Técnicas Fundamentales

Algoritmos codiciosos II

Segundo semestre 2022

IIC2283

Prof. Nicolás Van Sint Jan

Recordatorio: Problema de almacenamiento

Sea Σ un alfabeto.

Una **Σ-codificación** es una función $\tau : \Sigma \to \{0,1\}^*$ tal que $\tau(a) \neq \epsilon$ para todo $a \in \Sigma$.

La **extensión** $\hat{\tau}$ de una Σ-codificación τ a todas las palabras $w \in \Sigma^*$ se define como:

$$\hat{\tau}(w) = \begin{cases} \varepsilon & w = \varepsilon \\ \tau(a_1) \cdots \tau(a_n) & w = a_1 \cdots a_n \text{ con } n \geq 1 \end{cases}$$

Nos interesa que:

1. $\hat{\tau}$ sea **inyectiva**.
2. τ sea de **largo variable**.

¿Por qué nos interesa esto?

Recordatorio: Codificación libre de prefijos

Lema

Si existen $w_1, w_2 \in \Sigma^*$ tales que $w_1 \neq w_2$ y $\hat{\tau}(w_1) = \hat{\tau}(w_2)$, entonces existen $a, b \in \Sigma$ tales que $a \neq b$ y $\tau(a)$ es un prefijo de $\tau(b)$

Decimos que una Σ-codificación τ es **libre de prefijos** si para cada $a, b \in \Sigma$ tales que $a \neq b$ se tiene que $\tau(a)$ no es un prefijo de $\tau(b)$.

Corolario

Si τ es una codificación libre de prefijos, entonces $\hat{\tau}$ es una **función inyectiva**.

Recordatorio: Frecuencias relativas de los símbolos

Fije $w \in \Sigma^*$.

Para $a \in \Sigma$ definimos $\mathrm{fr}_w(a)$ como la **frecuencia relativa** de a en w, vale decir

$$\mathrm{fr}_w(a) = \frac{\#\text{ de apariciones de } a \text{ en } w}{|w|}$$

Para una Σ-codificación τ definimos el **largo promedio** para w como:

$$\mathrm{lp}_w(\tau) \quad = \quad \sum_{a \in \Sigma} \mathrm{fr}_w(a) \cdot |\tau(a)|$$

Problema de optimización a resolver

Dado $w \in \Sigma^*$, encontrar una Σ-codificación τ libre de prefijos que minimice el valor $\mathrm{lp}_w(\tau)$

Recordatorio: Frecuencias relativas de los símbolos

Fije $w \in \Sigma^*$.

Para $a \in \Sigma$ definimos $\mathrm{fr}_w(a)$ como la **frecuencia relativa** de a en w, vale decir

$$\mathrm{fr}_w(a) = \frac{\#\text{ de apariciones de } a \text{ en } w}{|w|}$$

Para una Σ-codificación τ definimos el **largo promedio** para w como:

$$\mathrm{lp}_w(\tau) \quad = \quad \sum_{a \in \Sigma} \mathrm{fr}_w(a) \cdot |\tau(a)|$$

Problema de optimización a resolver

Dado $w \in \Sigma^*$, encontrar una Σ-codificación τ libre de prefijos que minimice el valor $\mathrm{lp}_w(\tau)$

Recordatorio: Frecuencias relativas de los símbolos

Decimos que $f : \Sigma \to (0, 1)$ es una **función de frecuencias relativas** para Σ si se cumple que $\sum_{a \in \Sigma} f(a) = 1$

Dada una función f de frecuencias relativas para Σ y una Σ-codificación τ, el **largo promedio de τ para f** se define como:

$$\mathrm{lp}_f(\tau) \quad = \quad \sum_{a \in \Sigma} f(a) \cdot |\tau(a)|$$

Problema (correcto) de optimización a resolver

Dado f una función de frecuencias relativas para Σ, encontrar una Σ-codificación τ libre de prefijos que minimice el valor $\mathrm{lp}_f(\tau)$

