

Asymptotics and random sampling for BCI and BCK lambda terms.

O.Bodini ^{*}, D. Gardy [†], A. Jacquot^{*}

Abstract

This paper presents a bijection between combinatorial maps and a class of enriched trees, corresponding to a class of expression trees in some logical systems (constrained lambda terms). Starting from two alternative definitions of combinatorial maps: the classical definition by gluing half-edges, and a definition by non-ambiguous depth-first traversal, we derive non-trivial asymptotic expansions and efficient random generation of logic formulæ (syntactic trees) in the BCI or BCK systems.

Keywords: enumeration of lambda terms, random generation, BCI and BCK logical systems

Introduction and motivations

Quantitative logic deals with the evaluation of parameters characterizing different logical systems, with the goal of comparing their respective expressive powers from a statistical point of view. E.g., some former work [14, 15, 18] was concerned with the proportion of tautologies among all possible formulæ. In this paper we focus our attention on the expressivity of two intuitionist logical systems, named BCI and BCK from their axioms. More precisely, we wish to compute and compare the numbers of syntactic trees in both logical systems.

By the classical Curry-Howard isomorphism, these systems can be viewed as families of λ -terms: it is possible to associate a syntactic tree with a formula. Such syntactic trees are isomorphic to unlabelled planar unary–binary trees, enriched with links from unary nodes to leaves which correspond to the naming of variables. The families of λ -terms we consider can be defined from constraints on these links; the representation by enriched trees leads naturally to differential specifications for BCI and BCK classes.

^{*}LIPN, Institut Galilée Université Paris Nord et CNRS UMR 7030, 99 avenue Jean-Baptiste Clément, 93430 Villetaneuse, France; supported by Grant ANR MAGNUM (2010 BLAN 0204).

[†]PRISM, CNRS UMR 8144 et Université de Versailles Saint-Quentin en Yvelines, 45 avenue des Etats-Unis, 78035 Versailles Cedex, France; supported by Grant ANR Boole (ANR 09 BLAN 0011) and PHC AMADEUS Project 2391UD “Random Logical trees and related structures”.

Such families of enriched trees can also be viewed as rooted combinatorial maps, i.e., graphs embedded in an oriented surface without boundary, together with pointing on a half-edge. One of the main results of this paper is to present a one-to-one correspondence between some classes of maps and of enriched trees, and thus to obtain differential equations on the generating functions of these classes of maps. Roughly speaking, the embedding induces a cyclic order on the edges around a vertex, which classically enables a deterministic depth-first traversal. In this way we can define maps as “enriched” trees. In parallel, another description of maps is based on the classical representation of graphs by half-degree sequences. More precisely, a classical way to construct graphs is to take a list of half-degrees and to match the half-edges between them. This method has many applications, in particular in Network Analysis. For instance, the *pairing model* developed by Bollobas [5] in the 1960s is an effective way to generate interesting random graphs : we draw vertices with a certain distribution of degrees and match the half-edges between them. Similarly, the classical definition of combinatorial maps by a couple of permutations, intensively studied by Edmonds [10] in order to deal with properties of planar graphs, can be viewed as a matching of half-edges.

These two definitions of maps, when mixed with classical symbolic combinatorial tools, give the asymptotics of the numbers of BCI and BCK syntactic trees. In fact, the enumerative sequence of BCI formulæ corresponds to the Wright-Louchard-Takács sequence which appeared for example in [12, 21]. In passing, we obtain a closed-form solution for a non-trivial class of differential equations. In one subcase, this has already been done in [21]. In the same vein, we can sample BCI or BCK terms and enriched trees in time linear in the size of the structure.

The paper is organized as follows: we present in Section 1 a classical definition of labelled maps of any gender by permutations, corresponding to gluing of half-edges, then deduce from it a symbolic combinatorics specification of maps using an exponential version of the Hadamard product, introduced in [16]. This description will be the basis for asymptotic analysis, and for the definition of random generators. In Section 2 we give an alternative definition of maps based on a deterministic depth-first traversal, and we obtain a class of differential equations specifying maps. We introduce the BCI and BCK logical systems in Section 3, and derive asymptotic results (in this same section) and random samplers for them (in Section 4). A conclusion sums up our main results and presents some prospective work.

A preliminary version of this paper was presented in GASCOM 2010.

1 The classical definition of a map by gluing half-edges

1.1 Maps and permutations

Combinatorial maps have been first introduced in the 1960s to study oriented surfaces, in particular planar graphs [10, 6]. They can be seen as connected graphs (where loops and multi-edges are allowed) embedded in a oriented surface up to homotopy (continuous deformation of the surface). Due to the complexity of their symmetry groups, those objects are quite difficult to study, for instance their asymptotic analysis has been the subject of many papers [3, 1]. Both for this reason and for the applications we have in mind, we find it more convenient to deal with a (half-edge-)labelled version of combinatorial map: labelled maps are represented by a set of vertices and a set of edges with information on the order of incidence of half-edges around vertices (Fig. 1). This definition is usually formalized as follows, as introduced in [7]:

Definition 1.1. *A labelled combinatorial map (called map in this paper) is a triplet (E, α, β) such that:*

- E is a finite set of half-edges,
- α is a permutation on E ,
- β is an involution on E with no fixed point.
- α, β generate a transitive subgroup of the permutation group.

In this definition, a cycle in α correspond to a vertex represented by its incident half-edges seen counter clockwise and a transposition in β correspond to an edge, composed of two half-edges.

In this paper, we essentially focus on constrained labelled maps: a map (E, α, β) having its vertex degrees in a fixed set $D \subseteq \mathbb{N}^*$. Note that this condition is equivalent to *all cycles of α have their length in D* .

Let us anticipate somewhat on the second section: in order to remove ambiguity at the start of the depth-first traversal, we need to distinguish a half-edge. The notion of rooted maps then becomes natural: A *rooted map* is a map with a distinguished half-edge.

1.2 Symbolic combinatorial equation

Our aim here is to formulate the classical definition of maps in term of constructors from symbolic combinatorics, a framework useful for asymptotics and for random sampling. We refer the reader to the book “Analytic Combinatorics” [13] for a comprehensive survey. Similar work has been done on the sub-class of triangular rooted diagrams [20, 21].

