Combinatorics lecture notes (I).
Combinatorial maps and compact data structures

Gilles Schaeffer
http://www.lix.polytechnique.fr/Labo/Gilles.Schaeffer

Version of December 5, 2006

Preliminary remarks

These notes roughly correspond to the material of my 3 first lectures. The aim of these lectures is to present different directions of research in combinatorics, with enumeration and planarity as a common flavor.

Pictures are missing in this version of the notes... too bad for those who missed the lectures. There are probably also many inaccuracies (let's say errors...), please indicate them to me so that the next version improve.

1 First lecture : combinatorial maps

1.1 Permutations, cycle types and conjugacy classes

The symmetric group S_n, or group of permutations on $[n] = \{1, \ldots, n\}$, is the set of permutations of $[n]$ endowed with the composition product: the permutation $\sigma\rho$ is the permutation such that $(\sigma\rho)(i) = \sigma(\rho(i))$. Let us denote by $c(\sigma)$ the number of cycles of a permutation σ, and by $\varepsilon(\sigma)$ its parity $n - c(\sigma) \mod 2$ (a permutation σ is said *even* if $\varepsilon(\sigma) = 0$, *odd* otherwise). A permutation has *cyclic type* $1^{n_1}2^{n_2}\ldots k^{n_k}$ if it has n_i cycles of length i for all $i = 1, \ldots, k$.

Exercice 1 *Show that*

- *the number of transpositions (with cyclic type* $1^{n-2}2$*) is* $\binom{n}{2}$,
- *the number of circular permutations (with cyclic type* n*) is* $(n-1)!$,
- *the number of fix point free involutions (with c.t.* 2^m*, with* $n = 2m$*) is* $(2m-1)!! = \frac{(2m)!}{m!2^m}$.

More generally show that the number of permutations with cyclic type $1^{n_1}2^{n_2}\ldots k^{n_k}$ *is*

$$\frac{n!}{\prod_{i=1}^{k} n_i! i^{n_i}}.$$

Two permutations σ and ρ are said to be *conjugate* if there exists a permutation π such that $\rho = \pi\sigma\pi^{-1}$. The action of π on σ by conjugation is nicely observed on the two row representation of σ or on the cyclic representation (each entry i is replaced by $\pi(i)$). In particular two permutations are conjugate if and only if they have the same cyclic type: the set of permutations with cycle type $\nu = 1^{n_1}2^{n_2}\ldots k^{n_k}$ is denoted $\mathcal{C}_\nu$ and called a *conjugacy class*.

Example: consider the two permutations of $\mathcal{C}_{1^2 2 4}$

$$\sigma \quad = \quad (1,$$

Exercice 2 *Let σ be a permutation and τ be a transposition of S_n. The show that $c(\sigma\tau) = c(\sigma) \pm 1$. More precisely what can be the cyclic type of $\sigma\tau$ if σ has cyclic type ν?*

Exercice 3 *Let σ and ρ be two permutations of S_n. Show that the parities of σ, ρ and $\sigma\rho$ satisfy $\varepsilon(\sigma\rho) \equiv \varepsilon(\sigma) + \varepsilon(\rho) \ mod\ 2$.*

Problem 1 *Consider three cyclic types $\nu = 1^{n_1}2^{n_2}\dots p^{n_p}$, $\mu = 1^{m_1}2^{m_2}\dots q^{m_q}$ and $\lambda = 1^{\ell_1}2^{\ell_2}\dots r^{\ell_r}$. Does there exists a triple of permutations $\sigma\rho$ such that $\sigma \in \mathcal{C}_\nu$, $\rho \in \mathcal{C}_\mu$ and $\sigma\rho \in \mathcal{C}_\lambda$?*

It is an open problem to find a polynomial time algorithm to answer this question or to prove that the problem is NP-complete.

1.2 Chord diagrams and owers

A *chord diagram* is a set of m chords connecting the points $\{1, 2, \dots, 2m\}$ evenly placed on a circle. Chord diagrams can be encoded by fix point free involutions: α is associated to the chord diagram with chords (i, j) iff $j = \alpha(i)$. When chords do not cross each other, the diagram is called *planar*. Equivalently a fix point free involution is called *non-crossing* if α contains no pair of 2-cycles (i, j), (k, ℓ) such that $i < k < j < \ell$. (Convince yourself that α is non-crossing if and only if the associated chord diagram is planar).

Exercice 4 *Show that planar chord diagrams are counted by Catalan numbers (use a decomposition and generating functions, or a bijection with Dyck paths, or a bijection with plane trees).*

Assume that the fix point free involution α encode a planar chord diagram on $\{1, \dots, 2m\}$. Then the faces are described by the product of the involution α and the cycle $\gamma_{2m} = (1, \dots, 2m)$. (How?) What does the product $\alpha\gamma$ yield when the chord diagram is not planar?

Proposition 1 *A chord diagram α is planar if and only if $c(\gamma\alpha) = m + 1$.*

Proof hint: show that there exists i such that $\alpha = (i, i + 1)$ and prove by recurrence.

The *flower representation* of a chord diagram consists in drawing edges outside the disc instead of chords inside. Show that there exists a flower representation such that edges do not intersect if and only if the chord diagram is planar (hint: make a picture on the sphere instead of the plane).

Exercice 5 *A Hamiltonian cubic map consists of a circle with $2m$ points $\{1, \dots, 2m\}$ that are connected two by two by edges that can be either inside or outside the circle. (Strictly speaking one should call this "a cubic map with a distinguished hamiltonian cycle", but this terminology will only be explained later). What is the number of Hamiltonian cubic map with $2m$ vertices? Give an asymptotic equivalent of this number when m goes to infinity.*

Problem 2 *A Hamiltonian cubic map is bipartite if its vertices can be colored with two colors in such a way that each edge connects two vertices with different colors.*

One of the most fascinating currently open asymptotic enumerative problem is the following conjecture: the number of bipartite Hamiltonian cubic maps with $2m$ vertices is equivalent when m goes to infinity to

$$c\rho^m m^{-\frac{1+\sqrt{13}}{6}}$$

for some constants c and ρ.

