

AN OPTIMAL ALGORITHM TO GENERATE POINTED TRIVALENT DIAGRAMS AND POINTED TRIANGULAR MAPS

SAMUEL ALEXANDRE VIDAL

ABSTRACT. A trivalent diagram is a connected graph with cyclic orientations and degree conditions on its vertices. It is the combinatorial description of an unembedded trivalent ribbon graph. We shall describe and analyze an algorithm giving an *exhaustive* list of trivalent diagrams of a given size. The list being *non-redundant* in that no two diagrams of the list are isomorphic. The algorithm will be shown to have optimal performances in that the time necessary to generate a diagram will be shown to be bounded in the amortized sense. What is striking is that the bound is *uniform* on the size of the diagrams being generated. One objective of the paper is to provide a reusable theoretical framework for algorithms generating exhaustive lists of complex combinatorial structures with attention paid to the case of *unlabeled structures* and to those generators having the CAT property.

0. INTRODUCTION

Roughly speaking a trivalent diagram is a connected graph with degree conditions on its vertices and cyclic orientations on the edges adjacent to each vertex, it is the combinatorial description of an unembedded trivalent ribbon graph [23, 11] (*cf.* definitions 1.1 and 1.2 for a precise definition). We shall see (*cf.* section 1.2) that it can be described by a pair of permutations $(\sigma_{\bullet}, \sigma_{\circ})$ satisfying the conditions of *involutivity* $\sigma_{\circ}^2 = \text{id}$ and *triangularity* $\sigma_{\bullet}^3 = \text{id}$. The notion of pointed trivalent diagrams are also very useful, both to our study and to the target applications, so we take a special care to study them in detail.

0.1. Motivations. In a recent paper [27] we gave a complete classification of the subgroups of the modular group $\text{PSL}_2(\mathbb{Z})$ and their conjugacy classes by pointed trivalent diagrams and trivalent diagrams. A question one may ask is how to generate a complete list of such trivalent diagrams. Such a question is unavoidable : for a classification to be fully satisfactory one is indeed willing for a systematic way to enumerate all the particular instances of the objects being classified. It was soon realized that there were a connection with combinatorial maps. In this paper we clarify that point and give an application to generate exhaustive lists of triangular combinatorial maps.

2000 *Mathematics Subject Classification.* Primary 68R05, 68W05, 20E07, 05C30 ; Secondary 20E06, 05C85, 05A15.

The other sources of motivations to generate the trivalent diagrams come mainly from mathematical physics in connection with two-dimensionnal quantum gravity and the Witten-Kontsevich model [11]. Algebraic topology is also a source of motivation through triangular subdivisions of surfaces, knots, braids, links and tangles theory [23, 2]. It is also connected to the deformation theory of quantized Hopf algebras [4, 5]. The problem we solve is also relevant to the study of combinatorial maps as explained in section 5 and to the vast galoisian program of A. Grothendieck [8] as explained in hundreds of papers and books such as [17, 27, 19, 16]. As an application, we give in section 4 a way to generate a complete list describing all the sub-groups of a given finite index in the modular group $\mathrm{PSL}_2(\mathbb{Z})$ and a way to decide conjugacy relations among those subgroups. We show also in section 5, as a second application, how to generate an exhaustive list of *triangular maps* satisfying various criteria.

0.2. Problem Statement. We shall describe and analyze two algorithms, the first giving an *exhaustive* list of pointed trivalent diagrams of a given size (*cf.* definitions 1.3 and 1.4 below) and the second giving an exhaustive list of unpointed trivalent diagrams (definitions 1.1 and 1.2). Those lists being *non-redundant* in that no two diagrams of the lists are isomorphic. The algorithm for pointed diagrams will be shown to have optimal performances meaning that the time necessary to generate a diagram will be shown to be *bounded* in the amortized sense. What is striking is that the bound is *uniform* on the size of the diagrams being generated. One objective of the paper is to provide a reusable theoretical framework for algorithms generating exhaustive lists of complex combinatorial structures with attention paid to the case of *unlabeled structures* and to those generators having the CAT property.

0.3. What is a CAT Generator. The *CAT generator* expression stands as an acronym for *constant amortized time generator* meaning a generator of combinatorial structures that on the large seems to spend only a constant time generating each of the structures. The usual idea in such a generator is that one arranges so that passing from a structure to the next only takes a few modifications to be made. Sometimes though, it could take more modifications than usual and we don't usually have any upper bound in the amount of actual modifications that could be needed to pass from a structure to the next. When needs for large amounts of modifications tends to be significantly rare in comparison to small ones, we can sometimes prove that an *amortization effect* is going on. In such a situation, one cannot tell from the running time of the algorithm and the number of structure generated in the interval, the actual *size* of the structures being generated. Technically, one can summarize that amortization effect in saying that the total amount of time needed to generate n distinct structures is *asymptotically bounded* by a constant multiple of the number n of structures being generated, the word *constant* meaning that the bound is *uniform* on the size of the structures being generated.

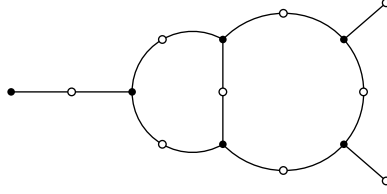


Figure 1. A trivalent diagram is conveniently described by a little diagram like the one above, hence the name. The actual cyclic orientation of the vertices are conveniently rendered implicit by adopting the standard trigonometric orientation of the figure.

1. TRIVALENT DIAGRAMS

Definition 1.1. A *trivalent diagram* Γ is given by three sets Γ_{\bullet} , Γ_{\circ} and Γ_{-} and two maps $\partial_{\bullet} : \Gamma_{-} \rightarrow \Gamma_{\bullet}$, $\partial_{\circ} : \Gamma_{-} \rightarrow \Gamma_{\circ}$, and two \mathbb{Z} -actions $+_{\bullet}, +_{\circ} : \Gamma_{-} \times \mathbb{Z} \rightarrow \Gamma_{-}$ given by $(a, n) \mapsto a +_{\bullet} n$ and $(a, n) \mapsto a +_{\circ} n$ satisfying, for all pair a, a' of distinct elements of Γ_{-} the following conditions,

$$\begin{aligned} a +_{\bullet} 3 &= a & \partial_{\bullet}(a +_{\bullet} 1) &= \partial_{\bullet}(a) & \partial_{\bullet}(a) = \partial_{\bullet}(a') &\Rightarrow \exists n, a +_{\bullet} n = a' \\ a +_{\circ} 2 &= a & \partial_{\circ}(a +_{\circ} 1) &= \partial_{\circ}(a) & \partial_{\circ}(a) = \partial_{\circ}(a') &\Rightarrow \exists n, a +_{\circ} n = a' \end{aligned}$$

The two maps ∂_{\bullet} and ∂_{\circ} being moreover assumed to be *surjective*.

The elements of Γ_{-} are the edges of the diagram, the elements of the two sets Γ_{\bullet} and Γ_{\circ} are the *black* and *white* vertices of the diagram. Each edge is adjacent to exactly one black vertex and one white vertex, those two vertices being given by the two maps ∂_{\bullet} and ∂_{\circ} respectively.

Definition 1.2. A morphism φ between two trivalent diagrams Γ and Γ' is a triple of applications $\varphi_{\bullet} : \Gamma_{\bullet} \rightarrow \Gamma'_{\bullet}$, $\varphi_{\circ} : \Gamma_{\circ} \rightarrow \Gamma'_{\circ}$ and $\varphi_{-} : \Gamma_{-} \rightarrow \Gamma'_{-}$, compatible to the three structure applications and the two group actions in that φ_{-} is both $+_{\bullet}$ and $+_{\circ}$ equivariant and the following two diagrams are commutative,

$$\begin{array}{ccc} \Gamma_{-} & \xrightarrow{\varphi_{-}} & \Gamma'_{-} \\ \partial_{\bullet} \downarrow & & \downarrow \partial_{\bullet} \\ \Gamma_{\bullet} & \xrightarrow{\varphi_{\bullet}} & \Gamma'_{\bullet} \end{array} \qquad \begin{array}{ccc} \Gamma_{-} & \xrightarrow{\varphi_{-}} & \Gamma'_{-} \\ \partial_{\circ} \downarrow & & \downarrow \partial_{\circ} \\ \Gamma_{\circ} & \xrightarrow{\varphi_{\circ}} & \Gamma'_{\circ} \end{array}$$

1.1. Pointed Trivalent Diagrams. The following notion play an important rôle in that dissertation and in the applications.

Definition 1.3. A trivalent diagram is said to be *pointed* if one of its edges is distinguished from the others.

A convenient way to describe the pointing of a diagram is to draw a little cross on its distinguished edge.

Definition 1.4. A *morphism* φ of pointed trivalent diagrams (Γ, a) and (Γ', a') is a morphism of the underlying diagrams (forgetting base points) which φ_- component is further assumed to send base point to base point *i.e.* for which we have the following relation $\varphi_-(a) = a'$.

Convention. From now on, without express mention of the contrary, trivalent diagrams are all assumed to be *connected*.

1.2. Permutational Representation. Trivalent diagrams have a convenient description in terms of two permutations, that proves to be useful to formulate various algorithms. To a trivalent diagram Γ one associate the set $X_\Gamma = \{ \mathbf{a}_a \}_{a \in \Gamma_-}$ (which is a copy of the set Γ_- of edges of Γ with canonical bijection associating to any edges a of Γ the element \mathbf{a}_a of X_Γ) and two permutations σ_\bullet and σ_\circ of X_Γ defined by the following relations,

$$\sigma_\bullet(\mathbf{a}_a) = \mathbf{a}_{a+\bullet,1} \quad \text{and} \quad \sigma_\circ(\mathbf{a}_a) = \mathbf{a}_{a+\circ,1}$$

The triple constituted of X_Γ , σ_\bullet and σ_\circ is by definition, the *permutational representation* of the diagram Γ . To state the functoriality of this operation we are first required to give a precise notion of morphism between the introduced objects.