We recall briefly here the basic notions we need: A *combinatorial class* is a pair $(\mathcal{A}, |\cdot|)$ (denoted \mathcal{A} when not ambiguous) where \mathcal{A} is a set of elements

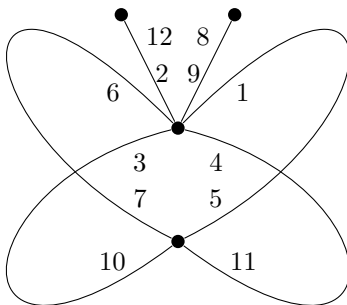


Figure 1: A representation of the map $([1..12], \alpha, \beta)$ with degrees in $[1, 5]$ where $\alpha = ((1, 9, 2, 6, 3, 4)(5, 7, 10, 11)(8)(12))$ and $\beta = ((1, 5)(2, 12)(3, 10)(4, 11)(6, 7)(8, 9))$.

and $|\cdot|$ the size function $\mathcal{A} \rightarrow \mathbb{N}$, with a finite number of objects of each size. Let \mathcal{A} be a combinatorial class. The ordinary generating function (OGF) is $A(x) = \sum_{n \geq 0} a_n x^n$, where a_n is the number of unlabelled objects of \mathcal{A} of size n .

We note $\hat{A}(x)$ the exponential generating function (EGF) of its labelled class: $\hat{A}(x) = \sum_{n \geq 0} \frac{\hat{a}_n}{n!} x^n$, where \hat{a}_n is the number of labelled objects of the labelled class of \mathcal{A} of size n .

A first question is to describe rooted labelled maps in terms of the classical constructors (\mathcal{Z} , \times , $+$, SEQ , SET , CYC , \bullet, \dots). Let us denote the class of maps by \mathcal{C} , where *the size of a map is the number of its half-edges*. With this size function, rooting a map corresponds to the pointing operation [13]. Indeed, each labelled map with $2n$ half-edges gives $2n$ rooted maps. Classically, \mathcal{A}^\bullet denotes the class of pointed objects from \mathcal{A} , and we thus seek a combinatorial specification for the class \mathcal{C}^\bullet of pointed (rooted) maps.

Definition 1.1 leads us to consider two permutations of the same labels, one for the permutation defining the half-edges around vertices and one for the permutation defining edges from half-edges. In order to define the construction of a map from two such permutations, we use the exponential Hadamard product, introduced in [16].

Definition 1.2. *Let \mathcal{A} and \mathcal{B} be two labelled combinatorial classes. An exponential Hadamard product of \mathcal{A} and \mathcal{B} , denoted by $\mathcal{A} \odot \mathcal{B}$, is the set of pairs $[\alpha, \beta]$ where $\alpha \in \mathcal{A}$, $\beta \in \mathcal{B}$ and $|\alpha| = |\beta|$ and which keeps the labelling of α and*

β . By definition $|\alpha, \beta| = |\alpha|$.

Let $\hat{A}(x)$ and $\hat{B}(x)$ be the exponential generating functions of the combinatorial classes \mathcal{A} and \mathcal{B} : $\hat{A}(x) = \sum_n \frac{\hat{a}_n}{n!} x^n$ and $\hat{B}(x) = \sum_n \frac{\hat{b}_n}{n!} x^n$. The exponential generating function of the exponential Hadamard product is $\hat{A} \odot \hat{B}(x) = \sum_n \frac{\hat{a}_n \hat{b}_n}{n!} x^n$.

We recall that $\text{CYC}_k(\mathcal{A})$ is a k -tuple of objects of \mathcal{A} up to circular permutation. For a set of positive integers K , we denote $\text{CYC}_K(\mathcal{A})$ the class of cycles of \mathcal{A} , with length in K : $\text{CYC}_K(\mathcal{A}) = \sum_{k \in K} \text{CYC}_k(\mathcal{A})$. So the classical operator CYC without any restriction on length can be seen as $\text{CYC} = \sum_{n \in \mathbb{N}^*} \text{CYC}_n = \text{CYC}_{\mathbb{N}^*}$.

Proposition 1.3. *Define \mathcal{C}_K as the class of labelled maps with vertex degrees in K . The class of sets of elements in \mathcal{C}_K is obtained as*

$$\text{SET}(\mathcal{C}_K) = \text{SET}(\text{CYC}_K(\mathcal{Z})) \odot \text{SET}(\text{CYC}_2(\mathcal{Z})). \quad (1)$$

Proof. A permutation can be viewed as a labelled set of cycles. A permutation with cycle lengths in K can be written $\text{SET}(\text{CYC}_K(\mathcal{Z}))$. Here an exponential Hadamard product enables us to put two structures (vertices and edges) on the set of half-edges. This leads to the result. \square

In the case $K = \{3\}$ the equation (1) already appears in [21].

Corollary 1.4. *The class \mathcal{C}_K^\bullet of rooted labelled maps with vertex degrees in K satisfies the following equation:*

$$(\text{SEQ}_K(\mathcal{Z}) \times \text{SET}(\text{CYC}_K(\mathcal{Z}))) \odot \text{SET}(\text{CYC}_2(\mathcal{Z})) = \mathcal{C}_K^\bullet \times \text{SET}(\mathcal{C}_K), \quad (2)$$

where

$$\text{SET}(\mathcal{C}_K) = \text{SET}(\text{CYC}_K(\mathcal{Z})) \odot \text{SET}(\text{CYC}_2(\mathcal{Z})).$$

Proof. We apply pointing to both terms of the equation (1):

$$\text{SET}(\text{CYC}_K(\mathcal{Z})) \odot \text{SET}(\text{CYC}_2(\mathcal{Z})) = \text{SET}(\mathcal{C}_K).$$

The reader can easily check that, for two combinatorial classes \mathcal{A} and \mathcal{B} , $(\mathcal{A} \odot \mathcal{B})^\bullet \simeq \mathcal{A}^\bullet \odot \mathcal{B} \simeq \mathcal{A} \odot \mathcal{B}^\bullet$. Hence the left-hand term becomes

$$(\text{SET}(\text{CYC}_K(\mathcal{Z})))^\bullet \odot \text{SET}(\text{CYC}_2(\mathcal{Z})).$$

Now, from the relations

1. $\mathcal{Z}^\bullet = \mathcal{Z}$
2. $(\text{SET}(A))^\bullet = A^\bullet \times \text{SET}(A)$
3. $(\text{CYC}_K(A))^\bullet = A^\bullet \times \text{SEQ}_{K-1}(A)$ with $K-1 = \{k-1 | k \in K\}$

we obtain

$$(\mathcal{Z} \times \text{SEQ}_{K-1}(\mathcal{Z}) \times \text{SET}(\text{CYC}_K(\mathcal{Z}))) \odot \text{SET}(\text{CYC}_2(\mathcal{Z}))$$

for the left-hand term, and

$$\mathcal{C}_K^\bullet \times \text{SET}(\mathcal{C}_K)$$

for the right-hand term; hence the result. \square

Remark 1.5. We can easily consider a map with vertex degrees in K , but whose root degree satisfies other constraints, since CYC_K and SEQ_K are independent. We shall use this fact in Section 3 to describe maps whose root has degree 2, and all other vertices have degree 3.