Exercice 6 *A multimeander is obtained by gluing together two chords diagrams with $2m$ vertices along their circle (one with chords, the other in flower representation). What is the number of such multimeander?*

Problem 3 *A meander is a multimeander such that deleting the circle the remaining edges form a connected (closed) curve. Another remarkable conjecture is that the number of meanders with $2m$ vertices is equivalent when m goes to infinity to*

$$c\rho^m m^{-\frac{29+\sqrt{145}}{12}}$$

for some (other) constants c and ρ

The *side gluing construction* on the flower representation goes as follows: replace the circle by a regular polygon with $2m$ sides so that each edge of the flower ends in the middle of a side; then glue the sides two by two according to these edges. Show that if the diagram is planar this constructs a sphere and that the sides create a tree on it. More generally what does this construct when the chord diagram is not planar? Introduce compact orientable surfaces of higher genus.

An alternative construction is obtained by considering *fatgraphs* or *rubber band graphs*. Fill in the central circle of the flower to form a disc and replace the outer edges by rubber bands attached to the disc (twist are not allowed; in other terms the rubber band we use has two sides that are painted with different colors and the gluings respect these colors). The net result of this construction is a compact surface with boundaries, which are homeomorphic to circles: in the planar case, this surface can be flattened on the plane and the boundary circles are also boundaries of the faces. Similarly in the general case, gluing a disc to each boundary circle of a rubber band graph, one constructs a compact surface without boundary with the graph drawn on it.

For example consider the flowers with two edges. Two of them are planar ($\rho = (1,2)(3,4)$ and $\rho = (1,4)(2,3)$). The third one, $\rho = (1,3)(2,4)$, is associated to a rubber band graph whose filling yields a torus. Make (mental and real) pictures!

1.3 Combinatorial maps

The rubber band construction can now be generalised to allow for several discs connected by rubber bands (the rubber we use still has two different colors on its two sides). Labeling the end points of the rubber bands leads to two permutations: σ whose cycles are obtained by reading labels around disc in counterclockwise direction, and ρ, a fix point free involution obtained by writing a 2-cycle (i,j) for each rubber band with endpoints i,j.

Conversely, to a pair (σ,ρ) of permutations of S_{2m} such that ρ is a fix point free involution, let us associate a rubber band graph: to every cycle $(s_1,\ldots,s_k)$ of σ associate a disc with labels $s_1,\ldots,s_k$ in counterclockwise direction, and for each 2-cycle (i,j) of ρ add a rubber band connecting labels i,j. A *combinatorial map* is a pair of such permutations such that the associated rubber band graph is connected. (Since we always consider connected rubber band graphs, it is usual and convenient to include the connectivity in the definition of maps.)

Exercice 7 *A plane tree can be seen as a rubber band graph: replace each vertex by a disc and edges by rubber bands. What are the combinatorial maps associated to plane trees?*

1.4 The action of a subgroup of permutations, transitivity

A set of permutations $\sigma_1,\ldots,\sigma_k$ generates a subgroup $\langle\sigma_1,\ldots,\sigma_k\rangle$ of S_n, consisting of all the permutations that can be obtained by multiplying the σ_i together.

Exercice 8 *Show that the set of all transpositions (i,j) generates S_n. Show that the subset of transpositions of the form $(i,i+1)$, $i=1,\ldots,n-1$ also generates S_n. Show that the permutation $(1,2,3,\ldots,n)$ and the transposition $(1,2)$ is also sufficient to generate S_n.*

What is the subgroup of permutations generated by 3-cycles (permutations with cyclic type $1^{n-3}3$)?

A subgroup G of S_n is said to *act transitively* on $\{1,\ldots,n\}$ if for all i, j there exists $\pi\in G$ such that $\pi(i)=j$. The notion of transitive action allows to express at connectivity at the level

of permutations: a pair of permutation/fix point free involution (σ, ρ) is a map iff the subgroup $\langle \sigma, \rho \rangle$ acts transitively on $\{1, \ldots, 2n\}$. (Why? Express the displacements on the map in terms of permutations.)

[*Side comments on the terminology.* Let G be a group and X a set, and consider an operation $\cdot$ of G on X: for all $\sigma \in G$ and $i \in X$, $\sigma \cdot i$ is an element of X. The operation $\cdot$ is called an *action* if it is coherent with the group product: for all σ, ρ in G and i in X, $(\sigma\rho) \cdot i = \sigma \cdot (\rho \cdot i)$. Obviously the permutations of S_n act on $\{1, \ldots, n\}$ by the operation $\sigma \cdot i = \sigma(i)$.

The *orbit* of an element i under the action of a subgroup G is the set of elements that can be reached from i by the action of elements of G: $j \in G(\{i\})$ if there exists a permutation $\rho \in G$ such that $\rho(i) = j$. A subgroup G of S_n acts *transitively* if it action has only one orbit $\{1, \ldots, n\}$.]

1. Topological maps and surface

When the rubber band graph can be drawn in the plane without crossings, the map is said to be a *planar map*. More generally, the construction of a rubber band graph yields again a compact surface with boundaries that are circles: filling these circles (that is attaching a disc to each circle) yields again a compact surface without boundary with the rubber band graph drawn on it. The result is called a *topological map*, that is a partition of a surface into vertices (the original discs), edges (the rubber bands), and faces (the filling discs). Equivalently a topological map can be viewed as a proper embedding of a graph G in a surface S such that $S \setminus G$ is a union of topological discs (the faces).

In view of this construction we define the number of vertices of a combinatorial map (σ, ρ) to be $c(\sigma)$, its number of edges to be $c(\rho)$ and its number of faces to be $c(\sigma\rho)$.

Exercice 9 *The degree of a face of a topological map is the number of sides of edges incident to this face (in particular a bridge connecting two otherwise not connected subgraphs counts for two in the degree of the incident face). What should be the definition of degrees of faces for combinatorial maps?*

Exercice 10 *Show that the number of maps with n edges, k vertices and ℓ faces is equal to the number of maps with n edges, ℓ vertices and k faces. (Either use combinatorial or topological maps.)*

One could also simply associate a graph to a pair (σ, ρ): the vertices are the cycles of σ and the edges connects vertices according to ρ. The rubber band graph or the topological map contains more information: the "circular order" of edges around each vertex and the faces are prescribed by the structure. In particular there are several maps with the same underlying graph.

