Introducción a los códigos y a la teoría de la información

Fernando Martínez
fernando.martinez@upc.edu

Departament de Matemàtiques • Universitat Politècnica de Catalunya

15 de febrero de 2022

Capítulos 1 y 2 de Introduction to Data Compression **Sayood, Khalid** Morgan Kaufmann, 2012, 4th ed.

Ejemplo

UTF-16/UTF-8 Codifican símbolos Unicode (lista reducida de símbolos en Wikipedia o lista completa Unicode 12.1). Los valores fijos al principio de cada byte garantizan la descodificación única, pues son distintos en función de la posición del byte en la cadena.

Rango Unicode	Valor escalar	UTF-16	UTF-8	Notas
U+000000 – U+00007F	0xxxxxxx	00000000 0xxxxxxx	0xxxxxxx	Equivalente a ASCII. Símbolos de un único byte con el bit más significativo igual a 0
U+000080 – U+0007FF	00000yyy yyxxxxxx	00000yyy yyxxxxxx	110yyyyy 10xxxxxx	Extensiones Latín, griego, cirílico...
U+000800 – U+00FFFF	zzzzyyyy yyxxxxxx	zzzzyyyy yyxxxxxx	1110zzzz 10yyyyyy 10xxxxxx	Símbolos chinos, japoneses, coreanos; símbolos matemáticos, formas geométricas...
U+010000 – U+10FFFF	000uuuuu zzzzyyyy yyxxxxxx	110110ww wwzzzzyy 110111yy yyxxxxxx (wwww = uuuuu - 1)	11110uuu 10uuzzzz 10yyyyyy 10xxxxxx	Símbolos musicales, piezas de ajedrez, cartas de baraja francesa, fichas de dominó, emojis...

Códigos

Definición

Alfabeto q-ario: Conjunto $A = \{a_1, \ldots, a_q\}$ de q elementos que reciben el nombre de letras.

Si no se especifica lo contrario, el alfabeto será binario, $q = 2$.

Definición

Palabra c de longitud $l(c) = k$, $k > 0$ entero, sobre el alfabeto A: Concatenación de k letras de A.

$$c = a_{i_1} a_{i_2} \ldots a_{i_k}, \qquad a_{i_j} \in A.$$

Definición

B^* conjunto de todas las concatenaciones de elementos del conjunto B.

Definición

Código $\mathcal{C}$ (q-ario) sobre un alfabeto A (q-ario): Subconjunto finito de palabras de A tal que:

$$\forall m \in \mathcal{C}^*, \text{ si } m = c_1 \dots c_r = \tilde{c}_1 \dots \tilde{c}_s,\ c_i, \tilde{c}_j \in \mathcal{C}$$

entonces

$$r = s \text{ y } c_i = \tilde{c}_i,\ i = 1, ..., r.$$

La descomposición de mensajes en palabras del código es única.

Definición

Los elementos de $\mathcal{C}^*$ reciben el nombre de mensajes.

Definición

Código de bloque: todas las palabras tienen la misma longitud

Ejemplo

- $\mathcal{C}_1 = \{001, 010, 100\}$

$$001001010100 = 001/001/010/100$$

- $\mathcal{C}_2 = \{10, 11, 01, 1011\}$

$$01101111 = 01/1011/11 = 01/10/11/11$$

- $\mathcal{C}_3 = \{0, 01, 011, 111\}$

$$011111111 = 011/111/111,$$

hasta que no recibimos el último bit no podemos descomponer en palabras.

- $\mathcal{C}_4 = \{0, 10, 110, 111\}$

$$111111110 = 111/111/110$$

a medida que recibimos una palabra podemos identificarla.

Definición

Código prefijo o instantáneo: ninguna palabra del código es prefijo de otra palabra del código.

1. Cualquier conjunto de palabras que tenga la propiedad de ser prefijo es código.

   D/. Veamos por inducción sobre r que si $c_1 \dots c_r = \tilde{c}_1 \dots \tilde{c}_s$ entonces $r = s$ y $c_i = \tilde{c}_i$.

   1. $r = 1$, $c_1 = \tilde{c}_1 \dots \tilde{c}_s$.
      - Si $s > 1$, $\tilde{c}_1$ prefijo de c_1, imposible!
      - Si $s = 1$, $\tilde{c}_1 = c_1$.
   2. Supongamos que es cierto hasta $r-1$, veamos que es cierto para r.

      $$c_1 \dots c_r = \tilde{c}_1 \dots \tilde{c}_s$$

      - Si $l(c_1) < l(\tilde{c}_1)$, c_1 prefijo de $\tilde{c}_1$, imposible!
      - Si $l(c_1) > l(\tilde{c}_1)$, $\tilde{c}_1$ prefijo de c_1, imposible!
      - Si $l(c_1) = l(\tilde{c}_1)$, $c_1 = \tilde{c}_1$ entonces $c_2 \dots c_r = \tilde{c}_2 \dots \tilde{c}_s$, por la H.I. $r - 1 = s - 1$ y $c_i = \tilde{c}_i$, $i = 2, ..., r$.

