

# Premières notions

- Spécifications
- Algorithme et correction
- Complexité temporelle
  
- Algorithme récursif
- Algorithme optimal
- Conclusion

# Spécification : problème du tri

Entrée : un tableau d'éléments

Sortie : une permutation triée par ordre croissant de ces même éléments

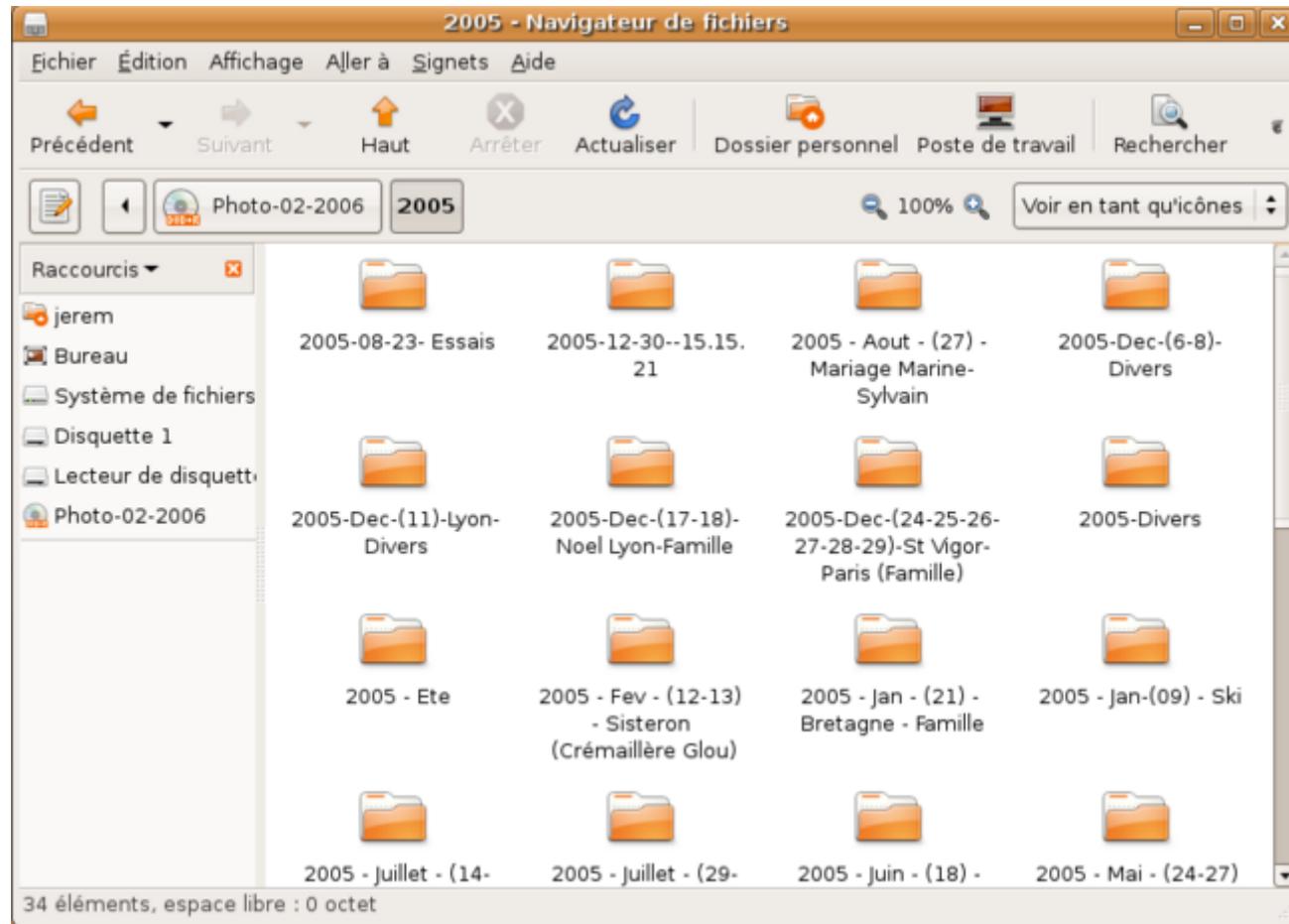
# Example

|    |   |    |   |   |
|----|---|----|---|---|
| 10 | 3 | 32 | 4 | 5 |
|----|---|----|---|---|