Consider the planar case. A planar graph is a graph that *can be* drawn in the plane. There are infinitely many drawings but some drawings share combinatorial properties like the degree of faces or the cyclic order of edges: a planar map describes those properties that are invariant by continuous deformation of the plane. To draw a planar graph, one thus choose one of the planar maps, that is, one fixes a cyclic order around vertices. In particular to a planar graph are associated maybe several but a finite number of planar maps. Observe however that choosing an arbitrary cyclic order for a planar graph can lead to non planar map, for instance on the torus.

Exercice 11 *Give exemples of planar graph that are associated to (only one/several/an exponential number of) planar map(s).*

1. The genus of a combinatorial map

The classical Euler formula says that for a connected planar graph drawn in the plane (that is, for a topological planar map), the numbers edges (e), vertices (v) and faces (f) satisfy:

$$v + f = e + 2.$$

This suggests the following definition: a combinatorial map (σ, ρ) with m edges (σ, ρ are in S_n with $n = 2m$) is planar if $c(\sigma) + c(\sigma\rho) = m + 2$.

Claim 2 *A combinatorial map is planar if and only if the associated topological map is planar, that is, if and only if it can be drawn properly in the plane.*

We will not prove this claim (it is a slight generalisation of Proposition 1, see however the discussion below).

Theorem 3 *Let σ and ρ be two permutations of S_n such that $\langle \sigma, \rho \rangle$ acts transitively on $\{1, \ldots, n\}$. Then $n + 2 - c(\sigma) - c(\rho) - c(\sigma\rho)$ is a non negative even integer.*

In view of this theorem we can define the *genus* of a pair of permutations (σ, ρ) as

$$g(\sigma, \rho) = \frac{n + 2 - c(\sigma) - c(\rho) - c(\sigma\rho)}{2}.$$

If (σ, ρ) is a map, then ρ is a fix point free involution, $n = 2m$ and $c(\rho) = m$, so that $g(\sigma, \rho) = \frac{1}{2}(m + 2 - c(\sigma) - c(\sigma\rho)) = \frac{1}{2}(m + 2 - v - f)$, where v and f are the numbers of vertices and faces of the map. In particular if $g(\sigma, \rho) = 0$ the map is planar (this is Claim 2). If $g(\sigma, \rho) = 1$, the map is on the torus (the surface of genus 1), and more generally the combinatorial genus gives the genus of the surface on which is drawn the associated topological map. We shall only prove here Theorem 3, that is the fact that the combinatorial genus is non negative (observe that the fact that it is even is an immediate consequence of Exercice 3).

[Further comment: the next topological step would be to recover the genus classification of compact orientable surfaces by proving that the surfaces associated to two combinatorial maps are homeomorphic if and only if these maps have the same (combinatorial) genus.]

Proof of the genus theorem.

Lemma 4 *Let σ be a permutation and $\tau = (a, b)$ be a transposition, then $c(\sigma\tau) = c(\sigma) + 1$ if a and b are in the same cycle of σ, $c(\sigma\tau) = c(\sigma) - 1$ otherwise.*

Lemma 5 *If $\langle \sigma, \rho \rangle$ acts transitively and ρ has more than 1 cycle, then there exists $\tau = (a, b)$ such that: a and b are in the same cycle of σ but in different cycle of ρ and $\langle \sigma\tau, \rho\tau \rangle$ acts transitively.*

Lemma 6 *By recurrence on $c(\rho)$, we have $c(\sigma) + c(\rho) \leq n + 1$ if $\langle \sigma, \rho \rangle$ acts transitively.*

Lemma 7 *By recurrence on $c(\sigma\rho)$, we have $c(\sigma) + c(\rho) + c(\sigma\rho) \leq n + 2$ if $\langle \sigma, \rho \rangle$ acts transitively.*

This last lemma concludes the proof.

Exercice 12 *The definition of combinatorial map can be extended to all pairs of permutations (σ, ρ) of S_n such that $\langle \sigma, \rho \rangle$ acts transitively (ρ does not have anymore to be a fix point free involution). The associated rubber band graphs are constructed using two sets of discs: white discs for cycles of σ and black discs for cycles of ρ; for each i in $[n]$ a rubber band is attached between the cycle of σ and the cycle of ρ containing i. The result is a bicolored rubber band graph, so these pairs of permutations are called bicolored combinatorial maps.*

What is the special case of bicolored maps that corresponds to maps? Write the Euler's formula for bicolored maps. Give an interpretation of the proof of the genus theorem in terms of bicolored maps.

2 Second lecture _ compact data structures for trees

2.1 Tree representations

Rooted plane trees (aka ordered trees) can be defined recursively as follows:

- A (finite) rooted plane tree T is a couple $(r, (u_1, \ldots, u_k))$ where r is called the root label of A, k is a non negative integer and the u_i are (finite) rooted plane trees.

From now on we shall only consider finite trees. We assume that the reader is familiar with the definitions of subtrees, vertices (inner nodes and leaves), edges, father, sibblings or sons, prefix order, depth first search, breadth first search, which can be found in any standard textbooks.

The recursive definition is readily translated into a data structure: (a picture would have been better here, but we get a piece of C-code instead...)

```
typedef struct plane_tree{
  int degree;
  plane_tree **sibblings;
  plane_tree *father;
} plane_tree;
```

With this structure a tree is given by a pointer `pT` with type `plane_tree*` and the degree (`pT->degree`) or the ith son (`pT->sibblings[i]`) is accessed in constant time.

Now suppose our trees are used only for left-right depth (or breadth) first search. A natural variant of the encoding is then to keep only father/first-son and next-brother relations:

```
typedef struct simplif_tree{
  simplif_tree *first_son;
  simplif_tree *next_brother;
  simplif_tree *father;
} simplif_tree;
```

Although this representation is less explicit, it should be clear that it correctly encodes the plane tree. Of course some queries are now more difficult to answer than with the first representation, like the degree of a vertex, or going from a vertex to its ith son ($i > 1$).

The second representation is clearly formally equivalent to the natural binary tree data structure: each cell can be seen as an internal node and each NULL pointer as a leaf of a complete binary tree. Combinatorially, the transformation that associates to a rooted plane tree the binary tree underlaying the second representation is called the *rotation correspondence*, and we write the following statement:

Theorem 8 *The rotation correspondence is a bijection between plane trees with n edges and complete binary trees with n internal nodes (and $n + 1$ leaves).*

While data structures based on pointers are natural from the algorithmic point of view, the recursive definition also naturally leads to representations of trees by words. The explicit expression of a tree, for instance

$$T = (a, ((b, ()), (c, ((d, ()), (e, ()))), (f, ())),$$

is somewhat cumbersome but can be simplified in several ways:

- Trees can be viewed as algebraic expressions, with vertex labels written as functionals:

  $$T = a(b(), c(d(), e()), f()).$$

  Usually this notation is further simplified by denoting leaves as constants:

  $$T = a(b, c(d, e), f).$$

  From this point of view it is natural to call *arity* the number of sibblings of a vertex (rather than calling it degree, as often seen; this convention is also coherent with the graph theory definition of the degree as the number of neighbors of a vertex, including its father).

