

Slide 1

0. Avant de partir en vacances

À la fin de la deuxième année...

Slide 2

- choisir un PA
- choisir un stage de recherche
- choisir une quatrième année
- choisir si on s'arrête en quatrième année ou si on continue par une formation par la recherche

LMD

Slide 3

À la fin de la deuxième année : fin du L

PA + stage + quatrième année = M

Les Programmes d'approfondissement

Informatique (Algorithmique et optimisation, Qualité du logiciel, Réseaux, Protection de l'information, Images)

Slide 4

Ingénierie des systèmes complexes (Systèmes embarqués, Systèmes d'information)

EE

Bio-informatique

La quatrième année

Slide 5

MPRI, COMASIC (X, Orsay, Thalès)

Télécom, Ensimag, Supélec, ...

Master à l'étranger

Slide 6

Ces différents choix doivent être cohérents et dessiner un projet personnel de formation ...

Slide 7

... qui doit être cohérent avec un projet professionnel

Deux questions :

- l'industrie, la recherche ou l'administration ?
- après un M ou après un D ?

La Taupe

Slide 8

Un système d'aide à l'indécision (horizon bouché par le concours)

Mais c'est fini

Droit à l'erreur, mais pas de ne pas choisir

[Y réfléchir dès ces vacances](#)

Slide 9

Le tri

Classer les données pour les retrouver

Slide 10

Liste ordre ou désordre : toujours une opération linéaire

Arbre ordre : toutes les opérations logarithmiques

Trier les données

Relations

Pré-ordre : relation réflexive et transitive

$$x \leq x$$

$$\text{si } x \leq y \text{ et } y \leq z \text{ alors } x \leq z$$

Ordre : pré-ordre et antisymétrique

$$\text{si } x \leq y \text{ et } y \leq x \text{ alors } x = y$$

Total : pré-ordre et deux éléments toujours comparables

$$x \leq y \text{ ou } y \leq x$$

Slide 11

Le tri

Un tableau qui contient n données

Une relation de **pré-ordre total**

Un tableau (le même) qui contient les mêmes données (même multiplicité) ordonnées

Slide 12

Avant - après

Slide 13

| | | | | | |
|-------|-------|-------|-------|-------|-------|
| (7,3) | (5,3) | (9,9) | (1,4) | (7,9) | (5,3) |
|-------|-------|-------|-------|-------|-------|

| | | | | | |
|-------|-------|-------|-------|-------|-------|
| (1,4) | (5,3) | (5,3) | (7,9) | (7,3) | (9,9) |
|-------|-------|-------|-------|-------|-------|

Slide 14

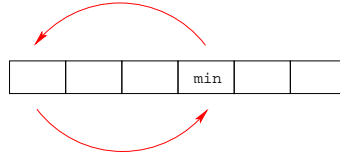
I. Le tri par sélection et le tri par insertion

Le tri par sélection

Slide 15

On construit petit à petit le **tableau final**

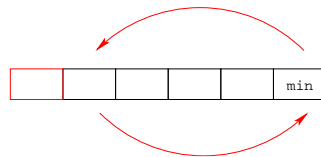
Son premier élément est un minimum



Le tri par sélection

Le deuxième, un minimum du reste

Slide 16



etc.

Slide 17

```
static void select (int [] t, int n) {  
  for (int i = 0; i < n; i = i + 1) {  
    int min = i;  
    for (int j = i + 1; j < n; j = j + 1) {  
      if (t[j] <= t[min]) min = j;}  
    int z = t[i]; t[i] = t[min]; t[min] = z;}}
```

Le tri par insertion

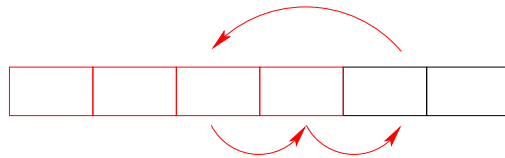
Au lieu de construire petit à petit le tableau final

Au on trie petit à petit le **tableau initial**

On trie les deux premiers éléments

Puis on insère le troisième, le quatrième, etc.