Definition 1.5. A *morphism* φ between two pairs of permutations $(\sigma_\bullet, \sigma_\circ)$ on two respective sets X and X' is an application $\varphi : X \rightarrow X'$ which is simultaneously equivariant to σ_\bullet and σ_\circ in the sense that the two following diagrams are commutative,

$$\begin{array}{ccc} X & \xrightarrow{\varphi} & X' \\ \sigma_\bullet \downarrow & & \downarrow \sigma_\bullet \\ X & \xrightarrow{\varphi} & X' \end{array} \qquad \begin{array}{ccc} X & \xrightarrow{\varphi} & X' \\ \sigma_\circ \downarrow & & \downarrow \sigma_\circ \\ X & \xrightarrow{\varphi} & X' \end{array}$$

Any morphism φ between two trivalent diagrams Γ and Γ' induce a morphism φ_\triangleleft between their permutational representations X_Γ and $X_{\Gamma'}$, defined for all \mathbf{a}_a in X_Γ by the following equation,

$$\varphi_\triangleleft(\mathbf{a}_a) = \mathbf{a}'_{\varphi_-(a)}$$

It is now easily seen that we have a simultaneous equivariance of the induced map φ_\triangleleft with respect to σ_\bullet and σ_\circ as required by definition 1.5 above :

$$\begin{array}{ll} \varphi_\triangleleft(\sigma_\bullet(\mathbf{a}_a)) = \varphi_\triangleleft(\mathbf{a}_{a+\bullet,1}) & \varphi_\triangleleft(\sigma_\circ(\mathbf{a}_a)) = \varphi_\triangleleft(\mathbf{a}_{a+\circ,1}) \\ = \mathbf{a}'_{\varphi_-(a+\bullet,1)} & = \mathbf{a}'_{\varphi_-(a+\circ,1)} \\ = \mathbf{a}'_{\varphi_-(a)+\bullet,1} & = \mathbf{a}'_{\varphi_-(a)+\circ,1} \\ = \sigma_\bullet(\mathbf{a}'_{\varphi_-(a)}) & = \sigma_\circ(\mathbf{a}'_{\varphi_-(a)}) \\ = \sigma_\bullet(\varphi_\triangleleft(\mathbf{a}_a)) & = \sigma_\circ(\varphi_\triangleleft(\mathbf{a}_a)) \end{array}$$

The induced maps are transitive in the following sense. Let Γ , Γ' and Γ'' be three trivalent diagrams and let $\varphi : \Gamma' \rightarrow \Gamma$ and $\varphi' : \Gamma'' \rightarrow \Gamma'$ be two morphisms of trivalent

diagrams, then we have both $(\text{Id}_\Gamma)_\triangleleft = \text{Id}_{X_\Gamma}$ and $(\varphi \circ \varphi')_\triangleleft = \varphi_\triangleleft \circ (\varphi')_\triangleleft$. The first statement is immediate, the second is a matter of simple rewriting,

$$\begin{aligned} (\varphi_\triangleleft \circ (\varphi')_\triangleleft)(\mathbf{a}''_a) &= \varphi_\triangleleft(\mathbf{a}'_{\varphi'_-(a)}) \\ &= \mathbf{a}_{\varphi_-(\varphi'_-(a))} \\ &= (\varphi \circ \varphi')_\triangleleft(\mathbf{a}''_a) \end{aligned}$$

The situation just described can be concisely summed up by saying that the process of taking the permutational representations of trivalent diagrams is functorial.

We shall now prove that this representation is *faithful* in that one can recover any diagram from its permutational representation. To that purpose, we shall now introduce a *reconstruction* operation and show that it is *reciprocal* to that of taking the permutational representation, in the precise sense that the reconstructed diagram $(X_\Gamma)^{diag}$ from the permutational representation X_Γ of a diagram Γ , is naturally isomorphic to Γ the diagram it comes from (theorem 1.1 below).

That reconstruction operation takes a pair of permutations $(\sigma_\bullet, \sigma_\circ)$ on a given set X , satisfying both triangularity $\sigma_\bullet^3 = \text{id}$ and involutivity $\sigma_\circ^2 = \text{id}$ properties, and it produces a trivalent diagram denoted by X^{diag} which sets of black vertices, white vertices and edges are the following,

$$X_\bullet^{diag} = X/\sigma_\bullet \qquad X_\circ^{diag} = X/\sigma_\circ \qquad X_-^{diag} = X$$

which two structure maps ∂_\bullet and ∂_\circ from X_-^{diag} to X_\bullet^{diag} and X_\circ^{diag} are the natural projections of those quotients and which two \mathbb{Z} -actions $+_\bullet$ and $+_\circ$ on X_-^{diag} are defined for all elements x of X and all integer n by the following relations,

$$x +_\bullet n = \sigma_\bullet^n(x) \quad \text{and} \quad x +_\circ n = \sigma_\circ^n(x)$$

Theorem 1.1. *For any trivalent diagram Γ , there exists a bijection φ_- natural with respect to Γ and simultaneously equivariant with respect to $+_\bullet$ and $+_\circ$ and two natural bijections φ_\bullet and φ_\circ all of which plug in the following commutative diagrams,*

$$\begin{array}{ccc} (X_\Gamma)_-^{diag} & \xrightarrow{\varphi_-} & \Gamma_- \\ \partial_\bullet \downarrow & & \downarrow \partial_\bullet \\ (X_\Gamma)_\bullet^{diag} & \xrightarrow{\varphi_\bullet} & \Gamma_\bullet \end{array} \qquad \begin{array}{ccc} (X_\Gamma)_-^{diag} & \xrightarrow{\varphi_-} & \Gamma_- \\ \partial_\circ \downarrow & & \downarrow \partial_\circ \\ (X_\Gamma)_\circ^{diag} & \xrightarrow{\varphi_\circ} & \Gamma_\circ \end{array}$$

Demonstration. The application φ_- which associate to \mathbf{a}_a the edge a is clearly natural, bijective and simultaneously equivariant with respect to $+_\bullet$ and $+_\circ$. Therefore, as ∂_\bullet and ∂_\circ are equivariant by definition to $+_\bullet$ and $+_\circ$ respectively, so are $\partial_\bullet \circ \varphi_-$ and $\partial_\circ \circ \varphi_-$. The application φ_\bullet (resp. φ_\circ) is induced between $(X_\Gamma)_\bullet^{diag} = \Gamma_-/\sigma_\bullet$ (resp. $(X_\Gamma)_\circ^{diag} = \Gamma_-/\sigma_\circ$) and Γ_\bullet by equivariance of $\partial_\bullet \circ \varphi_-$ with respect to σ_\bullet (resp. by equivariance of $\partial_\circ \circ \varphi_-$ with respect to σ_\circ).

Let's prove now that φ_\bullet and φ_\circ are both bijective. First, as $(\partial_\bullet \circ \varphi_-) : (X_\Gamma)_-^{diag} \rightarrow \Gamma_\bullet$ and $(\partial_\circ \circ \varphi_-) : (X_\Gamma)_-^{diag} \rightarrow \Gamma_\circ$ are surjective, the two applications φ_\bullet and φ_\circ are both surjective by commutativity of the diagrams. Let's show that φ_\bullet is injective.

Let x and x' be two elements of $(X_\Gamma)_\bullet^{diag}$ such that $\varphi_\bullet(x) = \varphi_\bullet(x')$, then by surjectivity of $\partial_\bullet : (X_\Gamma)_-^{diag} \rightarrow (X_\Gamma)_\bullet^{diag}$ there exist a and a' in Γ_- such that $x = \partial_\bullet(\mathbf{a}_a)$ and $x' = \partial_\bullet(\mathbf{a}_{a'})$ then,

$$\begin{aligned} \varphi_\bullet(\partial_\bullet(\mathbf{a}_a)) = \varphi_\bullet(\partial_\bullet(\mathbf{a}_{a'})) &\Rightarrow \partial_\bullet(\varphi_-(\mathbf{a}_a)) = \partial_\bullet(\varphi_-(\mathbf{a}_{a'})) \\ &\Rightarrow \partial_\bullet(a) = \partial_\bullet(a') \\ &\Rightarrow \exists n, a +_\bullet n = a' \\ &\Rightarrow \exists n, \sigma_\bullet^n(\mathbf{a}_a) = \mathbf{a}_{a'} \\ &\Rightarrow \partial_\bullet(\mathbf{a}_a) = \partial_\bullet(\mathbf{a}_{a'}) \end{aligned}$$

Thus $x = x'$ and φ_\bullet is injective. Let's show now that φ_\circ is also injective. Let x and x' be two elements of $(X_\Gamma)_\circ^{diag}$ such that $\varphi_\circ(x) = \varphi_\circ(x')$, then by surjectivity of $\partial_\circ : (X_\Gamma)_-^{diag} \rightarrow (X_\Gamma)_\circ^{diag}$ there exist a and a' in Γ_- such that $x = \partial_\circ(\mathbf{a}_a)$ and $x' = \partial_\circ(\mathbf{a}_{a'})$ then,

$$\begin{aligned} \varphi_\circ(\partial_\circ(\mathbf{a}_a)) = \varphi_\circ(\partial_\circ(\mathbf{a}_{a'})) &\Rightarrow \partial_\circ(\varphi_-(\mathbf{a}_a)) = \partial_\circ(\varphi_-(\mathbf{a}_{a'})) \\ &\Rightarrow \partial_\circ(a) = \partial_\circ(a') \\ &\Rightarrow \exists n, a +_\circ n = a' \\ &\Rightarrow \exists n, \sigma_\circ^n(\mathbf{a}_a) = \mathbf{a}_{a'} \\ &\Rightarrow \partial_\circ(\mathbf{a}_a) = \partial_\circ(\mathbf{a}_{a'}) \end{aligned}$$

Thus $x = x'$ and φ_\circ is also injective, which ends the demonstration. \square

Corollary. *There are a natural bijections between the cycles of the permutation σ_\bullet (resp. the permutation σ_\circ) and the black vertices of Γ (resp. the the white vertices of Γ). Those two natural bijections are both unique.*

Demonstration. It suffice to notice that the elements of $(X_\Gamma)_\bullet^{diag} = \Gamma_-/\sigma_\bullet$ and $(X_\Gamma)_\circ^{diag} = \Gamma_-/\sigma_\circ$ are nothing but the cycles of the two permutations σ_\bullet and σ_\circ respectively, then it become obvious that the bijections φ_\bullet and φ_\circ of the theorem satisfy the statement. Uniqueness is granted by construction. \square

1.3. Labeled vs. Unlabeled Diagrams. Historically, the dichotomy between *labeled* and *unlabeled* structures had been greatly clarified and properly emphasized by the discovery by A. Joyal of *combinatorial species* [10]. The subject was, and still is, a very prolific source of discovery from the Quebec school of combinatorics and from a growing community of researchers around the world. One must cite the wonderful book [6] by F. Bergeron, G. Labelle and P. Leroux, which gives a brilliant exposition of the whole subject.

On a given set of vertices X one can build different trivalent diagrams and pointed trivalent diagrams. Let's denote by $D_3(X)$ and $D_3^\bullet(X)$ the corresponding sets of structures, let's call X the *labeling alphabet* and let's talk about diagrams one can build on that set, to be *labeled diagrams* on X or diagrams *labeled* by X . Any bijection ϱ between two finite sets X and Y induce another bijection ϱ_* between the sets $D_3(X)$ and $D_3(Y)$ of pointed trivalent diagrams labeled by X and Y respectively. This induced bijection is the *relabeling operation*, from $D_3(X)$ to $D_3(Y)$, according to the bijection ϱ between the labeling sets. It is also referred as a *transport of structure* along the relabeling bijection ϱ . The same considerations applied to pointed

trivalent diagrams give rise to an induced bijection, also denoted by ϱ_* for simplicity, between the sets of $D_3^\bullet(X)$ and $D_3^\bullet(Y)$ of pointed trivalent diagrams labeled by X and Y .

