

# LE TRAITEMENT D'IMAGES

- IMPLÉMENTATION -

Jonathan Fabrizio

<http://jo.fabrizio.free.fr>

# Implémentation

FASTER – FASTER – FASTER



# Introduction

- Efficacité
  - Taille des images
  - Contrainte sur le temps de réponse
  - Contrainte sur le matériel (téléphone...)
- Solutions
  1. Bien penser ses algorithmes (et ses structures de données)
  2. Revoir son implémentation sur CPU
  3. Éventuellement envisager une implémentation sur GPU

# Introduction

- Efficacité
  - Taille des images
  - Contrainte sur le temps de réponse
  - Contrainte sur le matériel (téléphone...)
- Solutions
  1. Bien penser ses algorithmes (et ses structures de données)
  2. Revoir son implémentation sur CPU
  3. Éventuellement envisager une implémentation sur GPU

# Bien penser ses algorithmes

- Bien choisir les structures de données
  - Représentation des images
  - ...
- Repenser les algorithmes
  - FFT
  - ...

# Bien penser ses algorithmes

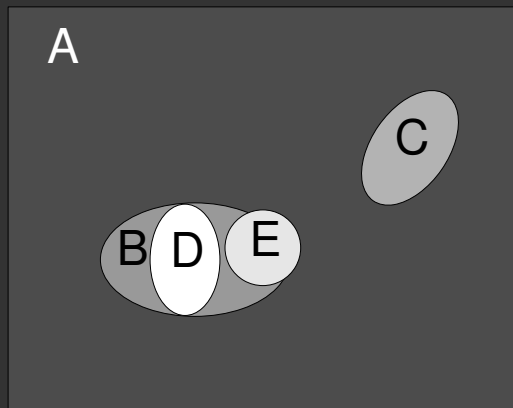
## Représentation des images

- Comment représenter une image
  - Matrice
  - Vecteur
  - Arbre/graph...
    - Max Tree, Min Tree
    - Tree of Shapes...
  - ...
- Matrice, vecteur : bien respecter le cache de la machine !

