

Obligatorio Combinatoria Analítica Rep. 2

Damián Ferencz

22 de Mayo de 2019

Ejercicio 28:

Parte 1:

Veamos como podemos, con las construcciones que tenemos disponibles, especificar el conjunto de pares de permutaciones del mismo largo \mathcal{P}^2 , donde, el peso está determinado por el largo de las permutaciones.

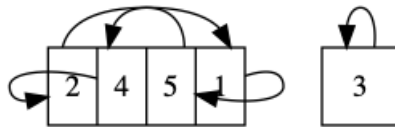
Lo primero que podemos hacer es pensar la primera permutacion con la estructura usual de una lista etiquetada, y la segunda como punteros entre las posiciones de la lista.

Podemos establecer la especificacion recursiva:

$$\mathcal{P}^2 = \{\epsilon\} + \mathcal{Z} \star (\mathcal{P}^2 + \Theta \mathcal{P}^2)$$

Veamos como se puede construir cualquier lista etiquetada con punteros de tamaño n :

- $\{\epsilon\}$ es el caso de la lista con un elemento y cuya posición se apunta a sí misma (corresponde al caso del par de permutaciones triviales con un elemento).
- $\mathcal{Z} \star \mathcal{P}^2$ corresponde a agregar un nuevo elemento con etiqueta que se apunta a si misma (y por lo tanto, la segunda permutacion tiene a n fijo)
- $\mathcal{Z} \star \Theta \mathcal{P}^2$ con posición distinguida p corresponde a agregar un nuevo elemento con etiqueta que apunta a la posición p , cambiando el viejo puntero a p para la posición n (y por lo tanto, la segunda permutacion manda p a n).



En el ejemplo de la figura, al agregar el nodo de la derecha, con etiqueta 3, podemos optar por la construcción $\mathcal{Z} \star \mathcal{P}^2$ y obtenemos el par de permutaciones $((\begin{smallmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 5 & 1 & 3 \end{smallmatrix}), (\begin{smallmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 2 & 3 & 5 \end{smallmatrix}))$. En cambio, si optamos por la construcción $\mathcal{Z} \star \Theta \mathcal{P}^2$ y elegimos la posición distinguida 4, el par de permutaciones quedaría $((\begin{smallmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 5 & 1 & 3 \end{smallmatrix}), (\begin{smallmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 1 & 2 & 3 & 4 \end{smallmatrix}))$.

Si llamamos $p(z) = \sum_{n=0}^{\infty} \frac{p_n}{n!} z^n$ a la FGE de \mathcal{P}^2 , se tiene

$$p(z) = 1 + z(p(z) + zp'(z))$$

Aplicando $[z^{n+1}]$ a ambos lados de la igualdad:

$$\frac{p_{n+1}}{(n+1)!} = \frac{p_n}{n!} + \frac{np_n}{n!}$$

Es decir,

$$\frac{p_{n+1}}{(n+1)!} = \frac{(n+1)p_n}{n!} = \frac{(n+1)!p_n}{n!^2}$$

Luego,

$$p_{n+1} = \frac{(n+1)!^2}{n!^2} p_n$$

Como $p_0 = 1$, deducimos $p_n = n!^2$, como hubieramos previsto.

Parte 2:

Sean $\Gamma = \{\Gamma_{\sigma\tau} : (\sigma, \tau) \in \mathcal{P}^2\}$ ¹ y $\Gamma^C = \{\Gamma_{\sigma\tau} \text{ conexo} : (\sigma, \tau) \in \mathcal{P}^2\}$.

Cada grafo de Γ puede entenderse como el conjunto formado por sus componentes conexas, luego se tiene la especificación $\Gamma = \text{SET}(\Gamma^C)$. Es claro que $\Gamma \cong \mathcal{P}^2$, luego la FGE de Γ es $g(z) = \sum_{n=0}^{\infty} \frac{n!^2 z^n}{n!} = \sum_{n=0}^{\infty} n! z^n$ y la de Γ^C es $\log(g(z))$.