- The *polnish notation* is based on the following observation: upon writting the arity of each vertex, parentheses can be recovered from the vertex symbols alone:

$$T = a_3 b_0 c_2 d_0 e_0 f_0.$$

When vertices are not labeled (*ie* all labels are identical), it is sufficient to write the arity, and this yields the *arity code* or *degree code* of the tree:

$$T = 3\,0\,2\,0\,0\,0.$$

A tree can obviously be recovered from its code by reading from left to right and inserting at the leftmost available position a vertex of arity i when a letter i is read. Any sequence on the alphabet $\mathbb{N}$ is of course not the code of a tree. It is however easy to check that a word $w = w_1 \dots w_n$ on $\mathbb{N}$ is a degree code if and only if

  - The letters involved form a coherent degree sequence: $\sum_{i=1}^{n} w_i = n - 1$ (why?)
  - The word w satisfies the Łukasiewicz property: for all $j < n$, $\sum_{i=1}^{j} (w_i - 1) \geq 0$.

The Łukasiewicz property expresses the fact that, during the reconstruction of the tree T, there must always be an available slot for inserting the next vertex.

A convenient way to visualize the Łukasiewicz property consists in representing the word w by a path from the origin with ith step $(1, w_i - 1)$. The fact that w is a coherent degree sequence is equivalent to the fact that this path ends at position $(n, -1)$, and the Łukasiewicz property is equivalent to the fact that the path stays on or above the horizontal axis until the last step.

- An opposite way to simplify algebraic expression when there are no labels on vertices consists in keeping only the parentheses:

$$T = (()(()())())$$

The *parenthesis code* of a tree with n vertices is a word with n opening and n closing parentheses such that the number of opening parenthesis is strictly larger than the number of closing ones in any strict prefix of the word.

Again these codes have a natural graphical representation: to a parenthesis code w of length $2n$ one associate a path from $(0, 0)$ to $(2n, 0)$ with ith step $(1, 1)$ if w_i is an opening parenthesis and $(1, -1)$ otherwise. Apart at its endpoints, this path lies always strictly above the horizontal axis.

(In this description of parenthesis codes, we have included the outer pair of parentheses that corresponds to the root vertex of the tree. In other presentations this pair is sometimes omitted, in which case the above characterisation of parenthesis codes must be rewritten without the occurrences of the word *strict*. In particular the standard definition of *Dyck path* allows those paths to return to the horizontal axis.)

Exercice 13 *Make explicit the transformations between all pairs of representations of a plane trees: recursive definition, first and second data structure, degree code and parenthesis code, Łukasiewicz and Dyck paths.*

Does the diagram of your $\binom{7}{2}$ *transformations commute?*

Exercice 14 *The* cycle lemma *says that given a word w of length n such that $\sum_{i=1}^{n} w_i = n - 1$ there is a unique way to factorize it as $w = uv$ with u non empty such that the word vu has the Łukasiewicz property.*

Prove the cycle lemma (using for instance the representation in terms of Łukasiewicz paths) and apply it to count the number of plane trees with n_i vertices of degree i.

2.2 Compact representations and entropy

Let us compare the space used to represent a rooted plane tree with n vertices in the various ways discussed above.

Obviously the most compact representation is the parenthesis code which uses $2n$ bits. This appears to be better than the degree code which consists of a list of n integers, and much better than the data structures which require a linear number of pointers. More precisely, from a theoretical point of view, the integers needed in the degree code are of order $O(n)$ so the cost of the degree code is $O(n \log n)$. Similarly the pointers that are used in data structure must allow to address n different cells, hence have size of order $\log n$ each.

We shall thus make a distinction between *compact* representations that use a linear space $O(n)$, and more explicit representations that use a super linear space (in general $O(n \log n)$). As we have seen, plane trees admit a compact representations, the parenthesis code.

How compact can be a representation of trees? Since the number of rooted plane trees with n nodes is $|T_n| = \frac{1}{n+1}\binom{2n}{n} \approx 4^n$, at least $\log_2 |T_n| = 2n + O(\log n)$ bits are needed to distinguish between them. More formally the following statement gives an "information theory" lower bound on the compacity of the encoding of a set of objets:

- Let $||C|| = \lceil \log_2 |C| \rceil$ be the *entropy* of a set of objets C with cardinality $|C|$. Then any binary code for the elements of C contains a code of length at least $||C||$. Equivalently for all $\phi : C \to \{0,1\}^*$ with $\phi(c) \neq \phi(c')$ if $c \neq c'$, there exists $c \in C$ such that $\phi(c) \geq ||C||$.

In the case of plane trees the length $2n$ of the parenthesis code is larger than the exact entropy of $|T_n|$, but when n goes to infinity these two quantities are asymptotically equivalent. We say that the parenthesis code is *asymptotically optimal* for plane trees with n vertices.

On the one hand we have seen that data structures that are direct translations of combinatorial structures using pointers cannot be compact since they require pointers of size $\log n$. On the other hand there is always an optimal representation for a set C, which consists in making a list of all the element of C and representing each element by its rank in this list. Besides the fact that this representation may be very hard to compute, it is not necessarily compact: for instance the permutations of $\{1, \ldots, n\}$ admit no compact representation since there are $n!$ such permutations and $\log(n!) \approx n \log n$.

We are interested in families of objects $\mathcal{C} = (C_n)$ such that the number $|C_n|$ of objects of size n is finite and the entropy $||C_n||$ is asymptotically linear in the size: there is a constant α_0, called the entropy per size unit of $\mathcal{C}$, such that $||C_n|| = \alpha_0 n + o(n)$. For instance plane trees have an entropy of 2 bits per vertex (2 bpv).

