In these notes we discuss techniques for counting, coding and sampling some classes of objects. We start by presenting several classes of objects counted by the Catalan sequence $C_n = \frac{1}{n+1} \binom{2n}{n}$. This is an occasion to present several bijective techniques for counting, and simply beautiful mathematics. We then discuss some algorithmic application of (bijective) counting: some coding and random sampling algorithms.

1 Some Catalan families

We start by defining three classes of objects, and then discuss the relation between them.

A plane tree (a.k.a. ordered tree) is a rooted tree in which the order of the children matters. Let $T_n$ be the set of plane trees with $n$ edges. The set $T_3$ is represented in Figure 1. A binary tree is a plane tree in which vertices have either 0 or 2 children. Vertices with 2 children are called nodes, while vertices with 0 children are called leaves. Let $B_n$ be the set of binary trees with $n$ nodes. The set $B_3$ is represented in Figure 2. A Dyck path is a lattice path (sequence of steps) made of steps +1 (up steps) and steps -1 (down steps) starting and ending at level 0 and remaining non-negative. Since the final level of a Dyck path is 0 the number of up steps and down steps are the same, and its length is even. Let $D_n$ be the set of Dyck paths with $2n$ steps. The set $D_3$ is represented in Figure 3.
Observe that there is the same number of elements in $T_3$, $B_3$ and $D_3$. This is no coincidence, as we will now prove that for all $n$, the sets $T_n$, $B_n$, $D_n$ have the same number of elements. We now use the notation $|S|$ to denote the cardinality of a set $S$. We will now prove that

$$|T_n| = |B_n| = |D_n| = \frac{1}{n+1}\binom{2n}{n}.$$ 

The number $\frac{1}{n+1}\binom{2n}{n}$ is the so-called $n$th Catalan number.

### 1.1 Counting Dyck paths

We first compute the number of Dyck paths. Let $\mathcal{P}_n^{(0)}$ be the set of paths of length $2n$ made of steps +1 steps and -1 steps starting and ending at level 0. Ending at level 0 is the same as having the same number of up steps and down steps, and any choice of order of such steps is allowed. Hence

$$\mathcal{P}_n^{(0)} = \binom{2n}{n}.$$

Now $\mathcal{D}_n$ is a subset of $\mathcal{P}_n^{(0)}$. It seems hard to find $|\mathcal{D}_n|$ because of the non-negativity constraint, but actually a trick will now allow us to compute the cardinality of the complement subset

$$\mathcal{D}_n \equiv \mathcal{P}_n^{(0)} \setminus \mathcal{D}_n.$$

Indeed we claim that $|\mathcal{D}_n| = \binom{2n}{n-1}$. To prove this claim we consider the set $\mathcal{P}_n^{(-2)}$ of paths of length $2n$ made of steps +1 steps and -1 steps starting at level 0 and ending at level -2. These paths have $n-1$ up steps and $n+1$ down steps, and any order of steps is possible, hence $|\mathcal{P}_n^{(-2)}| = \binom{2n}{n-1}$. So it suffices to give a bijection $f$ between $\mathcal{D}_n$ and $\mathcal{P}_n^{(-2)}$. This bijection is defined as follows: take a path $D$ in $\mathcal{D}_n$ consider the first time $t$ it reaches level $-1$. The path $f(D)$ is obtained from $D$ by flipping all the steps after time $t$ with respect to the line $y = -1$. An example is shown in Figure 4. We let the reader check that $f$ is a bijection between $\mathcal{D}_n$ and $\mathcal{P}_n^{(-2)}$. Since $f$ is a bijection we have $|\mathcal{D}_n| = |\mathcal{P}_n^{(-2)}| = \binom{2n}{n-1}$.

![Figure 4: The bijection $f$: the path $D \in \mathcal{D}$ in red, the path $f(D) \in \mathcal{P}_n^{(-2)}$ in black.](image)

By the preceding, we have

$$|\mathcal{D}_n| = |\mathcal{P}_n^{(0)}| - |\mathcal{D}_n| = \binom{2n}{n} - \binom{2n}{n+1} = \frac{(2n)!}{n!n!} - \frac{(2n)!}{(n+1)!(n-1)!}.$$
And by reducing to the same denominator we find

\[ |D_n| = \frac{(2n)!}{n+1!n!} = \frac{1}{n+1} \binom{2n}{n}, \]

as wanted.

### 1.2 Bijection between plane trees, binary trees and Dyck paths

We now present bijections between the sets \( T_n, B_n \) and \( D_n \).

