} = \{\Delta(\nu) \mid \nu \text{ inessential and } \Delta(\nu) \text{ maximal}\} \tag{5}$$

where, as usual, maximality is meant with respect to the subtree relation. Let a be the partial evaluation where all inessential variables are equal to 1. Then we have

$$[A \setminus \{\Delta_1, \ldots \Delta_p\}] = [A_{|a}] = [A]. \tag{6}$$

In the subsequent lemmas we will prove that A must lie in $\mathcal{N}$ by showing that enough variables repeat before a fixed left depth in A. $\square$

In order to complete the previous proof, we will decompose the set $\mathcal{P}_{f,4} \cup \mathcal{P}_{f,5}$ into the following disjoint subsets:

- $\mathcal{Q}_{f,0} = \{t \in \mathcal{P}_{f,4} \cup \mathcal{P}_{f,5}$ such that there are no inessential variables in $t\}$. This corresponds to the case $p = 0$ of (5).
- $\mathcal{Q}_{f,1} = \{t \in \mathcal{P}_{f,4} \cup \mathcal{P}_{f,5}$ such that all inessential variables are contained in the same maximal $\Delta(\nu)\}$. Here we have only one maximal subtree Δ_1 and thus we are in the case $p = 1$ of (5).
- $\mathcal{Q}_{f,2} = (\mathcal{P}_{f,4} \cup \mathcal{P}_{f,5}) \setminus (\mathcal{Q}_{f,0} \cup \mathcal{Q}_{f,1})$. This corresponds to the case $p \geqslant 2$ of (5).

Lemma 38 *For f different from True we have $\mathcal{Q}_{f,0} \subseteq \mathcal{N}$.*

Proof: In this case we have $\theta = 0$ and $\eta \geqslant 2$ and hence $A \in \mathcal{A}^{L(f)+2}_{L(f)+1}(\Gamma) \subseteq \mathcal{N}$. $\square$

Lemma 39 *For f different from True we have $\mathcal{Q}_{f,1} \subseteq \mathcal{N}$.*

Proof: In this case (6) reduces to $[A \setminus \Delta_1] = [A_{|a}] = [A]$. By definition of Δ_1 the tree $A \setminus \Delta_1$ contains only essential variables and has size at least $L(f)$. We subdivide this case according to the size of $A \setminus \Delta_1$.

- $A \setminus \Delta_1$ has size at least $L(f) + 2$. In the same way as in the case $p = 0$, we obtain that $A \in \mathcal{A}^{L(f)+2}_{L(f)+1}(\Gamma) \subseteq \mathcal{N}$.
- $A \setminus \Delta_1$ has size $L(f) + 1$. We denote the premises of Δ_1 by B_i and its goal by β. We consider two different cases:

  1. There exists i such that $r(B_i) \in \Gamma \cup \{\beta\}$. Then we have $L(f)+3$ occurrences of variables of $\Gamma \cup \{\beta\}$ whose left depths are all smaller than $L(f)+2$. Thus $A \in \mathcal{A}^{L(f)+3}_{L(f)+2}(\Gamma \cup \{\beta\}) \subseteq \mathcal{N}$.
  2. For all i, $r(B_i)$ is not an essential variable and different from β. Let b be an assignment of the inessential variables such that $\beta = 0$ and all other inessential variables are valuated to 1. Then $[\Delta_{1|b}] = 0$ and it follows that $[\Delta^2_{1|b}] = 1$ (Δ^2_1 exists because otherwise Δ_1 would be a premise of the whole tree and under b, the whole tree would compute 1). Thus $[\Delta^2_{1|b}] = 1$ and $[A_{|b}] = f$. Now the goal of Δ^2_1 is an essential variable; let us denote it by x. Remark that Δ^2_1 does not contain another essential variable since $[A \setminus \Delta^2_1] = f$. Consider an assignment $\tilde{b}$ of the inessential variables satisfying $\beta = 1$; in this case $[\Delta^2_{1|\tilde{b}}] = x$ and $[A_{|\tilde{b}}] = f$. Lemma 17 implies that A is reducible to $A \setminus \Delta^2_1$, a contradiction.

- $A \setminus \Delta_1$ has size $L(f)$. We denote the premises of Δ_1 by B_i and its goal by β.

1. For all i, $r(B_i) \notin \Gamma \cup \{\beta\}$. By taking $\beta = 0$ and all other inessential variables equal to 1, Δ_1 computes 0 and then Δ_1^2 computes 1. So $A \setminus \Delta_1^2$ computes f with at most $L(f) - 1$ occurrences of essential variables (because the goal of Δ_1^2 is an essential variable). This is absurd.
2. There exist $i \neq j$ such that $r(B_i)$ and $r(B_j)$ are both in $\Gamma \cup \{\beta\}$. Then there are $L(f)+3$ occurrences of variables of $\Gamma \cup \{\beta\}$ in A with left depth at most $L(f)+1$. Thus $A \in \mathcal{A}_{L(f)+1}^{L(f)+3}(\Gamma \cup \{\beta\}) \subseteq \mathcal{N}$.
3. There exists a unique i such that $r(B_i) \in \Gamma \cup \{\beta\}$. Let us suppose first that B_i is reduced to a leaf. If $B_i = \beta$, then Δ_1 is a simple tautology; it follows that A is reducible, absurd. Thus B_i is an essential variable x. Let b be the assignment of the inessential variables such that $\beta = 0$ and all other inessential variables evaluate to 1. Under b, Δ_1 computes $\bar{x}$, and A computes f. Now consider another assignment $\tilde{b}$ with $\beta = 1$; under $\tilde{b}$, Δ_1 computes 1, and A computes f. Using Lemma 18, we conclude that A is a expansion of $A \setminus \Delta_1$ of type "premise x"; this is a contradiction.

   Hence B_i is not reduced to a leaf. Let $B_i = C_1 \dots, C_\ell \to \gamma$ with $\ell \geqslant 1$ and $\gamma \in \Gamma \cup \{\beta\}$. We consider different subcases now:

   – If there exists j such that $r(C_j) \in \Gamma \cup \{\beta\}$, then A contains $L(f)+3$ occurrences of variables of $\Gamma \cup \{\beta\}$ with left depth at most $L(f)+2$. It follows that $A \in \mathcal{A}_{L(f)+2}^{L(f)+3}(\Gamma \cup \{\beta\}) \subseteq \mathcal{N}$.