Outline

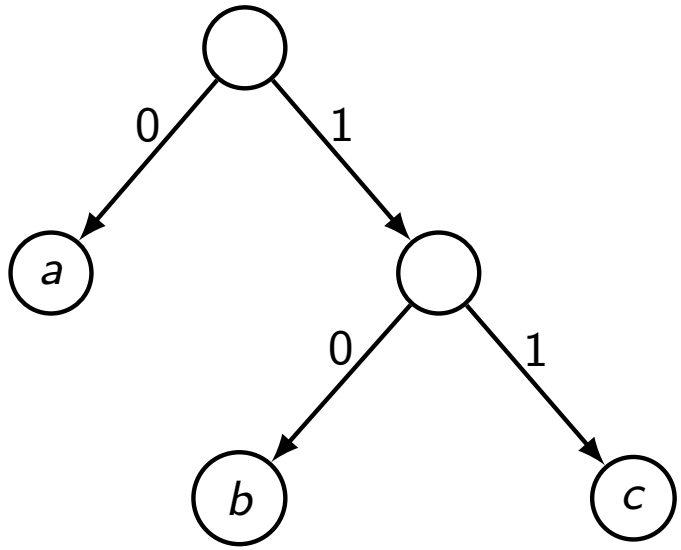
Algoritmos codiciosos: Codificaciones (cont.)

Outline

Algoritmos codiciosos: Codificaciones (cont.)

Un ingrediente fundamental: codificaciones como árboles

Representamos la codificación $\tau(a) = 0$, $\tau(b) = 10$, $\tau(c) = 11$ como un **árbol binario**:



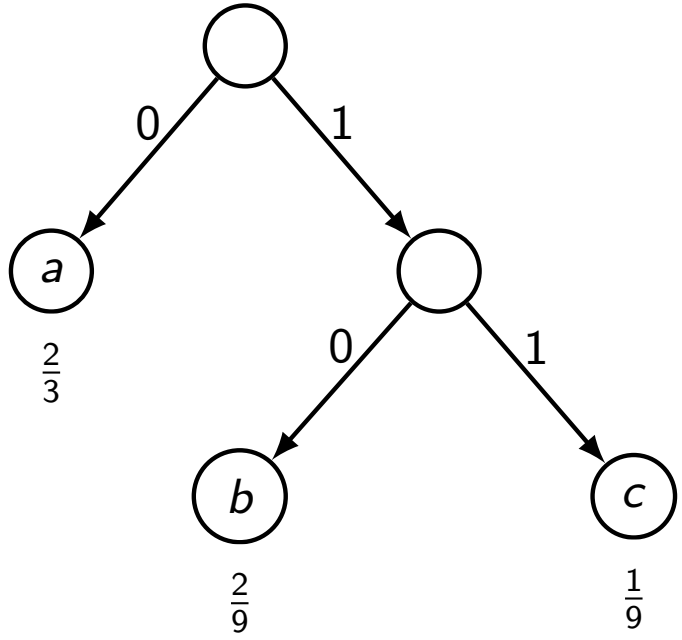
Un ingrediente fundamental: codificaciones como árboles

Si una Σ-codificación τ es **libre de prefijos**, entonces el árbol que la representa satisface las siguientes propiedades.

- Cada hoja tiene como etiqueta un elemento de Σ, y estos son los únicos nodos con etiquetas
- Cada símbolo de Σ es usado exactamente una vez como etiqueta
- Cada arco tiene etiqueta 0 ó 1
- Si una hoja tiene etiqueta e y las etiquetas de los arcos del camino desde la raíz hasta esta hoja forman una palabra $w \in \{0,1\}^*$, entonces $\tau(e) = w$

Codificaciones como árboles y las frecuencias relativas

Podemos agregar al árbol binario que representa una codificación τ la información sobre las **frecuencias relativas** dadas por una función f:



Llamamos a este árbol abf(τ, f).

Alguna propiedades importantes

Ejercicios

Sea f una función de frecuencias relativas para Σ y τ una Σ-codificación libre de prefijos que minimiza la función $\mathrm{lp}_f(x)$.

1. Sean u y v dos hojas en $\mathrm{abf}(\tau, f)$ con etiquetas a y b, respectivamente. Demuestre que si el camino de la raíz a u es más corto que el camino de la raíz a v, entonces $f(a) \geq f(b)$

2. Demuestre que cada nodo interno en $\mathrm{abf}(\tau, f)$ tiene dos hijos

3. Sean $a, b \in \Sigma$ tales que $a \neq b$, $f(a) \leq f(b)$ y $f(b) \leq f(e)$ para todo $e \in (\Sigma \setminus \{a, b\})$. Demuestre que existe una Σ-codificación τ' libre de prefijos tal que $\mathrm{lp}_f(\tau') = \mathrm{lp}_f(\tau)$ y las hojas con etiquetas a y b en $\mathrm{abf}(\tau', f)$ son hermanas
   - Vale decir, existe $w \in \{0,1\}^*$ tal que $\tau'(a) = w0$ y $\tau'(b) = w1$

Calculando el mínimo de $\mathrm{lp}_f(x)$

Vamos a ver un **algoritmo codicioso** para calcular una Σ-codificación τ libre de prefijos que minimiza $\mathrm{lp}_f(x)$

- El algoritmo calcula la **codificación de Huffman**.