What precedes leads to the consideration of the *Joyal Functors* D_3 and D_3^\bullet of the two combinatorial species of *trivalent diagrams* and *pointed trivalent diagrams* respectively. In the formalism of Joyal, two labeled structures are said to be *conjugated* or *isomorphic* if they coincide modulo the relabeling operation. An unlabeled structure is then just a conjugacy class of labeled structures. Let's denote by $\tilde{D}_3[n]$ and $\tilde{D}_3^\bullet[n]$ the sets of unlabeled trivalent diagrams, *unpointed* and *pointed* respectively. In precise terms, the symmetric group \mathfrak{S}_n acts via relabeling on the set of structures labeled by $\{1, \dots, n\}$ and the sets $\tilde{D}_3(n)$ and $\tilde{D}_3^\bullet(n)$ can be seen as the quotient sets of this induced group action.

$$\tilde{D}_3(n) \stackrel{\text{def.}}{=} D_3(\{1, \dots, n\})/\mathfrak{S}_n \quad \text{and} \quad \tilde{D}_3^\bullet(n) \stackrel{\text{def.}}{=} D_3^\bullet(\{1, \dots, n\})/\mathfrak{S}_n$$

We call the corresponding natural projections,

$$\begin{aligned} \pi_n : D_3(\{1, \dots, n\}) &\rightarrow D_3(\{1, \dots, n\})/\mathfrak{S}_n \\ \pi_n : D_3^\bullet(\{1, \dots, n\}) &\rightarrow D_3^\bullet(\{1, \dots, n\})/\mathfrak{S}_n \end{aligned}$$

the *condensation maps* of the combinatorial species D_3 and D_3^\bullet . The idea being that labeled structures form vast 'clouds' composed of a myriad of structures with differences between them varying from *pointless* and *insignificant* if the structures are conjugated, to *essential* and *noteworthy* if they are not conjugated, *unlabeled structures* ultimately standing as *the rare* and *the interesting*.

2. CHARACTERISTIC LABELING

A *characteristic labeling* is the choice of a unique representative in every conjugacy class of structure. In other terms, a characteristic labeling can be seen as a *natural section* to the condensation map π . The characteristic labeling that we like are those which are *computable*. We like them *even more* if there is an *efficient* way to compute them.

Pointed trivalent diagrams have the enjoyable property to possess many characteristic labelings that are computable with efficient algorithms. This situation is to be contrasted with that of *general graphs*. No algorithm is known to decide in polynomial time whether two given graphs are isomorphic, and having an efficient algorithm computing characteristic labeling of general graphs would render that particular problem *trivial*. We shall precise that, what makes trivalent diagrams particular in that respect, is not much in them being trivalent, but more in them being cyclically oriented at their vertices. Indeed, general graphs having only trivalent vertices still suffer from the above problem.

What we give now, is a succinct description of an algorithm producing a characteristic labeling of pointed trivalent diagrams Γ and having linear time complexity in the number of edges of Γ . The idea is the following, build a rooted planar binary tree T by depth-first traversal of the *edges* of the diagram (not the vertices, I insist

on the *edges*). Given a particular edge a of Γ , the two directions that are explored from it, are given by the two σ_\bullet and σ_\circ operations on the set of edges. We take care to never revisit a previously visited edge and we label the edges of Γ by numbers from 1 to n according to the order of their appearance in the depth-first traversal.

Algorithm 1: VISIT ($x : X$)

```

1 begin
2   if visited[x] then return
3   visited[x] ← true
4    $\ell_0[x] \leftarrow c$ 
5    $\ell_1[c] \leftarrow x$ 
6    $c \leftarrow c + 1$ 
7   VISIT ( $s_0[x]$ )
8   VISIT ( $s_1[x]$ )
9 end
```

Algorithm 2: RELABEL ($x : X$)

```

1 begin
2    $c \leftarrow 1$ 
3   for  $i \in X$  do
4      $\_visited[i] \leftarrow false$ 
5     VISIT ( $x$ )
6   for  $k \in \{1, \dots, n\}$  do
7      $t_0[k] \leftarrow \ell_0[s_0[\ell_1[k]]]$ 
8      $t_1[k] \leftarrow \ell_0[s_1[\ell_1[k]]]$ 
9 end
```

2.1. Implementation. We need as global data, seven arrays as follows and an integer c ,

$$\begin{aligned}
visited &: X \rightarrow \text{Bool} \\
\ell_0 &: \{1, \dots, n\} \rightarrow X \\
\ell_1 &: X \rightarrow \{1, \dots, n\} \\
s_0, s_1 &: X \rightarrow X \\
t_0, t_1 &: \{1, \dots, n\} \rightarrow \{1, \dots, n\}
\end{aligned}$$

Algorithm 1 which is an auxiliary recursive program computing the transport bijections ℓ_0 and ℓ_1 . Algorithm 2 is the main entry point of the relabeling process. It does the initialization job (line 2 to 4) and the actual relabeling of the input diagram (line 6 to 8). It takes as input a trivalent diagram labeled with the elements of the set X and pointed by the element x of X . which is described by the arrays s_0 and s_1 and the element $x \in X$ are descriptions of the input diagram via its associated two permutations σ_\bullet and σ_\circ (*cf.* section 1.2). The output diagram is encoded by the two arrays t_0 and t_1 in the very same fashion. The *visited* array is used to remember the positions already visited by the relabeling process. The integer c serves as a counter

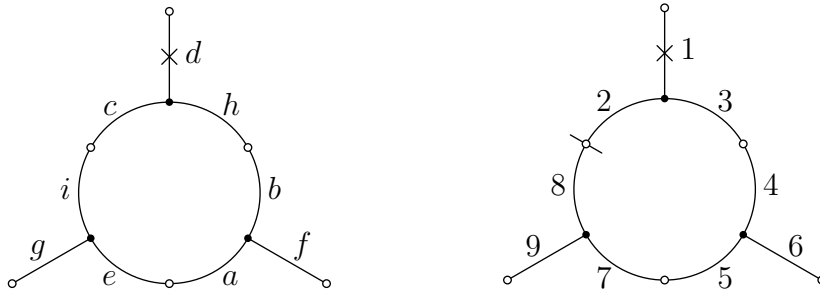


Figure 2. If one gives as input to the relabeling algorithm (algorithm 2) the pointed diagram shown on the left with an arbitrary initial labeling on the arbitrary alphabet $X = \{a, b, c, d, e, f, g, h, i\}$, it produces the characteristic relabeling shown on the right with numbers from 1 to 9 according to the depth-first traversal order of algorithm 1. One shall notice the *natural cutting* between the edges labeled by 2 and 8 that arise from the depth-first traversal.

to label the vertices in the order they are encountered, ℓ_0 and ℓ_1 are internal arrays describing the mutual inverse transport bijections between the input diagram and the output diagram.

2.2. Adequacy. The idea behind that algorithm is quit simple and present no difficulty excepting the *actual proof* of the relabeling being *characteristic*. There is two ways to do the proof, one is conceptual by nature, the other is more technical. The particular description of the algorithm is itself part of that former argument. We shall give both arguments because preferring one or the other is simply a mater of taste. Let's give the conceptual argument first.

One could have taken the input diagram to be labeled by the set $\{1, \dots, n\}$ then shown that the output labeled diagram stay unchanged if one conjugate the input labeled diagram according to any permutation of the labeling set. Such a proof would typically look rather technical if not difficult. Instead, we rather *abstract* the labeling alphabet of the input diagram to be an ordinary set X having exactly n distinct elements, and this requirement being the only assumption made on X . In particular, we make absolutely no assumption on the elements of X or on any structure that this particular set may carry.

A moment thought may convince the reader that *abstracting* the input label set to X and making no assumption whatsoever on the elements of that particular set indeed guaranties the required invariance, but this argument is undeniably subtle and may seems a hand waving argument to most people, so we give now another proof avoiding such considerations.

Theorem 2.1. *Algorithm 2 produce a characteristic relabeling of the connected pointed trivalent diagrams of size n .*

Demonstration. A permutation σ of the input label set X induce a conjugacy of the two input permutations s_0 and s_1 yielding two permutations $s'_0 = \sigma \cdot s_0 \cdot \sigma^{-1}$ and $s'_1 = \sigma \cdot s_1 \cdot \sigma^{-1}$. Now, putting $\ell_0(x) = c$ and $\ell'_0(x') = c$ according to line 4 of

algorithm 1 with $x' = \sigma(x)$ and varying x yields $\ell'_0 = \ell_0 \cdot \sigma^{-1}$. Similarly, considering line 5 of the same algorithm, we get $\ell'_1 = \sigma \cdot \ell_1$. Permutations t_0, t'_0, t_1 and t'_1 verifying the following identities (by line 6-8 of algorithm 2),

$$\begin{aligned} t_0 &= \ell_0 \cdot s_0 \cdot \ell_1 & t'_0 &= \ell'_0 \cdot s'_0 \cdot \ell'_1 \\ t_1 &= \ell_0 \cdot s_1 \cdot \ell_1 & t'_1 &= \ell'_0 \cdot s'_1 \cdot \ell'_1 \end{aligned}$$

and substituting $\ell'_0, \ell'_1, s'_0,$ and s'_1 for their above values, yields by a mutual cancellation of the σ 's,

$$\begin{aligned} t'_0 &= (\ell_0 \cdot \sigma^{-1}) \cdot (\sigma \cdot s_0 \cdot \sigma^{-1}) \cdot (\sigma \cdot \ell_1) = t_0 \\ t'_1 &= (\ell_0 \cdot \sigma^{-1}) \cdot (\sigma \cdot s_1 \cdot \sigma^{-1}) \cdot (\sigma \cdot \ell_1) = t_1 \end{aligned}$$

Thus proving the required invariance of the result. □

3. GENERATING ALGORITHM

Let's imagine that along the way of exploring a particular pointed trivalent diagram with algorithm 2 of section 2, we emit a sequence of events describing the particular cycles of the permutations t_0 and t_1 we encounter at each stage of the traversal. Those events typically saying for example : there we reach a new unforeseen black vertex (forward connection) and we label its adjacent edges $c, c + 1, c + 2,$ or there we reach a previously visited white vertex (backward connection), or there we reach an unforeseen white vertex, etc...

One shall easily convince oneself that such a sequence of events, relying only on the execution march of the algorithm and not on the particular labeling of the input diagram, is in fact *characteristic* to the diagram. If sufficiently detailed, that sequence of events can be used to unambiguously characterize pointed trivalent diagrams. The idea now, would be to consider a rooted planar tree with leaves labeled by pointed trivalent diagrams and with edges labeled by events in such a way that the sequence of events one gets along any branch from the root to a leaf is the very sequence of events that unambiguously characterize the corresponding pointed trivalent diagram.

We now get a usable principle of generation if we require two further properties. First *exhaustivity*, meaning that every conjugacy class of pointed trivalent diagram gets represented on a particular leaf of the tree, then *non-redundancy*, meaning that every such conjugacy class gets represented *just once*. Assuming that we spend only a constant time on each node of that tree and that the number of those nodes is linearly bounded by the number of its leaves, this would provide a constant amortized time algorithm to generate pointed trivalent diagrams.

To ease the memory requirements of the generator, we won't actually build the generation tree in memory. It will instead be realized in the calling pattern between the procedures of the generating program. Also, the program would be more useful if it generates the diagrams in permutational form instead of a sequence of events describing it. This mean that we have to carry around a partial diagram that gets built along the way of exploring the generation tree, each generating event

completing that description and each backtrack reverting the particular changes we have made.

3.1. Implementation. Fundamentally, the present algorithm generate by backtrack all the possible executions of the relabeling algorithm 2 of section 2 which itself is a backtrack on all the possible paths inside the input pointed trivalent diagram. This leads to a *tricky* double-backtrack algorithm. Such a technique needs two stacks to store the intermediate values of the two backtracks at work. One of the two stacks can be rendered implicit in the recursive formulation of the algorithm but the other is necessarily explicit.