   Por lo tanto, $r = s$ y $c_i = \tilde{c}_i$, $i = 1, ..., r$.
2. Los conjuntos de palabras de igual longitud son códigos de bloque.
3. En un código prefijo podemos identificar las palabras al recibirlas.

Definición

Sea $S = \{s_1, ..., s_n\}$ un conjunto de símbolos y $\mathcal{C} = \{c_1, ..., c_n\}$ un código.
Una codificación f de S en $\mathcal{C}$ es una aplicación biyectiva $f : S \longrightarrow \mathcal{C}$.

Ver `cara-cruz.ipynb`

Teorema de Kraft

Teorema

Sea $\mathcal{C}$ un código q-ario, entonces

$$\sum_{i=1}^{n} \frac{1}{q^{l_i}} \leq 1 \tag{1}$$

siendo l_i, $i = 1...n$, las longitudes de las palabras del código.

Notas:

- La desigualdad 1 recibe el nombre de desigualdad de Kraft-McMillan.
- Que se cumpla la desigualdad de Kraft-McMillan para un conjunto es una condición necesaria pero no suficiente para que dicho conjunto sea un código.

Demostración.

Tomaremos $q = 2$ y definimos $\alpha \equiv \sum_{i=1}^{n} \frac{1}{2^{l_i}}$. Observemos que:

- Si $\alpha > 1$, α^k crece exponencialmente; si α^k no crece exponencialmente entonces $\alpha \leq 1$.

$$\begin{aligned} \alpha^k &= \left(\sum_{i=1}^{n} 2^{-l_i}\right)^k = \left(\sum_{i=1}^{n} 2^{-l_i}\right) \cdots \left(\sum_{i=1}^{n} 2^{-l_i}\right) \\ &= \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \cdots \sum_{i_k=1}^{n} 2^{-l_{i_1}} \cdots 2^{-l_{i_k}} = \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \cdots \sum_{i_k=1}^{n} 2^{-\left(l_{i_1}+\cdots+l_{i_k}\right)} \end{aligned}$$

$l_{i_1} + \cdots + l_{i_k}$ es la longitud de k palabras del código; este valor será:

- mayor o igual que k ya que $l_i \geq 1$,
- menor o igual que $k \cdot l$ siendo l la longitud de la palabra más larga.

Por lo tanto

$$\alpha^k = \sum_{j=k}^{k \cdot l} A_j 2^{-j}$$

siendo A_j el número de mensajes de k palabras cuya suma de longitudes es j.
Notemos que $A_j \leq 2^j$, entonces $\alpha^k = \sum_{j=k}^{k \cdot l} A_j 2^{-j} \leq \sum_{j=k}^{k \cdot l} 2^j 2^{-j} = \sum_{j=k}^{k \cdot l} 1 \leq k\,l$.
O sea $\alpha^k \leq k\,l$, y por lo tanto α^k sólo puede crecer linealmente con k. Al no crecer exponencialmente podemos concluir $\alpha \leq 1$. □

Teorema (Kraft)

Sea A un alfabeto q-ario y $0 < l_1 \leq l_2 \leq \cdots \leq l_n$ enteros tales que verifican la desigualdad de Kratf (1). Entonces existe un código q-ario sobre A prefijo con n palabras de longitudes $0 < l_1 \leq l_2 \leq \cdots \leq l_n$.

Demostración.

Tomaremos $q = 2$. Lo demostraremos por inducción sobre n, el número de palabras del código.
•**n = 1**: $c_1 = 0 \dots 0$, $\mathcal{C} = \{c_1\}$
•**n ≥ 2**: $0 < l_1 \leq l_2 \leq \cdots \leq l_n$ con $\sum_{i=1}^{n} \frac{1}{2^{l_i}} \leq 1$.
Por H.I. existe un código prefijo $\tilde{\mathcal{C}} = \{c_1, \dots, c_{n-1}\}$ con longitudes $0 < l_1 \leq l_2 \leq \cdots \leq l_{n-1}$ (es obvio que $\sum_{i=1}^{n-1} \frac{1}{2^{l_i}} < 1$).
Veamos que existe una palabra c_n de longitud l_n tal que

1. c_i no es prefijo de c_n, $i = 1, ..., n-1$
2. c_n no es prefijo de c_i, $i = 1, ..., n-1$

Contemos:

- # de palabras de longitud l_n con c_i como prefijo: $2^{l_n - l_i}$,
- # de palabras de longitud l_n con algún c_i como prefijo: $\sum_{i=1}^{n-1} 2^{l_n - l_i}$,
- # de palabras de longitud l_n que NO tienen algún c_i como prefijo: $2^{l_n} - \sum_{i=1}^{n-1} 2^{l_n - l_i}$.