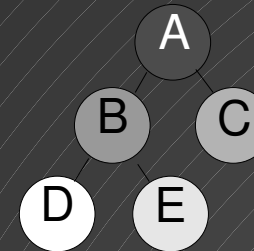
# Bien penser ses algorithmes

## Représentation des images

- Comment représenter une image
  - Max Tree



Image

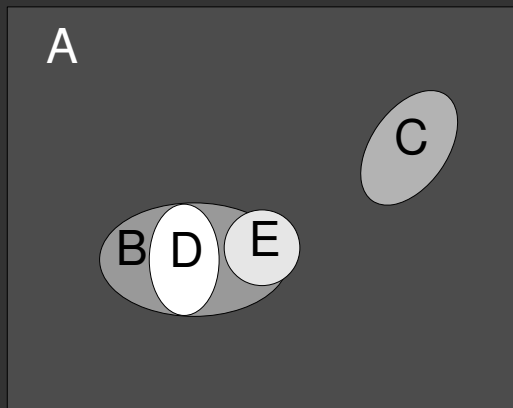


Max-Tree correspondant

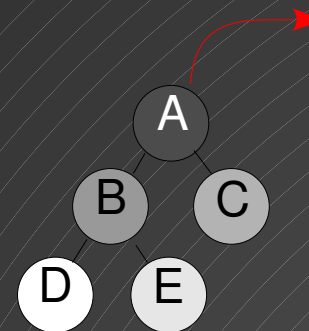
# Bien penser ses algorithmes

## Représentation des images

- Comment représenter une image
  - Max Tree



Image



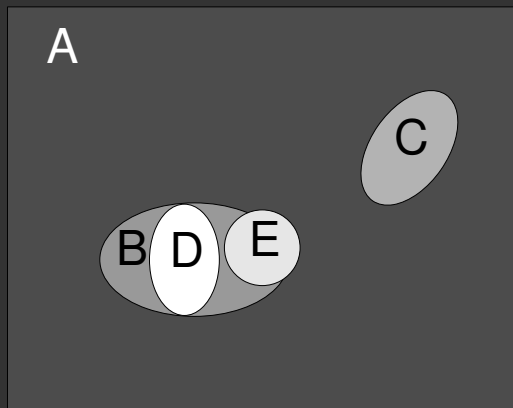
Max-Tree correspondant



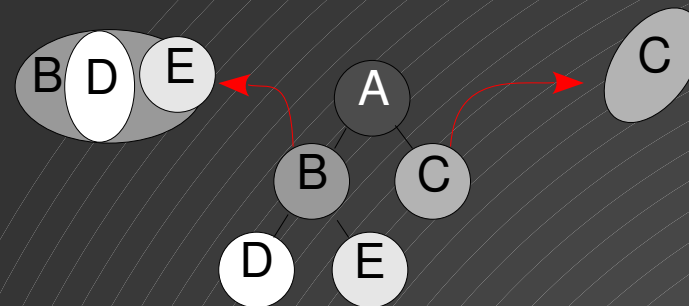
# Bien penser ses algorithmes

## Représentation des images

- Comment représenter une image
  - Max Tree



Image

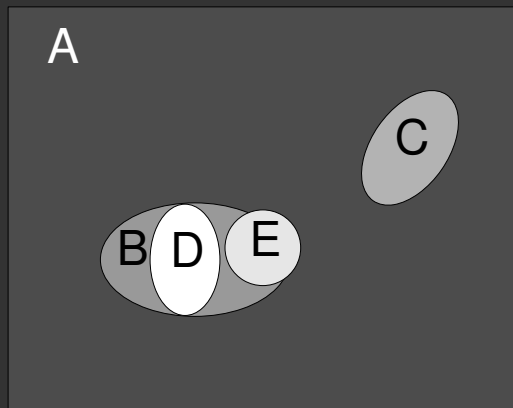


Max-Tree correspondant

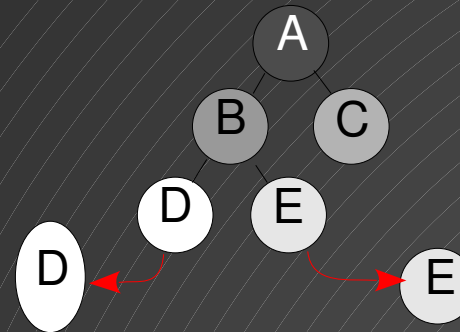
# Bien penser ses algorithmes

## Représentation des images

- Comment représenter une image
  - Max Tree



Image



Max-Tree correspondant

# Bien penser ses algorithmes

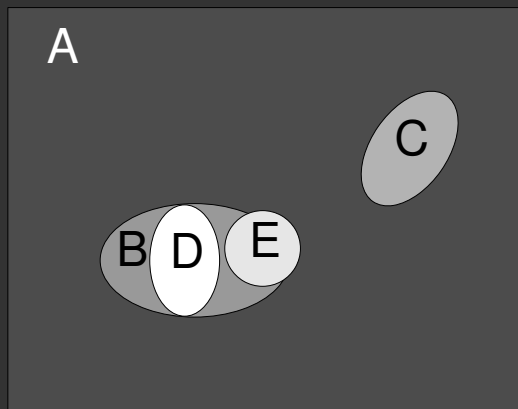
## Représentation des images

- Exemple :
  - Calcul de l'ouverture ultime
    - Long dans le cas général
    - Solution : utilisation du max-tree

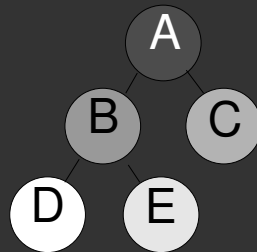
# Bien penser ses algorithmes

## Représentation des images

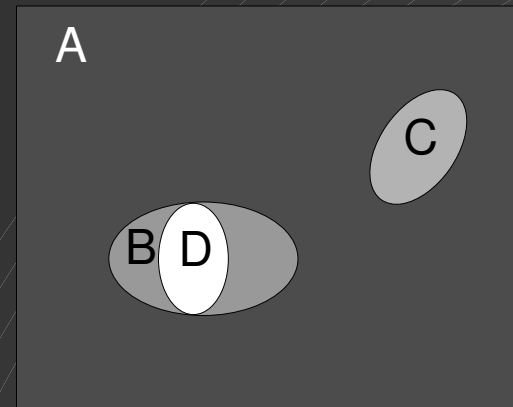
- Exemple :
  - Calcul de l'ouverture ultime