We refer to that stack (modeling the parameter stack of the recursive algorithm 1 of section 2) through the following self-explanatory methods (reference to the stack object itself being removed in the code, for the sake of brevity), the last two methods being just suggestive alias to *insertion* and *removal* primitives.

$$\begin{aligned} \text{PUSH} &: \text{Integer} \times \text{Stack} \rightarrow \text{Stack} \\ \text{POP} &: \text{Stack} \rightarrow \text{Integer} \times \text{Stack} \\ \text{STACKISEMPTY} &: \text{Stack} \rightarrow \text{Bool} \\ \text{MASK, REVEAL} &: \text{Integer} \times \text{Stack} \rightarrow \text{Stack} \end{aligned}$$

All of those methods can have constant execution time implementations, for example through a doubly linked list and an array to have both, constant iteration time needed line 8 of algorithm 4 and random access needed for MASK and REVEAL to have constant execution time. Note that no *membership* test is needed, although it could have been implemented with constant execution time with the help of just an additional boolean array.

Manipulation of the partial diagram is done through the four EMIT procedures, building cycles of size 1, 2 and 3 in the corresponding *black* σ_{\bullet} or *white* σ_{\circ} permutation of the diagram and their four REVERT counterparts doing just the converse. The entry point of the algorithm is the GENERATE procedure (algorithm 3). The OUTPUT method called line 4 of the RECURSE procedure (algorithm 9) is a user-defined procedure that serves as an *outlet* to the algorithm. It can be used for instance to do printing jobs or to collect some statistics on pointed trivalent diagrams. In section 5 and 4 below, we give natural bijections to the combinatorial species of triangular maps and to that of subgroups of the modular group.

3.2. Adequacy. There is at least *two* properties to be proved, first *non-redundancy* and then *exhaustivity*. The non-redundancy property is a simple consequence of the characteristicness of the sequence of the depth-first traversal (theorem 2.1). But exhaustivity still remains to be proved, it results by induction from a local exhaustivity property. To explain what we mean by local exhaustivity we shall take a close look at the DISPATCH procedure (algorithm 4).

Algorithm 3: GENERATE ($$)

```
1 begin
2   if  $MaxSize \geq 1$  then
3      $c \leftarrow 2$ 
4     EMITBLACK1CYCLE (1)
5     DISPATCH (1)
6   if  $MaxSize \geq 3$  then
7      $c \leftarrow 4$ 
8     EMITBLACK3CYCLE (1, 2, 3)
9     PUSH (1)
10    PUSH (2)
11    DISPATCH (3)
12 end
```

Algorithm 4: DISPATCH ($s : integer$)

```
1 local  $t : integer$ 
2 begin
3   if  $c + 3 \leq MaxSize + 1$  then
4     TRYFOREWARD ( $s$ )
5   if  $c + 1 \leq MaxSize + 1$  then
6     TRYCLOSEDBLACK ( $s$ )
7     TRYCLOSEDBLACK ( $s$ )
8   for  $t \in STACK$  do
9     MASK ( $t$ )
10    TRYBACKWARD ( $s, t$ )
11    REVEAL ( $t$ )
12 end
```

Algorithm 5: TRYFORWARD ($s : integer$)

```
1 begin
2   EMITBLACK3CYCLE ( $c, c + 1, c + 2$ )
3   PUSH ( $c + 1$ )
4   EMITWHITE2CYCLE ( $s, c$ )
5    $c \leftarrow c + 3$ 
6   DISPATCH ( $c - 1$ )
7    $c \leftarrow c - 3$ 
8   REVERTWHITE2CYCLE ( $s, c$ )
9   POP ( $\phantom{}$ )
10  REVERTBLACK3CYCLE ( $c, c + 1, c + 2$ )
11 end
```

Algorithm 6: TRYBACKWARD ($s, t : integer$)

```
1 begin
2   | EMITWHITE2CYCLE ( $s, t$ )
3   | RECURSE ()
4   | REVERTWHITE2CYCLE ( $s, t$ )
5 end
```

Algorithm 7: TRYCLOSEDWHITE ($s : integer$)

```
1 begin
2   | EMITWHITE1CYCLE ( $s$ )
3   | RECURSE ()
4   | REVERTWHITE1CYCLE ( $s$ )
5 end
```

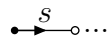
Algorithm 8: TRYCLOSEDBLACK ($s : integer$)

```
1 begin
2   | EMITWHITE2CYCLE ( $s, c$ )
3   | EMITBLACK1CYCLE ( $c$ )
4   |  $c \leftarrow c + 1$ 
5   | RECURSE ()
6   |  $c \leftarrow c - 1$ 
7   | REVERTBLACK1CYCLE ( $s$ )
8   | REVERTWHITE2CYCLE ( $s, c$ )
9 end
```

Algorithm 9: RECURSE ()

```
1 local  $k : integer$ 
2 begin
3   | if STACKISEMPTY () then
4   |   | OUTPUT ()
5   | else
6   |   |  $k \leftarrow \text{POP} ()$ 
7   |   | DISPATCH ( $k$ )
8   |   | PUSH ( $k$ )
9 end
```

The situation at the start of the DISPATCH procedure is the following, we have an edge labeled by s and we ask for all the possibilities that one can encounter in visiting the next adjacent edge in the black to white direction,

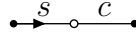


Algorithms 5, 6, 7 and 8 handle the following four cases.

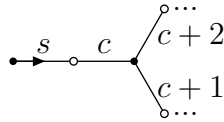
- 1) TRYCLOSEDWHITE (s) generates all the diagrams having an univalent white vertex at the current position.



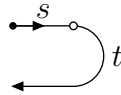
- 2) TRYCLOSEDBLACK (s) generates all the diagrams having a bivalent white vertex and an univalent black vertex at the current position. It labels with the current value of the counter c the previously unvisited edge.



- 3) TRYFORWARD (s) generates all the diagrams having a bivalent white vertex at the current position and going on with an unvisited trivalent black vertex whose adjacent edges are labeled by c , $c + 1$ and $c + 2$.



- 4) TRYBACKWARD (s, t) generates all the diagrams having a bivalent white vertex at the current position and going on with an already visited edge labeled by t .



We claim that those four cases exhausts the local possibilities, which is readily verified by considering all the possibilities of adjacency for the edge labeled s , and that this local exhaustivity guaranties by induction, the full exhaustivity of the generating algorithm.

4. FIRST APPLICATION : MODULAR GROUP

We recall that the modular group $\text{PSL}_2(\mathbb{Z})$ is the group of integer matrices with unit determinant,

$$\text{PSL}_2(\mathbb{Z}) = \left\{ \pm \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathcal{M}_2(\mathbb{Z}) / \pm \text{Id} \mid ad - bc = 1 \right\}$$

There are many possible finite presentations for this group and we shall stick to the following,

$$\text{PSL}_2(\mathbb{Z}) = \langle A, B \mid A^2 = B^3 = 1 \rangle$$

with A and B being the following two matrices,

$$A = \pm \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \quad \text{and} \quad B = \pm \begin{pmatrix} 1 & 1 \\ -1 & 0 \end{pmatrix}$$

for it renders explicit the following isomorphism,

$$\mathrm{PSL}_2(\mathbb{Z}) \simeq \mathbb{Z}/2\mathbb{Z} * \mathbb{Z}/3\mathbb{Z}$$

4.1. Displacements Groups. The modular group acts naturally on the set of edges of any trivalent diagrams. This action is generated by the two *elementary moves*,

$$a \cdot A = a +_{\circ} 1 \quad \text{and} \quad a \cdot B = a +_{\bullet} 1$$

The elementary move A acts by exchanging positions of the two adjacent edges of any bivalent white vertex and by fixing the only adjacent edge of any univalent white vertex. Similarly, the elementary move B acts by cyclically exchanging the three adjacent edges of any trivalent black vertex and by fixing the only adjacent edge to any univalent black vertex.



Figure 3. Here is in picture the result of the action of the two elementary moves A and B on the various sorts of edges.

Given any trivalent diagram Γ , the two elementary moves just described generate a group Φ_Γ called the *displacement group* of Γ . It is easily verified that it is the quotient group of $\mathrm{PSL}_2(\mathbb{Z})$ by the kernel of the group action $\rho : \mathrm{PSL}_2(\mathbb{Z}) \rightarrow \mathfrak{S}_{\Gamma_-}$. The modular group has therefore a universal status with respect to that construction, it can be considered as the *universal* group of displacements for the species of trivalent diagrams. If one restricts attention to finite trivalent diagrams, the profinite completion of $\mathrm{PSL}_2(\mathbb{Z})$ would be a more appropriate candidate for that purpose.

4.2. Unrooted Planar Binary Trees. One can associate to any (unrooted) planar binary tree T a connected and acyclic trivalent diagram Γ , called its *enriched barycentric subdivision* $\Gamma = T^{sb+}$, by putting an extra white vertex in the middle of every edges of T . The set of directed edges of T and that of undirected edges of Γ are in bijection in two natural ways.

There is another famous presentation of the modular group, it is given by two generators S and T and two relations as follows,

$$\mathrm{PSL}_2(\mathbb{Z}) = \langle S, T \mid S^2 = (ST)^3 = 1 \rangle$$

with S and T being the following two matrices,

$$S = \pm \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \quad \text{and} \quad T = \pm \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

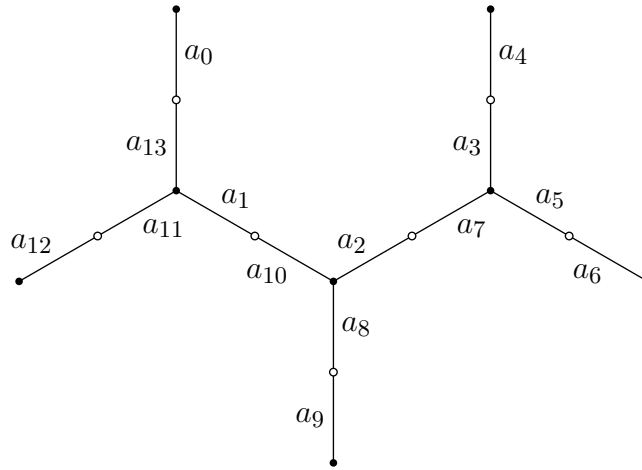


Figure 4. We see on this example the result of iterating the elementary move T on the edges of a binary tree. The edges are labeled by a_k where $a_{k+1} = T \cdot a_k$. This can be used to implement depth-first traversals in a purely iterative way.

The conversion between the two presentations is done through the application of the following rules,

$$\begin{array}{ll} A \rightarrow S & S \rightarrow A \\ B \rightarrow (ST)^2 & T \rightarrow AB^{-1} \end{array}$$

Here are two basic criterion relating *connectedness* and *acyclicity* of finite trivalent diagrams to the transitivity of the action of some displacement group :

- 1) A finite trivalent diagram Γ is *connected* if and only if, its displacement group Φ_Γ acts transitively on its set of edges.
- 2) If it is a *tree* then the subgroup Ψ_Γ , of its displacements generated by the elementary move T , acts transitively on its set of edges.

There is a natural bijection between trivalent diagrams having no univalent white vertex and those having no univalent black vertex. It works by removing every univalent black vertex and the adjacent edges in one direction and by growing every univalent white vertex with a new edge and a new univalent black vertex in the other direction. This bijection is compatible with *connectedness* and *acyclicity* thus restricts to the class of planar binary trees and we recover the classical bijection between complete and incomplete planar binary trees.