By definition the entropy is an increasing function with respect to inclusion: if $B_n \subset C_n$ then $||B_n|| \leq ||C_n||$, and this inclusion can of course be strict. For instance the set B_n of binary trees with $n = 2m+1$ vertices (m inner nodes and $m+1$ leaves) is a subset of the set T_n of plane trees with n vertices. Since $B_n = \frac{1}{m+1}\binom{2m}{m}$, $||B_n|| = n + o(n)$ so that the entropy of the subclass B_n is 1 bpv, half of the entropy 2 bpv of T_n. As a consequence an encoding can be optimal for a class and not for a subclass: the parenthesis code of a binary tree with $n = 2m+1$ vertices has length $2n = 4m+2$, which is asymptotically optimal if the tree is considered as member of the familly of plane trees (with asymptotic entropy $2n$) but which is not optimal if the tree is considered as member of the familly of binary trees (with asymptotic entropy $2m \approx n$).

Conversely the degree code appears not to be asymptotically optimal in general but for a binary tree with n vertices (m nodes, $n+1$ leaves), this code consists of a word with m letters 2 and $m+1$ letters 0, *ie* it uses $n = 2m+1$ bits: the degree code is asymptotically optimal for the familly of binary trees.

Exercice 15 *Give a compact representation for the set of permutations of* $\{1, \ldots, n\}$ *that avoid the pattern* 132.

Problem 4 *The Stanley Wilf conjecture (proved by Marcus and Tardos) says that given a permutation* σ *there exists a constant* c *such that the number of permutations of* $\{1, \ldots, n\}$ *avoiding the pattern* σ *is bounded by* c^n.

However it is an open problem to give a reasonable compact representation for the set of permutations avoiding a given pattern σ. (Here reasonable means that encoding and decoding the permutation must be done by a polynomial time algorithms: making a list and using the rank is not reasonable.)

Observe that a solution of this problem would yield a constructive proof of the Stanley Wilf conjecture.

2.3 Query complexity

Some representations we call *data structures*, other *codes*, what is the difference? The answer should be rather clear: codes are compact but not very explicit, they are suited to data storage or data transmission, while data structures are convenient to use the object, typically to navigate from one vertex to another in a tree, to describe and implement algorithms.

However, a more precise answer depends on the type of *queries* one has to deal with: any representation can be considered as an encoding (more or less compact) or as a data structure (allowing to answer queries more or less efficiently).

Let us consider the following queries on trees:

- First son: given a vertex, return its first son.
- ith son: given a vertex return its ith son.
- degree: given a vertex, what is its degree?

As already mentioned the complexity of answering each of these query depend on the representation. In fact already the way in which a vertex is given depends on the representation: for the two data structures above, a vertex is given by a pointer on a cell of type `plane_tree` or `simplif_tree` respectively; for the degree code a vertex is given by its index in prefix order (which is also the position of the corresponding letter in the code); for the parethesis code a vertex can be given by the position of the corresponding opening parenthesis in the code.

For the 3 queries above, we obviously get the following worst case complexity:

	`plane_tree`	`simplif_tree`	degree code	parenthesis
first son	$O(1)$	$O(1)$	$O(1)$	$O(1)$
ith son	$O(1)$	$O(i)$	$O(n)$	$O(n)$
degree	$O(1)$	$O(i)$	$O(1)$	$O(n)$

As expected, what is gained in compacity is lost in efficiency for queries. Conversely, one can improve the efficiency for a specific query by adding information (like the degree in `simplif_tree`).

Exercice 16 *Is it possible to answer in constant time the adjacency query: given two vertices, decide whether they are adjacent.*

2.4 Compact data structure

In view of the previous discussion a natural problem is to find good trade-off between compactness and query efficiency. Ideally one seeks a representation that is simultaneously compact (ideally asymptotically optimal with respect to the entropy) and allows to answer the relevant queries efficiently (ideally in constant time).

Let us consider an example of such a trade-off in our test case of plane trees. The data structure we will describe is based on enriching the parenthesis code with $O(n)$ (to keep compactness) so as to improve the complexity of the standard queries above.

The main issue with the parenthesis code is the matching query: given the position i of an opening parenthesis, return the position of the corresponding closing parenthesis. A trivial improvment on the parenthesis code thus consists in adding an auxiliary table that yields for each position i the matching position j, but of course this requires a space of order $n \log n$.

In order to keep the compactness we wish to fill only partially the auxiliary table: we store the positions of a selection of pairs of parenthesis, which we call the *shortcuts*. Given a function $f(n)$ such that $0 < f(n) < n$, the selection of the shortcuts is based on the construction of an auxiliary graph:

- Initialize the auxialiary graph to be the graph on $\{1, \ldots, 2n\}$ with edges $(i, i+1)$. Then read the parenthesis code from right to left.
- When an opening parenthesis is read at position i with closing parenthesis at position j, the shortcut (i, j) is added in the auxiliary table and in the auxiliary graph if the distance between vertices i and j in the auxiliary graph is larger than $f(n)$.

Exercice 17 *Show that given a parenthesis code and its shortcuts, the query* matching *can be answered in time $O(f(n))$.*

Show that the number of shortcuts added to the structure is $O(n/f(n))$. (Hint: show that the auxiliary graph is planar and more precisely that there is a representation of the graph such that $f(n) - 2$ vertices are hidden from the infinite face each time a shortcut is added.)

Propose a range of $f(n)$ and a representation of the auxiliary table of shortcuts such that the resulting structure is compact. Is it asymptotically optimal?

This improved parenthesis code does not allow to answer the matching query in constant time. During the last few years this problem has raised lots of interest and the design of succinct data structres in general is an active area in algorithmics: although theoretical solutions have been found, they tend to be very complex and it is still a major open problem to propose a simple representation granting optimal space and constant query time even for plane trees.

3 Third lecture : enumeration of paths, trees and maps

This lecture is concerned with enumeration with ordinary generating function.

3.1 Trees and algebraic languages

The natural way to decompose a tree is to break it into pieces that are themselves trees of the same type. For instance the following decomposition is a variant of the recursive definition of rooted planar trees: a rooted plane tree consists of

- a single root vertex,
- or a root edge connecting two rooted plane trees: the subtree rooted at the leftmost son of the root, and the tree consisting of all other vertices.