Image



Max-Tree correspondant



Image

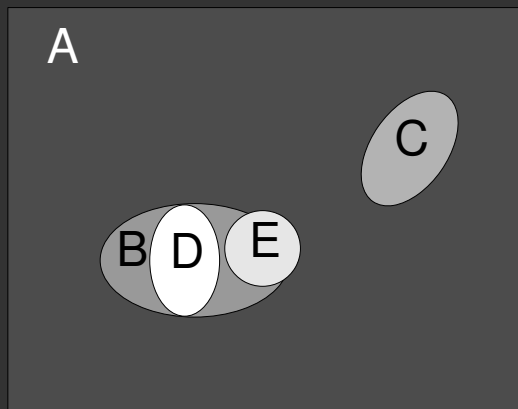


Max-Tree correspondant

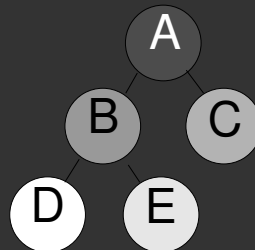
# Bien penser ses algorithmes

## Représentation des images

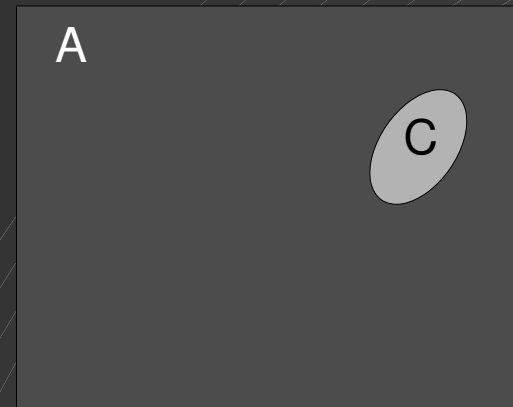
- Exemple :
  - Calcul de l'ouverture ultime