Por definición de probabilidad en espacios finitos, se tiene:

$$\pi_n = P(\{(\sigma, \tau) \in \mathcal{P}_n^2 : \Gamma_{\sigma\tau} \in \Gamma^C\}) = \frac{n! [z^n] \log(g(z))}{n! [z^n] g(z)} = \frac{[z^n] \log(g(z))}{n!}$$

Parte 3:

Nos disponemos a probar $n!^2 \pi_n = (n-1)! I_n$, o, equivalentemente, que $(n-1)! I_n = n! \log(g(z)) = \Gamma_n^C$. Llamemos $(\gamma_n)_{n \in \mathbb{N}}$ a la sucesión que cuenta Γ_n^C y $(i_n)_{n \in \mathbb{N}}$ la correspondiente a \mathcal{I} .

Recordemos que la clase \mathcal{I} se define implícitamente con la ecuación $\mathcal{P} = \text{SEQ}(\mathcal{I})$. Ahora, desarrollando un paso la definición de SEQ, esto es lo mismo que escribir:

$$\mathcal{P} = \{\epsilon\} + \mathcal{I} \times \mathcal{P}$$

Dicha especificación se traduce en una ecuación de FGOs:

$$\sum_{k=0}^{\infty} n! z^n = 1 + \left(\sum_{k=0}^{\infty} k! z^k \right) \left(\sum_{k=1}^{\infty} i_k z^k \right)$$

Luego, si $n \geq 1$, al aplicar $[z^n]$ obtenemos $n! = \sum_{k=0}^n i_k (n-k)!$. Entonces, podemos afirmar que la sucesión $(i_n)_{n \in \mathbb{N}}$ queda determinada por

$$\begin{cases} i_0 = 0 \\ i_n = n! - \sum_{k=1}^{n-1} i_k (n-k)! \end{cases}$$

¹Estamos asumiendo que las aristas se etiquetan con la permutación a la que corresponden. Luego $\Gamma_{\sigma\tau} \neq \Gamma_{\tau\sigma}$. Además, las aristas son dirigidas

Esta recurrencia, que se desprende de la especificación que recién mostramos para \mathcal{P} , resulta de entender cualquier permutación indescomponible de largo n como unión de dos permutaciones $P(1) \cup P'$ siendo $P(1)$ la permutación indescomponible más pequeña que contiene al 1 y P' la permutación restante.

Inspirado por ésta recurrencia, y notando A_k^n a los arreglos de n en k , observamos que $(\gamma_n)_{n \in \mathbb{N}}$ queda determinada por la siguiente:

$$\begin{cases} \gamma_0 = 0 \\ \gamma_n = (n!)^2 - \sum_{k=1}^{n-1} \gamma_k A_{n-k}^{n-1} (n-k)! \end{cases}$$

Ésta recurrencia también proviene de una especificación, pero es un poco engorrosa así que prefiero relatarla: Cualquier grafo $\Gamma_{\sigma\tau}$ disconexo puede verse como una unión no trivial $\Gamma_{\sigma\tau}(1) \cup \Gamma'_{\sigma\tau}$, donde $\Gamma_{\sigma\tau}(1)$ es la componente conexa que contiene al vértice con etiqueta 1 y $\Gamma'_{\sigma\tau}$ el grafo con los vértices restantes, sin restricciones. Ahora bien, si $|\Gamma_{\sigma\tau}(1)| = k$, tenemos A_{n-k}^{n-1} formas de elegir las etiquetas de $\Gamma'_{\sigma\tau}$ y a la vez una de las permutaciones en dicho grafo. Luego de ésto, tenemos $(n-k)!$ formas de elegir la otra permutación en $\Gamma'_{\sigma\tau}$. Por último observemos que las $n-1$ etiquetas utilizadas en $\Gamma_{\sigma\tau}(1)$ quedaron determinadas y hay γ_k formas de elegir $\Gamma_{\sigma\tau}(1)$.