El algoritmo tiene los ingredientes mencionados de un algoritmo codicioso.

1. **Función objetivo a minimizar**: $\mathrm{lp}_f(x)$
2. **Función de selección**: elige los dos símbolos de Σ con menor frecuencia relativa, los coloca como hermanos en el árbol binario que representa la Σ-codificación óptima, y continua la construcción con el resto de los símbolos de Σ

Será necesario realizar una demostración para probar que el algoritmo es correcto.

Algoritmo de Huffman

Sea $f : \Sigma \to (0, 1)$ una **función de frecuencias relativas**. Supongamos que:

1. Representamos a las funciones como conjuntos de pares ordenados
2. f tiene al menos dos elementos en el dominio.

El siguiente algoritmo obtiene la Σ-codificación τ de Huffman (la que minimiza $\mathrm{lp}_f(\tau)$).

Algoritmo de Huffman

CalcularCodificaciónHuffman(f)

Sea Σ el dominio de la función f

if $\Sigma = \{a, b\}$ **then return** $\{(a, 0), (b, 1)\}$

else

Sean $a, b \in \Sigma$ tales que $a \neq b$, $f(a) \leq f(b)$ y
$f(b) \leq f(e)$ para todo $e \in (\Sigma \setminus \{a, b\})$

Sea c un símbolo que **no** aparece en Σ

$g := (f \setminus \{(a, f(a)), (b, f(b))\}) \cup \{(c, f(a) + f(b))\}$

$\tau^\star :=$ **CalcularCodificaciónHuffman**(g)

$w := \tau^\star(c)$

$\tau := (\tau^\star \setminus \{(c, w)\}) \cup \{(a, w0), (b, w1)\}$

return τ

La correctitud de **CalcularCodificaciónHuffman**

Teorema

Si f es una función de frecuencias relativas, Σ es el dominio de f y $\tau = \textbf{CalcularCodificaciónHuffman}(f)$, entonces τ es una Σ-codificación libre de prefijos que minimiza la función $\mathrm{lp}_f(x)$.

Demostración

Vamos a realizar la demostración por inducción en $|\Sigma|$.

Si $|\Sigma| = 2$, entonces la propiedad se cumple trivialmente.

- ¿Por qué?

Suponga entonces que la propiedad se cumple para un valor $n \geq 2$, y suponga que $|\Sigma| = n + 1$.

La demostración del teorema

Demostración

Sean a, b, c, g, $\tau^\star$ y τ definidos como en el código de la llamada **CalcularCodificaciónHuffman**(f), y sea Γ el dominio de g

Como $|\Gamma| = n$ y $\tau^\star =$ **CalcularCodificaciónHuffman**(g), por hipótesis de inducción tenemos que $\tau^\star$ es una Γ-codificación libre de prefijos que minimiza la función $\mathrm{lp}_g(x)$

Dada la definición de τ es simple verificar las siguientes propiedades:

- τ es una codificación libre de prefijos
- Para cada $e \in (\Sigma \setminus \{a, b\})$ se tiene que $\tau(e) = \tau^\star(e)$
- $|\tau(a)| = |\tau(b)| = |\tau^\star(c)| + 1$

La demostración del teorema

Demostración

Por contradicción suponga que τ NO minimiza el valor de la función $\mathrm{lp}_f(x)$

- Vale decir, existe una Σ-codificación τ' libre de prefijos tal que τ' minimiza la función $\mathrm{lp}_f(x)$ y $\mathrm{lp}_f(\tau') < \mathrm{lp}_f(\tau)$

Por la definición de a, b y los ejercicios anteriores podemos suponer que existe $w \in \{0,1\}^*$ tal que $\tau'(a) = w \cdot 0$ y $\tau'(b) = w \cdot 1$.