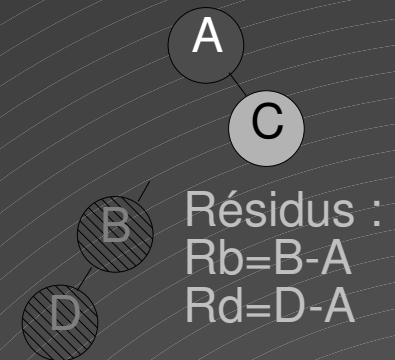
Image



Max-Tree correspondant



Image

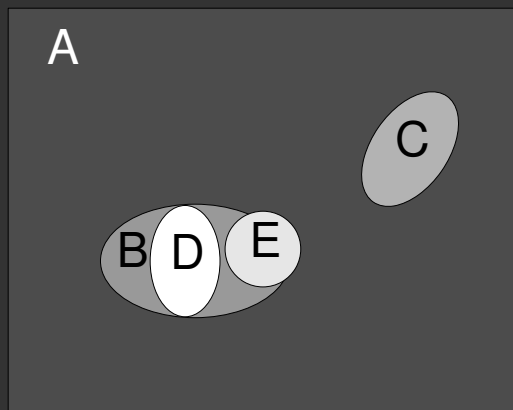


Max-Tree correspondant

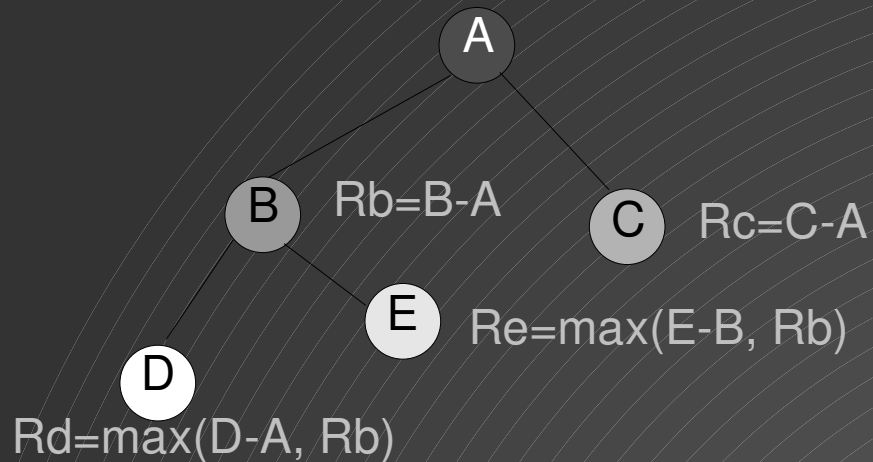
# Bien penser ses algorithmes

## Représentation des images

- Exemple :
  - Calcul de l'ouverture ultime



Image

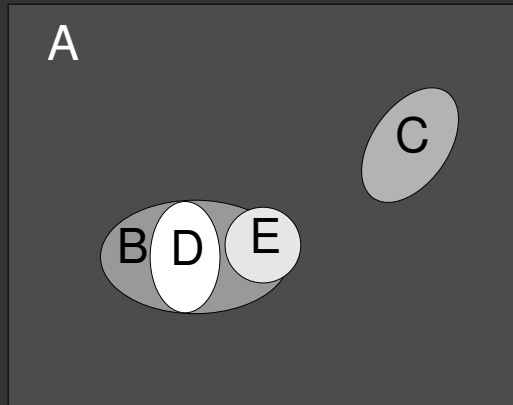


Max-Tree correspondant

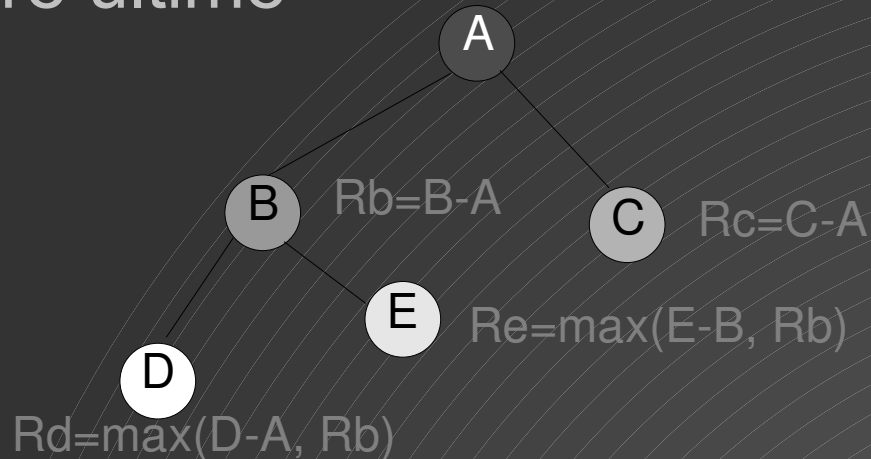
# Bien penser ses algorithmes

## Représentation des images

- Exemple :
  - Calcul de l'ouverture ultime



Image



max-tree correspondant

```
UO(node, parent_level, max_contrast) {  
    node.r = max(parent_level - node.level, max_contrast)  
    for all child c  
        UO(c, node.level, r)  
}
```

# Bien penser ses algorithmes

## Représentation des images

- Exemple :
  - Calcul de l'ouverture ultime
  - Résultats :

Format	128x128	256x256	512x512	1024x1024	2048x2048
Nb of pixels	16384	65536	262144	1048576	4194304
Time (ms)	0,18	2,39	12,01	52,04	235,53

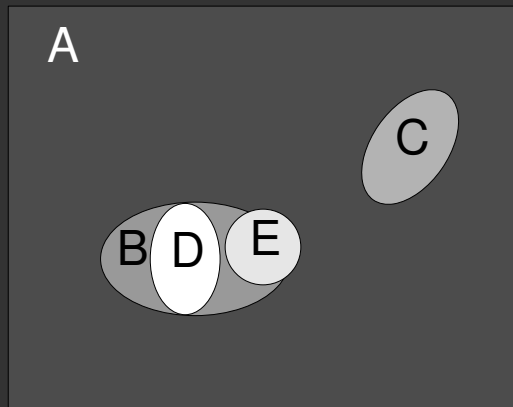
- Un problème difficile à la base est rendu plus simple et plus rapide en changeant le codage de l'image.



