Cut and Choose - Teoría de juegos

Bernardo Daniel Dávalos

6 de diciembre de 2022

INFORME

Enunciado. Se tienen dos jugadores $I = \{1, 2\}$ y una torta de tres sabores etiquetados por $X = \{1, 2, 3\}$. El jugador 1 realiza un corte conexo sabiendo la preferencia de 2, y dos elige el que más le conviene. Los cortes pueden realizarse solo de forma vertical, ya que la representación de la torta es una proyección unidimensional. Los jugadores 1 y 2 (se referirán a ellos por J_1 y J_2 para abreviar) tienen asociados una función de preferencia $f_i : A \subset X \to [0, 10]$ (fija para todo el estudio), la cual se obtiene «preguntando» sus gustos (ver Cuadro 1) en una escala del 1 al 10.

Jug.	**Sabor 1**	**Sabor 2**	**Sabor 3**
1	3	5	8
2	6	0	10

Cuadro 1: Valores de los gustos de cada jugador

Como nos interesa saber las preferencias de los jugadores siguiendo la condición probabilidad, es decir, la suma de los eventos sea 1, normalizamos los valores. Por eso se utiliza la función de pago se ve reflejada por $\mu_i(p_k) = \frac{f_i(p_k)}{s}$, donde $s = \sum_{j=1}^{T} f_i(p_j)$. De aquí que la función de pagos se convierte en $u_1(1) = 0{,}1875$, $u_1(2) = 0{,}3125$ y $u_1(3) = 0{,}5$ para J_1, mientras que $u_2(1) = 0{,}375$, $u_2(2) = 0$ y $u_2(3) = 0{,}625$ para J_2. De manera que $\sum_k u_1(k) = \sum_k u_2(k) = 1$.

El primer jugador debe cortar la torta de manera que ambos jugadores se beneficien, pero buscando el mayor beneficio propio. El juego «no tiene memoria», por lo que no se aplica una «venganza» del segundo jugador para realizar un corte más justo de parte del primero, por tanto puede hacerse de manera tal que $\mu_1(p_k) \geq \mu_2(p_k)$.

Método. Dado que representamos una torta por una concatenación de sabores $S = \{1, 2, 3\}$ de la forma $L(n) = \{s \in S^* : longitud(s) = n\}$, donde n representa la longitud de la cadena s, podemos representarla mediante un vector $s = (s_1, \ldots, s_n)$. Por un ejercicio sabemos que el juego de Cut and Choose es proporcional (para dos jugadores), por lo que $u_i(A_i) \geq \frac{1}{2}$, y además es libre de envidia, por lo que aseguramos que los cortes sean beneficiosos para cada jugador, es decir, de forma tal que

$$u_1(A_1) \geq u_1(A_2) \quad \text{y} \quad u_2(A_2) \geq u_2(A_1) \tag{1}$$

donde claramente $A_1 \cup A_2 = X$ y $A_1 \cap A_2 = \emptyset$, ya que el corte es conexo. La forma de asegurar el mejor corte para J_1 es, considerando la partición $\mathcal{P}(X) \ni A_1, A_2$, y con (1), además que $u_1(A_1) = \max\limits_{A \in \mathcal{P}(X)} \{u_1(A) : u_2(A) \geq u_2(X \setminus A)\}$.

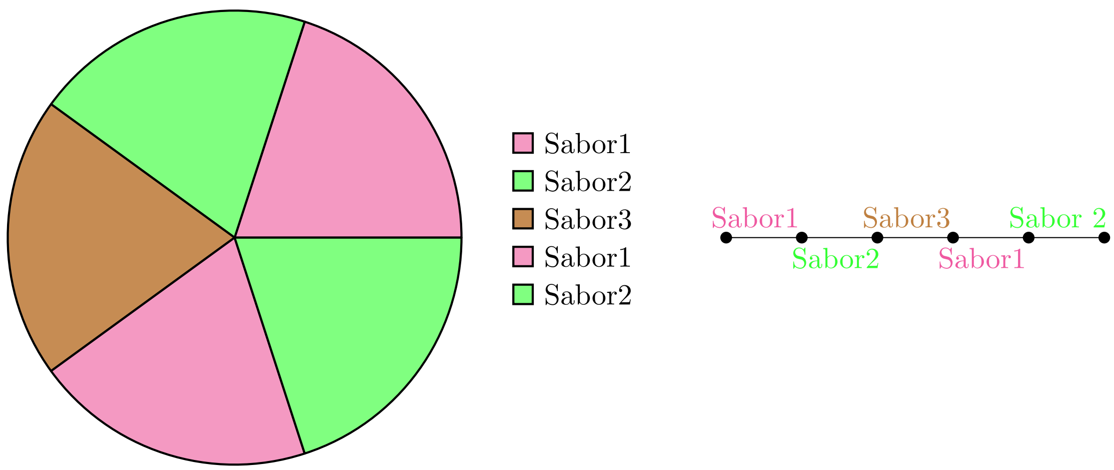

Figura 1: El pastel del ejemplo puede ser representado por el segmento

Código. El programa que simule el corte está escrito en **Prolog**, en el archivo corte.pl. Los valores de T y N se generan de forma pseudo-aleatoria entre $1 \leq T \leq 10$ y $3 \leq N \leq 10$ en `leer(T,N)`, donde T es el tamaño de la torta y N es la cantidad de iteraciones que se realiza. Los datos de preferencia de J_1 y J_2 están precargados y normalizados como `u1(N)` y `u2(N)`, que generará la lista normalizada de cada jugador. La torta se representa mediante una lista de los sabores 1, 2 o 3; es decir, si tenemos una torta como la Figura 1, entonces podemos representar mediante la lista `[1,2,3,1,2]`.

Dada una torta, la clausula `elegir(Torta,P,S1,S2)` retorna todas las particiones `P=(P1,P2)` (donde `[P1|P2]=Torta`), los pagos para J_1 `S1=(S1P1,S1P2)` que representa al par $(u_1(P_1), u_1(P_2))$, y lo mismo `S2` para J_2. En tanto la cláusula `mejor_eleccion(Torta,P,S)` retorna las particiones `P=(P1,P2)` y los pagos `S=(S1,S2)` para las condiciones antedichas, dadas en (1). Se puede ejecutar cada ciclo de tamaños T y N aleatorios con la entrada `go.` en la consulta del programa.

Conclusiones. Se evidencia que al final de cada ciclo de tortas de tamaño T y N iteraciones, hay una gran diferencia de beneficios $B_1 \ll B_2$, donde B_1, B_2 son los beneficios de J_1 y J_2, respectivamente, en este caso. Esto ocurre principalmente porque ambos jugadores prefieren con casi el mismo valor el sabor 3, luego como J_2 elige primero, por lo general tendrá más beneficio que J_1. Probablemente sea más aleatorio al cambiar los valores de la matriz dada en el cuadro 1, pero casi siempre $B_1 < B_2$.