Como

$$2^{l_n} - \sum_{i=1}^{n-1} 2^{l_n - l_i} = 2^{l_n} \left(1 - \sum_{i=1}^{n-1} 2^{-l_i} \right) > 0$$

existe alguna palabra de longitud l_n, c_n, que no tiene como prefijo a ninguna palabra c_i.
Además, como $l_n \geq l_i$ c_n no puede ser prefijo de ninguna otra palabra c_i.
El código buscado es $\mathcal{C} = \tilde{\mathcal{C}} \cup \{c_n\}$. □

Corolario

Dado un código siempre podemos encontrar otro código prefijo cuyas palabras tengan las mismas longitudes.

Corolario

Dado un código prefijo con $\sum_{i=1}^{n} \frac{1}{2^{l_i}} < 1$ *siempre podemos añadir una palabra de longitud máxima de forma que el nuevo conjunto de palabras siga siendo un código prefijo.*

Si no se dice lo contrario supondremos que todos los códigos son prefijos.

Si usamos un código cuyas palabras son de longitud variable tenemos el problema de guardarlo: sería necesario almacenar las longitudes de las palabras* así como las palabras del código. Por ejemplo si quisiéramos codificar una cadena de ADN

$$\{(A, 2, 01), (T, 1, 1), (C, 3, 000), (G, 4, 001)\}$$

Para no enviar el código, este se genera canónicamente con las longitudes (Ver RFC 1951 DEFLATE Compressed Data Format Specification, 3.2.2. Use of Huffman coding in the deflate format):

I) Lexicográficamente las palabras más cortas preceden a las más largas.

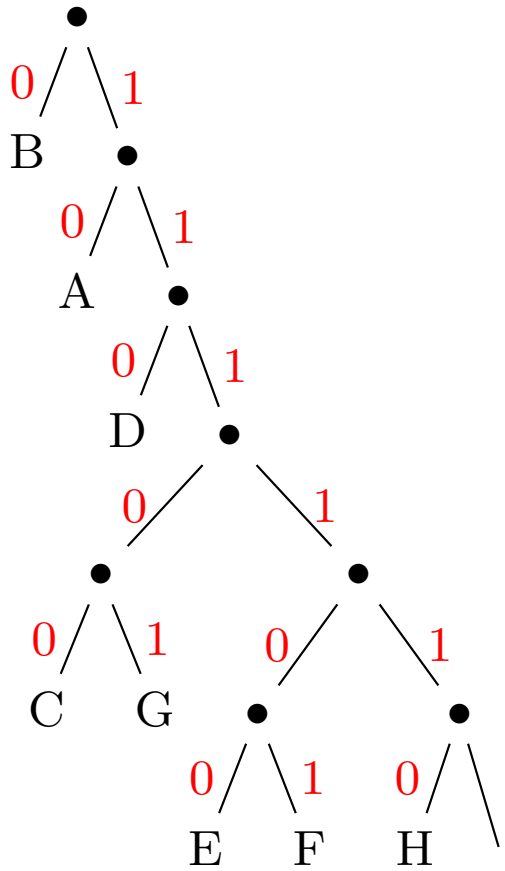
II) Todas las palabras del mismo tamaño tienen valores consecutivos lexicográficamente y se asignan en el mismo orden que los símbolos que representan.

Coloquialmente: El árbol crece por la derecha, se poda por la izquierda.

*Es necesario enviar la longitud para saber qué bits hay que tomar ya sea porque todos se envían concatenadamente y/o porque la *unidad mínima* es el byte.

Ejemplo

Un código $\mathcal{C} = \{A, B, C, D, E, F, G, H\}$ cuyas palabras tiene longitudes $\{2, 1, 5, 3, 6, 6, 5, 6\}$ se construiría:



El código reconstruido será $\{10, 0, 11100, 110, 111100, 11101, 111110\}$.