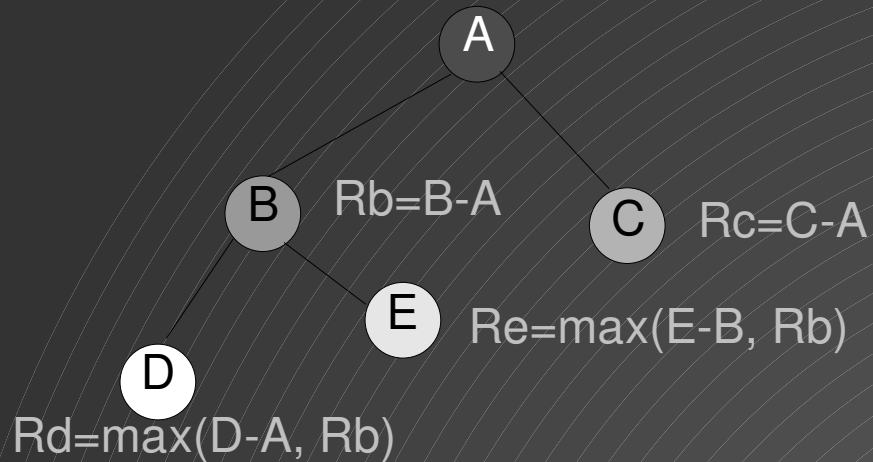
# Bien penser ses algorithmes

## Représentation des images

- Remarque :
  - Calcul de l'ouverture ultime



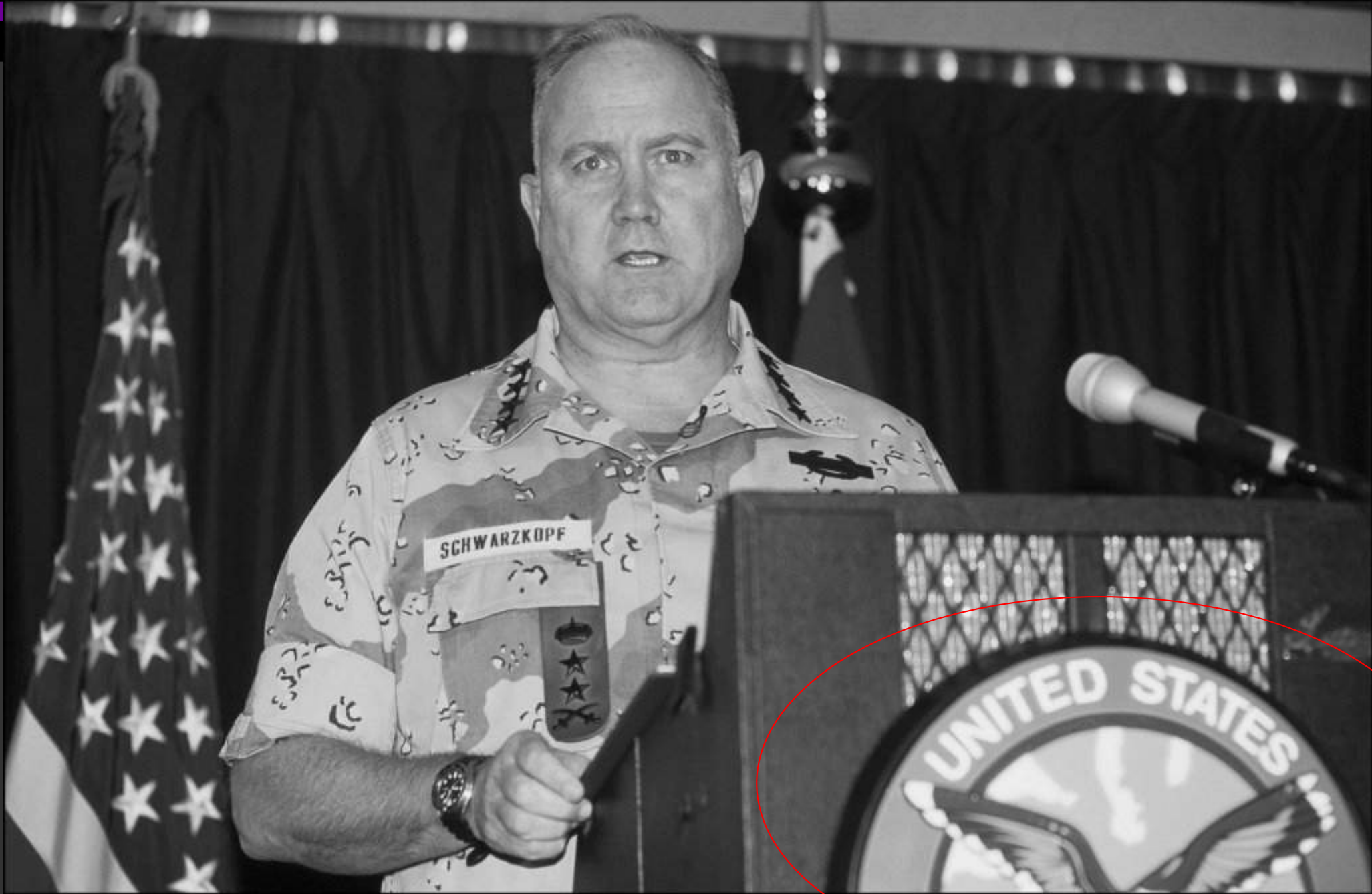
Image



max-tree correspondant

- Calcul possible sur une partie de l'image (une branche de l'arbre) !

# Bien penser ses algorithmes



# Bien penser ses algorithmes

[image  
éclaircie]



# Bien penser ses algorithmes

[image éclaircie]



# Bien penser ses algorithmes

- Repenser les algorithmes
  - Exemple FFT
  - Implémentation des filtres
  - L'image intégrable

# Bien penser ses algorithmes

- Calcul rapide de la DFT
  - FFT (1965 – Cooley et Tukey) (Gauss 1805??)

# Bien penser ses algorithmes

- Calcul rapide de la DFT
  - FFT (1965 – Cooley et Tukey) (Gauss 1805??)
  - The DFT :
$$X(l) = \sum_{k=0}^{N-1} x(k) e^{\frac{-2j\pi kl}{N}}, l=0, \dots, N-1$$
    - N complex mults, N-1 complex add **pour chaque l**.
      - $O(N^2)$