In other terms we have the following bijective decomposition of the set of plane trees $\mathcal{T}$:

$$\mathcal{T} \equiv \{v\} + \{e\} \times \mathcal{T} \times \mathcal{T}$$

Let $T(z)$ denote the (ordinary) generating function of trees with respect to the number of edges: by definition,

$$T(z) = \sum_{t \in \mathcal{T}} z^{|t|},$$

where $|t|$ denote the number of edges of a tree $t \in \mathcal{T}$. Then in view of the previous decomposition,

$$\begin{aligned} T(z) &= t^0 + \sum_{(t_1,t_2) \in \mathcal{T}^2} z^{1+|t_1|+|t_2|} \\ &= 1 + z \sum_{t_1 \in \mathcal{T}} z^{|t_1|} \sum_{t_2 \in \mathcal{T}} z^{|t_2|} \\ &= 1 + zT(z)^2. \end{aligned}$$

This equation for $T(z)$ allows to compute the first terms of the expansion of $T(z)$ by iteration $T(z) = O(1) = 1 + O(z) = 1 + z + O(z^2) = 1 + z + 2z^2 + O(z^3)$: indeed it is equivalent to a recurrence on the coefficients of $T(z)$, as can be seen by extracting the coefficient of z^n on both sides.

Since the equation $T = 1 + zT^2$ satisfied by the series $T(z)$ is just a quadratic equation, it can be explicitely solved: the roots of this second order equation are $T_0(z) = \frac{1\pm\sqrt{1-4z}}{2z} = \frac{1}{z} - 1 - z - 2z^2 + O(z^3)$ and $T_1(z) = \frac{1-\sqrt{1-4z}}{2z} = 1 + 2z + 5z^2 + O(z^3)$, and $T(z)$ is identified as $T_1(z)$ by comparing the first terms of their expansions (in particular here T_0 is not finite at 0). Using the standard Taylor expansion of $(1-z)^\alpha$ at $z = 0$, we get

$$T(z) = \sum_{n\geq 0} \frac{1}{n+1}\binom{2n}{n} z^n.$$

There is a more general technic to extract the coefficients of a formal power series $f(z)$ that satisfies an equation of the form

$$f(z) = z\phi(f(z)),$$

for some formal power series $\phi(u)$ with $\phi(0) \neq 0$: the *Lagrange inversion* theorem states that

$$[z^n]f(z) = \frac{1}{n}[u^{n-1}]\phi(u)^n.$$

Upon setting $U(z) = T(z) - 1$, we have $U(z) = z(1 + U(z))^2 = z\phi(U(z))$ with $\phi(u) = (1+u)^2$. Hence $[z^n]f(z) = \frac{1}{n}[u^{n-1}](1+u)^{2n} = \frac{1}{n}\binom{2n}{n-1} = \frac{1}{n+1}\binom{2n}{n}$.

Exercice 18 *Complete k-ary trees are rooted plane trees with vertices of arity k and 0 only. Write a functional equation for their generating series in terms of the number of inner vertices and use the Lagrange inversion theorem to compute the number of such trees with n inner vertices.*

Exercice 19 *The weighted generating function of Łukasiewicz words is*

$$L(z) = \sum_{w\in\mathcal{L}} \phi_w\, z^w,$$

where $\phi_w = \phi_0^{|w|_{x_0}}\phi_1^{|w|_{x_1}}\cdots\phi_k^{|w|_{x_k}}$ with $|w|_{x_i}$ denotes the number of letter x_i in w. In other terms ϕ_w is the commutative image of w by the map $x_i \to \phi_i$.

– Show that $L(z)$ is the generating function of rooted plane trees according to the number of edges and to some weight ϕ_t.

– Show that $L(z)$ satisfies the equation $L(z) = z\phi(L(z))$ with $\phi(u) = \sum_{i\geq 0}\phi_i u^i$.

– Observe that $\phi(u)^n$ is the weighted generating function of words of length n on the alphabet $\{x_i\}_i \geq 0$. What is given by $[u^{n-1}]\phi(u)^n$?

– Apply the cycle lemma to Łukasiewicz words to prove the Lagrange inversion formula.

– Generalize the above argument to show that

$$[z^n](f(y))^k = \frac{k}{n}[u^{n-k}]\phi(u)^n$$

and more generally give an expression of $[z^n]\psi(f(y))$ for any given power series $\psi(u)$.

Chord diagrams can be decomposed in a similar way as trees: a chord diagram on $\{1,\ldots,2n\}$ with at least one chord can be decomposed at the chord connecting the vertex 1 to a vertex $2i$ (obviously with even index) into two chord diagrams on the vertex sets $\{1,2,\ldots,2i\}$ and $\{1,2i,2i+1,\ldots,2n\}$. If $i = 1$ or $i = n$, chord diagrams without chords are obtained that need not be further decomposed.

The generating function $D(z)$ of chord diagrams with respect to the number of chords therefore satisfies

$$D(z) = \sum_{d\in\mathcal{D}} z^{|d|} = z(1 + D(z))^2,$$

and once again Catalan numbers are recovered.

Exercice 20 *A dissection of the n-gon is formed of a regular polygon with n vertices and a set of non intersecting straight chords that connect these vertices, without restriction on the number of chords ending at a vertex. A triangulation is a dissection such that all faces are triangles.*

– Show that there are at most $n-3$ chords in a dissection and that this maximal number of chords is reached for triangulations.

– Compute the average degree of vertices in a triangulation of a n-gon (hint: use Euler's formula).

– Use generating function to prove that the number of triangulations of the n-gon is again given by the Catalan numbers. Propose a direct bijection between these triangulations and binary trees.

– Compute the generating function of dissections of the n-gon with respect to size n of the polygon. Use the Lagrange inversion formula to express the number of such dissections of the n-gon as a sum of simple coefficients.

The approach we have used in this section consists in decomposing an object into smaller *independant* objects of the *same type*. It leads recurrence involving convolutions, or, in terms of generating functions, to algebraic equations. This approach applies to non ambiguous algebraic structures, or more generally to *decomposable structures* in the sense of Flajolet's lectures.

3.2 Constrained paths and linear equations with catalytic variables

Let us now consider a second approach that consists in writing linear equations. The enumeration of walks in sectors of the plane typically leads to such decompositions.