4.3. Classification Principle. To any connected pointed trivalent diagram, one can moreover associate the subgroup of the $\text{PSL}_2(\mathbb{Z})$ consisting of elements that are

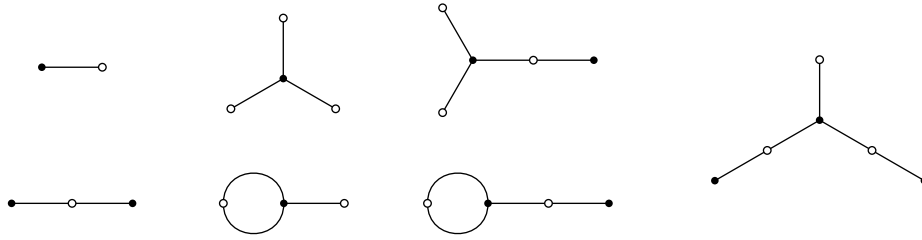


Table 1. Trivalent diagrams of size up to five.

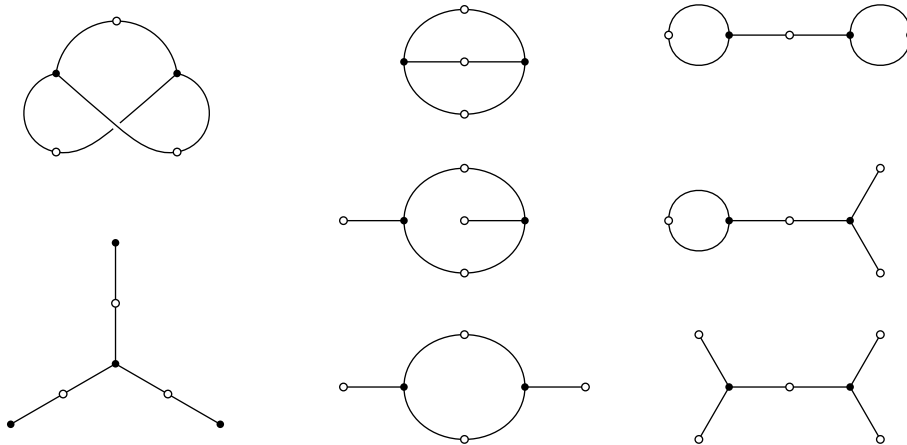


Table 2. Trivalent diagrams of size six.

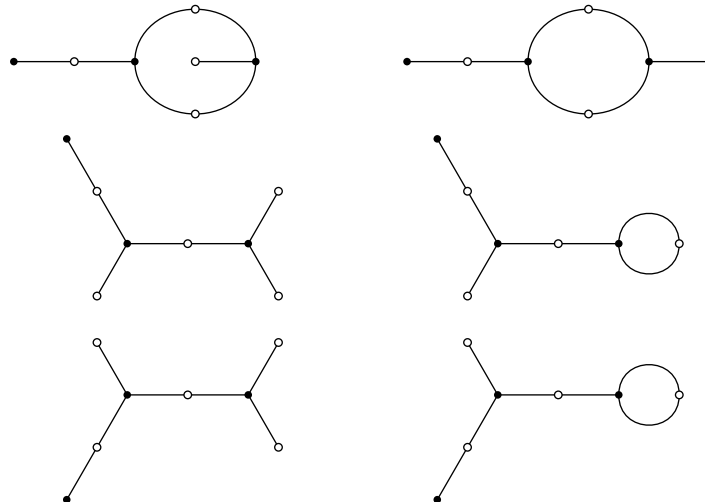


Table 3. Trivalent diagrams of size seven.

fixing the distinguished edge of the diagram. We proved in [27] that this correspondence is one to one and we gave a counting in the form of a generating series that agree perfectly with the number of structures generated by algorithm 3.

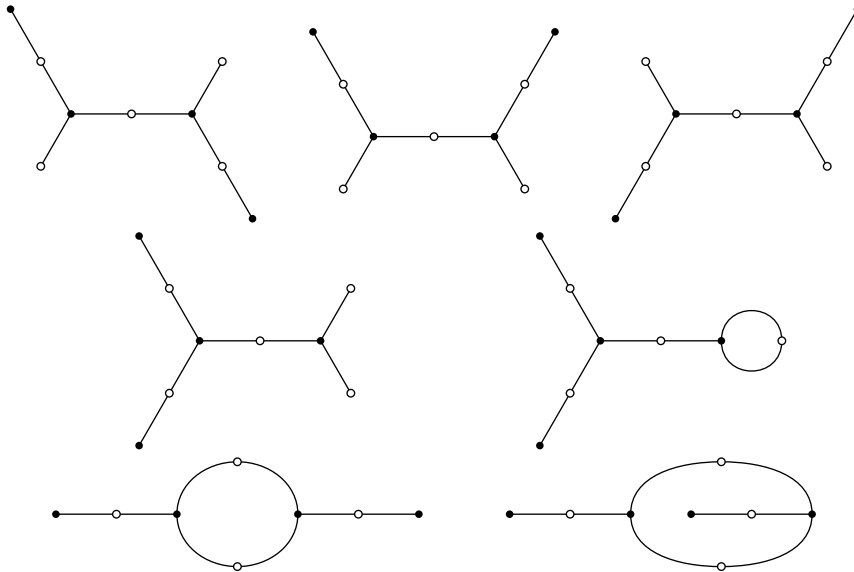


Table 4. Trivalent diagrams of size eight.

Now, if one changes the distinguished edge of a pointed trivalent diagram, the corresponding subgroups gets conjugated. We have moreover proved that two subgroups in the modular group are conjugated if and only if the associated pointed trivalent diagrams only differ by the position of their distinguished edges. It follows that the unpointed trivalent diagrams correspond in a one to one fashion to the conjugacy classes of subgroups of the modular group.

We gave in [27] the exhaustive list of trivalent diagram of size up to nine. That list was computed by hand in a non-systematic fashion. One intent of algorithm 3 of section 3.1 is to permit a retroactive validation of both those tables and the associated generating series that we reproduce here in tables 6 and 7, they are parts of the online encyclopedia of integer sequences [20] under the references (A005133) and (A121350).

5. SECOND APPLICATION : TRIANGULAR MAPS

By an (oriented) *triangular map* we mean a finite polyhedral structure composed of vertex, directed edges, and oriented triangular faces with an adjacency relation among them. Suggestively enough a directed edge, also called an *arc*, is bordered by an ordered pair of vertices, we call its *origin* and its *destination* respectively, such that triangular faces are each bordered by a cycle of three arcs whose destination of whom coincide with the origin of their following arc in cyclic order.

The following definition is useful in grasping the adjacency relations of a combinatorial map but insufficient because lacking some traversal informations such as the cyclic orientation of the faces.

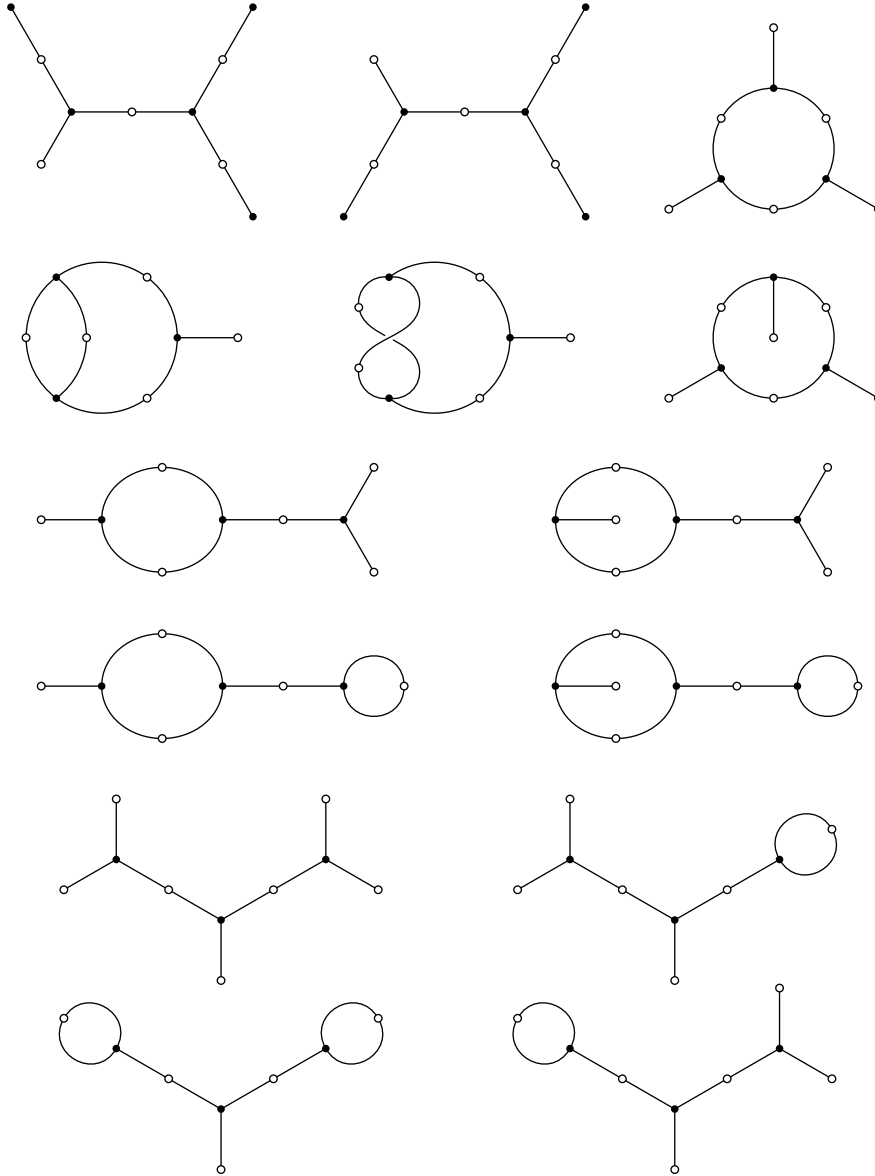


Table 5. Trivalent diagrams of size nine.

Definition 5.1. A *combinatorial pre-map* Λ is given by three sets Λ_0 , Λ_1 and Λ_2 and five applications $s, t : \Lambda_1 \rightarrow \Lambda_0$, $\ell, r : \Lambda_1 \rightarrow \Lambda_2$ and $\cdot^{-1} : \Lambda_1 \rightarrow \Lambda_1$ satisfying the following conditions for all elements a of Λ_1 ,

$$\begin{array}{lll}
 s(a^{-1}) = t(a) & \ell(a^{-1}) = r(a) & (a^{-1})^{-1} = a \\
 t(a^{-1}) = s(a) & r(a^{-1}) = \ell(a) & a^{-1} \neq a
 \end{array}$$

the four maps s , t , ℓ and r are further assumed to be *surjective*.