Finalmente, veamos por inducción que $(k-1)!i_{k+1} = \gamma_k \forall k \geq 1$. Para eso, supongamos que vale para todo $k \leq n$, con $n \geq 2$ ². Veamos que vale para n :

El lado derecho de la recurrencia para γ_n se puede escribir en términos de los i_k y esta claro que habremos terminado si probamos $(n-1)!i_{n+1} = (n!)^2 - \sum_{k=1}^{n-1} (k-1)!i_{k+1}A_{n-k}^{n-1}(n-k)!$

Pero ésto último es igual a $(n!)^2 - \sum_{k=1}^{n-1} i_{k+1}(n-1)!(n-k)! = (n!)^2 - \sum_{k=2}^n i_k(n-1)!(n-k+1)!$. Así que basta ver que

$$i_{n+1} = n!n - \sum_{k=2}^n i_k(n-k+1)!$$

Pero esto se cumple sencillamente porque $i_{n+1} = (n+1)! - \sum_{k=1}^n i_k(n-k+1)! = (n!n + n!) - \sum_{k=2}^n i_k(n-k+1)! - n! = n!n - \sum_{k=2}^n i_k(n-k+1)!$

²El caso base se cumple trivialmente pues $0!i_2 = \gamma_1 = 1$

Ejercicio 11:

Resulta claro que $\cosh(z) = \sum_{i=0}^{\infty} \frac{1}{(2i)!} z^{2i}$, es decir, la FGE de $\text{SET}_2(\mathcal{Z})$, la clase etiquetada de los conjuntos con una cantidad par de elementos. Análogamente, $\sinh(z) = \sum_{i=0}^{\infty} \frac{1}{(2i+1)!} z^{2i+1}$, luego es la FGE de $\text{SET}_{2+1}(\mathcal{Z})$.

$\cosh(z)^N$ es la FGE de $\text{SEQ}_N(\text{SET}_2(\mathcal{Z}))$, y asociando a cada conjunto la posición donde se encuentra en la secuencia, podemos ver que es una especificación para $\mathcal{MP}_N = \{f : \{1, \dots, k\} \rightarrow \{1, \dots, N\} : k \in \mathbb{N} \text{ y } |f^{-1}(n)| \text{ es par } \forall n \in \mathbb{N}\}$. Es análogo ver que $\sinh(z)^N$ es la FGE de $\text{SEQ}_N(\text{SET}_{2+1}(\mathcal{Z}))$, y que ésta es una especificación para \mathcal{ML}_N ³.

Observemos que cada $f \in \{1, \dots, N\}^{\{1, \dots, k\}}$ puede ser entendida como una evolución de configuraciones de cámaras en k tiempos discretos, si pensamos a $\{1, \dots, N\}$ como el conjunto de pelotas, y $f^{-1}(n)$ como todos los tiempos donde la pelota n cambia de cámara. Estamos asumiendo, como dice la letra, que todas las pelotas comienzan en la cámara A , por lo que la repartición de las pelotas en cada momento queda determinada por la series de cambios discretos. Luego, \mathcal{MP}_N corresponde al conjunto de evoluciones donde todas las pelotas terminan en la cámara A , y \mathcal{ML}_N las evoluciones que dejan llena la cámara B .

Cada elemento de $\text{SEQ}_\ell(\text{SET}_2(\mathcal{Z})) \star \text{SEQ}_{N-\ell}(\text{SET}_{2+1}(\mathcal{Z}))$ con tamaño k puede ser pensado como una cuaterna (f, g, A, B) , con $(f, g) \in \mathcal{MP}_l \times \mathcal{ML}_{N-\ell}$ y (A, B) partición de $\{1, \dots, k\}$. Sea (n_1, \dots, n_r) la ordenación de A y (m_1, \dots, m_{k-r}) la ordenación de B . Podemos pensar nuestra cuaterna como la evolución de configuraciones que en tiempos n_i cambia de cámara la pelota $f(i)$ y que en tiempos m_i cambia de cámara la pelota $g(i) + \ell$. Es decir, logramos establecer una correspondencia entre $\text{SEQ}_\ell(\text{SET}_2(\mathcal{Z})) \star \text{SEQ}_{N-\ell}(\text{SET}_{2+1}(\mathcal{Z}))$ y las evoluciones donde las primeras l pelotas terminan en A y las restantes en B . La generatriz de dichas evoluciones es por lo tanto $(\cosh z)^\ell (\sinh z)^{N-\ell}$.