Slide 18



La complexité du tri par sélection et insertion

```
static void select (int [] t, int n) {  
  for (int i = 0; i < n; i = i + 1) {  
    ...  
    for (int j = i + 1; j < n; j = j + 1) {...}  
    ...}}}
```

Slide 19

Le corps de la boucle la plus interne exécuté

$(n-1) + (n-2) + \dots + 1 + 0$ fois

Équivalent à n^2 à une constante près (quadratique)

Idem pour le tri par insertion

Slide 20

II. Le tri rapide et le tri fusion

Diviser pour régner

Slide 21

Récurrance : décomposer n en $n-1$ et 1

Décomposer n en $n/2$ et $n/2$

Le tri rapide

Sélection : quel est le premier élément du tableau final ?

Rapide : quelle est la première moitié du tableau final ?

Slide 22

Un élément quelconque du tableau : le pivot p

On sépare les éléments $\leq p$ et $\geq p$

On trie séparément (récursivement)

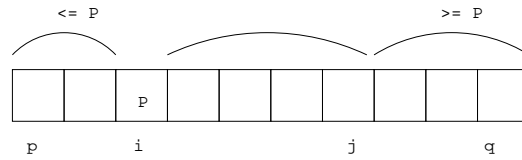
On assemble

Le tri rapide

On trie la partie du tableau comprise entre les indices p et q

La phase de séparation

Slide 23



Slide 24

```
static void rapide (final int [] t,
                   final int p, final int q) {
    if (p < q) {
        int i = p;
        int j = q;
        while (i < j) {
            if (t[i+1] <= t[i]) {
                int z=t[i];t[i]=t[i+1];t[i+1]=z;i=i+1;}
            else {int z=t[j];t[j]=t[i+1];t[i+1]=z;j=j-1;}}
        rapide(t,p,i-1);
        rapide(t,i+1,q);}}
```

Slide 25

Complexité en moyenne

Temps pour exécuter la boucle majorée par $a(n+1)$

$$C_n = a(n+1) + \frac{2}{n} \sum_{k=0}^{n-1} C_k$$

$$(n+1)C_{n+1} - nC_n = 2a(n+1) + 2C_n$$

$$\frac{C_{n+1}}{n+2} = \frac{2a}{n+2} + \frac{C_n}{n+1}$$

Slide 26

Complexité en moyenne

$$\frac{C_n}{n+1} = C_0 + 2a \sum_{k=2}^{n+1} 1/k$$

C_n asymptotiquement maj. par $n \ln(n)$ à une constante près

Complexité en moyenne

Dans le pire des cas : la partition des $n-1$ éléments dégénère en $0 + (n-1)$ (sélection : quadratique)

Le tri fusion

Au lieu de raisonner sur le tableau final on raisonne sur le tableau à trier

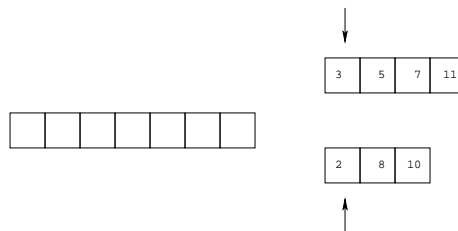
On le divise en deux parties **de même taille**

On les trie séparément (récursivement)