The elements of the tree sets Λ_0 , Λ_1 , and Λ_2 are the *vertices*, the *arcs* (directed edges), and *faces* of the combinatorial map respectively. The two applications s and t map any arc a to its *origin* $s(a)$ and *destination* $t(a)$. The two applications ℓ and

$$\begin{aligned}
\tilde{D}_3^\bullet(t) = & t + t^2 + 4t^3 + 8t^4 + 5t^5 + 22t^6 + 42t^7 + 40t^8 + 120t^9 + 265t^{10} + 286t^{11} \\
& + 764t^{12} + 1729t^{13} + 2198t^{14} + 5168t^{15} + 12144t^{16} + 17034t^{17} + 37702t^{18} \\
& + 88958t^{19} + 136584t^{20} + 288270t^{21} + 682572t^{22} + 1118996t^{23} \\
& + 2306464t^{24} + 5428800t^{25} + 9409517t^{26} + 19103988t^{27} + 44701696t^{28} \\
& + 80904113t^{29} + 163344502t^{30} + 379249288t^{31} + 711598944t^{32} \\
& + 1434840718t^{33} + 3308997062t^{34} + 6391673638t^{35} + 12921383032t^{36} \\
& + 29611074174t^{37} + 58602591708t^{38} + 119001063028t^{39} \\
& + 271331133136t^{40} + 547872065136t^{41} + 1119204224666t^{42} \\
& + 2541384297716t^{43} + 5219606253184t^{44} + 10733985041978t^{45} \\
& + 24300914061436t^{46} + 50635071045768t^{47} + 104875736986272t^{48} \\
& + 236934212877684t^{49} + 499877970985660t^{50} + o(t^{50})
\end{aligned}$$

Table 6. Order fifty development of the generating series $\tilde{D}_3^\bullet(t)$ giving as the coefficient of t^n the number of connected pointed trivalent diagrams with n edges (A005133) which is also the number of index n subgroups in the modular group $\text{PSL}_2(\mathbb{Z})$.

$$\begin{aligned}
\tilde{D}_3(t) = & t + t^2 + 2t^3 + 2t^4 + t^5 + 8t^6 + 6t^7 + 7t^8 + 14t^9 + 27t^{10} + 26t^{11} \\
& + 80t^{12} + 133t^{13} + 170t^{14} + 348t^{15} + 765t^{16} + 1002t^{17} + 2176t^{18} \\
& + 4682t^{19} + 6931t^{20} + 13740t^{21} + 31085t^{22} + 48652t^{23} + 96682t^{24} \\
& + 217152t^{25} + 362779t^{26} + 707590t^{27} + 1597130t^{28} + 2789797t^{29} \\
& + 5449439t^{30} + 12233848t^{31} + 22245655t^{32} + 43480188t^{33} \\
& + 97330468t^{34} + 182619250t^{35} + 358968639t^{36} + 800299302t^{37} \\
& + 1542254973t^{38} + 3051310056t^{39} + 6783358130t^{40} + 13362733296t^{41} \\
& + 26648120027t^{42} + 59101960412t^{43} + 118628268978t^{44} \\
& + 238533003938t^{45} + 528281671324t^{46} + 1077341937144t^{47} \\
& + 2184915316390t^{48} + 4835392099548t^{49} + 9997568771074t^{50} + o(t^{50})
\end{aligned}$$

Table 7. Order fifty development of the generating series $\tilde{D}_3(t)$ giving as the coefficient of t^n the number of connected unpointed trivalent diagrams with n edges (A121350) which is also the number of conjugacy classes of index n subgroup in the modular group $\text{PSL}_2(\mathbb{Z})$.

r map any arc a to its *left-hand face* $\ell(a)$ and *right-hand face* $r(a)$. Finally, the application \cdot^{-1} maps any arc a to its *inverse* a^{-1} obtained by reversing its direction.

Definition 5.2. A *morphism* φ between two combinatorial pre-maps Λ and Λ' is a triple of applications $\varphi_0 : \Lambda_0 \rightarrow \Lambda'_0$, $\varphi_1 : \Lambda_1 \rightarrow \Lambda'_1$ and $\varphi_2 : \Lambda_2 \rightarrow \Lambda'_2$ compatible to the five structure applications in the sense that the following diagrams are commutatives.

$$\begin{array}{ccc}
\Lambda_1 & \xrightarrow{\varphi_1} & \Lambda'_1 \\
\downarrow s,t & & \downarrow s,t \\
\Lambda_0 & \xrightarrow{\varphi_0} & \Lambda'_0
\end{array}
\quad
\begin{array}{ccc}
\Lambda_1 & \xrightarrow{\varphi_1} & \Lambda'_1 \\
\downarrow \ell,r & & \downarrow \ell,r \\
\Lambda_2 & \xrightarrow{\varphi_2} & \Lambda'_2
\end{array}
\quad
\begin{array}{ccc}
\Lambda_1 & \xrightarrow{\varphi_1} & \Lambda'_1 \\
\downarrow \cdot^{-1} & & \downarrow \cdot^{-1} \\
\Lambda_1 & \xrightarrow{\varphi_1} & \Lambda'_1
\end{array}$$

5.1. Cyclic Orientation. In a given combinatorial pre-map Λ , the *inner border* of a face f is the set $\ell^{-1}(f) = \{a \in \Lambda_1 \mid \ell(a) = f\}$ of arcs having f as their left-hand face. A combinatorial pre-map is said to be *strictly triangular* if each of its faces has exactly three arcs in its inner border. The following definition describes the traversal information lacking to a triangular combinatorial pre-map to fully describe a triangular map.

Definition 5.3. A (*strictly*) *triangular map* Λ is a strictly triangular combinatorial pre-map together with a $\mathbb{Z}/3\mathbb{Z}$ -action $\Lambda_1 \times \mathbb{Z}/3\mathbb{Z} \rightarrow \Lambda_1$ given by $(a, n) \rightarrow a + n$ acting simply and transitively on the inner-border of every faces.

Definition 5.4. A *morphism* φ between two triangular maps Λ and Λ' is a morphism of the underlying combinatorial pre-maps with its φ_1 component further assumed to be equivariant with respect to the corresponding $\mathbb{Z}/3\mathbb{Z}$ -actions, meaning $\varphi_1(a+1) = \varphi_1(a) + 1$ for all arc a of Λ .

5.2. Associated Trivalent Diagram. The *adjacency diagram* of a triangular map Λ is the trivalent diagram, denoted Λ^{adj} , which sets of white vertices, black vertices and edges are the following,

$$\Lambda_{\circ}^{adj} = \{\mathbf{a}_a\}_{a \in \Lambda_1^*} \quad \Lambda_{\bullet}^{adj} = \{\mathbf{b}_f\}_{f \in \Lambda_2} \quad \Lambda_{-}^{adj} = \{\mathbf{c}_a\}_{a \in \Lambda_1}$$

where Λ_1^* is the set of undirected edges of Λ , and which structure applications $\partial_{\circ} : \Lambda_{-}^{adj} \rightarrow \Lambda_{\circ}^{adj}$, $\partial_{\bullet} : \Lambda_{-}^{adj} \rightarrow \Lambda_{\bullet}^{adj}$ and group actions $+_{\circ}, +_{\bullet} : \Lambda_{-}^{adj} \times \mathbb{Z} \rightarrow \Lambda_{-}^{adj}$ are defined by the following relations,

$$\begin{array}{ll}
\partial_{\circ}(\mathbf{c}_a) = \mathbf{a}_{\pi(a)} & \mathbf{c}_a +_{\circ} 1 = \mathbf{c}_{a^{-1}} \\
\partial_{\bullet}(\mathbf{c}_a) = \mathbf{b}_{\ell(a)} & \mathbf{c}_a +_{\bullet} 1 = \mathbf{c}_{a+1}
\end{array}$$

This operation is *functorial* for it is easily extended to morphisms of triangular maps by the following process. To any morphism φ between two triangular maps Λ and Λ' , we associate a morphism denoted φ^{adj} between the corresponding adjacency diagrams Λ^{adj} and $(\Lambda')^{adj}$ defined by the following relations,

$$\varphi_{\circ}^{adj}(\mathbf{a}_a) = \mathbf{a}'_{\varphi_1^*(a)} \quad \varphi_{\bullet}^{adj}(\mathbf{b}_f) = \mathbf{b}'_{\varphi_2(f)} \quad \varphi_{-}^{adj}(\mathbf{c}_a) = \mathbf{c}'_{\varphi_1(a)}$$

Functoriality should be obvious by careful inspection.

The \cdot^{adj} functor we get by what precedes is full and faithful, but not essentially surjective because trivalent diagrams we get as the adjacency diagram of a trivalent

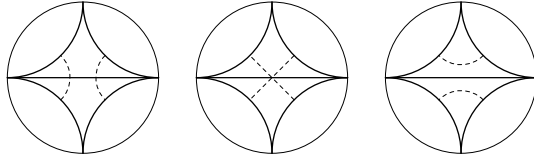


Table 8. The three triangular maps with two faces.

map Λ have no univalent white vertex nor univalent black vertex. Let's call *regular* a trivalent diagram having no univalent vertex, the \cdot^{adj} functor is essentially surjective on the full subcategory of regular trivalent diagrams, and so,

Theorem 5.1. *The \cdot^{adj} functor realize an equivalence of categories between the category of triangular maps and the full subcategory of regular trivalent diagrams.*

Demonstration. To prove this theorem we shall describe a *reconstruction* operation, which associate to any regular trivalent diagram Γ a triangular map denoted Γ^{map} , with functorial property and such that for all regular trivalent diagrams Γ and triangular maps Λ , one have two natural *reciprocity isomorphisms* as follows,

$$(\Gamma^{map})^{adj} \underset{nat.}{\simeq} \Gamma \quad \text{and} \quad (\Lambda^{adj})^{map} \underset{nat.}{\simeq} \Lambda$$

We shall first introduce some notations. Let's call Ψ_Γ the subgroup of Φ_Γ generated by the elementary move T (cf. section 4.2) and lets call $\pi : \Gamma_- \rightarrow \Gamma_-/\Psi_\Gamma$ the natural projection. The application induced between Γ_-/Ψ_Γ and Γ'_-/Ψ_Γ by an equivariant map $\varphi : \Gamma_- \rightarrow \Gamma'_-$ will be denoted φ_Ψ . The sets of vertices, edges and faces of the reconstructed map are the following,

$$\Gamma_0^{map} = \{ \mathfrak{d}_x \}_{x \in \Gamma_-/\Psi_\Gamma} \quad \Gamma_1^{map} = \{ \mathfrak{e}_a \}_{a \in \Gamma_-} \quad \Gamma_2^{map} = \{ \mathfrak{f}_y \}_{y \in \Gamma_\bullet}$$

The five structure maps $s, t : \Gamma_1^{map} \rightarrow \Gamma_0^{map}$, $r, \ell : \Gamma_1^{map} \rightarrow \Gamma_2^{map}$ and $\cdot^{-1} : \Gamma_1^{map} \rightarrow \Gamma_1^{map}$ and the group action $+$: $\Gamma_1^{map} \times \mathbb{Z} \rightarrow \Gamma_1^{map}$ of the reconstructed map are given by the following equations,

$$\begin{aligned} s(\mathfrak{e}_a) &= \mathfrak{d}_{\pi(a)} & \ell(\mathfrak{e}_a) &= \mathfrak{f}_{\partial_\bullet(a)} & \mathfrak{e}_a^{-1} &= \mathfrak{e}_{a+\circ 1} \\ t(\mathfrak{e}_a) &= \mathfrak{d}_{\pi(a+\circ 1)} & r(\mathfrak{e}_a) &= \mathfrak{f}_{\partial_\bullet(a^{-1})} & \mathfrak{e}_a + 1 &= \mathfrak{e}_{a+\bullet 1} \end{aligned}$$

The construction then extends to morphisms in the sense that any morphism φ between to regular trivalent diagrams Γ and Γ' induces a morphism φ^{map} between the two reconstructed maps Γ^{map} and $(\Gamma')^{map}$ which three components are the following,

$$\varphi_0^{map}(\mathfrak{d}_x) = \mathfrak{d}_{\varphi_-, \Psi(x)} \quad \varphi_1^{map}(\mathfrak{e}_a) = \mathfrak{e}_{\varphi_-(a)} \quad \varphi_2^{map}(\mathfrak{f}_y) = \mathfrak{f}_{\varphi_\bullet(y)}$$

The functoriality of the reconstruction operation and the two reciprocity isomorphisms should be clear by careful inspection. End of the demonstration. \square

Remark. The content of the above theorem is nothing but a specific notion of *Poincaré duality*.