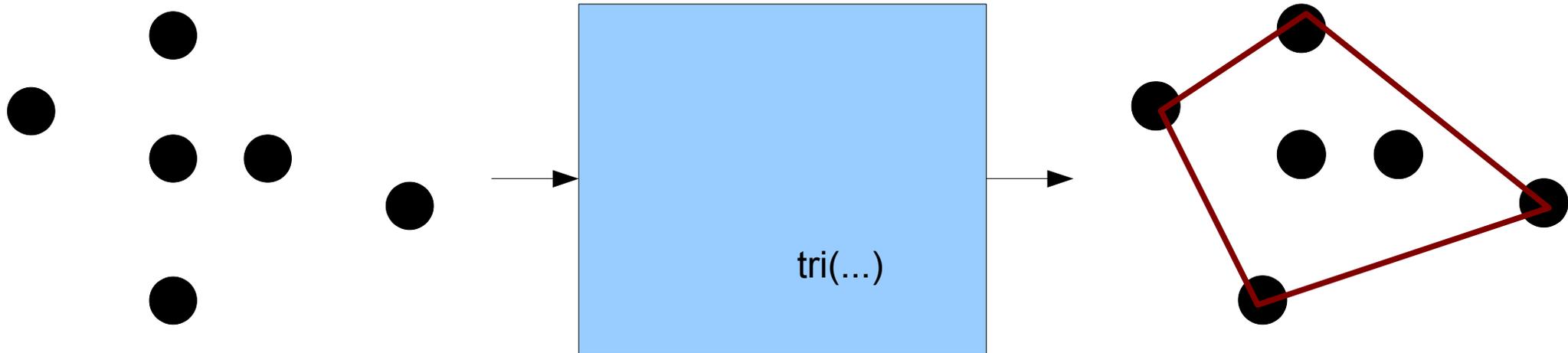


|   |   |   |    |    |
|---|---|---|----|----|
| 3 | 4 | 5 | 10 | 32 |
|---|---|---|----|----|

# Applications



# Applications : utilisé dans d'autres algorithmes (enveloppe convexe)



# Instructions élémentaires

$x := \langle \text{expression} \rangle$

$T[i] := \langle \text{expression} \rangle$

$1+3$

$T[2i]$

$x \leq 5$

$x \leq T[i]$

# Structures de contrôle

**Si** <condition>

...

**Si** <condition>

...

**Sinon**

...

**Pour**  $i := \dots$  à  $\dots$

...

**Pendant que** <condition>

...

# Procédures et fonctions

**Procédure** machin(x, y)

...

**a := truc(..., ...)**

**Fonction** truc(x, y)

**appeler** machin(x, y)

**retourner** <expression>

# Du pseudo-code !

Pour  $i := 1$  à  $n$

...

Pour  $i$  parcourant  $\{1, \dots, n\}$

...

$x := \langle \text{expression} \rangle$

$x \leftarrow \langle \text{expression} \rangle$

# Tri insertion. A nous de jouer !

Entrée : T tableau d'éléments

Sortie : à la fin T est une permutation triée par ordre croissant de ces même éléments

**Procédure triInsertion(T)**

??

Si <condition>

...

Pour i := ... à ...

...

x := <expression>

Pendant que <condition>

...

Si <condition>

...

Sinon

...

# Terminaison

**Procédure triInsertion(T)**

...

**Procédure truc()**

**Pendant que 1 = 1**

**x := 1**

# Validité

Entrée : T tableau d'éléments

Sortie : à la fin T est une permutation triée par ordre croissant de ces même éléments



**Théorème**

...

**Procédure triInsertion(T)**

...



**Démonstration**

...

# Complexité en temps

**Procédure triInsertion(T)**

...

- Pire des cas
- Cas moyen
- Meilleur des cas

- $O(f(n))$
- $\Theta(f(n))$

Attendriez-vous autant de temps ?

# On peut faire mieux !

Entrée : T tableau d'éléments

Sortie : à la fin T est une permutation triée par ordre croissant de ces mêmes éléments

**Fonction triFusion(T)**

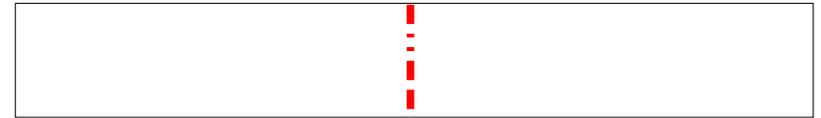
**Retourner ...**

# Procédure récursive

Entrée : ...

Sortie : ...

T =



**Fonction** triFusion(T[1..n])

...

**Retourner** ...

# Fusion

Entrée : ...

A =

Sortie : ...