Ver `01_Codificacion.ipynb`

Entropía

Definición

Una fuente (sin memoria) $\mathcal{S} = (S, \mathcal{P})$ es un par $S = \{s_1, ..., s_n\}$, conjunto de símbolos, y $\mathcal{P}$, distribución de probabilidad (ddp) para S, $(\mathcal{P}(s_i) = p_i > 0, \sum p_i = 1)$.

Podemos pensar una fuente como una caja negra que emite símbolos s_i con probabilidad p_i, que es independiente de lo que ha emitido con anterioridad.

En la práctica, no siempre es así, sino que muchas veces sí que depende de la historia. Por ejemplo, si los símbolos son letras de un texto la probabilidad de que después de aparecer una letra `q` aparezca una `u` es casi 1.

Recordemos la definición de codificación:

Definición

Sea $S = \{s_1, ..., s_n\}$ un conjunto de símbolos y $\mathcal{C} = \{c_1, ..., c_n\}$ un código.
Una codificación f de S en $\mathcal{C}$ es una aplicación biyectiva $f : S \longrightarrow \mathcal{C}$.

Definición

Longitud media de una codificación f:

$$\tilde{l} = \sum l(f(s_i))\, p_i = \sum l(c_i)\, p_i$$

Aunque $\mathcal{S}$ y $\mathcal{C}$ sean las mismas, $\tilde{l}$ depende de f.

Supondremos codificaciones que asignan a los símbolos más probables las palabras del código más cortas.

Es esta situación diremos, que $\tilde{l} = \sum l(c_i)\, p_i$ es la **longitud media del código**.

Objetivo

Dada una fuente hallar un código tal que su longitud media sea mínima.

Imaginemos la siguiente situación:

1. Dos personas tienen cada una de ellas una caja negra que emite símbolos de una fuente $\mathcal{S} = (\{\Omega, \Lambda\}, \{0{,}99, 0{,}01\})$.
2. Las cajas negras van emitiendo símbolos y dichas personas juegan a ver quién es capaz de nombrar primero los dos símbolos de la fuente.
3. Supongamos que el jugador A ha visto Ω y el jugador B ha visto Λ
4. Si hay que apostar por el jugador que ganará, ¿por cual apostaríais?

El jugador B tiene más *información* sobre la fuente.

Definición

Denotaremos por $I(p)$ la *información* obtenida de una fuente cuando emite un símbolo con una probabilidad p.

Requisitos para $I(p)$:

I) $I(p) \geq 0$,

II) $I(p)$ función continua,

III) $I(p_1 p_2) = I(p_1) + I(p_2)$.

Teorema

$I(p)$, $0 < p \leq 1$, satisface los requisitos anteriores si y solamente si

$$I(p) = K \log \frac{1}{p}, \qquad K \text{ constante positiva.}$$

Tomaremos la base del logaritmo igual a 2 y $K = 1$. En este caso la información se mide en bits.

Definición

$$I(p) = \log_2 \frac{1}{p} = \log \frac{1}{p} = -\log p$$

Definición

Entropía de una fuente: promedio de la información de los símbolos de una fuente:

$$H(\mathcal{S}) = \sum p_i I(p_i) = \sum p_i \log \frac{1}{p_i} = -\sum p_i \log p_i$$

También es usual escribir $H(p_1, ..., p_n)$.

Nota: Si $p_i = 0$, $p_i \log(p_i) \equiv 0$,

Ejemplo

I) Suceso seguro $H(1) = 0$.

II) Sucesos equiprobables, $p_i = 1/n$, $H(p_1, ..., p_n) = \log n$.
Si $n = 2^k$, $H(p_1, ..., p_n) = k$

III) $H(\frac{31}{32}, \frac{1}{32}) = 0,20$. Si codificamos:

- 1 símbolo, $\tilde{l}(\mathcal{C}) = 1$, 1 bit por símbolo.
- 2 símbolos: ++:0, +-:10, -+=110, −:111 las probabilidades son ++:$\frac{31}{32}\frac{31}{32}$, +-:$\frac{31}{32}\frac{1}{32}$, -+:$\frac{1}{32}\frac{31}{32}$, −:$\frac{1}{32}\frac{1}{32}$ y $\tilde{l}(\mathcal{C}) = 1,09$, o sea $0,55$ bits por símbolo.
- 3 símbolos con 0, 100, 101, 110, 11100, 11110, 11101, 11111 entonces $\tilde{l}(\mathcal{C}) = 1,19$, o sea $0,40$ bits por símbolo.

Ver `cara-cruz.ipynb`

Observaciones:

I) La entropía mide la *incertidumbre* de la fuente.

II) La entropía se puede interpretar como el número medio de bits necesarios para representar los símbolos de la fuente.