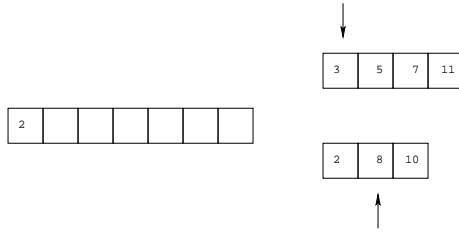
On les **fusionne**

Slide 27

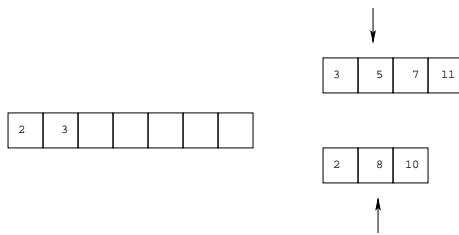
Slide 28



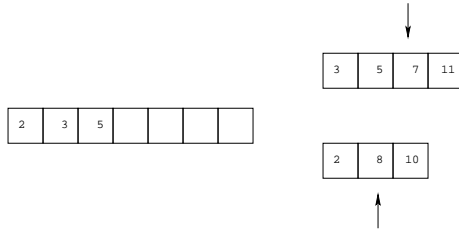
Slide 29



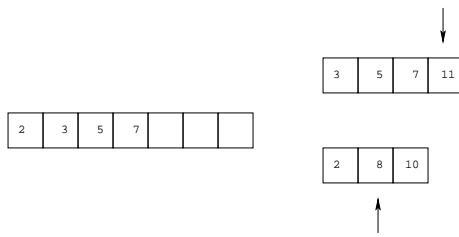
Slide 30



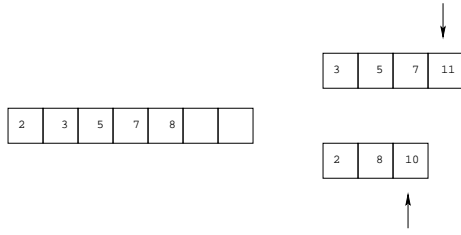
Slide 31



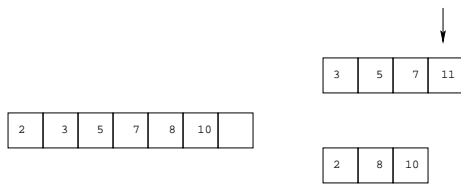
Slide 32



Slide 33



Slide 34



Slide 35

| | | | | | | |
|---|---|---|---|---|----|----|
| 2 | 3 | 5 | 7 | 8 | 10 | 11 |
|---|---|---|---|---|----|----|

| | | | |
|---|---|---|----|
| 3 | 5 | 7 | 11 |
|---|---|---|----|

| | | |
|---|---|----|
| 2 | 8 | 10 |
|---|---|----|

Complexité

$$C_n \leq a(n + 1) + 2 C_{n/2}$$

$$\leq 2 a n \log_2(n)$$

Slide 36

Asymptotiquement majorée par $n \ln(n)$ dans le pire des cas

Mais la fusion demande un tableau auxiliaire

Quatre algorithmes

Sélection, insertion : quadratiques

Slide 37

Rapide : quasi-linéaire en moyenne mais quadratique dans le pire des cas

Fusion : quasi-linéaire dans le pire des cas, mais deux fois plus de mémoire

Slide 38

III. Le tri en tas

Une nouvelle idée

Insérer un à un dans une **file de priorité** (tas)

Puis récupérer les maximums un à un

$n \ln(n)$ dans le pire des cas

Peut se faire avec un tableau unique

Slide 39

La phase d'insertion



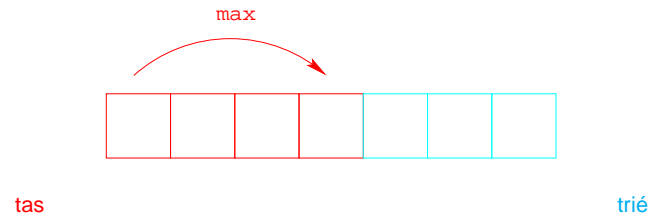
Slide 40

tas

en attente d'insertion

La phase de suppression

Slide 41



Slide 42

```
static void tas (int [] t, int n) {  
    for (int i = 0; i < n; i = i + 1) {insert(t,i);} }  
    for (int i = n - 1; i >=0; i = i - 1) {  
        int a = t[0];supress(t,i);t[i] = a;}}
```

Slide 43

IV. La borne inférieure

Faire mieux que $n \ln(n)$

Par exemple n ?

Slide 44

Impossible si

- le tableau initial est une permutations quelconque
- la seule opération permise est la [comparaison](#)