2 Non-ambiguous depth-first traversal and enriched trees

Depth-first traversal (DFT) is a classical algorithm for exploring graphs in computer science and graph theory; see for example [17], [8]. The algorithm can be summed up in a few words as *Visit the root, then recursively visit each child in turn*. It yields a spanning tree of the graph; the exact tree depends of the order in which we visit the successors of a node. Here we want a non-ambiguous version of a depth-first traversal of the graph (a graph then gives a single spanning tree); to this effect we use the cyclic order of half-edges around the vertices.

DFT also enables us to give a new symbolic combinatorial description of maps, using a differential operation to link a leaf to another vertex of the spanning tree.

2.1 Right-depth-first spanning tree

We are interested here in depth-first spanning trees of maps. A *depth-first spanning tree* of a map is the tree formed by a maximal depth-first traversal walk from the root (by convention, the first half-edge followed after the root r is $\beta(r)$).

A *right-depth-first spanning tree* is the depth-first spanning tree where, each time we get a choice for the following half-edge, we choose the next one in the permutation order, i.e., for a map (E, α, β) the half-edge visited after $e \in E$ is either $\alpha(e)$ or $\beta(e)$, depending of the parity of the size of the current tree.

Note that, once the first half-edge is fixed, this traversal is totally deterministic. An important consequence is the rigidity (defined below) of rooted maps.

Now we can partition the edges of a map in two parts: the edges belonging to the right-depth-first tree T , called *tree edges*, and the edges not in T , called *back edges* (see Fig. 2).

2.2 Rigidity and unlabelled rooted maps

The construction presented in Section 1.1 defines labelled rooted maps. However, in the sequel we shall work with unlabelled rooted maps: such maps are much simpler to study than unrooted ones, due to *rigidity*.

Definition 2.1. *A combinatorial class \mathcal{A} is rigid if each unlabelled object of size n corresponds to $n!$ labelled objects, ie, $\hat{C}(x) = C(x)$.*

This means that, starting from an unlabelled object of size n , we can obtain the $n!$ corresponding labelled objects just by labeling it in the $n!$ possible ways (there is no symmetry).

Theorem 2.2. *The class \mathcal{C}^\bullet is rigid.*

Proof. The proof follows directly from the existence of a non-ambiguous depth first traversal. \square

Corollary 2.3. *For all $K \subseteq \mathbb{N}^*$ the class \mathcal{C}_K^\bullet is rigid.*

Proof. \mathcal{C}_K^\bullet is a sub-class of \mathcal{C}^\bullet . \square

2.3 L-morphing

We now modify our maps as follows, in order to obtain a new symbolic specification. Starting from a map M , we associate to it a map \tilde{M} where we put a new vertex on each back edge. In the new map, of the two edges thus created, only the first one belongs to the right-depth-first tree (see Fig. 2). In this way, each back edge in \tilde{M} is preceded by a vertex of degree 2.

Let \mathcal{C}^\bullet be the class of rooted unlabelled maps and \mathcal{D}^\bullet the class of rooted unlabelled maps where the origin vertices of back edges are of degree 2. Let $\mathcal{C}_{n,r}^\bullet$ (resp. $\mathcal{D}_{m,r}^\bullet$) be the set of maps in \mathcal{C}^\bullet (resp. \mathcal{D}^\bullet) with n half-edges and r back edges (resp. m half-edges and r back edges).

Definition 2.4. *Let $M = (E, \alpha, \beta) \in \mathcal{C}^\bullet$, with $|E| = n$. Let T and R be respectively the sets of its tree edges and its back edges, with $r = |R|$. We define a new map $\tilde{M} = (\tilde{E}, \tilde{\alpha}, \tilde{\beta})$ as follows:*

- For each edge $\{e, e'\} \in R$ with $e' = \beta(e)$, we define two half-edges f and f' , such that $\tilde{\alpha}(f) = f'$, $\tilde{\alpha}(f') = f$, $\tilde{\beta}(e) = f$, $\tilde{\beta}(f) = e$, $\tilde{\beta}(e') = f'$, $\tilde{\beta}(f') = e'$. These new half-edges form a set F ; we set $\tilde{E} = E \cup F$.
- $\forall e \in E, \tilde{\alpha}(e) = \alpha(e)$.
- $\forall (e, e') \in T, \tilde{\beta}(e) = \beta(e) = e'$.

The transformation from M to \tilde{M} is called a *L-morphing*.

Lemma 2.5. *For all n and r , the L-morphing is a bijection from $\mathcal{C}_{n,r}^\bullet$ to $\mathcal{D}_{n+2r,r}^\bullet$.*

Proof. It is easy to see that $\tilde{M} \in \mathcal{D}_{n+2r,r}^\bullet$. The L-morphism is invertible. Here is the description of the inverse bijection : the half-edges of F are the last half-edge before a external edge and the first half-edge of an external edge in the right-depth-first tree of \tilde{M} . By deleting all pairs of such half edges and linking the other part of the concerned edge (in β), we get back M . \square

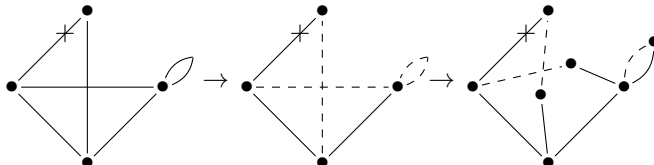


Figure 2: From right to left: a unlabelled rooted map, its partition in 3-edges (plain) and back edges (dashed) and its transformation by the L-morphism. The cross mark the root.

2.4 A differential equation

The non-ambiguous partition of map edges into tree edges and back edges, described in Section 2.1, gives us a new way to specify rooted maps. We do not give the specification directly on the class of rooted maps, but on an equivalent class \mathcal{M}^\bullet , which we now define.

Definition 2.6. *The class \mathcal{M}^\bullet is the image of unlabelled rooted maps by the one-to-one L-morphing.*

Although the elements of \mathcal{M}^\bullet are actually directed graphs, they are akin to trees: they can be viewed as trees to which we add some edges from internal nodes to leaves, and we shall use the term *enriched tree*, or simply *tree*, when speaking about them.