# Bien penser ses algorithmes

- Calcul rapide de la DFT

$$X(l) = \sum_{k=0}^{N-1} x(k) e^{\frac{-2j\pi kl}{N}}, l=0, \dots, N-1$$

- Exploiter la symétrie

$$W_N = e^{\frac{-2j\pi}{N}}$$

$$W_N^{k(N-n)} = W_N^{-kn} = (W_N^{kn})^* \quad (W_N^{kN} = 1)$$

$$W_N^{k(n)} = W_N^{k(N+n)} = W_N^{(k+N)n}$$



# Bien penser ses algorithmes

- Calcul rapide de la DFT
  - On suppose  $N = 2^m$

$$X(l) = \sum_{k \text{ pair}} x(k) W_N^{lk} + \sum_{k \text{ impair}} x(k) W_N^{lk}$$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) e^{\frac{-2j\pi 2kl}{N}} + \sum_{k=0}^{N/2-1} x(2k+1) e^{\frac{-2j\pi 2(k+1)l}{N}}$$

# Bien penser ses algorithmes

- Calcul rapide de la DFT
  - On suppose  $N = 2^m$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) e^{\frac{-2j\pi 2kl}{N}} + \sum_{k=0}^{N/2-1} x(2k+1) e^{\frac{-2j\pi 2(k+1)l}{N}}$$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) W_N^{2kl} + \sum_{k=0}^{N/2-1} x(2k+1) W_N^{(2k+1)l}$$