   – If there exist $j \neq j'$ such that $r(C_j) = r(B_{j'})$ or $r(C_j) = r(C_{j'})$. Let $\delta = r(C_j)$. Then A contains $L(f)+4$ occurrences of variables of $\Gamma \cup \{\beta, \delta\}$ with left depth at most $L(f)+2$. It follows that $A \in \mathcal{A}_{L(f)+2}^{L(f)+4}(\Gamma \cup \{\beta, \delta\}) \subseteq \mathcal{N}$.

   – If all $r(C_j)$ are distinct and for all j, $r(C_j) \notin \Gamma \cup \{\beta\} \bigcup_{j'} \{r(B_{j'})\}$. If C_1 is reduced to a leaf, consider the following assignment: $C_1 = 0$, $\beta = 0$ and all other inessential variables equal to 1. Then Δ_1 computes 0, so Δ_1^2 computes 1, and $A \setminus \Delta_1^2$ computes f with at most $L(f) - 1$ occurrences of essential variables, absurd. Hence C_1 is not reduced to a leaf; let $C_1 = D_1, \dots, D_m \to \delta$, with $m \geqslant 1$ and δ an inessential variable. If there exists j such that $r(D_j) \in \Gamma \cup \{\beta, \delta\}$, then $A \in \mathcal{A}_{L(f)+3}^{L(f)+4}(\Gamma \cup \{\beta\}) \subseteq \mathcal{N}$.

     Otherwise, if there exist $j \neq j'$ such that $r(D_j) = r(B_{j'})$ or $r(D_j) = r(C_{j'})$, or $r(D_j) = r(D_{j'})$: let ϵ denote this repeated inessential variable. Then A contains $L(f)+4$ occurrences of variables of $\Gamma \cup \{\beta, \epsilon\}$ with left depth at most $L(f)+3$. It follows that $A \in \mathcal{A}_{L(f)+3}^{L(f)+4}(\Gamma \cup \{\beta, \epsilon\}) \subseteq \mathcal{N}$.

     If we are not in the previous cases, every D_j computes 0 by evaluating its goal to 0 and all its subgoals to 1 (recall that the subgoals of D_j are the goals of its premises): there is no conflict because all are different. So for this assignment C_1 computes 0, B_i evaluates to 1 and at last the whole subtree Δ_1 evaluates to 0. It follows that $A \setminus \Delta_1^2$ computes f with at most $L(f) - 1$ occurrences of essential variables, which is absurd.

□

Lemma 40 *For f different from True we have $\mathcal{Q}_{f,2} \subseteq \mathcal{N}$.*

Proof: As in the previous lemma our starting point is (6), and we conclude that $A \setminus \{\Delta_1, \dots \Delta_p\}$ contains only essential variables and has size at least $L(f)$. Again we consider different cases according to the size of $A \setminus \{\Delta_1, \dots \Delta_p\}$.

1. $A \setminus \{\Delta_1, \dots \Delta_p\}$ has size at least $L(f)+2$. Then $A \in \mathcal{A}_{L(f)+1}^{L(f)+2}(\Gamma) \subseteq \mathcal{N}$.
2. $A \setminus \{\Delta_1, \dots \Delta_p\}$ has size $L(f)+1$. Let S be the set of goals and subgoals of the subtrees $\{\Delta_1, \dots, \Delta_p\}$. If there exists a node of S labelled with an essential variable, then $A \in$

$\mathcal{A}_{L(f)+2}^{L(f)+2}(\Gamma) \subseteq \mathcal{N}$. Otherwise, if there exist two nodes of S labelled with the same inessential variable α, then $A \in \mathcal{A}_{L(f)+2}^{L(f)+3}(\Gamma \cup \{\alpha\}) \subseteq \mathcal{N}$. Finally, if we are not in the previous cases, we are able to find an assignment a of the inessential variables such that all Δ_i are computing 0 under a. Then the tree $A \setminus \{\Delta_1^2, \ldots, \Delta_p^2\}$ computes f; since it must be of size at least $L(f)$, it follows that all Δ_i^2 are equal to the same subtree. Let $x = r(\Delta_1^2)$ (x is necessarily an essential variable). We shall prove that A is obtained from $A \setminus \Delta_1^2$ by a single valid expansion of the type "goal x". Indeed, put all inessential variables to 1. Then Δ_1^2 evaluates to x and A computes f. Now choose a (complete) assignment of the inessential variables such that all Δ_i evaluate to 0 (it exists, because there are no repetition in the labels of S). Then Δ_1^2 evaluates to 1, and the whole tree computes f. By Lemma 17, A is reducible, this case is not possible.

3. $A \setminus \{\Delta_1, \ldots \Delta_p\}$ has size $L(f)$. Again, let S be the set of the goals and the subgoals of the subtrees $\{\Delta_1, \ldots, \Delta_p\}$. If S contains at least two occurrences of essential variables then A belongs to $\mathcal{A}_{L(f)+1}^{L(f)+2}(\Gamma) \subseteq \mathcal{N}$. Else if there exists one occurrence of an essential variable and two occurrences of the same inessential variables α in S then A belongs to $\mathcal{A}_{L(f)+1}^{L(f)+3}(\Gamma \cup \{\alpha\}) \subseteq \mathcal{N}$. Else if there exist three occurrences of the same inessential variable α or four occurrences of two different inessential variables α and β in S then A belongs to $\mathcal{A}_{L(f)+1}^{L(f)+3}(\Gamma \cup \{\alpha\}) \subseteq \mathcal{N}$ or $\mathcal{A}_{L(f)+1}^{L(f)+4}(\Gamma \cup \{\alpha, \beta\}) \subseteq \mathcal{N}$. There are three other cases. *First case:* S contains a single essential variable and no repetition among the inessential variables. Let Δ_1 be the subtree containing a subgoal labelled with an essential variable. There exists an assignment of the inessential variables such that Δ_1 computes 1 and all the other Δ_i compute 0. So we conclude that the tree $A \setminus \{\Delta_1, \Delta_2^2, \ldots, \Delta_p^2\}$ computes f with at most $L(f) - 1$ occurrences of essential variables (since $p \geqslant 2$). This is absurd. *Second case:* there exists no essential variable among S but there are exactly two occurrences of the same inessential variable. Then there exists an assignment of the inessential variables such that all the Δ_i compute 0 or a single Δ_j computes 1 and all the other compute 0. So we conclude that the tree $A \setminus \{\Delta_1^2, \ldots, \Delta_p^2\}$ (or $A \setminus \{\Delta_1^2, \ldots, \Delta_{j-1}^2, \Delta_j, \Delta_{j+1}^2, \ldots, \Delta_p^2\}$ in the other case) computes f, with at most $L(f) - 1$ occurrences of essential variables. This is absurd. *Third case:* there exists no essential variable and no repetition among S. Then there exists an assignment of inessential variables such that all the Δ_i compute 0. We conclude that the tree $A \setminus \{\Delta_1^2, \ldots, \Delta_p^2\}$ computes f with at most $L(f) - 1$ occurrences of essential variables. This is absurd.