Roughly speaking, we interpret the back edges of an element of \mathcal{M}^\bullet as links from each leaf adjacent to a back edge, to one of its ancestors. We thus need to distinguish leaves to get a combinatorial specification. This can be done by marking leaves with a neutral sticker \mathcal{U} . Then each link corresponds to a differential operation on \mathcal{U} and represents a back edge.

Definition 2.7. *Let \mathcal{A} be a combinatorial class with objects marked by \mathcal{U} . Let \mathcal{A}_u be the set of objects of \mathcal{A} with at least one mark.*

The class $\frac{\partial \mathcal{A}}{\partial \mathcal{U}}$ is the class obtained from objects of \mathcal{A}_u , by pointing at one of the atoms marked by \mathcal{U} , then removing the mark on the chosen atom.

Let \mathcal{A} be a combinatorial class with atoms marked by \mathcal{U} . Let $a_{n,l}$ be the number of objects of \mathcal{A} of size n with l marks, and $A(x,u)$ the bivariate generating function of \mathcal{A} : $A(x,u) = \sum_{a,l \in \mathbb{N}} a_{n,l} x^n u^l$. Then the bivariate generating

function of $\frac{\partial \mathcal{A}}{\partial \mathcal{U}}$ is $\frac{\partial A(x,u)}{\partial u}$.

Our interest actually lies, not in the whole set \mathcal{M}^\bullet , but in a specific subset whose vertices have degree 2 or 3.

Theorem 2.8. *Let \mathcal{D} be the combinatorial class of rooted unlabelled maps, where the origin vertices (in DFT) of back edges are of degree 2 and marked by \mathcal{U} , with the number of vertices as size function:*

$$\mathcal{D} = \mathcal{Z}\mathcal{U} + \sum_{\omega \in a.(a+b)^*} \mathcal{Z} \times \mathcal{D}^\omega,$$

$$\text{where } \mathcal{D}^\omega = \begin{cases} \varepsilon & \text{if } \omega = \varepsilon; \\ \mathcal{D} \times \mathcal{D}^{\omega'} & \text{if } \omega = \omega'a; \\ \frac{\partial \mathcal{D}^{\omega'}}{\partial \mathcal{U}} & \text{if } \omega = \omega'b. \end{cases}$$

Proof. Let M be a map of \mathcal{D} , with T its non-ambiguous depth-first traversal spanning tree. The first outgoing edge of an internal node a of T is always in T . Indeed, when we visit for the first time a node a in the DFT, none of the sub-trees growing from a has yet been visited, so one of the edges outgoing from a is in T . As we choose first the next edge in the cyclic order, this edge will be in T . The other edges can be of any type. An edge of the tree leads to a sub-tree in \mathcal{D} . A back edge leads to a marked leaf of a previous sub-tree, which we visited previously in the DFT, and corresponds to a differential operation on a mark. As a node can be linked to any number of leaves by as many back edges, we can have several differential operations. □

The generating function can be directly deduced from Theorem 2.8.

It can be seen that, for the class \mathcal{D}_K of maps where each internal node of the right depth-first tree has $k \in K$ outgoing edges, the equation becomes:

$$\mathcal{D}_K = \mathcal{Z}\mathcal{U} + \sum_{\substack{\omega \in a.(a+b)^* \\ |w| \in K}} \mathcal{Z} \times \mathcal{D}_K^\omega.$$

Example 2.9. *The table of Figure 3 shows the possible types of nodes for the class \mathcal{D}_4 , where each internal node has 4 outgoing edges. It yields the following differential equation on \mathcal{U} :*

$$\begin{aligned} \mathcal{D}_4 &= \mathcal{Z} \left(\mathcal{U} + \mathcal{D}_4^4 + \frac{\partial \mathcal{D}_4^3}{\partial \mathcal{U}} + \frac{\partial \mathcal{D}_4^2}{\partial \mathcal{U}} \mathcal{D}_4 + \frac{\partial \mathcal{D}_4}{\partial \mathcal{U}} \mathcal{D}_4^2 + \frac{\partial^2 \mathcal{D}_4^2}{\partial \mathcal{U}^2} + \frac{\partial \frac{\partial \mathcal{D}_4}{\partial \mathcal{U}} \mathcal{D}_4}{\partial \mathcal{U}} + \frac{\partial^2 \mathcal{D}_4}{\partial \mathcal{U}^2} \mathcal{D}_4 + \frac{\partial^3 \mathcal{D}_4}{\partial \mathcal{U}^3} \right) \\ &= \mathcal{Z} \left(\mathcal{U} + \mathcal{D}_4^4 + 6\mathcal{D}_4^2 \frac{\partial \mathcal{D}_4}{\partial \mathcal{U}} + 4\mathcal{D}_4 \frac{\partial^2 \mathcal{D}_4}{\partial \mathcal{U}^2} + 3 \left(\frac{\partial \mathcal{D}_4}{\partial \mathcal{U}} \right)^2 + \frac{\partial^3 \mathcal{D}_4}{\partial \mathcal{U}^3} \right) \end{aligned}$$

3 BCI and BCK logical systems

We now are ready to define formally the classes of λ -terms we shall consider.










Node	Term	Definition
	zU	Unlinked leaf.
	\mathcal{D}_4^4	Node with 4 children in T .
	$z \frac{\partial \mathcal{D}_4^3}{\partial U}$	Ternary node in T with a back edge in 4th position, pointing towards a leaf of an arbitrary child.
	$z \frac{\partial \mathcal{D}_4^2}{\partial U} \mathcal{D}_4$	Ternary node in T with a back edge in 3rd position, pointing towards a leaf of one of its first two children.
	$z \frac{\partial \mathcal{D}_4}{\partial U} \mathcal{D}_4^2$	Ternary node in T with a back edge in 2nd position, pointing towards a leaf of its first child.
	$z \frac{\partial^2 \mathcal{D}_4^2}{\partial U^2}$	Binary node in T with 2 back edges in 3rd and 4th positions pointing towards 2 leaves in arbitrary children.
	$z \frac{\partial(\frac{\partial \mathcal{D}_4}{\partial U} \mathcal{D}_4)}{\partial U}$	Binary node in T with 2 back edges in 2nd and 4th positions, pointing towards 2 leaves, resp. in the 1st child and an arbitrary one.
	$z \frac{\partial^2 \mathcal{D}_4}{\partial U^2} \mathcal{D}_4$	Binary node in T with 2 back edges in 2nd and 3rd positions, both pointing towards a leaf of its first child
	$z \frac{\partial^3 \mathcal{D}_4}{\partial U^3}$	Unary node with 3 back edges all three of them pointing to a leaf of its only child.