Consider first the set of all walks in the plane with steps in $\{(1,0),(0,1),(-1,0),(0,-1)\}$, which we rename o, n, e, s. We consider these walks up to translation (or equivalently starting from $(0,0)$). Obviously such a walk is either empty (no steps) or formed of a last step following a (smaller) walk. Formally the language of all walks satisfies

$$\mathcal{L} = 1 + \mathcal{L} \cdot (o + n + e + s).$$

The generating function of these walks according to the length $|w|$,

$$L(z) = \sum_{w \in \mathcal{L}} z^{|w|}$$

therefore satisfies the equation

$$L(z) = 1 + 4zL(z),$$

so that $L(z) = \sum_{n \geq 0} 4^n z^n$, a result that could clearly have been obtained directly.

In order to apply this idea more generally one must be able to characterize the paths obtained after the removal of the last step. For instance let us apply this technics to Dyck paths.

Recall that a Dyck path is a path of length $2n$ with steps in $\mathcal{S} = \{(1,1),(1,-1)\}$ starting from $(0,0)$, ending at $(2n,0)$, and never visiting points with negative ordinates. If we intend to remove the last step we will need to consider paths which end at ordinate $j > 0$. Given a path w let $j(w)$ be the ordinate of its endpoint and let $\mathcal{F}$ be the set of paths starting from $(0,0)$ with steps in $\mathcal{S}$ and never visiting points with negative ordinates. The generating function of these paths with respect to their length and ordinate of the endpoint is

$$F(z;y) = \sum_{w \in \mathcal{F}} z^{|w|} y^{j(w)},$$

and the removal of the last step yields the following equation:

$$\begin{aligned} F(z;y) &= 1 + \sum_{w=w'u \in \mathcal{F}} z^{|w'|+1} y^{j(w')+1} + \sum_{w=w'd \in \mathcal{F}} z^{|w'|+1} y^{j(w')-1} \\ &= zy \sum_{w' \in \mathcal{F}} z^{|w'|} y^{j(w')} + \frac{z}{y} \sum_{w' \in \mathcal{F} \setminus \mathcal{F}_0} z^{|w'|} y^{j(w')} \\ &= 1 + zyF(z;y) + \frac{z}{y}(F(z;y) - F(z;0)), \end{aligned}$$

where $\mathcal{F}_0$ denotes the set of paths of $\mathcal{F}$ with endpoint at level 0, that is Dyck paths. The equation $F(z;y) = 1 + zyF(z;y) + \frac{z}{y}(F(z;y) - F(z;0))$ may not seem sufficient to determine $F(z;y)$ since it involves two unknown functions $F(z;y)$ and $F(z;0)$. However it does allow to compute iteratively the first terms of $F(z;y)$ as a power series in z, since it is equivalent to a linear reccurence on the (polynomial) coefficients $f_n(y) = [z^n]F(z;y)$ (extract the coefficient of z^n on both sides): for $n \geq 1$,

$$f_n(y) = yf_{n-1}(y) + \frac{1}{y}(f_{n-1}(y) - f_{n-1}(0))$$

with initial term $f_0(y) = 1$. However writing explicitely the recurrence does not help very much to solve it. As we shall see, the functional equation on $F(z;u)$ is easier to deal with, and, with some practice, also easier to write down.

The equation satisfied by $F(z,u)$ is linear in $F(z,u)$ and can be rewritten as follows:

$$(zy^2 + z - y)F(z;y) = zF(z;0) - y.$$

This equation has the generic form

$$K(z;y) \cdot F(z;y) = Q(z;y,f(z)),$$

where $K(z;y)$ is an explicit function of y and the right hand side may only involve unknown functions in z. Equations of this form can be solved in a systematic way using the so-called *kernel method*. Assume that there exists a series $Y(z)$ such that the substitution $F(z;Y(z))$ makes sense as a formal power series in z, and such that $K(z;Y(z)) = 0$. Then the substitution of $Y(z)$ into the equation above cancels the left hand side and yields a new equation $Q(z;Y(z),f(z)) = 0$ which determines $f(z)$.

In our example the kernel is a polynomial $K(z,Y) = zY^2 + z - Y$ which has two roots $Y_\pm(z) = \frac{1 \pm \sqrt{1-4z^2}}{2z}$. Observe that

$$Y_+(z) = \frac{1}{z} - z - z^3 + O(z^5), \quad \text{while} \quad Y_-(z) = z + z^3 + O(z^5),$$

so that substituting $y = Y \pm (z)$ in

$$F(z;y) = \sum_{n>0} f_n(y)z^n$$

yields a meaningful power series only for $Y_-(z)$ (try to compute the first coefficients of z in the substitution $F(z;Y_+(z))$). The formal power series solution $Y_-(z)$ is essentially again the Catalan generating series and its substitution in the initial equation cancels the left hand side and yields:

$$F(z;0) = Y(z)/z = \frac{1 - \sqrt{1-4z^2}}{2z^2} = 1 + z^2 + 2z^4 + 5z^6 + O(z^8),$$

allowing to recover the enumeration of Dyck paths by Catalan numbers. As a byproduct we also obtain an expression for $F(z;y)$:

$$F(z;y) = \frac{Y(z) - y}{zy^2 + z - y}, \quad \text{where } Y(z) = z(1 + Y(z)^2).$$

Exercice 21 *Apply the kernel method to paths with steps in $\mathcal{S}_k = \{(1,k),(1,-1)\}$ that satisfy the Łukasiewicz condition.*

Exercice 22 *(Harder) Write a functional equation describing the last step decomposition of paths with steps in $\mathcal{S} = \{(1,3),(1,-2)\}$ that start at $(0,0)$, never visit points with negative coordinates and end on the horizontal axis. Adapt the kernel method to solve this equation (hint: how many roots of the kernel can be substituted in the equation?).*

Let us apply a similar approach to chords diagrams, now viewed in the flower representation: we wish to decompose chord diagrams by removing only one edge, without cutting the diagram into two pieces.

In the flower representation consider the edge starting at vertex 1 (the root edge) and let us call outerface the face on the right hand side of this edge. Removing the root edge obviously yields again a chord diagram but several different diagrams of size n produce in this way the same diagram of size $n-1$. In order to describe the set of preimages of a diagram of size $n-1$ we need to know in how many places the initial root edge could end. The initial root edge originates in the middle of the side connecting vertex $2n-2$ and 1 of the diagram of size $n-1$ and ends on any side of the polygon that is incident to the outerface. Let us called the degree of the outerface this number of incident sides of the polygon (which is also the number of chords incident to the outerface).