□

The previous three lemmas complete the proof of Proposition 37.

Proof of Theorem 7: Of course each tree computing f falls in a set obtained by an arbitrary number of expansions of an irreducible tree computing f. That is, the set of trees computing f is exactly $\mathcal{F}_k(f) = E^*(\mathcal{M}_f \cup \mathcal{P}_{f,1} \cup \mathcal{P}_{f,2} \cup \mathcal{P}_{f,3} \cup \mathcal{P}_{f,4} \cup \mathcal{P}_{f,5})$.

Because $E(\mathcal{M}_f) \subseteq \mathcal{F}_k(f)$, the lower bound on $\mu_k(f)$ is easily obtained from Lemma 32. Let us show the upper bound on $\mu_k(f)$ now.

Lemmas 34, 35, 36 and Proposition 37 together with Proposition 29 show that $\bar{\mu}_k(E^*(\mathcal{P}_{f,1} \cup E(\mathcal{P}_{f,2}) \cup \mathcal{P}_{f,3} \cup \mathcal{P}_{f,4} \cup \mathcal{P}_{f,5})) = O(1/k^{L(f)+2})$. Moreover, since $|\mathcal{P}_{f,2}| = O(1)$, it holds that $\mu_k(\mathcal{P}_{f,2}) = 0$. Hence $\bar{\mu}_k(E^*(\bigcup_{1 \leqslant i \leqslant 5} \mathcal{P}_{f,i})) = O(1/k^{L(f)+2})$.

In the same way, $E^*(\mathcal{M}_f) = \mathcal{M}_f \cup E(\mathcal{M}_f) \cup (E^{\geqslant 2}(\mathcal{M}_f) \setminus E(\mathcal{M}_f))$. Since $|\mathcal{M}_f| = O(1)$, we have $\mu_k(\mathcal{M}_f) = 0$. Moreover, Lemma 33 yields $E^{\geqslant 2}(\mathcal{M}_f) \setminus E(\mathcal{M}_f) \subseteq \mathcal{N}$ and it follows by Proposition 29 that $\bar{\mu}_k(E^{\geqslant 2}(\mathcal{M}_f) \setminus E(\mathcal{M}_f)) = O(1/k^{L(f)+2})$. The upper bound on $\mu_k(f)$ is proved. □

Let us provide some bounds on the integer $\lambda(f)$ now.

Proposition 41 *Let f be a Boolean function different from True, and let ℓ be its number of essential variables. It holds that*

$$2(2L(f) - 1)|\mathcal{M}_f| \leqslant \lambda(f) \leqslant (1 + 2\ell)(2L(f) - 1)|\mathcal{M}_f|.$$

Proof: Let M be a minimal tree computing f and let ν be a node of M. Since M is of size $L(f)$, the tree M has $2L(f)-1$ nodes in total. Thus all we have to show is that the number λ_ν of types of valid expansions in the node ν of M satisfies $2 \leqslant \lambda_\nu \leqslant 1+2\ell$.

We obtain the upper bound simply by counting all possible types of expansions: 1 for the tautology type, ℓ of goal type, and ℓ of premise type – recall from Lemma 31 that no expansion of type "goal α" or "premise α" with respect to an inessential variable α is valid in M.

The lower bound is obtained by remarking that, besides the tautology type of expansion which is of course valid in ν, the expansion of type "premise x" is also valid in ν, where x is the goal of M. Indeed, let A be the tree obtained from M by replacing the subtree B rooted at ν by $C \to B$, where C is any tree with a premise equal to x. Of course $[A_{|x=0}] = [M_{|x=0}]$ because $[C_{|x=0}] = 1$. Moreover, $[A_{|x=1}] = [M_{|x=1}] = 1$ because x is the goal of M. Thus $[A] = [M]$ and we conclude that the expansion of type "goal x" is valid in ν. $\square$

6 Read-once functions

We consider here the case of *read-once* functions, i.e., functions f with $L(f)$ essential variables. An alternative definition is a function whose minimal trees contain no repetition of variables. A tree in which all leaves are labelled by distinct variables is called a read-once tree. Notice that read-once trees are exactly minimal trees computing read-once functions.

Let $\mathcal{R}_{c,k}$ be the set of read-once trees of size c over $\{x_1, \ldots, x_k\}$. Our aim is to estimate both the limiting ratio of all read-once functions of a fixed complexity c over $\{x_1, \ldots, x_k\}$ and the average limiting ratio of a read-once function of complexity c. A symmetry argument allows to focus on $\mathcal{R}(x_1, \ldots, x_c)$, the set of read-once trees of size c on $\{x_1, \ldots, x_c\}$.

We first define the notion of equivalence under recursive permutation of premises, denoted as $\equiv$. Two trees A and B of left depth 0 are equivalent if and only if they are equal. Two trees $A = (A_1, \ldots, A_p, \alpha)$ and $B = (B_1, \ldots, B_q, \beta)$ of positive left depth are equivalent if and only if $p = q$, $\alpha = \beta$ and there exists a permutation π of $\{1, \ldots, p\}$ such that $A_i \equiv B_{\pi(i)}$ for all $i \in \{1, \ldots, p\}$.

Lemma 42 *Let A be a read-once tree computing f. The set of all read-once trees computing f is exactly $\{B \mid B \equiv A\}$; i.e. it is the set of trees obtained from A by recursive permutation of premises.*

Proof: The result is obtained by an induction on the size of the read-once tree. Let A and B be two minimal trees of f. The initial step is obvious because two trees of size 1 that compute the same function f are equal, and so they are equivalent. Now suppose that all read-once trees of size at most r computing the same function are equivalent. Let $A = (A_1, \ldots, A_p, \alpha)$ and $B = (B_1, \ldots, B_q, \beta)$ be two read-once trees of size $r+1$ such that both compute the same function f. Let us first study the goals of A and B. Let a be the assignment such that $\alpha = 0$ and all other variables are valuated to 1. For this assignment, A evaluates to 0. Since both trees compute f, B evaluates to 0 too. Hence the goal of B must be valuated to 0. Since α is the single variable valuated to 0, it holds that $\beta = \alpha$.