Figure 3: Possible types of nodes for an element of \mathcal{D}_4 .

Definition 3.1. *The class of λ -terms is recursively defined as follows. Let A and B be λ -terms, x a name of variable.*

- the term x represents a free variable of length 1;
- the term $A B$ is an application of B to A , of length $l(A) + l(B)$;
- the term $\lambda x.A$ is the lambda abstraction of the variable x , of length $l(A)+1$, which bounds every free occurrence of x in A .

The syntactic tree of a λ -term is a rooted planar tree to which we add some edges: a lambda abstraction is a unary node; an application is a binary node; a variable is a leaf; and the declaration of the variable name is a distinguished edge between the leaf and a unary node.

The *size function* we consider on λ -terms is their length, which is equal to the total number of nodes of its syntactic tree.

We next consider the BCK and BCI systems for intuitionist logic, which were introduced in [19]. We refer the reader to this paper for a complete definition of the logical axioms B, C, and K (I is the identity) and give below their characterization in terms of the set of lambda-terms they define.

Definition 3.2. *BCK (resp. BCI) is the class of λ -terms without free variables, and where each lambda abstraction links at most (resp. exactly) one variable. The size on these classes is defined as the number of nodes in the corresponding syntactic trees.*

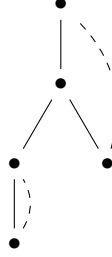


Figure 4: The syntactic tree of the λ -term $\lambda x.((\lambda y.y)x)$

This propriety allows us to forget variables names, since the variables that occur in the leaves are linked with their abstraction (see Fig. 4).

BCI tree always have $3n + 2$ nodes : there are n internal nodes, $n + 1$ leaves and an equal number of unary nodes.

The main theorem of this section is the following:

Theorem 3.3. *The number of syntactic trees of size p which can be build in the BCI system is*

$$[x^p]T(x) = \begin{cases} 0 & \text{if } p \not\equiv 2 \pmod{3}; \\ \frac{6n(6n/e)^n}{\sqrt{2\pi n}}(1 + o(1)) & \text{if } p = 3n + 2 \end{cases}$$

So, $T(x)$ is the OGF for BCI terms.

The end of this section is devoted to the proof of this theorem.

3.1 Isomorphism between BCI/BCK and maps

Theorem 3.4. *The class BCI is in bijection with the class of unlabelled rooted maps with root vertex degree of 2 and all other vertex degrees equal to 3. A BCI term of size $3k + 2$ corresponds to a map of size $6k + 2$.*

Proof. We have this equation on the class of BCI: $T = \mathcal{Z}u + \mathcal{Z}T^2 + \mathcal{Z}\frac{\partial T}{\partial u}$. Indeed, a syntactic tree T of a term in BCI has four different types of nodes:

- A root of degree 2: it has two children if it is an application, or a child and a link if it is a lambda abstraction;
- leaves of degree 2 (the parent and the link to the abstraction);
- application nodes, of degree 3 (the parent and the two children);
- lambda abstraction of degree 3 (the parent, the child and the link to a leaf containing the variable).

By L-morphing, BCI terms are isomorphic to maps with degree 3 and root degree 2. \square

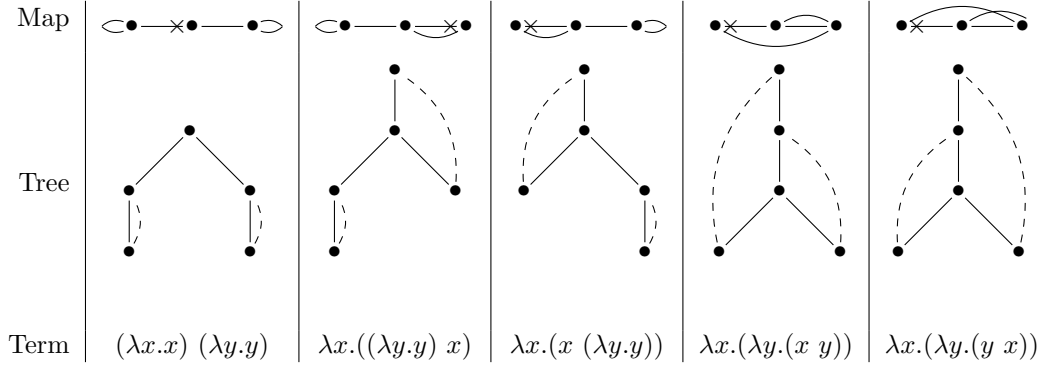


Figure 5: Maps of size 8 and their corresponding BCI terms of size 5

In the same vein, the class BCK is isomorphic to the unlabelled pointed product of a sequence of atoms, and a map with vertex degrees in $\{2, 3\}$. Indeed, lambda abstraction nodes are now of degree 2 (if not linked to a leaf) or 3 (if linked). We thus obtain the BCK terms where the root is either an application or a lambda abstraction of a variable that does appear in the term. We need to add the terms whose root is a lambda abstraction of a variable not present in the term. We obtain such terms either by adding a sequence of lambda abstractions, without corresponding variables, before the root of a λ -term of the first type, or by considering that the root has specific allowed degrees $K_r = \{1, 2\}$. Thus:

Theorem 3.5. *The class BCK is in bijection with the class of unlabelled rooted maps with root degree of 1 or 2 and all other vertex degrees equal to 2 or 3. In this correspondence, a BCK term of size $k + 2$ is associated with a map of size $2k$.*

3.2 Asymptotic number of terms in BCI and BCK

3.2.1 Asymptotic number of syntactic trees in BCI

To determine the asymptotic behaviour of the number of syntactic trees in BCI, we start from the combinatorial equation (2), and obtain readily the equation for their generating function.