Let us therefore introduce the generating function of chord diagrams with respect to the number of edges and the degree of the outerface (recall our convention that the outerface is the face on the right of the edge starting from vertex 1):

$$C(z;u) = \sum_{c \in \mathcal{C}} z^{|c|} u^{d(c)},$$

where $|c|$ denotes the number of edges and $d(c)$ the degree of the outerface of a diagram c of the set $\mathcal{C}$ of chord diagrams. In order to write an equation for $C(z;u)$ we reformulate the summation as a sum over the reduced diagrams, obtained by removing the root edge:

$$C(z;u) \quad = \quad 1 + \sum_{c' \in \mathcal{C}} \sum_{c \vdash c'} z^{|c|} u^{d(c)}.$$

By construction $|c| = |c'|+1$, and a given c' is obtained $d(c')+1$ times, more precisely from chord diagrams with outerface of degree $1, 2, \ldots, d(c')+1$ respectively. Hence:

$$\begin{aligned} C(z;u) &= 1 + \sum_{c' \in \mathcal{C}} z^{|c'|+1}(u + u^2 + \cdots + u^{d(c')} + u^{d(c')+1}) \\ &= 1 + \frac{zu}{1-u} \sum_{c' \in \mathcal{C}} z^{|c'|}(1 - u^{d(c')+1}) \\ &= 1 + \frac{zu}{1-u}(C(z;1) - uC(z;1)). \end{aligned}$$

The resulting equation can be written in the form

$$(1 - u - zu^2)C(z,u) = 1 - u + zuC(z;1)$$

and solved by the kernel method (yes we find again Catalan numbers...).

The variable y and u in the two previous examples are called in the literature *catalytic variables* because they are introduced in the problem as a tool to allow for the equation to be written (they catalyze the reaction of decomposition...). In this terms, the kernel method is a technic to solve linear equations with one catalytic variable. The resolution of catalytic equations with more than one catalytic variable is currently a very active domain of research.

3.3 Maps and polynomial equations with catalytic variables

Our aim is now to count planar maps and we intend to apply the previous tools. In order to start a decomposition we need a starting point, so we consider *rooted planar maps*: a planar map is rooted if one edge, the *root* is distinguished and oriented, its origin is called the *root vertex* and the face on its right is called the *root face*. By convention rooted planar maps should always be drawn in the plane using the root face as infinite face.

In view of the previous section our strategy should be clear: remove the root edge... Given a rooted planar map, this strategy yields three cases:

- The map has one vertex, one face, and no edge, no decomposition is possible/needed.
- The removal of the root edge disconnects the map into two connected components. Let $\mathcal{M}_1$ the set of such maps.
- The removal of the root edge does not disconnect the map. Let $\mathcal{M}_2$ the set of such maps.

On the first hand, observe that

$$\mathcal{M}_1 \equiv \{e\} \times \mathcal{M} \times \mathcal{M}$$

Indeed the two connected components can be canonically rooted so that they are seen as elements of $\mathcal{M}$ and conversely, given two rooted planar maps a map of $\mathcal{M}_1$ is easily constructed by connecting their root vertices by an edge. Observe that that the orientation of the root edge corresponds in this decomposition to the fact that the two compoments are ordered (the bijection is with couples of maps, not with pairs of maps).

On the other hand, the deletion of the root edge of a map m of $\mathcal{M}_2$, yields a rooted map m' with $|m'| = |m| - 1$ and the number of preimages of a map m' is, as in the case of chord diagrams, given by the degree of the root face of m' plus one. More precisely, a map m' with root face of degree $d(m')$ has $d(m') + 1$ preimages with respective root face degree $1, 2, \ldots, d(m') + 1$.

This suggests again to consider the generating function of rooted planar maps with respect to the number of edges and the degree of the root face:

$$M(z;u) = \sum_{m \in \mathcal{M}} z^{|m|} u^{d(m)}.$$

Then the root edge deletion yields

$$\begin{aligned} M(z;u) &= 1 + zu^2 M(z;u)^2 + \sum_{m' \in \mathcal{M}} z^{|m'|+1}(u + u^2 + \ldots + u^{d(m')+1}) \\ &= 1 + zu^2 M(z;u)^2 + \frac{zu}{1-u}(M(z;1) - uM(z;u)). \end{aligned}$$

The kernel method cannot be applied directly to this equation since it is not linear. However the equation can be put in the form

$$P(M(z;u), u, z, f(z)) = 0$$

where $P(M, u, z, f)$ a polynomial and where only one unknown, $M(z;u)$ depends on the variable u. Deriving this equation with respect to u we get

$$M'_u(z;u)P'_1(M(z;u), u, z, f(z)) + P'_2(M(z;u), u, z, f(z)) = 0$$

so assuming that there exists a series $U(z)$ such that the substitution $m(z) = M(z, U(z))$ make sense and such that $P'_1(m(z), U(z), z, f(z)) = 0$ we obtain the following system of three equations for the three unknowns $(m(z), U(z), f(z))$:

$$\left\{ \begin{array}{c} P(m(z), U(z), z, f(z)) = 0 \\ P'_1(m(z), U(z), z, f(z)) = 0 \\ P'_2(m(z), U(z), z, f(z)) = 0, \end{array} \right.$$

where $P(m, u, z, f)$ is an explicit polynomial. If this system is not degerate it determines the three series $m(z)$, $U(z)$ and $f(z)$ and an equation for $f(z)$ can be obtained by elimination. In the case of our equation for maps the system is

$$\left\{ \begin{array}{c} (1-u)zu^2m^2 + (u - 1 - zu^2)m + (1 - u + zuf) = 0 \\ 2(1-u)zu^2m + (u - 1 - zu^2) = 0 \\ 2zum^2 - 3zu^2m^2 + (1 - 2zu)m + (zf - 1) = 0. \end{array} \right.$$

Using the second equation, m can be expressed in terms of u and f, and using the other two equations an equation for u is derived. Putting these expressions together and using the Lagrange inversion theorem yields the number of rooted planar maps with n edges

$$[z^n]M(z,1) = \frac{2}{n+2}\frac{3^n}{n+1}\binom{2n}{n}.$$

This remarkably simple result suggests that there must be a deep relation between the rooted planar maps and trees, and a more elegant proof than the above computation. In the next lecture we will present such a relation based on the properties of breadth first search on maps, and discuss some algorithmic consequences for coding and random sampling of maps.