5.3. Exhaustive Generation of Triangular Maps. To adapt the generator algorithm to produce only *regular* pointed trivalent diagrams and thus *pointed triangular maps*, it suffice to remove the call to algorithms 8 and 7 from line 6 and 7 of the DISPATCH procedure (algorithm 4) and the lines 2 to 5 from algorithm 3. Those

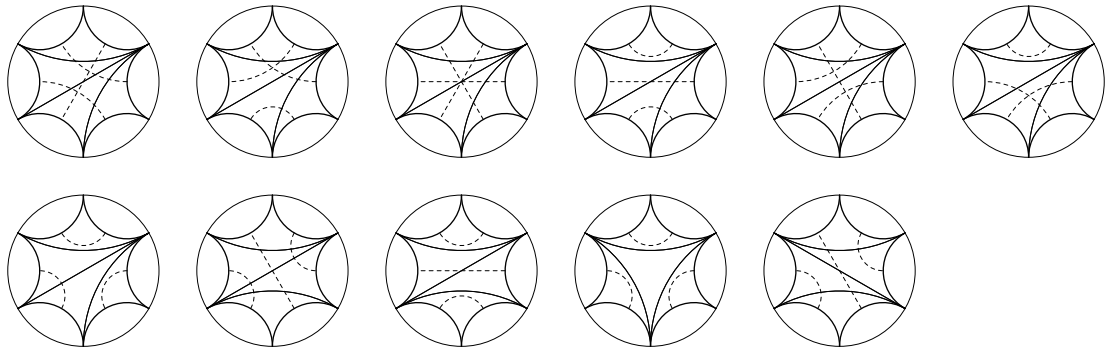


Table 9. The eleven triangular maps with four faces.

two removals preserve the CAT property, we thus get this way a constant amortized time generator for pointed triangular maps, as announced.

The tables 8 to 11 shows exhaustive lists of *unpointed* triangular maps produced from the output of the generator algorithm. Those tables arise from a two steps process. First, generate the full list of pointed regular diagrams of a given size, then, remove the duplicated diagrams from the list one obtain by forgetting the base point. An easy and efficient way to do that is to put a linear order on the set of pointed trivalent diagrams of a given size, then to remove from the list the trivalent diagrams that are not minimal in their conjugacy class (two pointed diagrams being *conjugated* by definition, when they differ only by the position of their base point).

The interpretation of the drawings of tables 8 to 11 deserves some explanations. For that purpose let's adopt a geographical terminology. Forgetting for a moment the surrounding circle and the dashed lines of the drawings, the triangular regions are called the *countries* of the maps, the plain lines are the *boundaries* of their adjacent countries. One can distinguish the boundaries that are bordering two distinct countries (the *inner boundaries*) from those that are bordering a single country (the *outer boundaries*). The roads of the maps are symbolized by dashed lines connecting, in a two by two fashion, the outer boundaries of the maps.

To produce them, we have considered the planar rooted binary tree of the depth-first traversal of algorithm 1. As already noted, this algorithm provides natural cuttings for the associated trivalent diagrams. Those cuttings arise as what we previously called *backward connections*. In contrast, the edges of the traversal tree correspond to what we previously called *forward connections*. In the graphical representation, we use an embedding of the produced triangulated polygon in the Poincaré disc model of the hyperbolic plane as it seems the natural setting for generic non-overlapping triangular tilings. The surrounding circle around each figure is of course irrelevant to the structures.

The tables 12 and 13 give the number of pointed triangular maps and unpointed triangular maps in the form of generating series. They are also part of [20] under the references (A062980) and (A129114). Their computation is very similar to that

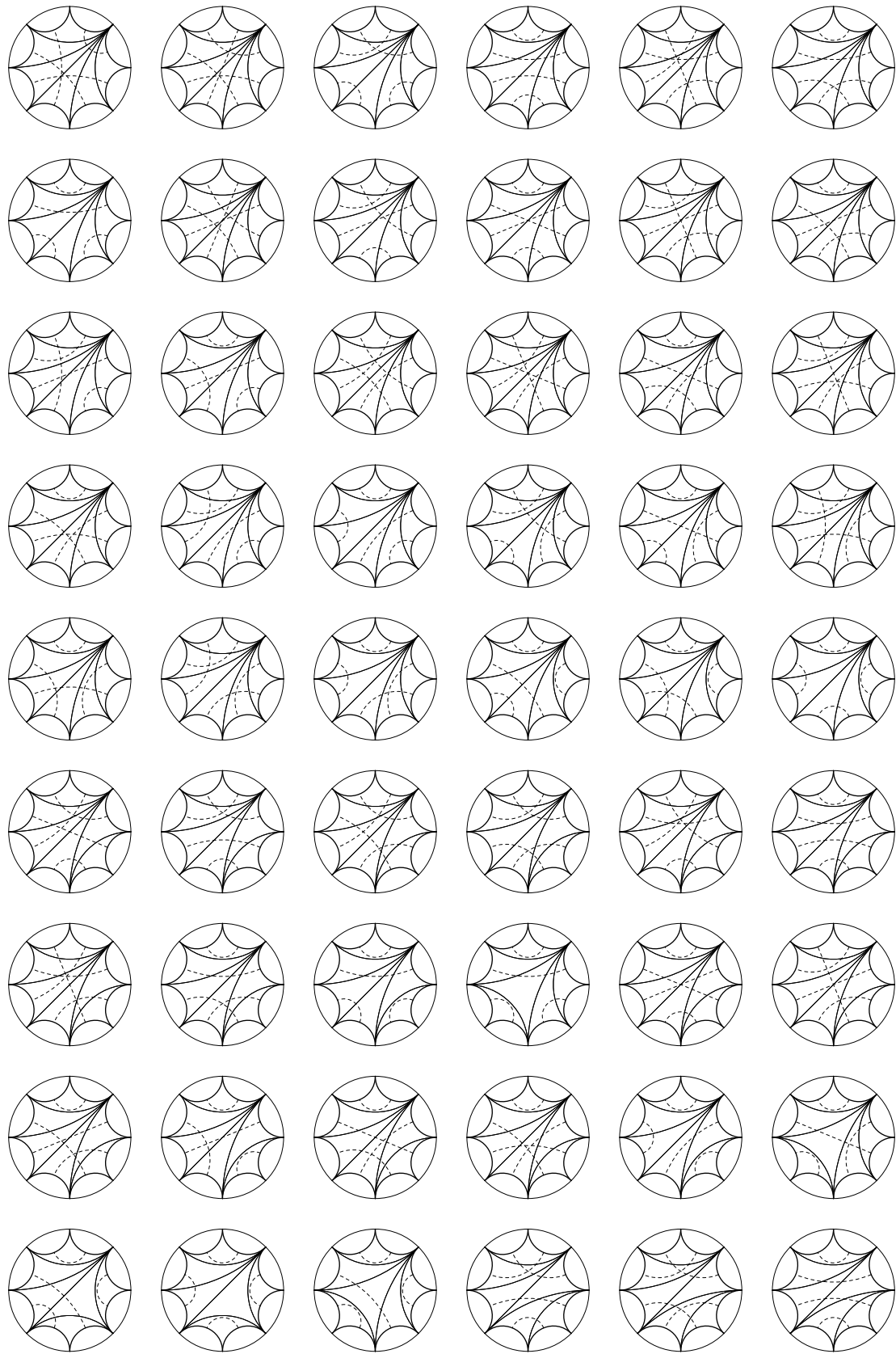


Table 10. The eighty one triangular maps with six faces (first part).

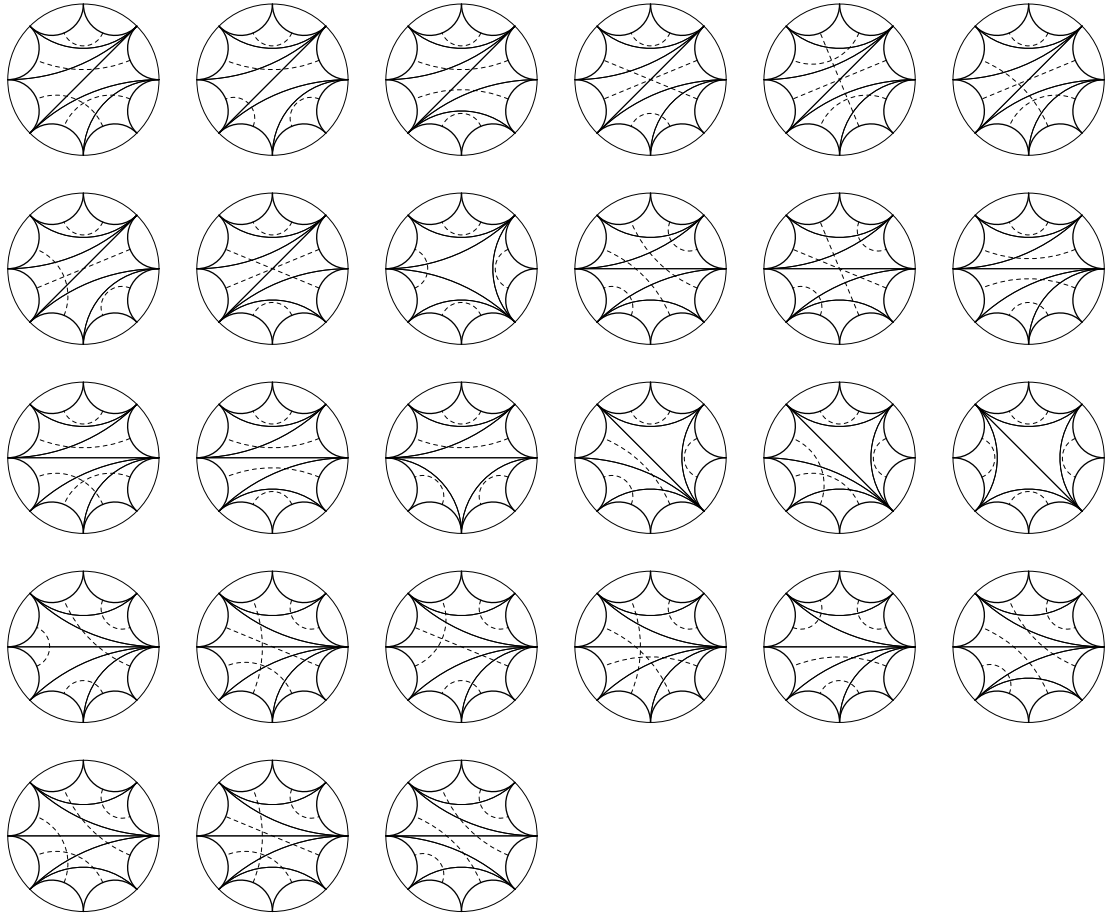


Table 11. The eighty one triangular maps with six faces (last part, continued from table 10).

of tables 6 and 7 which is explained in detail in [27]. We shall explain in [26] in a unified fashion how one can compute generating series for pointed and unpointed unlabeled maps of various kind and in [25] the unexpected relation of this sequence to the asymptotics of the Airy function.