Para obtener $E_n^{[\ell]}$ basta tomar cualquier cuaterna (f, g, A, B) y considerar todas las quintuplas (f, g, A, B, K) , donde $K \subseteq \{1, \dots, N\}$ con $|K| = \ell$ representa el conjunto de pelotas que terminan en A (las restantes son las que terminan en B). Es decir, si $\{\alpha_1, \dots, \alpha_\ell\}$ es la ordenación de K y $\{\beta_{\ell+1}, \dots, \beta_N\}$ la ordenación de $\{1, \dots, N\} - K$, la evolución queda determinada por la función de cambios:

$$h(x) = \begin{cases} \alpha_{f(i)} & \text{si } x = n_i \\ \beta_{g(i)+\ell} & \text{si } x = m_i \end{cases}$$

Luego la generatriz de $E_n^{[\ell]}$ es $\binom{N}{\ell} (\cosh z)^\ell (\sinh z)^{N-\ell}$.

³La misma definición de \mathcal{MP}_n pero cambiando por $|f^{-1}(n)|$ impar

Ejercicio 15:

La estrategia que proponemos para los prisioneros es la siguiente:

El prisionero i , al llegar a la cabina, elige dar vuelta la carta que está guardada en la gaveta i . A partir de ahí, siempre decide abrir la gaveta asociada al número indicado en la carta que recién dio vuelta. Continúa hasta abrir cincuenta gavetas o encontrar su propia carta.

Ahora, nos disponemos a probar que con ésta estrategia, la probabilidad de que los 100 prisioneros hallen su carta excede 0.30 -suponiendo que la elección de la permutación asociada a la distribución de cartas es aleatoria-.

Consideremos $\sigma \in \mathcal{P}_{100}$ la permutación asociada a la distribución de cartas en gavetas.

La primera observación es que al abrir las gavetas utilizando la estrategia descrita, el prisionero i abre exactamente las cartas $\{\sigma^k(i) : k \in \{1, \dots, 50\}\}$, es decir, los primero 50 elementos del ciclo de i asociado a σ , que notamos $\langle i \rangle_\sigma$. Entonces, podemos afirmar lo siguiente:

La estrategia propuesta será exitosa para la distribución asociada a σ si y solo si todos los ciclos que conforman σ tienen a lo sumo 50 elementos.

Es claro que $\{\sigma \in \mathcal{P} : \sigma \text{ tiene algún ciclo de tamaño mayor a } 50\}$ puede identificarse con la clase combinatoria etiquetada $\mathcal{P}' = \text{SEQ}_{\geq 1}(\text{CYC}_{>50}(\mathcal{Z})) \star \text{SEQ}(\text{CYC}_{\leq 50}(\mathcal{Z}))$. Por lo tanto, la probabilidad de éxito de la estrategia es $1 - \frac{|\mathcal{P}'_{100}|}{|\mathcal{P}_{100}|}$

La cantidad $|\mathcal{P}'_{100}|$ se obtiene de manera sencilla. Debemos elegir un $k \geq 51$, un conjunto de k etiquetas, un ciclo con esas k etiquetas, y una permutación con las restantes $n - k$ etiquetas. $|\mathcal{P}'_{100}| = \sum_{k=51}^{100} \binom{100}{k} (k-1)! (100-k)! = \sum_{k=51}^{100} \frac{n!}{k}$ Luego, contamos $1 - \frac{\sum_{k=51}^{100} \frac{n!}{k}}{n!} = 1 - \sum_{k=51}^{100} \frac{1}{k} \approx 0.312$.

Como algo anecdótico, simulé en Python en el orden de 100000 corridas de la situación, aplicando la estrategia. El número de éxitos es significativamente mayor al teórico -0.334-, lo que muestra las limitaciones del algoritmo de generación pseudo-aleatoria de permutaciones utilizado

Código de Simulación Ejercicio 15:

```
1 from numpy.random import permutation
2 success = 0
3 for j in range(10000):
4     k = 0
5     box = permutation([i for i in range(100)])
6     while k < 3:
7         count = 0
8         current = k
9         found = False
10        while count < 50:
11            if box[current] == k:
12                found = True
13                break
14            current = box[current]
15            count = count + 1
16        if not found:
17            break
18        if k == 2:
19            success = success + 1
20        k = k + 1
21 print(success)
```