# Bien penser ses algorithmes

- Calcul rapide de la DFT
  - On suppose  $N = 2^m$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) W_N^{2kl} + \sum_{k=0}^{N/2-1} x(2k+1) W_N^{(2k+1)l}$$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) (W_N^2)^{kl} + W_N^l \sum_{k=0}^{N/2-1} x(2k+1) (W_N^2)^{kl}$$

$$W_N^2 = W_{N/2}$$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) e^{\frac{-2j\pi 2kl}{N}} + e^{\frac{-2j\pi l}{N}} \sum_{k=0}^{N/2-1} x(2k+1) e^{\frac{-2j\pi 2kl}{N}}$$

# Bien penser ses algorithmes

- Calcul rapide de la DFT
  - On suppose  $N = 2^m$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) W_N^{2kl} + \sum_{k=0}^{N/2-1} x(2k+1) W_N^{(2k+1)l}$$


$$X(l) = \sum_{k=0}^{N/2-1} x(2k) W_{N/2}^{kl} + W_N^l \sum_{k=0}^{N/2-1} x(2k+1) W_{N/2}^{kl}$$

$$W_N^2 = W_{N/2}$$

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) e^{\frac{-2j\pi 2kl}{N}} + e^{\frac{-2j\pi l}{N}} \sum_{k=0}^{N/2-1} x(2k+1) e^{\frac{-2j\pi 2kl}{N}}$$

# Bien penser ses algorithmes

- Calcul rapide de la DFT

$$X(l) = \sum_{k=0}^{N/2-1} x(2k) W_{N/2}^{kl} + W_N^l \sum_{k=0}^{N/2-1} x(2k+1) W_{N/2}^{kl}$$


- N/2 DFT des échantillons pairs, N/2 DFT des échantillons impairs.

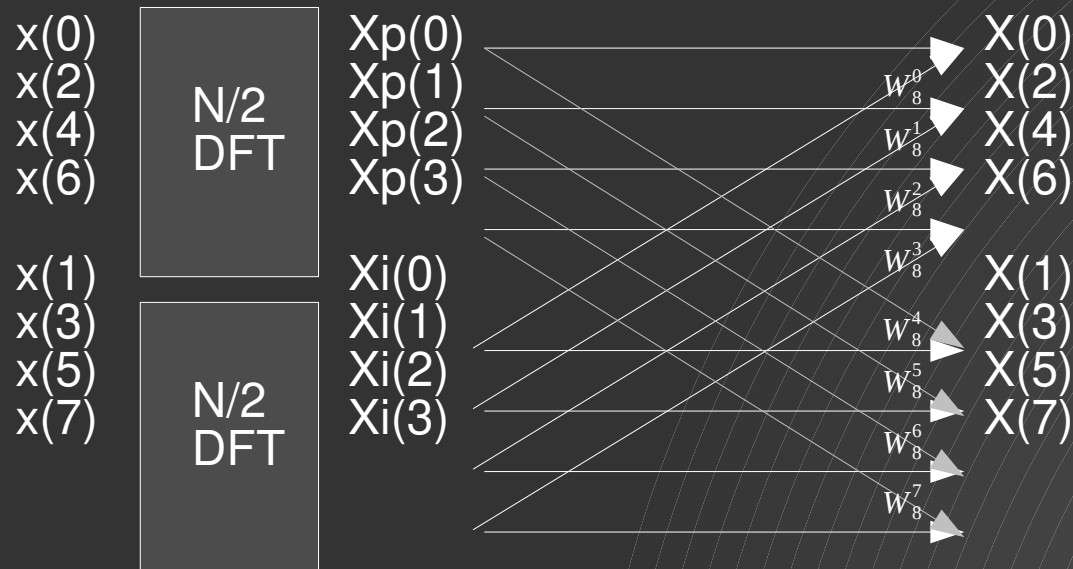
$$X(l) = X_p(l) + W_N^l X_i(l)$$

- Somme de deux DFTs de N/2 échantillons

# Bien penser ses algorithmes

- Calcul rapide de la DFT

$$X(l) = X_p(l) + W_N^l X_i(l)$$



- $2(N/2)^2 + N$  multiplications

# Bien penser ses algorithmes

- Calcul rapide de la DFT
  - $N/2, N/4, \dots, N/2^p=1$
  - 1 :  $2(N/2)^2 + N = N^2/2 + N$
  - 2 :  $2(2(N/4)^2 + N/2) + N = N^2/4 + 2N$
  - ...
- $N \log_2 N$