Now we focus on the premises of A and B. Let us denote by α_i the goal of A_i and β_i the goal of B_i. Assume that $\{\alpha_1, \ldots, \alpha_p\} \neq \{\beta_1, \ldots, \beta_q\}$. By symmetry, we shall assume there exists $\beta_\ell \in \{\beta_1, \ldots, \beta_q\} \setminus \{\alpha_1, \ldots, \alpha_p\}$. Let a be the partial assignment such that $\alpha = \beta_\ell = 0$ and all other variables are valuated to 1. Under this assignment, A evaluates to 0 while B evaluates to 1. This is a contradiction. It follows that $\{\alpha_1, \ldots, \alpha_p\} = \{\beta_1, \ldots, \beta_q\}$, and of course $p = q$ since A and B are read-once.

Let π be the permutation of $\{1, \ldots, p\}$ such that $\alpha_i = \beta_{\pi(i)}$ for all i. Let $j \in \{1, \ldots, p\}$. Let a be the assignment such that $\alpha = \alpha_j = 0$ and all other variables are valuated to 1. Under this assignment, A computes $\bar{A}_j$ and B computes $\bar{B}_{\pi(j)}$. These two subtrees must compute the same function, are read-once and of size smaller than $r+1$: by induction, we conclude that $A_j \equiv B_{\pi(j)}$. We have proved that $A \equiv B$. $\square$

Lemma 42 allows to count the number of read-once functions of complexity c over the variables $\{x_1, \ldots, x_c\}$. Indeed, equivalence classes of read-once trees over $\{x_1, \ldots, x_c\}$ under recursive permutation of premises are in one-to-one correspondence with labelled Cayley trees with c leaves [9]. Thus there are c^{c-1} read-once functions over $\{x_1, \ldots, x_c\}$.

For an expression A, we define $\lambda^{taut}(A)$ to be the number of valid expansions by tautologies in A. In the same way, we define the number of valid expansions of type goal and premise $\lambda^{goal}(A)$ and $\lambda^{prem}(A)$. Of course $\lambda(A) = \lambda^{taut}(A) + \lambda^{goal}(A) + \lambda^{prem}(A)$. Then we define $\lambda^t(\mathcal{R}(x_1, \ldots, x_c)) = \sum_{A \in \mathcal{R}(x_1,\ldots,x_c)} \lambda^t(A)$, the number of expansions of type t of all read-once trees depending on all the variables $\{x_1, \ldots, x_c\}$. Note that for a read-once tree A, the three quantities $\lambda^t(A)$ do not depend on the labelling of A but only on its shape – see Lemmas 44 and 43.

Just as in any tree, expansions of type tautology are possible in all (internal or external) nodes of a read-once tree. It follows that $\lambda^{taut}(\mathcal{R}(x_1, \ldots, x_c)) = c!(2c-1)C_{c-1}$.

In order to evaluate $\lambda^{prem}(\mathcal{R}(x_1, \ldots, x_c))$ and $\lambda^{goal}(\mathcal{R}(x_1, \ldots, x_c))$, we first need to characterize valid expansions of type "goal" and "premise" in read-once trees. Notice that a read-once tree is a minimal tree, so Lemma 31 ensures that a valid expansion in a read-once tree must be with respect to an essential variable (i.e., a variable that appears in this tree).

Lemma 43 *An expansion of type "premise α" is valid in the node ν of a read-once tree A if and only if ν lies in a left subtree of A with goal α.*

Proof: Suppose there exists a leaf τ in A labelled by α, and let ν be a node of $\Delta(\tau)$. Consider a tree A' obtained by an expansion of type "premise α" in ν; we denote by τ' the node of A' corresponding to the node τ after this expansion. If $\alpha = 1$, since $\Delta(\tau')$ evaluates to 1, it is clear that $[A'_{|\alpha=1}] = [A_{|\alpha=1}] = 1$. If $\alpha = 0$, the left subtree added in the expansion obviously evaluates to 1; hence $[\Delta(\tau')_{|\alpha=0}] = [\Delta(\tau)_{|\alpha=0}]$ and it follows that $[A'_{|\alpha=0}] = [A_{|\alpha=0}]$. We have shown that the expansion of type "premise α" is valid in ν.

Assume now that the expansion of type "premise α" is valid in a node ν of A. Let τ be the leaf of A labelled by α. Suppose by contradiction that $\nu \notin \Delta(\tau)$. Consider the tree A' obtained by replacing $\Delta(\nu)$ with $(\alpha \to \beta) \to \Delta(\nu)$ in A, for some variable β not appearing in A. The function $[A_{|\alpha=1}]$ depends on the goal of $\Delta(\nu)$, while $[A_{|\alpha=1,\beta=0}]$, which should be equal to $[A_{|\alpha=1}]$, does not. This is absurd. $\square$

Lemma 44 *An expansion of type "goal α" is valid in the node ν of a read-once tree A if and only if ν is not labelled by α, and lies in a left subtree of A with a premise α.*

Proof: Assume that there exists a left leaf τ in A, labelled by α and let ν be a node of $\Delta^2(\tau) \setminus \{\tau\}$. It is easily checked that the expansion of type "goal α" is valid in ν, by considering both values $\alpha = 0$ and $\alpha = 1$ in turn.

Now suppose that there exists a node ν in A where an expansion of type "goal α" is valid. The variable α must appear in A by Lemma 31; let τ be the leaf of A labelled by α. Let A' be the tree obtained by replacing ν with $\alpha \to \nu$ in A. Let $\beta = r(\Delta(\nu))$. First notice that $\beta \neq \alpha$: indeed, if we had $\alpha = \beta$, $[A'] = [A]$ would not depend on α.