We first present a bijection \( \Phi \) between plane trees and Dyck paths as follows: given any tree \( T \) in \( T_n \), perform a depth-first search of the tree \( T \) (as illustrated in Figure 5) and define \( \Phi(T) \) as the sequence of up and down steps performed during the search. A Dyke path is obtained from \( T \) because \( \Phi(T) \) has \( n \) up steps and \( n \) down steps (one step in each direction for each edge of \( T \)), starts and end at level 0 and remains non-negative. Because \( \Phi \) is a bijection between \( T_n \) and \( D_n \), we conclude

\[ |T_n| = |D_n| = \frac{1}{n+1} \binom{2n}{n}. \]

![Figure 5: A plane tree \( T \) and the associated Dyck path \( \Phi(T) \). The depth-first search of the tree \( T \) is represented graphically by a tour around the tree (drawn in orange).](image)

We now present a bijection \( \Psi \) between binary trees and Dyck paths. Let \( B \) be a binary tree in \( B_n \). The tree \( B \) has \( n \) nodes. It can be shown that it has \( n+1 \) leaves (do it!). We can perform a depth-first search of the tree \( B \) and make a up step the first time we encounter each node and a down step each time we encounter a leaf. This makes a path with \( n \) up steps and \( n+1 \) down steps. The last step is a down step and we ignore it. We denote by \( \Psi(B) \) the sequence of \( n \) up steps and \( n \) down steps obtained in this way. An example is represented in Figure 6. It is actually true that \( \Psi(B) \) is always a Dyck path and that \( \Psi \) is a bijection between \( B_n \) and \( D_n \). We omit the proof of these facts. Since the sets \( B_n \) and \( D_n \) are in bijection we conclude

\[ |B_n| = |D_n| = \frac{1}{n+1} \binom{2n}{n}. \]

### 2 Coding

Let \( S \) be a finite set of objects. A coding function for the set \( S \) is a function which associate a distinct binary sequence \( f(s) \) to each element \( s \) in \( S \). The binary sequence \( f(S) \) is called code of \( S \). Here are lower bounds for the length of codes.

Count-3
Figure 6: A binary tree $B$ and the associated Dyck path $\Psi(B)$. The depth-first search of the tree $B$ is represented graphically by a tour around the tree (drawn in orange).

**Lemma 1.** If $S$ contains $N$ elements then at least one of the codes has length greater or equal to $\lfloor \log_2(N) \rfloor$. If one consider the uniform distribution for elements in $S$ then the codes have length at least $\log_2(N) - 2$ in average.

**Exercise:** Prove Lemma 1 for $N = 2^k - 1$.

**Example 1: coding permutations.** Let $S_n$ be the set of permutations of $\{1, 2, \ldots, n\}$. We now discuss a possible coding function $f$ for the set $S_n$. Recall that for any integer $i$, the binary representation of $i$ is $\left\lceil \log_2(i + 1) \right\rceil$. Thus each number $i \in \{1, 2, \ldots, n\}$ can be represented uniquely by binary sequences of length exactly $\left\lceil \log_2(n + 1) \right\rceil$: it suffice to take their binary representations and add a few 0 in front if necessary to get this length. Let $\pi \in S_n$ be a permutation seen as a sequence of distinct numbers $\pi = \pi_1 \pi_2 \ldots \pi_n$. One can define $f(\pi)$ as the concatenation of the binary sequences (of length $\left\lceil \log_2(n + 1) \right\rceil$) corresponding to each number $\pi_1 \pi_2 \ldots \pi_n$. Then the length of the code $f(\pi)$ is $n \left\lceil \log_2(n + 1) \right\rceil \sim n \log_2(n)$. We can recover the permutation from the code: if one has the code, it can cut it in subsequences of length $\log_2(n + 1)$ each and then recover the numbers $\pi_1 \pi_2 \ldots \pi_n$ making the permutation. Is it an efficient coding? Well according to Lemma 1 we cannot achieve codes shorter than $\log_2(n!) - 2$ in average. Moreover, $\log_2(n!) \sim n \log_2(n)$. Therefore our coding function $f$ has length as short as possible asymptotically.

**Example 2: coding Dyck paths.** Consider the set $D_n$ of Dyck path of length $2n$. There is an easy way of coding a Dyck path $D \in D_n$ by a binary sequence of length $2n$. Simply encode down steps by “0” and up steps by “1” this give a binary sequence $f(D)$ of length $2n$. Could we hope for shorter codes? Certainly it would be possible to get a code of length $2n - 2$ because the first step is an up step and the last step is a down step, so these could be ignored. But could we do better than $2n + o(n)$ (where the “little o” notation means that the expression divided by $n$ goes to zero as $n$ goes to infinity)? We have seen that the set $D_n$ has cardinality $N = \frac{2n!}{n!(n+1)!}$. Using the Stirling formula

$$n! \sim \sqrt{2n \pi} \left( \frac{n}{e} \right)^n$$

one gets $\log_2(n!) = n \log_2(n) - n \log_2(e) + o(n)$. Hence one can compute

$$\log_2(N) = \log_2(2n!) - \log_2(n!) - \log_2((n + 1)!) = 2n + o(n).$$

Therefore, by Lemma 1 one cannot encode Dyck paths by codes of length less than $2n + o(n)$ on average. So our naive coding is asymptotically optimal. Observe that this also gives a way of coding plane trees or binary trees optimally.
3 Random sampling

Let $S$ be a finite set of objects. A (uniformly random) sampling algorithm for the set $S$ is an algorithm which outputs an element in $S$ uniformly at random from $S$. Here we suppose we dispose of a perfect random generator for integers. More precisely, let us suppose that one can generate a uniformly random integer in $\{1, 2, \ldots, n\}$ for any integer $n$.

