Proyecto Final

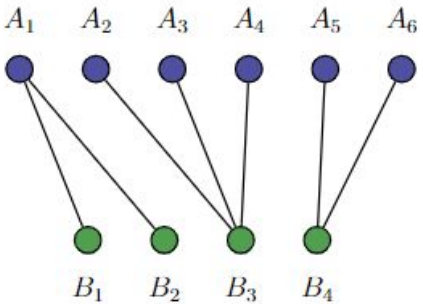
Rodrigo Céspedes
Gabriel Spranger
Benjamín Díaz

Introducción: El problema

Entrada:

$$A = [0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0]$$
$$B = [0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0]$$

Salida:



Primera Parte del Proyecto

Algoritmo Voraz

Tomar la solución óptima local

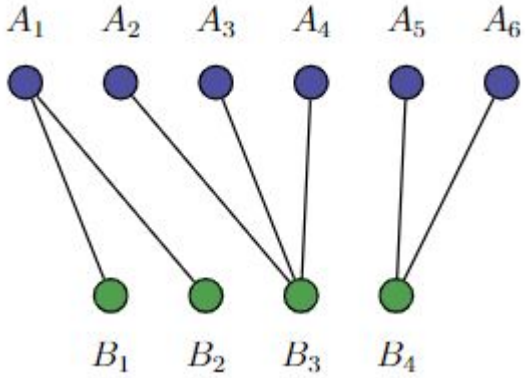
Desición voraz:

Analizar los bloques de tal manera que los valores sumados de un subconjunto de bloques en A tenga el valor más cercano a su análogo en B.

m = cantidad de bloques en A
n = cantidad de bloques en B

Lógica del algoritmo:

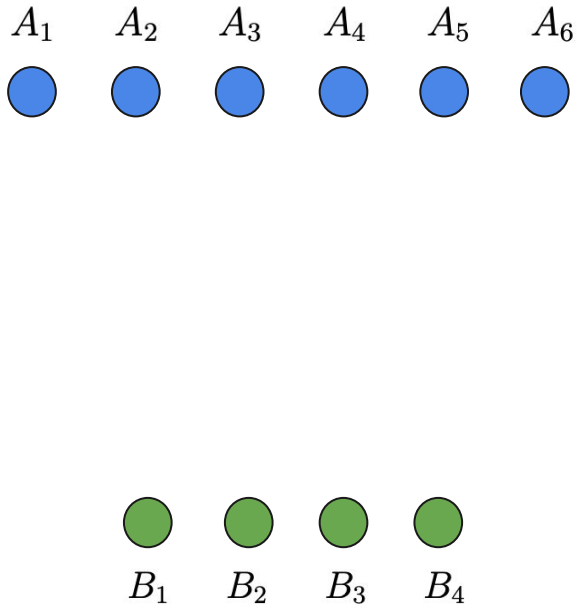
$$A = [0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0]$$
$$B = [0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0]$$



|Grupos en A|
≈
|Grupos en B|

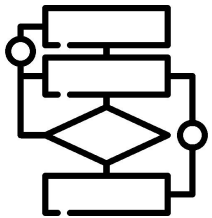
Recurrencia

$$OPT(i,j) = \begin{cases} \frac{A_1}{B_1} & \text{si } i = 1 \text{ y } j = 1 \\ \frac{A_1 + A_2 + \ldots + A_i}{B_1} & \text{si } j = 1 \text{ y } i > 1 \\ \frac{A_1}{B_1 + B_2 + \ldots + B_j} & \text{si } i = 1 \text{ y } j > 1 \\ \min \left\{ \begin{array}{l} \min_{k=j-1}^{1} \{ OPT(i-1,k) + \frac{A_i}{B_{k+1} + \ldots + B_j} \}, \\ \min_{k=1}^{i-1} \{ OPT(k,j-1) + \frac{A_{k+1} + \ldots + A_i}{B_j} \} \end{array} \right. & \text{caso contrario} \end{cases}$$



Algoritmos desarrollados

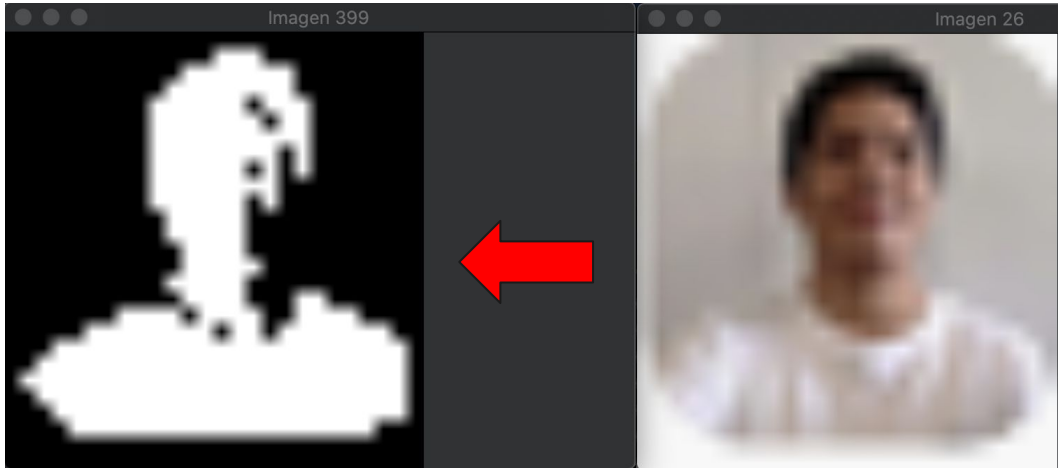
- Algoritmo greedy $O(\max(m,n))$
- Algoritmo recursivo $\Omega(2^{max\{m,n\}})$
- Algoritmo memoizado $O(m^2n)$
- Programación dinamica $O(m^2n)$
- Programación dinámica mejorada $O(m^2n)$



Segunda Parte del Proyecto

Lectura de Imágenes

- Usamos OpenCV.
- Tres métodos: 601, 709 y 240. Cada uno con distintas constantes.
- Ecuación: $Y = c_1R + c_2G + c_3B$
- Si **Y (brightness) < umbral**, entonces 1, sino 0.



Algoritmo Dinámico Mejorado

- Encontramos el matching con menor "peso promedio":

$$\mu = \sum_{i=1}^{n} a_i / \sum_{i=1}^{m} b_i$$

- Donde a_i es el tamaño del i-ésimo bloque en A.
- Donde b_i es el tamaño del i-ésimo bloque en B.

$$\bar{w}(D) = |w(D) - \mu|$$

Peso promedio

Peso del matching

Transformación de Matrices

- Voraz.
- Dinámico.
- Dinámico Mejorado.

Transformación de Imágenes

- Utilizando el matching.

- Pixel por pixel.

Demo de la Animación

¿Qué algoritmo produce la mejor animación?

- Algoritmo Voraz.
- Algoritmo de Programación Dinámica.
- Algoritmo de Programación Dinámica Mejorado.

Gracias

¿Preguntas?

¿Por qué O(m^2n)?

```
for i = 2 to n do                    n veces
    for j = 2 to m do                m veces
        for k = j-1 to 1 do          m veces (upper-bound)
```

¿Por qué $\mathcal{O}(pq^3)$?

- Se tiene una matriz p x q.
- **p** arreglos de 0s y 1s de tamaño **q**, cada uno.
- Algoritmo dinámico y dinámico mejorado corre en O(m^2n)).
- **m < q** y **n < q**.
- Ambos corren también en O(q^3).
- Como el algoritmo corre **p** veces, este corre en $\mathcal{O}(pq^3)$.