Theorem 3.6. *Let $\mathcal{C}_{2-3}^\bullet$ be the class of the class of unlabelled rooted maps with root vertex degree of 2 and all other vertex degrees equal to 3. The generating function for $\mathcal{C}_{2-3}^\bullet$ is*

$$\mathcal{C}_{2-3}^\bullet(x) := x^3 \cdot \frac{d}{dx} \ln(e^{x^3/3} \odot e^{x^2/2}).$$

Proof. Let \mathcal{C}_3 the class of (connected) maps where all vertices have degree 3. From (1),

$$e^{\mathcal{C}_3(x)} = e^{x^3/3} \odot e^{x^2/2},$$

which gives $C_3(x) = \ln(e^{x^3/3} \odot e^{x^2/2})$. Now if we first derive, then multiply by x , the exponential generating function $C_3(x)$, this is the generating function analog of the combinatorial operation of adding as root a new vertex of degree 2 (this creates 2 new half-edges). Starting from an element of \mathcal{C}_3 , this gives an element of $\mathcal{C}_{2-3}^\bullet$. So,

$$C_{2-3}^\bullet(x) = x^3 \frac{d}{dx} \ln(e^{x^3/3} \odot e^{x^2/2}).$$

□

Standard computations give:

$$e^{x^3/3} \odot e^{x^2/2} = \sum_{n \geq 0} \frac{(6n)!}{(3n)!(2n)!2^{3n}3^{2n}} x^{6n}.$$

Applying Stirling's formula, we get:

$$[x^{6n}](e^{x^3/3} \odot e^{x^2/2}) = \frac{(6n/e)^n}{\sqrt{2\pi n}} \left(1 - \frac{1}{18n} + \frac{1}{648n^2} + \frac{143}{1399680n^3} + O(n^{-4})\right).$$

Lemma 3.7. *We have the following relation :*

$$[x^{6n}]C_3(x) = [x^{6n}]e^{C_3(x)} - \frac{5}{6}[x^{6n-6}]e^{C_3(x)} + O([x^{6n-12}]e^{C_3(x)}).$$

Proof. The corner stone is the following theorem due to Bender [2]:

Suppose $A(x) = \sum_{n>0} a_n x^n$ and $F(x, y) = \sum_{i,k} f_{i,k} x^i y^k$ satisfy:

i) $a_{n-1} = o(a_n)$;

ii) for some $r > 0$, $\sum_{k=r}^{n-r} |a_k a_{n-k}| = O(a_{n-r})$;

iii) $F(x, y)$ is analytic at $(0, 0)$.

Define the two series $B(x) = F(x, A(x))$ and $D(x) = F_y(x, A(x))$ where $F_y(x, y) = \frac{\partial}{\partial y} F(x, y)$. Then

$$[x^n]B(x) = \sum_{k=0}^{n-1} [x^k]D(x)a_{n-k} + O(a_{n-r}).$$

In our case, we choose $F(x, y) = \ln(1 + y)$ (Note that we do not need a bivariate function) and $A(x) = e^{x^{1/2}/3} \odot e^{x^{1/3}/2} - 1$. So, $a_n = [x^{6n}](e^{x^3/3} \odot e^{x^2/2} - 1)$ and the conditions i) and iii) are easily checked. The proof of ii) follows from the observation that, for every r , $\sum_{k=r}^{n-r} |a_k a_{n-k}| = O(a_{n-r})$ due to the super-exponential growth of the coefficients a_n . We have

$$D(x) = 1 - 5/6 x - \frac{335}{72} x^2 - \frac{74285}{1296} x^3 + O(x^4).$$

Thus, applying Bender's result with $r = 2$ then gives Lemma 3.7. □

Theorem 3.8. For $n = 2 \pmod 3$, the number of syntactic trees of size n in BCI has the following asymptotic expansion as n tends to infinity:

$$BCI_n = \frac{\sqrt[3]{2}\sqrt{6}}{2n^{1/6}\sqrt{\pi}} \left(\frac{2n}{e}\right)^{n/3} \left(1 - \frac{1}{2n} + O(n^{-2})\right).$$

Proof. Previous computations show that the coefficient $[x^n]T(x)$ is non-null only for $n = 2 \pmod 3$, and using lemma 3.7, we obtain that

$$[x^{3k+2}]T(x) = [x^{6k+2}]C(x) = 6k \frac{(6k/e)^k}{\sqrt{2\pi k}} \left(1 - \frac{7}{36k} + O(k^{-2})\right),$$

where $T(x)$ is the OGF of the combinatorial class of BCI formulæ with size equal to the number of nodes of its syntactic tree, i.e., the length of the λ -term. The asymptotic behaviour of BCI_n is then obvious. \square

Remark 3.9. As P. Flajolet pointed to us, the papers [11, 21, 12] list various occurrences of the sequence $[z^n]T(z)$ in combinatorics, including the asymptotic enumeration of connected graphs by excess, moments of path length and inversions in trees, moments of the area below (discrete or continuous) excursions, number of trivalent diagrams... The asymptotic expression for BCI_n in Theorem 3.8 can also be directly obtained by solving the Riccati differential equation

$$T = zu + zT^2 + z \frac{\partial T}{\partial u},$$

for which we have an explicit solution $T(x) = \frac{1}{2x} - \frac{\text{Ai}'(1/4x^2)}{\text{Ai}(1/4x^2)}$, with Ai the Airy function.

3.2.2 Asymptotic number of syntactic trees in BCK

In the same vein as above, we start from the equation on the generating function $Y(x)$ for the rooted maps with degrees in $\{2, 3\}$:

$$Y(x) := x \cdot \frac{d}{dx} \ln(e^{x^3/3+x^2/2} \odot e^{x^2/2}).$$

We first have to evaluate the asymptotic expansion of the coefficients of the entire series $e^{x^2/2+x^3/3}$. This question exactly fits the framework of saddle-point methods [13]. A fastidious but straightforward computation gives

$$[x^n]e^{x^2/2+x^3/3} \sim \frac{1}{\sqrt{6\pi n}} \left(\frac{e}{n}\right)^{1/3n} e^{1/18-1/6n^{1/3}+1/2n^{2/3}}.$$

We again apply Bender's theorem and obtain:

$$[x^n] \ln(e^{x^2/2+x^3/3} \odot e^{x^2/2}) \sim n^{n/6} \frac{1}{\sqrt{3\pi n}} e^{1/18-1/6n+1/2n^{2/3}-1/6n^{1/3}}.$$

Now the bijection described in the beginning of this section yields the OGF $S(x)$ for the combinatorial class of BCK formulæ, with size equal to the number of nodes of the syntactic tree associated to a formula, i.e. the length of the λ -term. This generating function satisfies: $S(x) = \frac{1}{1-x}(x^2 + x^2Y(\sqrt{x}))$ where $Y(x) := x \cdot \frac{d}{dx} \ln(e^{x^3/3+x^2/2} \odot e^{x^2/2})$. By standard singularity analysis (the pole at $x = 1$ of $1/(1-x)$ is not dominant), we obtain:

Theorem 3.10. *The number of syntactic trees of size n in BCK logical system admits the following asymptotic equivalent as n tends to infinity:*

$$BCK_n \sim [x^n]S(x) \sim \frac{e^{1/18} \sqrt[3]{2}}{\sqrt{6\pi} n^{1/6}} \left(\frac{2n}{e}\right)^{n/3} e^{1/2(2n)^{2/3} - 1/6(2n)^{1/3}}.$$

Corollary 3.11. *The ratio between the number BCK_n of syntactic trees of size n in BCK system and the number BCI_n of syntactic trees of size n in BCI system has a sub-exponential growth: for $n \equiv 2 \pmod{3}$,*

$$\frac{BCI_n}{BCK_n} \sim 3 e^{-2^{-1/3} n^{2/3} + 1/6 \sqrt[3]{2} n^{1/3} - 1/18}.$$

Proof. Take $n = 3k + 2$. Then the asymptotic behaviour of BCI_n comes from Theorem 3.8 and that of BCK_n from Theorem 3.10. \square

Remark 3.12. The class BCK is isomorphic to the class obtained from BCI when we substitute a non-empty sequence (possibly of size 1) of nodes to every node, i.e. we group a sequence of unlinked abstraction nodes and its child into one single node. The generating function of BCK is then obtained from the generating function of BCI through the substitution of $\frac{z}{1-z}$ to z . However, the asymptotic behaviour of BCK_n seems harder to obtain from this definition than from the map characterization.

4 Random sampling

4.1 Sampler for BCI terms

We can obtain a random sampler for BCI directly from the specification

$$(\mathcal{Z}^2 \times \text{SET}(\text{CYC}_3(\mathcal{Z}))) \odot \text{SET}(\text{CYC}_2(\mathcal{Z})) = \mathcal{Z}^2 \text{SET}^\bullet(\mathcal{C}_3).$$

A simple way to obtain a permutation of length $3n$ of 3-cycles is to sample a labelled sequence (*i.e.* a permutation written in line) of length $3n$ and to form a cycle from each set of 3 consecutive numbers. For example, drawing the sequence 296734185 is equivalent to drawing the permutation (296)(734)(185). As cycles have fixed length, this generates a permutation uniformly among all permutations of 3-cycles of length $3n$, and the corresponding algorithm works in linear time. The same argument goes for drawing an involution without fixed point, i.e. a permutation of 2-cycles. Now sampling an object of $\mathcal{Z}^2 \times$

$\text{SET}(\text{CYC}_3(\mathcal{Z}))$ is just drawing two distinguished labels and a permutation of 3-cycles among the remaining labels.

Algorithm 1: Γ_{BCI}

Input: An integer $6n + 2$, with n even.

Output: a BCI term.

- 1 Draw two labelled sequences α and β of size $6n + 2$.
 - 2 Write α as a product of two atoms and of cycles of length 3. The first label of α is the pointed half-edge.
 - 3 Write β as a product of cycles of length 2.
 - 4 **if** the map $(\{1, \dots, 6n + 2\}, \alpha, \beta)$ is connected **then**
 - 5 Forget labels.
 - 6 Transform the map by the L-morphism.
 - 7 **return** the BCI term.
 - 8 **else**
 - 9 Restart Γ_{BCI} .
-

Example 4.1. A BCI syntactic tree for $n = 1$.

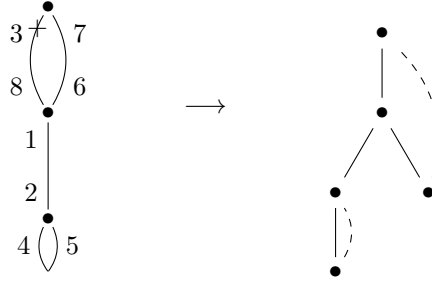


Figure 6: A way to draw the λ -term $\lambda x.((\lambda y.y)x)$

Drawing the sequences $(3, 7, 1, 6, 8, 4, 2, 5)$ and $(8, 3, 6, 7, 4, 5, 1, 2)$ gives the permutations $\alpha = (3, 7, (1, 8, 6)(4, 2, 5))$ (the edges 3 and 7 for which there is no cyclic order correspond to the root) and $\beta = ((8, 3)(6, 7)(4, 5)(1, 2))$. We let the reader check that this leads to the element of \mathcal{D} in Figure 4.1.

Proposition 4.2. *Algorithm 1 provides a uniform sampler that generates a syntactic tree of BCI in expected linear time (wrt size).*

Proof. The uniformity comes directly from the rigidity (see Definition 2.1). Generating permutations and partitioning them into fixed-length parts can easily be done in linear time, as well as right depth-first traversal. There remains the question of the connectivity. By Lemma 3.7, we have asymptotically almost surely only one component, and the expected number of rejections is constant. Thus the sampler returns a syntactic tree, i.e. a lambda-term in BCI, in expected linear time and without any precomputation. \square

Remark 4.3. For an approximate-size sampler, we simply return the pointed L -morphism of the component without any rejection.

4.2 Uniform random sampler for BCK terms

Generating an element of \mathcal{D} where we allow the degree of a node to vary, turns out to be more complicated than generating an element of BCI. Indeed, it is no longer possible to draw an exact-size labelled sequence and to write it in a unique way as a permutation of cycles of differing lengths, and we need a more sophisticated tool to generate such a random permutation uniformly. We use here Boltzmann samplers [9], which are uniform approximate-size samplers automatically build from a specification. The final object is not drawn with Boltzmann probability, but every object of the same size has the same probability to appear. This implies some linear rejection, since using an exponential Hadamard product requires a permutation of even size.

Algorithm 2: Γ_{tree}

Input: A parameter x , a minimal size n_m , a maximal size n_M , the set D_r of allowed root degrees and the set D of allowed degrees for non-root vertices.

Output: A syntactic tree

- 1 Draw α a object in $\text{SEQ}_{D_r}(\mathcal{Z}) \times \text{SET}(\text{CYC}_D(\mathcal{Z}))$ by a labelled Boltzmann sampler of parameter x .
- 2 **if** $|\alpha| \equiv 1 \pmod 2$ **or** $|\alpha| < n_m$ **or** $|\alpha| > n_M$ **then**
- 3 | Restart Γ_{tree}
- 4 **else**
- 5 | Draw a labelled sequence β of size $|\alpha|$.
- 6 Write β as a set of cycles of length 2.
- 7 **if** the map $(\{1, \dots, |\alpha|\}, \alpha, \beta)$ is connected **then**
- 8 | Forget labels.
- 9 | Transform the map by the L -morphism.
- 10 | **return** the term.
- 11 **else**
- 12 | Restart Γ_{tree} .