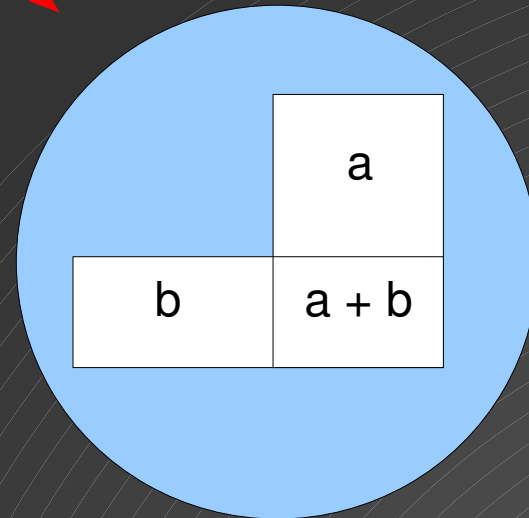
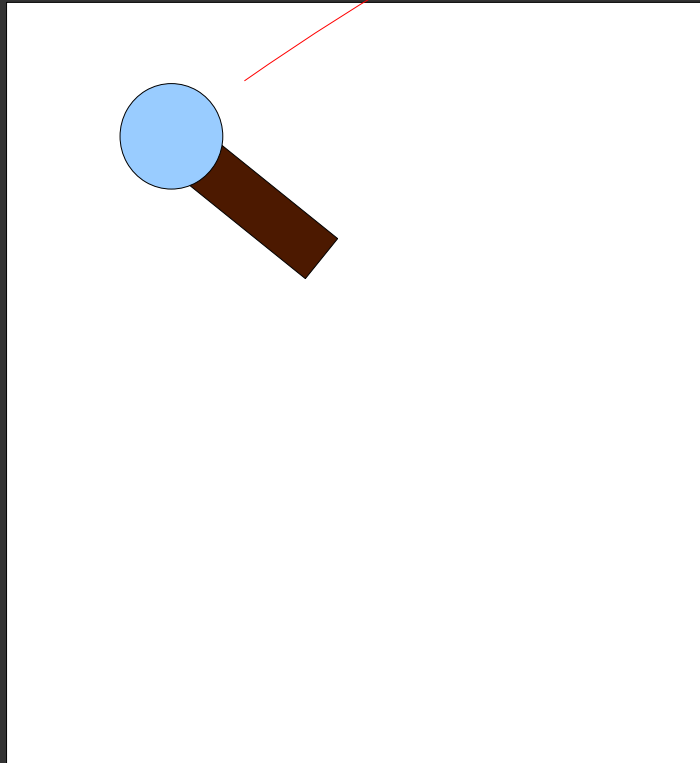
# Bien penser ses algorithmes

- Résultat :
  - DFT  $O(N^2)$
  - FFT  $O(N \log_2 N)$



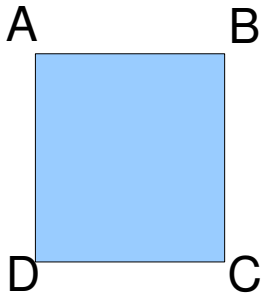
# Bien penser ses algorithmes

- L'image intégrable



# Bien penser ses algorithmes

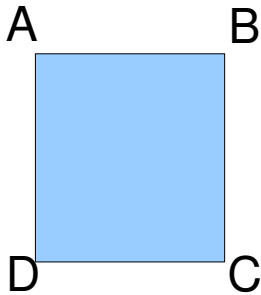
- L'image intégrable



Aire(ABCD) =

# Bien penser ses algorithmes

- L'image intégrable



$$\text{Aire(ABCD)} = C - B - D + A$$

# Bien penser ses algorithmes

- Implémentation des filtres
  - Décomposition des convolutions (filtres séparables)
    - $N \times N \rightarrow N + N$
  - Prise en compte des bordures ?