Assume now that τ is a right leaf. In this case, $[A_{|\alpha=0}]$ still depends on β while $[A'_{|\alpha=0}]$ does not: this is absurd. Hence τ is a left leaf. We have already noticed that $(r(\Delta(\nu)) \neq \alpha$: this ensures that $\nu \neq \tau$. It remains to prove that $\nu \in \Delta^2(\tau)$. Assume now that $\nu \notin \Delta^2(\tau)$. Again, $[A'_{|\alpha=0}]$ does not depend on β any more in this case, while $[A_{|\alpha=0}]$ does. This is absurd. $\square$

Lemma 45 *The number of valid expansions of type premise of all read-once functions depending on $\{x_1, \ldots, x_c\}$ is*

$$\lambda^{prem}(\mathcal{R}(x_1, \ldots, x_c)) = c!\ 4^{c-1}.$$

Proof: Since the number of valid expansions of type "premise" of a read-once tree does not depend on its labelling, we define the number $\lambda^{prem}(T)$ for an unlabelled tree T as $\lambda^{prem}(A)$ where A is any read-once labelling of T. For an integer n, let $p_n = \sum_{|T|=n} \lambda^{prem}(T)$.

For every internal or external node ν, we denote by $\delta_\ell(\nu)$ its left depth, and $\delta(\nu)$ its depth. For a node ν, there are exactly $\delta_\ell(\nu)+1$ (right) leaves τ such that $\nu \in \Delta(\tau)$. Using Lemma 43, we conclude that there exist exactly $\delta_\ell(\nu)+1$ valid expansions of type "premise" valid in ν. Hence $p_n = \sum_{|T|=n}\sum_{\nu\in T}(\delta_\ell(\nu)+1) = (2n-1)C_{n-1} + q_n$, where $q_n = \sum_{|T|=n}\sum_{\nu\in T}\delta_\ell(\nu)$. By a symmetry argument obtained by considering mirror images of trees, $q_n = (1/2)r_n$ with $r_n = \sum_{|T|=n}\sum_{\nu\in T}\delta(\nu)$. This quantity r_n is the sum of the depth of all (internal and external) nodes over all trees of size n. Let us decompose $r_n = r_n^{int} + r_n^{ext}$ according to the contribution of internal and external nodes. It is easily obtained by induction on the size that in a tree of size n, the contribution of external nodes is equal to the contribution of internal nodes plus $2(n-1)$. This gives $r_n^{ext} = r_n^{int} + 2(n-1)C_{n-1}$. Hence, $r_n = 2r_n^{int} + 2(n-1)C_{n-1}$. It is known that $r_n^{int} = 4^{n-1} - (3n-2)C_{n-1}$ – see [7, Theorem 5.3, p. 242]. We have obtained $p_n = 4^{n-1}$. Since a Catalan tree of size c can be labelled with all $\{x_1,\dots,x_c\}$ in $c!$ ways, we get the result. □

Lemma 46 *The number of valid expansions of type goal of all read-once trees depending on* $\{x_1,\dots,x_c\}$ *is*

$$\lambda^{goal}(\mathcal{R}(x_1,\dots,x_c)) = c!\left(4^{c-1} - \binom{2c-2}{c-1}\right).$$

Proof: As already noted, the number of valid expansions of type goal of a read-once tree does not depend on its labelling. For an unlabelled Catalan tree T, we shall denote by $\lambda^{goal}(T)$ the number of expansions of any read-once tree of shape T.

Let $g_n = \sum_{|T|=n} \lambda^{goal}(T)$ where the sum is over all (unlabelled) Catalan trees of size n. We shall first prove that this sequence is defined by the following induction:

$$\begin{cases} g_1 = 0 \\ g_n = C_{n-1}(2n-2) + \sum_{t=2}^{n-1} C_{n-t}g_t \ \text{ for } n>1 \end{cases}$$

Obviously $g_1 = 0$. Let $n>1$. Given a tree T, we denote by $p(T)$ the number of premises of T, and by $T_1,\dots,T_{p(T)}$ its premises. The number of premises of size t of the tree T is denoted by $p^t(T)$. For each premise of size one labelled by α, expansions of type "goal α" are valid in every node except in the premise itself (Lemma 44). We have:

$$\lambda^{goal}(T) = p^1(T)(2n-2) + \sum_{i=1}^{p(T)} \lambda^{goal}(T_i).$$

Summing over all trees of size n, we get:

$$g_n = (2n-2)\sum_{|T|=n} p^1(T) + g_n^{>1}$$

where

$$g_n^{>1} = \sum_{|T|=n}\sum_{i=1}^{p(T)} \lambda^{goal}(T_i).$$

It is a folklore result that the average number of premises of size t in a Catalan tree of size n is equal to $\bar{p}_t(n) = C_{t-1}C_{n-t}/C_{n-1}$. Hence we get $g_n = (2n-2)C_{n-1} + g_n^{>1}$. Let us evaluate $g_n^{>1}$, by regrouping premises by their size (we consider only premises of size greater than one since $\lambda^{goal}(A) = 0$ when $|A| = 1$):

$$g_n^{>1} = \sum_{|T|=n}\sum_{i=1}^{p(T)} \lambda^{goal}(T_i) = \sum_{|T|=n}\sum_{t=2}^{n-1}\sum_{i=1}^{p(T)} 1_{|T_i|=t}\lambda^{goal}(T_i) = \sum_{t=2}^{n-1}\sum_{|T|=n}\sum_{i=1}^{p(T)} 1_{|T_i|=t}\lambda^{goal}(T_i).$$

By a symmetry argument, it holds that:

$$\sum_{|T|=n} \sum_{i=1}^{p(T)} 1_{|T_i=t|} \lambda^{goal}(T_i) = \left(\sum_{|T|=n} p^t(T) \right) \bar{g}_t.$$

where $\bar{g}_t = g_t / C_{t-1}$. Hence we get $g_n^{>1} = \sum_{t=2}^{n-1} \bar{p}_t(n) C_{n-1} \bar{g}_t$. Using the average number of premises of fixed size in a tree recalled above, we get $g_n^{>1} = \sum_{t=2}^{n-1} C_{n-t} g_t$.