As another byproduct of the the exhaustive list obtained from the generating algorithm, one can get the precise number of pointed and unpointed triangular map having a given genus and a given number of triangular faces. Tables 14 and 15 summarize those results for small number of faces. Recently, M. Krikun [12] kindly communicated us recurrence relations satisfied by the entries of table 14 which he obtained by a clever recursive decomposition of pointed triangular maps. Those recurrence relations make it possible to evaluate easily those numbers without running the generator algorithm.

Some lines of those two tables were previously known. For instance, the first line of table 14 is the number of spherical rooted triangular maps by the number of its faces [14]. The first line of table 15 is its unpointed counterpart. It is computed by impressive closed formulae in a recent paper by Liskovets, Gao and Wormald [24].

$$\begin{aligned}
\tilde{T}_3^\bullet(t) = & 5t^6 + 60t^{12} + 1105t^{18} + 27120t^{24} + 828250t^{30} + 30220800t^{36} \\
& + 1282031525t^{42} + 61999046400t^{48} + 3366961243750t^{54} \\
& + 202903221120000t^{60} + 13437880555850250t^{66} + 970217083619328000t^{72} \\
& + 75849500508999712500t^{78} + 6383483988812390400000t^{84} \\
& + 575440151532675686278125t^{90} + 55318762960656722780160000t^{96} \\
& + 5649301494178851172304968750t^{102} \\
& + 610768380520654474629120000000t^{108} \\
& + 69692599846542054607811528918750t^{114} \\
& + 8370071726919812448859648819200000t^{120} + o(t^{120})
\end{aligned}$$

Table 12. Development of the generating series $\tilde{T}_3^\bullet(t)$, up to order a hundred twenty. It gives as the coefficient of t^{6n} the number of connected unpointed unlabeled triangular maps with n arcs, thus $n/2$ undirected edges and $n/3$ triangular faces (A062980). If we note a_n that coefficient, the recurrence is as follows : $a_1 = 5$ and for $n \geq 1, a_{n+1} = (6n + 6)a_n + \sum_{k=1}^{n-1} a_k a_{n-k}$.

$$\begin{aligned}
\tilde{T}_3(t) = & 3t^6 + 11t^{12} + 81t^{18} + 1228t^{24} + 28174t^{30} + 843186t^{36} + 30551755t^{42} \\
& + 1291861997t^{48} + 62352938720t^{54} + 3381736322813t^{60} \\
& + 203604398647922t^{66} + 13475238697911184t^{72} + 972429507963453210t^{78} \\
& + 75993857157285258473t^{84} + 6393779463050776636807t^{90} \\
& + 576237114190853665462712t^{96} + 55385308766655472416299110t^{102} \\
& + 5655262782600929403228668176t^{108} \\
& + 611338595145132827847686253456t^{114} \\
& + 69750597724332100283681465962492t^{120} + o(t^{120})
\end{aligned}$$

Table 13. Order a hundred twenty development of the generating series $\tilde{T}_3(t)$ giving the number of connected unpointed unlabeled triangular maps with n arcs, thus $n/2$ undirected edges and $n/3$ triangular faces (A129114).

The diagonal terms of those two tables also received close attention. For instance in [9] Harer and Zagier computed the Euler-Wall characteristic of the mapping class group of once pointed genus g closed oriented surfaces by a remarkable combinatorial reduction of the problem in which pointed combinatorial maps with one vertex are counted by genus yielding the diagonal sequence of the first table 1, 105, 50050, 56581525, The diagonal sequence of the second table : 1, 9, 172, 1349005, gives the number of unpointed triangular maps of genus g with only one vertex. It has been studied at depth in the article [13] by A. Vdovina and R. Bacher.

	2	4	6	8	10	12	14
0	4	32	336	4096	54912	786432	11824384
1	1	28	664	14912	326496	7048192	150820608
2	0	0	105	8112	396792	15663360	544475232
3	0	0	0	0	50050	6722816	518329776
4	0	0	0	0	0	0	56581525

Table 14. The number of *pointed* triangular maps by genus (horizontally) and number of faces (vertically).

	2	4	6	8	10	12	14
0	2	6	26	191	1904	22078	282388
1	1	5	46	669	11096	196888	3596104
2	0	0	9	368	13448	436640	12974156
3	0	0	0	0	1726	187580	12350102
4	0	0	0	0	0	0	1349005

Table 15. The number of *unpointed* triangular maps by genus (horizontally) and number of faces (vertically).

6. CONCLUDING REMARKS AND PERSPECTIVES

The generating algorithm presented in this paper (section 3) may receive trivial adaptations to generate wider classes of diagrams and combinatorial maps, possibly with prescribed degree lists for vertices or faces. Basically, it can be simply generalized to produce any connected pair of permutations with prescribed cyclic types, up to simultaneous conjugacy.

Another way to extend the study, would be to modify the DISPATCH procedure (algorithm 4 of section 3) to generate not an exhaustive cover of the partial cases, but instead a single case of them picked at random. This would result, in a fairly straightforward fashion, in a random sampler algorithm of the corresponding combinatorial structures instead of an exhaustive generator. The difficulty there, is to precompute precise conditional probability tables in order to control the probability distribution of the generated structures by bayesian techniques.

Such tables of conditional probabilities could be computed with the help of generating series techniques, namely by following the particular recursive structure of the algorithm and translating this recursive structure in functional equations on the generating series. This appeal for a further investigation and will be dealt with in a subsequent paper.

7. ACKNOWLEDGEMENTS

I am grateful to professors D. Bar Nathan, P. Flajolet, F. Hivert, M. Huttner, M. Krikun, M. Petitot, B. Salvy, G. Shaeffer, N. Thiery, and D. Zvonkine for useful discussions and warm encouragements.

REFERENCES

- [1] R. Nedela A. Mednykh. Enumeration of unrooted maps of a given genus. *J. Comb. Theory, Ser. B*, 96(5):706–729, 2006.
- [2] D. Bar-Nathan. On associators and the grothendieck-teichmuller group I. *Selecta Mathematica, New Series*, 4:183–212, 1998.
- [3] R. Cori. *Un code pour les graphes planaires et ses applications*, volume 27 of *Astérisque*. Société Mathématique de France, 1975.
- [4] V. G. Drinfel’d. Quasi-hopf algebras. *Leningrad Math. J.*, 1:1419–1457, 1990.
- [5] V. G. Drinfel’d. On quasitriangular quasi-hopf algebras and a group closely connected with $\text{Gal}(\bar{\mathbb{Q}}/\mathbb{Q})$. *Leningrad Math. J.*, 2:829–860, 1991.
- [6] P. Leroux F. Bergeron, G. Labelle. *Théorie des espèces et combinatoire des structures arborescentes*. LACIM Montréal, 1994.
- [7] P. Leroux F. Bergeron, G. Labelle. *Combinatorial Species and Tree-like Structures*. Cambridge University Press, 1998. English edition of [6].
- [8] A. Grothendieck. Esquisse d’un programme. In P. Lochak L. Schneps, editor, *Geometric Galois Actions Vol. I*, number 242 in London Math. Soc. Lecture Notes, pages 5–48. Cambridge Univ. Press, 1997.
- [9] D. Zagier J. Harer. The euler characteristic of the moduli space of curves. *Invent. Math.*, 85:457–486, 1986.
- [10] A. Joyal. Une théorie combinatoire des séries formelles. *Adv. Math.*, 42:1–82, 1981.
- [11] M. Kontsevich. Intersection theory on the moduli space of curves and the matrix airy function. *Commun. Math. Phys.*, 147:1–23, 1992.
- [12] M. Krikun. Enumeration of triangulations by genus (incomplete draft). Private communication, 2007.
- [13] A. Vdovina R. Bacher. Counting 1-vertex triangulations of oriented surfaces. *Discrete Math.*, 246(1-3):13–27, 2002.
- [14] P. J. Schellenberg R. C. Mullin, E. Nemeth. The enumeration of almost cubic maps. In R. C. Mullin et al., editor, *Proceedings of the Louisiana Conference on Combinatorics*, volume 1 of *Graph Theory and Computer Science*, pages 281–295, 1970.
- [15] A. Machì R. Cori. Maps, hypermaps and their automorphisms : a survey I, II, III. *Expos. Math.*, 10(5):403–427, 429–447, 449–467, 1992.
- [16] A. Douady R. Douady. *Algèbre et théories galoisiennes*. Cassini, France, 2003.
- [17] L. Schneps. Dessins d’enfants on the Riemann sphere. In P. Lochak L. Schneps, editor, *The Grothendieck Theory of Dessins d’Enfant*, number 200 in London Math. Soc. Lecture Notes, pages 5–48. Cambridge Univ. Press, 1994.
- [18] J.P. Serre. *Trees*. Springer Monographs in Math. Springer-Verlag, 2003.
- [19] A.K. Zvonkine S.K. Lando. *Graphs on Surfaces and Their Applications*. Springer-Verlag, 2004.
- [20] N. J. A. Sloane. The on-line encyclopedia of integer sequences. Available on the net at : <http://www.research.att.com/~njas/sequences/>, 2005.
- [21] A. B. Lehman T. R. S. Walsh. Counting rooted maps by genus. *J. Comb. Th.*, 13:122–141 and 192–218, 1972.
- [22] A. B. Lehman T. R. S. Walsh. Counting rooted maps by genus III. *J. Comb. Th.*, 18:222–259, 1975.
- [23] W. Thurston. Three-dimensional manifolds, kleinian groups and hyperbolic geometry. *Bull. Amer. Math. Soc., New Series*, 6:357–381, 1982.

- [24] N. Wormald V. A. Liskovets, Z. Gao. Enumeration of unrooted odd-valent regular planar maps. To appear, 2005.
- [25] S. A. Vidal. Asymptotics of the airy function and triangular maps decomposition. In preparation.
- [26] S. A. Vidal. Multiparametric enumeration of unrooted combinatorial maps. In preparation.
- [27] S. A. Vidal. Sur la classification et le dénombrement des sous-groupes du groupe modulaire et de leurs classes de conjugaison. *Pub. IRMA, Lille*, 66(II):1–35, 2006. Preprint : <http://arxiv.org/abs/math.CO/0702223>.
- [28] T. R. S. Walsh. *Combinatorial Enumeration of Non-Planar Maps*. PhD thesis, Univ. of Toronto, 1971.
- [29] T. R. S. Walsh. Generating nonisomorphic maps without storing them. *SIAM Journal on Algebraic and Discrete Methods*, 4(2):161–178, 1983.