B =

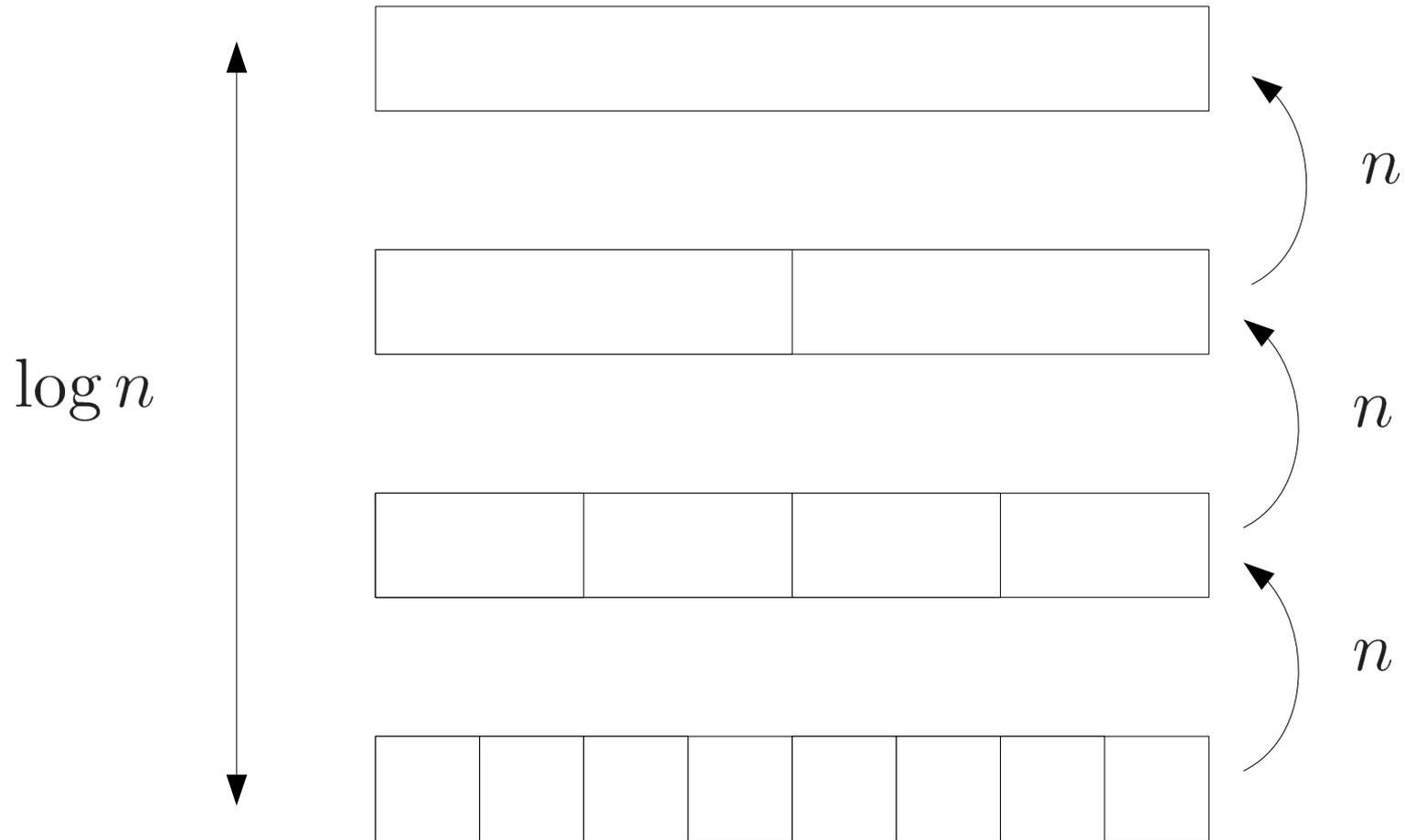
**Function** fusion(A[1..a], B[1..b])

...