Theorem 4.4. Algorithm 2 returns a term whose size belongs to $[(1 - \varepsilon)n, (1 + \varepsilon)n]$ in expected linear time, for a fixed size n and a given tolerance in size ε (approximate-size sampler). The drawing is uniform for objects of the same size.

Proof. The probability that a Boltzmann sampler of parameter x generates an object of even size in $\text{SEQ}_{D_r}(\mathcal{Z}) \times \text{SET}(\text{CYC}_D(\mathcal{Z}))$ is $\gamma(x) = \frac{1}{2}(1 + \frac{f(-x)}{f(x)})$, where $f(x) = \sum_{k \in D_r} x^k \cdot \exp(\sum_{k \in D} x^k)$. Now, as x tends to $+\infty$ this quantity $\gamma(x)$ has a limit in $[1/2, 1]$. The expected number of rejections due to the parity is thus constant. After this rejection phase, the drawing of an object in

$\text{SEQ}_{D_r}(\mathcal{Z}) \times \text{SET}(\text{CYC}_D(\mathcal{Z}))$ with size in $[(1 - \varepsilon)n, (1 + \varepsilon)n]$ is done in expected linear time (this result is proved in [9]). The last rejection phase for testing the connexity of the map also requires an average constant time. \square

Proposition 4.5 (On the random generation of BCK syntactic trees). *The random generator of Algorithm 2, for $D = \{2, 3\}$ and $D_r = \{1, 2\}$ gives an approximate-size uniform random sampler for BCK syntactic trees.*

Proof. From Theorems 3.5 and 4.4. \square

Conclusion

We have presented in this paper a “differential” definition of unlabelled rooted combinatorial maps, related to BCI and BCK terms. Switching between this unlabelled specification and the classical labelled specification (by gluing half-edges) allows us to derive the asymptotic behaviour of the number of BCI and BCK terms, and to present random samplers for them. As a surprising (to us) corollary, we proved that the expressivity of BCK, measured by the number of terms of size n we can build, is more than polynomial, but sub-exponential and has order $o(e^{(2n)^{2/3}/2})$.

Enumerating BCI and BCK terms is a first step towards understanding their statistical properties, e.g., their depth or average profile. Future work should also consider more general lambda-terms; although the direct analysis of unrestricted lambda-terms seems quite involved, a natural extension of the work presented here would allow a bounded number of arcs back to any abstraction node [4]. It might also be worthwhile to consider how more complicated constructions with the exponential Hadamard product can be interpreted as differential operations, and to characterize a class of differential equations amenable to this method.

Acknowledgements. We would like to thank M. Zaionc for presenting this problem to us, P. Flajolet for his helpful remarks and for pointing to us the references [11, 12], and A. Bacher for drawing Fig. 1.

References

- [1] C. Banderier, P. Flajolet, G. Schaeffer, and M. Soria. Random maps, coalescing saddles, singularity analysis, and Airy phenomena. *Random Structures and Algorithms*, 19(3-4):194–246, 2001.
- [2] E. A. Bender. Asymptotic methods in enumeration. *SIAM*, 16(4):485–515, 1974.
- [3] E. A. Bender and E. R. Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3):296 – 307, 1978.

- [4] O. Bodini, D. Gardy, B. Gittenberger, and A. Jacquot. Enumeration of lambda-terms of bounded arity. Technical report. In preparation.
- [5] B. Bollobas. *Random Graphs*. Cambridge University Press, 2001.
- [6] G. Chapuy, É. Fusy, O. Giménez, and M. Noy. On the diameter of random planar graphs. In *21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA'10), DMTCS Proceedings*, pages 65–78, 2010.
- [7] R. Cori. Un code pour les graphes planaires et ses applications. *Astérisque*, 27, 1975.
- [8] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
- [9] P. Duchon, P. Flajolet, G. Louchard, and G. Schaeffer. Boltzmann samplers for the random generation of combinatorial structures. *Combinatorics, Probability and Computing*, 13:577–625, 2004.
- [10] J. Edmonds. A combinatorial representation for polyhedral surfaces. *Notices Amer. Math. Soc.*, 7, 1960.
- [11] P. Flajolet and G. Louchard. Analytic variations on the Airy distribution. *Algorithmica*, 31(3):361–377, 2001.
- [12] P. Flajolet, P.V. Pobleto, and A. Viola. On the analysis of linear probing hashing. *Algorithmica*, 22:490–515, 1998.
- [13] P. Flajolet and R. Sedgewick. *Analytic combinatorics*. Cambridge University Press, Cambridge, 2009.
- [14] H. Fournier, D. Gardy, A. Genitrini, and M. Zaionc. Classical and intuitionistic logic are asymptotically identical. In *16th Annual Conference on Computer Science Logic (EACSL), volume 4646 of Lecture Notes in Computer Science*, pages 177–193, 2007.
- [15] A. Genitrini and J. Kozik. Quantitative comparison of intuitionistic and classical logics - full propositional system. In *International Symposium on Logical Foundations of Computer Science (LFCS'09), volume 5407 of Lecture Notes in Computer Science*, pages 280–294, Florida, USA, January 2009.
- [16] A. Joyal. Une théorie combinatoire des séries formelles. *Adv. Math.*, 42:1–82, 1981.
- [17] D. E. Knuth. *The Art of Computer Programming: Fundamental Algorithms*, volume 1. Addison Wesley, Reading, Massachusetts, 3rd edition, July 1997.

- [18] M. Moczurad, J. Tyszkiewicz, and M. Zaionc. Statistical properties of simple types. *Mathematical Structures in Computer Science*, 10(5):575–594, 2000.
- [19] M. Sagastume. Bounded commutative BCK logic and Lukasiewicz logic. *Logic and Philosophy of the Formal Sciences: A Festschrift for Itala M. Loffredo D'Ottaviano*, 28(2), 2005.
- [20] S. A. Vidal. An optimal algorithm to generate rooted trivalent diagrams and rooted triangular maps. *Theoretical Computer Science*, 411:2945–2967, 2010.
- [21] S. A. Vidal and M. Petitot. Counting rooted and unrooted triangular maps. *Journal of Non-linear Systems and Applications*, pages 51–57, 2010.