Consider now the generating function $g(z) = \sum_n g_n z^n$. Induction equations established above yield the following functional equation:

$$g(z) = 2zC'(z) - 2C(z) + \frac{C(z)g(z)}{z} - g(z),$$

where $C(z)$ enumerates Catalan trees. Of course $C(z) = (1 - \sqrt{1-4z})/2$ and we get:

$$g(z) = \frac{z(1 - \sqrt{1-4z})}{1-4z}.$$

Finally $g_c = [z^c]g(z) = 4^{c-1} - c \cdot C_{c-1}$. Furthermore we can label the leaves of an (unlabelled) Catalan tree of size c in $c!$ different ways with all variables from $\{x_1, \ldots, x_c\}$, so $\lambda^{goal}(\mathcal{R}(x_1, \ldots, x_c)) = c!(4^{c-1} - c \cdot C_{c-1})$. $\square$

Proposition 47 *Let c be a fixed integer. The limiting ratio of all read-once functions of complexity c satisfies, when k tends to infinity:*

$$\mu_k(\mathcal{R}_{c,k}) = \left(\frac{1}{2} + (1 - c^{-1}) \frac{\binom{2c-2}{c-1}}{4^c} \right) \frac{1}{k} + O\left(\frac{1}{k^2} \right).$$

The average limiting ratio of a read-once function of complexity c over k variables is equal to:

$$\frac{\mu_k(\mathcal{R}_{c,k})}{|[\mathcal{R}_{c,k}]|} = \left(\frac{c}{2} + (c-1) \frac{\binom{2c-2}{c-1}}{4^c} \right) \frac{c!}{c^c} \frac{1}{k^{c+1}} + O\left(\frac{1}{k^{c+2}} \right).$$

Proof: Let c be a fixed integer. Let us first compute $\mu_k(\mathcal{R}(x_1, \ldots, x_c))$. We have already noticed that $\lambda^{taut}(\mathcal{R}(x_1, \ldots, x_c)) = c!(2c-1)C_{c-1}$. Thus the results from Lemma 46 and 45 give us the number of valid expansions of all read-once trees on all $\{x_1, \ldots, x_c\}$:

$$\lambda(\mathcal{R}(x_1, \ldots, x_c)) = \lambda^{taut}(\mathcal{R}(x_1, \ldots, x_c)) + \lambda^{goal}(\mathcal{R}(x_1, \ldots, x_c)) + \lambda^{prem}(\mathcal{R}(x_1, \ldots, x_c)).$$

Then Theorem 7 gives:

$$\begin{aligned} \mu_k(\mathcal{R}(x_1, \ldots, x_c)) &= \frac{\lambda(\mathcal{R}(x_1, \ldots, x_c))}{4^c \ k^{c+1}} + O\left(\frac{1}{k^{c+2}} \right) \\ &= \left(\frac{1}{2} + (1 - c^{-1}) \frac{\binom{2c-2}{c-1}}{4^c} \right) c! \frac{1}{k^{c+1}} + O\left(\frac{1}{k^{c+2}} \right). \end{aligned}$$

By symmetry, $\mu_k(\mathcal{R}_{c,k}) = \binom{k}{c} \ \mu_k(\mathcal{R}(x_1, \ldots, x_c))$, which gives the first result of the lemma. Furthermore, since the number of read-once functions of complexity c on $\{x_1, \ldots, x_c\}$ is c^{c-1}, we immediately get the average limiting ratio of read-once functions of complexity c on $\{x_1, \ldots, x_c\}$. By symmetry, it is equal to the average limiting ratio of read-once functions of complexity c on $\{x_1, \ldots, x_k\}$, which gives the second equation stated in the lemma. $\square$

7 Branching processes

We shall consider the probability distribution on Boolean functions induced by a distribution on trees given by a critical Galton Watson process, with nodes uniformly and independently labelled at random among $\{x_1, \ldots, x_k\}$. In this model, the probabilities that a node has 0 or 2 sons are equal to 1/2 and the labelling of all leaves are chosen uniformly at random among $\{x_1, \ldots, x_k\}$ and mutually independent. It is known that a tree is almost surely finite in this model [1].

This probability distribution has been introduced in [4] on *And/Or* trees and can be obviously adapted to the case of implication trees (here the labelling of the internal node is not at random because there is a single label). So for a tree A, we get:

$$\pi_k(A) = \mathbb{P}(\text{structure of } A) \cdot \mathbb{P}(\text{labelling of } A) = \frac{1}{2^{2|A|-1}\ k^{|A|}}.$$

In this model, notice that the probability $\pi_k(\mathcal{A})$ is well defined for any subset of trees $\mathcal{A}$. We define the probability of a given function f as $\pi_k(f) = \pi_k(\{A \in \mathcal{F}_k \mid [A] = f\}) = \sum_{[A]=f} \pi_k(A)$.

Proposition 48 *The probability of tautologies satisfies:*

$$\pi_k(\mathit{True}) = \frac{1}{2k} + O\left(\frac{1}{k^2}\right).$$

Proof: We adapt the computations of [12] to the branching process model. Let us recall: the *simple tautologies* are trees such that one of their premise is equal to their goal. Let us denote $\mathcal{S}_k$ the set of all simple tautologies. A tree T with goal α is not a simple tautology if and only if all its premises are different from α. Summing over all $\alpha \in \{x_1, \ldots, x_k\}$, we obtain:

$$\begin{aligned}\pi_k(\mathcal{F}_k \setminus \mathcal{S}_k) &= \sum_{\alpha}\sum_{p=0}^{\infty} \frac{1}{2^p}\left(1-\frac{1}{2k}\right)^p \frac{1}{2k} \\ &= 1 - \frac{1}{2k+1}.\end{aligned}$$

Hence we conclude:

$$\pi_k(\mathcal{S}_k) = \frac{1}{2k} + O\left(\frac{1}{k^2}\right).$$

A tree is a *simple non tautology* if the goals of all its premises are different from the goal α of the whole tree. Let SN_k be the set of all these trees.

$$\begin{aligned}\pi_k(SN_k) &= \sum_{\alpha}\sum_{p=0}^{\infty} \frac{1}{2^p}\left(1-\frac{1}{k}\right)^p \frac{1}{2k} \\ &= 1 - \frac{1}{k} + O\left(\frac{1}{k^2}\right).\end{aligned}$$

This shows the result for $k = 1$.

Let us assume $k \geqslant 2$ now. We need to define the *less simple non tautologies*. For α and β two distinct variables, let us define the set of trees $LN_k^{\alpha,\beta}$ as the set of trees $T = T_1, \ldots, T_q, \ldots, T_p \to \alpha$ (with all $1 \leqslant q \leqslant p$) satisfying the following properties. The subtree T_q has goal α and contains at least one premise; the first premise $T_{q,1}$ of T_q has goal β, and all premises of $T_{q,1}$ (if any) have a goal different from α and β. At last, for all $i \in \{1, \ldots, p\} \setminus \{q\}$, $r(T_i) \notin \{\alpha, \beta\}$. The set of less simple non tautologies is defined as $LN_k = \bigcup_{\alpha \neq \beta} LN_k^{\alpha,\beta}$.