- Nótese que $w \neq \varepsilon$ puesto que $|\Sigma| \geq 3$.

A partir de τ' defina la siguiente Γ-codificación τ'':

$$\tau'' \quad = \quad (\tau' \setminus \{(a, \tau'(a)), (b, \tau'(b))\}) \cup \{(c, w)\}$$

Tenemos que τ'' es una Γ-codificación libre de prefijos.

- ¿Por qué?

La relación entre $\mathsf{lp}_g(\tau^\star)$ y $\mathsf{lp}_f(\tau)$

$$
\begin{aligned}
\mathsf{lp}_g(\tau^\star) &= \sum_{e \in \Gamma} g(e) \cdot |\tau^\star(e)| \\
&= \left(\sum_{e \in (\Gamma \setminus \{c\})} g(e) \cdot |\tau^\star(e)| \right) + g(c) \cdot |\tau^\star(c)| \\
&= \left(\sum_{e \in (\Sigma \setminus \{a,b\})} f(e) \cdot |\tau(e)| \right) + (f(a) + f(b)) \cdot |\tau^\star(c)| \\
&= \left(\sum_{e \in (\Sigma \setminus \{a,b\})} f(e) \cdot |\tau(e)| \right) + f(a) \cdot |\tau^\star(c)| + f(b) \cdot |\tau^\star(c)| \\
&= \left(\sum_{e \in (\Sigma \setminus \{a,b\})} f(e) \cdot |\tau(e)| \right) + f(a) \cdot (|\tau(a)| - 1) + f(b) \cdot (|\tau(b)| - 1) \\
&= \left(\sum_{e \in \Sigma} f(e) \cdot |\tau(e)| \right) - (f(a) + f(b)) \\
&= \mathsf{lp}_f(\tau) - (f(a) + f(b))
\end{aligned}
$$

La relación entre $\mathsf{lp}_g(\tau'')$ y $\mathsf{lp}_f(\tau')$

$$
\begin{aligned}
\mathsf{lp}_g(\tau'') &= \sum_{e \in \Gamma} g(e) \cdot |\tau''(e)| \\
&= \left(\sum_{e \in (\Gamma \setminus \{c\})} g(e) \cdot |\tau''(e)| \right) + g(c) \cdot |\tau''(c)| \\
&= \left(\sum_{e \in (\Sigma \setminus \{a,b\})} f(e) \cdot |\tau'(e)| \right) + (f(a) + f(b)) \cdot |\tau''(c)| \\
&= \left(\sum_{e \in (\Sigma \setminus \{a,b\})} f(e) \cdot |\tau'(e)| \right) + f(a) \cdot |\tau''(c)| + f(b) \cdot |\tau''(c)| \\
&= \left(\sum_{e \in (\Sigma \setminus \{a,b\})} f(e) \cdot |\tau'(e)| \right) + f(a) \cdot (|\tau'(a)| - 1) + f(b) \cdot (|\tau'(b)| - 1) \\
&= \left(\sum_{e \in \Sigma} f(e) \cdot |\tau'(e)| \right) - (f(a) + f(b)) \\
&= \mathsf{lp}_f(\tau') - (f(a) + f(b))
\end{aligned}
$$

Obteniendo una contradicción

Demostración

Tenemos entonces que

$$\begin{aligned} \mathrm{lp}_g(\tau'') &= \mathrm{lp}_f(\tau') - (f(a) + f(b)) \\ &< \mathrm{lp}_f(\tau) - (f(a) + f(b)) \\ &= \mathrm{lp}_g(\tau^\star) \end{aligned}$$

Concluimos entonces que $\tau^\star$ NO minimiza la función $\mathrm{lp}_g(x)$, lo cual contradice la hipótesis de inducción. $\square$

Dos comentarios finales

La siguiente función calcula la **codificación de Huffman** teniendo como entrada una palabra w:

CalcularCodificaciónHuffman(w)
    if $w = \varepsilon$ **then return** $\emptyset$
    else
        $\Sigma :=$ conjunto de símbolos mencionados en w
        if $\Sigma = \{a\}$ **then return** $\{(a, 0)\}$
        else return CalcularCodificaciónHuffman(fr_w)

¿Cuál es la complejidad de **CalcularCodificaciónHuffman**(f)?

R: $\mathcal{O}(n \log n)$ (considerando que $|f| \in \Theta(|\Sigma|)$).