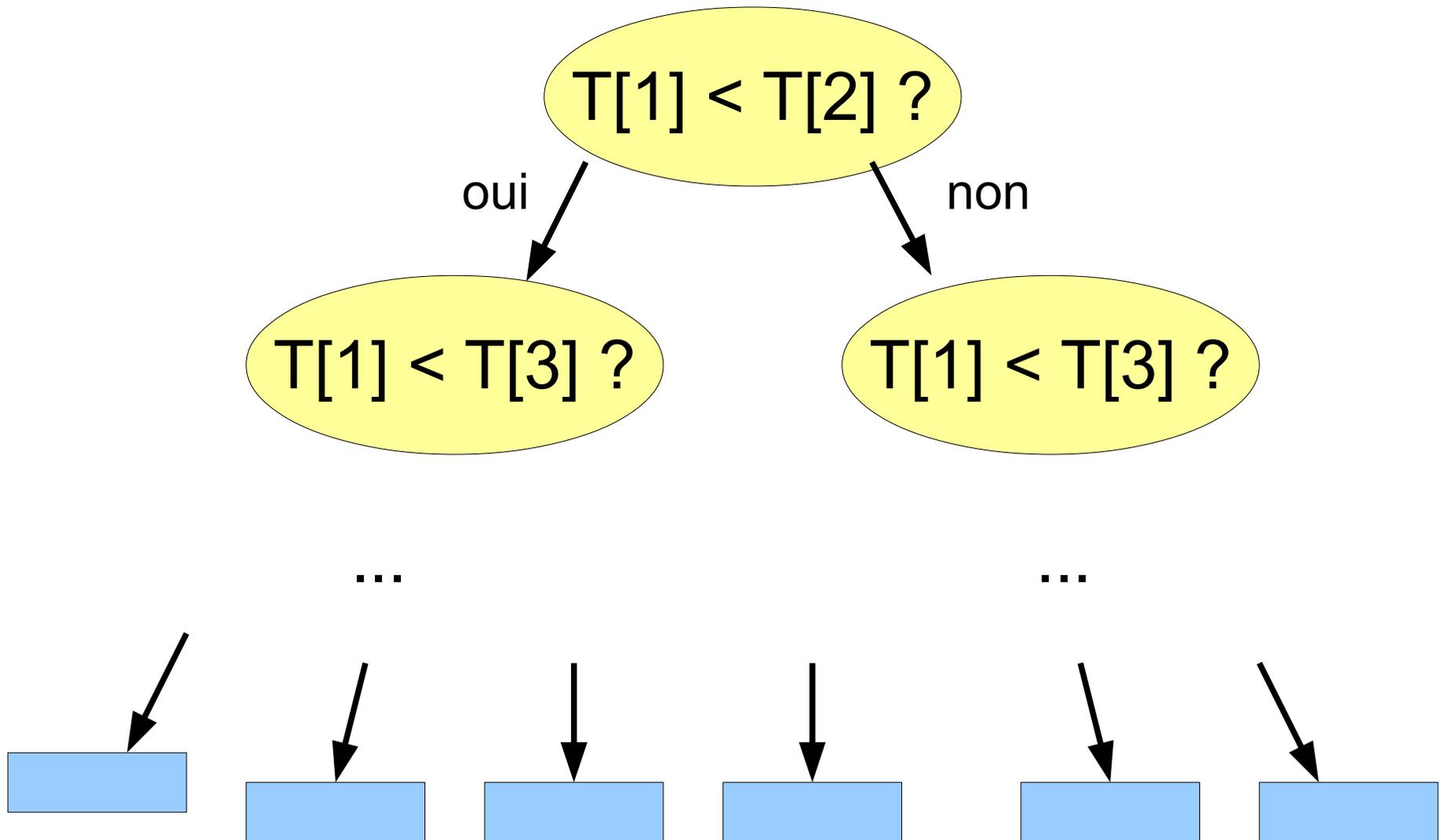
**Retourner** ...

# Terminaison et validité

# Complexité



# Optimalité de la complexité



# Qualités d'un algorithme de tri

- En place
- Stable
- Efficace sur les tableaux presque triés
- Localité

# Autres algorithmes de tri

|             | Pire des cas   | En moyenne   | En place | Stable |
|-------------|----------------|--------------|----------|--------|
| Insertion   | $O(n^2)$       | $O(n^2)$     | ✓        | ✓      |
| Tri fusion  | $O(n \ln n)$   | $O(n \ln n)$ | ✗*       | ✓      |
| Tri rapide  | $O(n^2)$       | $O(n \ln n)$ | ✓        | ✗*     |
| Tri Shell   | $O(n \ln^2 n)$ | ?            | ✓        | ✗      |
| Tri par tas | $O(n \ln n)$   | $O(n \ln n)$ | ✓        | ✗      |

# Plan ALGO1

## Premières notions

- 1) Diviser pour régner
- 2) Types abstraits et implémentations linéaires
- 3) Ensembles totalement ordonnées et arbres
- 4) File de priorité
- 5) Graphes
- 6) Graphes
- 7) Programmation gloutonne
- 8) Programmation dynamique
- 9) Programmation linéaire et réductions
- 10) NP-complétude
- 11) Quelques problèmes NP-complets

# ALGO2... un voyage

- D'autres machines : machines parallèles, quantiques, probas
- D'autres analyses : complexité en moyenne, coûts amortis
- Etc.