Example 1: sampling permutations. How to sample a permutation in $S_n$? Here is a solution written in pseudo-code.

<table>
<thead>
<tr>
<th>Input</th>
<th>an integer $n$.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Initialize an array $V$ of size $n$ with value $i$ at position $i$ for $i = 1 \ldots n$.</td>
</tr>
<tr>
<td></td>
<td>• For $i = 1$ to $n$ do</td>
</tr>
<tr>
<td></td>
<td>Choose an integer $r$ uniformly at random in ${i, i + 1, \ldots, n}$.</td>
</tr>
<tr>
<td></td>
<td>Swap the values at position $i$ and $r$ in $V$.</td>
</tr>
<tr>
<td>Output</td>
<td>the array $V$.</td>
</tr>
</tbody>
</table>

The output of the above algorithm is an array of number which corresponds to a uniformly random permutation. Indeed, the first number of the array is chosen uniformly in $\{1, 2, \ldots, n\}$, the second number in the array is chosen uniformly randomly from the remaining numbers etc. Thus the above algorithm is indeed a sampling algorithm for the set $S_n$.

Example 2: sampling Dyck paths. Sampling Dyck paths is a bit more difficult. We will need to first define an algorithm for sampling paths from another set. Let $P_n^{(-1)}$ be the set of paths of length $2n + 1$ with steps $+1$ and $-1$ starting at level 0 end ending at level $-1$. Hence a path $P \in P_n^{(-1)}$ has steps "+1" and $n + 1$ steps ",-1" in any order. Here is a sampling algorithm for the set $P_n^{(-1)}$.

<table>
<thead>
<tr>
<th>Input</th>
<th>an integer $n$.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Initialize an array $V$ of length $2n + 1$ with value 1 in the first $n$ entries and value $-1$ in the remaining $n + 1$ entries.</td>
</tr>
<tr>
<td></td>
<td>• For $i = 1$ to $2n + 1$ do</td>
</tr>
<tr>
<td></td>
<td>Choose an integer $r$ uniformly at random in ${i, i + 1, \ldots, 2n + 1}$.</td>
</tr>
<tr>
<td></td>
<td>Swap the values at position $i$ and $r$ in $V$.</td>
</tr>
<tr>
<td>Output</td>
<td>the array $V$.</td>
</tr>
</tbody>
</table>

Because the algorithm randomly permutes the steps $+1$ and $-1$, it indeed outputs a uniformly random path in $P_n^{(-1)}$.

Now we will show how to obtain an Dyck path $D \in D_n$ from a path $P \in P_n^{(-1)}$. The trick we will use is known as the cycle lemma. Let $P \in P_n^{(-1)}$. Let $\ell \leq 0$ be the lowest level of the path $P$, and let $t$ be the first time the level $\ell$ is reached. This decomposes $P$ as $P_1P_2$ where $P_1$ is the path before time $t$ and $P_2$ is the path after time $t$. Now consider the path $P_2P_1$. This path is ending with a $-1$ step. Then we define $g(P)$ as the path obtained from $P_2P_1$ by ignoring the last step. The mapping $g$ is illustrated in Figure 7. The path $g(P)$ has $n$ up steps and $n$ down steps so it ends at level 0. In fact we claim that it is a Dyck path. Here is an even stronger claim.

**Lemma 2.** For any path $P$ is $P \in P_n^{(-1)}$, the path $g(P)$ is a Dyck path. So $g$ maps the set $P_n^{(-1)}$ to the set $D_n$. Moreover, any Dyck path in $D_n$ is the image of exactly $2n + 1$ paths in $P_n^{(-1)}$. 

Count-5
Figure 7: A path $P \in \mathcal{P}_n^{(-1)}$ and the resulting $g(P)$.

We will not prove this Lemma. However we argue that this gives a way of sampling Dyck paths. Indeed, by the above algorithm, one can sample a path $P$ in $\mathcal{P}_n^{(-1)}$, and then apply the mapping $g$ to obtain a Dyck path $g(P)$. Since every path in $\mathcal{P}_n^{(-1)}$ has the same probability of being sampled and every Dyck path in $\mathcal{D}_n$ has the same number of preimages, every Dyck path in $\mathcal{D}_n$ has the same probability of being sampled. We have thus found a sampling algorithm for Dyck paths. Observe that this also gives a way of sampling plane trees or binary trees.