Given a tree $T \in LN_k^{\alpha,\beta}$, let us denote by $q(T)$ the integer such that $T_{q(T)}$ is the only premise of T with goal α. Let us first compute the probability of all possible $T_{q(T),1}$ of trees from $LN_k^{\alpha,\beta}$:

$$\pi_k(\{T_{q(T),1} \mid T \in LN_k^{\alpha,\beta}\}) = \sum_{p=0}^{\infty} \frac{1}{2^p}\left(1-\frac{2}{k}\right)^p \frac{1}{2k} = \frac{1}{k+2}.$$

Now, we compute the probability of all possible $T_{q(T)}$:

$$\pi_k(\{T_{q(T)} \mid T \in LN_k^{\alpha,\beta}\}) = \frac{1}{2}\pi_k(\{T_{q(T),1} \mid T \in LN_k^{\alpha,\beta}\})\frac{1}{k} = \frac{1}{2k(k+2)}.$$

Finally:

$$\begin{aligned} \pi_k(LN_k) &= \sum_{\alpha\neq\beta}\left(\sum_{p=0}^{\infty}\frac{1}{2^p}\left(1-\frac{2}{k}\right)^p\right)\frac{1}{2}\pi_k(\{T_{q(T)} \mid T \in LN_k^{\alpha,\beta}\})\left(\sum_{p=0}^{\infty}\frac{1}{2^p}\left(1-\frac{2}{k}\right)^p\right)\frac{1}{2k} \\ &= \frac{1}{2k}+O\left(\frac{1}{k^2}\right). \end{aligned}$$

See paper [12] to get a proof that all trees from $SN_k \cup LN_k$ are not tautologies. We have proved that $\pi_k(\mathit{True}) = 1/2k + O(1/k^2)$. □

The fact that iterated expansions of the set of trees with many repetitions of variables at a small left depth has a small probability still holds in the model of branching processes. This is proved in the following lemma – recall the definitions of $\mathcal{A}_q^p(\mathcal{V})$ and $\mathcal{B}_q^p(\mathcal{V})$ from Section 4.

Lemma 49 *For a fixed set of variables $\mathcal{V}$ and two integers p and q, $\pi_k(E^*(\mathcal{A}_q^p(\mathcal{V}))) = O(1/k^p)$.*

Proof: It is sufficient to prove that $\pi_k(X(\mathcal{B}_q^p(\mathcal{V}))) = O(1/k^p)$. Then Lemma 23 gives the result.

Let $B \in \mathcal{B}_q^p(\mathcal{V})$. We recall that the structure of a tree is obtained by a critical branching process and the labelling of each leaf is at uniform random among $\{x_1,\ldots,x_k\}$ and independent. So we get:

$$\pi_k(X(B)) = \pi_k(B)\left(\sum_{i=0}^{\infty}\frac{1}{2^i}\right)^{2|B|-1} \leqslant 2^{pq+1}\pi_k(B)$$

since $|B| \leqslant pq+1$. Moreover, B has p variables from the fixed set $\mathcal{V}$, so $\pi_k(B) = O(1/k^p)$. At last, $|\mathcal{B}_q^p(\mathcal{V})| = O(1)$ gives $\pi_k(X(\mathcal{B}_q^p(\mathcal{V}))) = O(1/k^p)$. □

Now, we can state the result on the asymptotic probability of a function different from *True*.

Proposition 50 *For a Boolean function f different from True,*

$$\pi_k(f) = \frac{|\mathcal{M}_f|}{2^{2L(f)-1}k^{L(f)}} + O\left(\frac{1}{k^{L(f)+1}}\right).$$

Proof: Recall that $\mathcal{F}_k(f)$ denotes the set of trees computing f. Obviously it holds that:

$$\mathcal{M}_f \subseteq \mathcal{F}_k(f) \subseteq \mathcal{M}_f \cup E^*(E(\mathcal{M}_f) \cup \mathcal{P}_{f,1} \cup \ldots \cup \mathcal{P}_{f,5})).$$

Of course $\pi_k(\mathcal{M}_f) = |\mathcal{M}_f|/(2^{2L(f)-1}k^{L(f)})$. This provides the lower bound on $\pi_k(f)$.

We focus on $E^*(E(\mathcal{M}_f)\cup\mathcal{P}_{f,1}\cup\ldots\cup\mathcal{P}_{f,5})$ now. Recall the definition of $\mathcal{N}$ from Section 5. By Lemma 49, it holds that $\pi_k(E^*(\mathcal{N})) = O(1/k^{L(f)+2})$. First recall that $\mathcal{P}_{f,1}$ and $\mathcal{P}_{f,3}$ are empty (Lemmas 34 and 35). Then Lemmas 33, 36 and Proposition 37, together with the bound on $\pi_k(E^*(\mathcal{N}))$, show that

$$\pi_k((E^{\geqslant 2}(\mathcal{M}_f) \setminus E(\mathcal{M}_f)) \cup E^+(\mathcal{P}_{f,2}) \cup E^*(\mathcal{P}_{f,4} \cup \mathcal{P}_{f,5})) = O(1/k^{L(f)+2}).$$

Now trees of $\mathcal{P}_{f_2}$ are of size $L(f)+1$ and without essential variables, so any tree $P \in \mathcal{P}_{f_2}$ has probability $\pi_k(P) = O(1/k^{L(f)+1})$. Since $|\mathcal{P}_{f_2}| = O(1)$, this gives $\pi_k(\mathcal{P}_{f_2}) = O(1/k^{L(f)+1})$. At last, it is easily shown, along the same lines as in the proof of Lemma 32, that $\pi_k(E(\mathcal{M}_f)) = O(1/k^{L(f)+1})$. This ends the proof of the upper bound on $\pi_k(f)$. □

8 Conclusion

When considering the limiting ratio of a Boolean function, e.g., in the system of implication, it may not be enough to know that the limiting ratio exists, and one may naturally wish for some numerical information. For a fixed, (very) small number of Boolean variables, explicit computation of the limiting ratios is feasible by writing, then solving, an algebraic system; see [14] for an overview of the mathematical technology involved and [4] for the application to And/Or trees. However, the fact that size of the system grows exponentially in k severely restricts hand-made evaluation. For a moderate number of variables, very recent results on explicit solving of algebraic systems [28] give us hope to extend the numerical computations a little bit farther. But exact computation will eventually fail, even for a "reasonable" number of Boolean variables. Then we turn to asymptotic analysis; this is where our result comes in.

We should also mention that Theorem 1 requires us to specify the Boolean function, and does not hold uniformly over *all* Boolean functions; hence we are still unable to compute the average complexity of a Boolean function chosen according to this probability distribution. Further work is required before we can either verify or invalidate the Shannon effect for this non-uniform probability distribution.

Acknowledgments. We are grateful to Jakub Kozik for fruitful discussions about this problem.

References

[1] K. B. Athreya and P. E. Ney. *Branching Processes*. Springer, 1972.

[2] R. B. Boppana. Amplification of probabilistic boolean formulas. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 20–29, 1985.

[3] A. Brodsky and N. Pippenger. The boolean functions computed by random boolean formulas or how to grow the right function. *Random Structures and Algorithms*, 27:490–519, 2005.

[4] B. Chauvin, P. Flajolet, D. Gardy, and B. Gittenberger. *And/Or* trees revisited. *Combinatorics, Probability and Computing*, 13(4-5):475–497, July-September 2004.

[5] William F. Dowling and Jean H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *J. Logic Programming*, 1(3):267–284, 1984.

[6] M. Drmota. Systems of functional equations. *Random Structures and Algorithms*, 10(1-2):103–124, 1997.

[7] P. Flajolet and R. Sedgewick. *An introduction to the Analysis of Algorithms*. Addison-Wesley, 1996.

[8] P. Flajolet and R. Sedgewick. Analytic combinatorics: Functional equations, rational and algebraic functions. Technical Report 4103, INRIA, January 2001.

[9] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[10] H. Fournier, D. Gardy, and A. Genitrini. Balanced And/Or trees and linear threshold functions. In *Proceedings of the 5th SIAM Workshop on Analytic and Combinatorics*, pages 51–57, 2009.

[11] H. Fournier, D. Gardy, A. Genitrini, and B. Gittenberger. Complexity and limiting ratio of boolean functions over implication. In *MFCS*, pages 347–362, 2008.

[12] H. Fournier, D. Gardy, A. Genitrini, and M. Zaionc. Classical and intuitionistic logic are asymptotically identical. In *Proceedings of CSL*, pages 177–193, 2007.

[13] H. Fournier, D. Gardy, A. Genitrini, and M .Zaionc. Tautologies over implication with negative literals. *Mathematical Logic Quarterly*, 56(4):338–396, 2010.

[14] D. Gardy. Random boolean expressions. In *Colloquium on Computational Logic and Applications*, volume AF, pages 1–36. DMTCS Proceedings, 2006.

[15] A. Genitrini and J. Kozik. Quantitative comparison of intuitionistic and classical logics - full propositional system. In *LFCS*, pages 280–294, 2009.

[16] A. Gupta and S. Mahajan. Using amplification to compute majority with small majority gates. *Computational Complexity*, 6(1):46–63, 1997.

[17] Alfred Horn. On sentences which are true of direct unions of algebras. *J. Symbolic Logic*, 16:14–21, 1951.

[18] Z. Kostrzycka. On the density of truth of implicational parts of intuitionistic and classical logics. *Journal of Applied Non-Classical Logics*, 13(2), 2003.

[19] Z. Kostrzycka. On the density of truth in modal logics. In *Mathematics and Computer Science*, volume AG, pages 161–170, Nancy (France), September 2006. DMTCS Proceedings.

[20] Z. Kostrzycka. On asymptotic divergency in equivalential logics. *Mathematical Structures in Computer Science*, 18:311–324, 2008.

[21] Z. Kostrzycka and M. Zaionc. Statistics of intuitionistic versus classical logics. *Studia Logica*, 76(3):307–328, 2004.

[22] J. Kozik. Subcritical pattern languages for And/Or trees. In *Fifth Colloquium on Mathematics and Computer Science*, Blaubeuren, Germany, september 2008. DMTCS Proceedings.

[23] S. P. Lalley. Finite range random walk on free groups and homogeneous trees. *The Annals of Probability*, 21, 1993.

[24] H. Lefmann and P. Savický. Some typical properties of large And/Or Boolean formulas. *Random Structures and Algorithms*, 10:337–351, 1997.

[25] G. Matecki. Asymptotic density for equivalence. *Electronic Notes in Theoretical Computer Science*, 140:81–91, 2005.

[26] M. Moczurad, J. Tyszkiewicz, and M. Zaionc. Statistical properties of simple types. *Mathematical Structures in Computer Science*, 10(5):575–594, 2000.

[27] J. B. Paris, A. Vencovská, and G. M. Wilmers. A natural prior probability distribution derived from the propositional calculus. *Annals of Pure and Applied Logic*, 70:243–285, 1994.

[28] C. Pivoteau, B. Salvy, and M. Soria. Combinatorial Newton iteration to compute Boltzmann oracle. In *Fifth Colloquium on Mathematics and Computer Science*, Blaubeuren, Germany, september 2008. DMTCS Proceedings.

[29] S. Reith and K. W. Wagner. The complexity of problems defined by boolean circuits. In *The Mathematical Foundation of Informatics*, pages 141–156, 2000.

[30] P. Savický. Random Boolean formulas representing any Boolean function with asymptotically equal probability. *Discrete Mathematics*, 83:95–103, 1990.

[31] Uwe Schöning. *Logic for computer scientists*. Modern Birkhäuser Classics. Birkhäuser Boston Inc., Boston, MA, english edition, 2008. Translated from the 1987 German original.

[32] L. Valiant. Short monotone formulae for the majority function. *Journal of Algorithms*, 5:363–366, 1984.

[33] A. Woods. On the probability of absolute truth for And/Or formulas. *Bulletin of Symbolic Logic*, 12(3), 2005.

[34] A. R. Woods. Coloring rules for finite trees, and probabilities of monadic second order sentences. *Random Structures and Algorithms*, 10(4):453–485, 1997.

[35] M. Zaionc. Statistics of implicational logic. *Electronic Notes in Theoretical Computer Science*, 84:205–216, 2003.

[36] M. Zaionc. On the asymptotic density of tautologies in logic of implication and negation. *Reports on Mathematical Logic*, 39:67–87, 2005.