Slide 45

Temps majoré par $a \cdot n$

Nombre de comparaisons majoré par $a \cdot n$

Nombre de comportements possibles majoré par $2^{a \cdot n}$

Or pour n assez grand $n! > 2^{a \cdot n}$

Deux permutations traitées de manière identique : contradiction

Plus généralement

La complexité du tri est bornée inférieurement par $\ln(n!)$

$$\ln(n!) = \sum_{k=1}^n \ln(k)$$

Slide 46

99% des termes $> \ln(n / 100) = \ln(n) - \ln(100)$

$$\ln(n!) \geq (99 / 100) n (\ln(n) - \ln(100))$$

Asymptotiquement minoré par $n \ln(n)$ à une constante près

Le tri fusion et le tri en tas sont **optimaux**

Les hypothèses

Slide 47

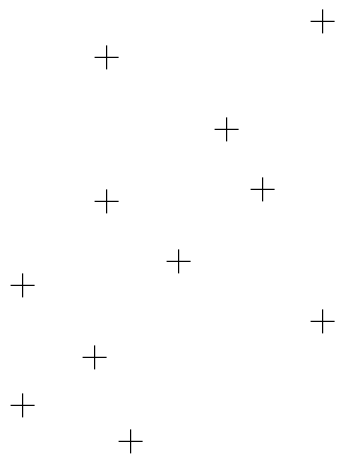
- le tableau initial est une permutations quelconque
- la seule opération permise est la **comparaison**

sont nécessaires toutes les deux

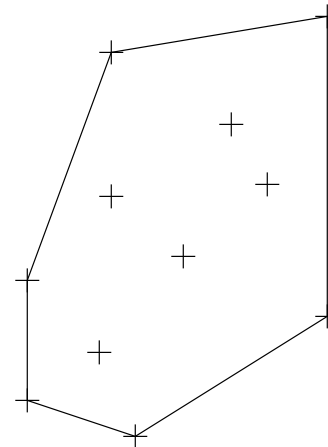
Trier par abscisse croissante des points d'abscisse : 1, 2, 3, 4, ... :
linéaire

Slide 48

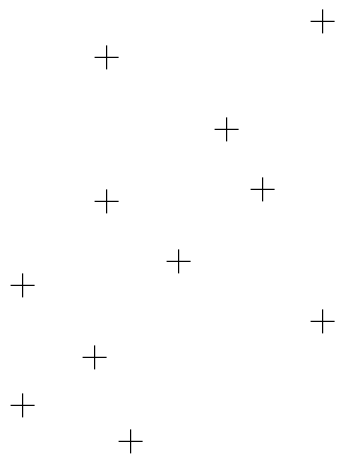
V. Une application en géométrie algorithmique



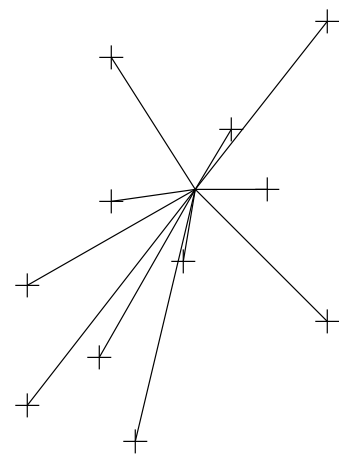
Slide 49



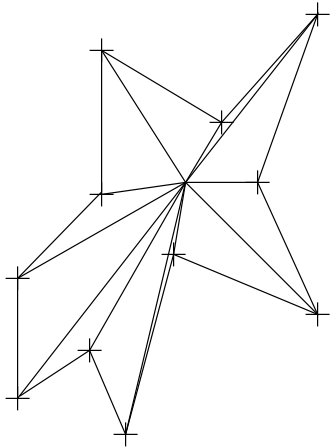
Slide 50



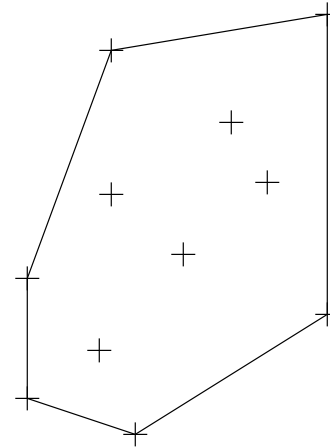
Slide 51



Slide 52

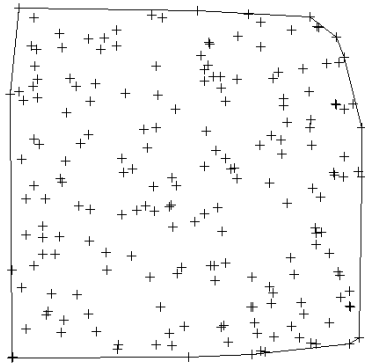


Slide 53



Slide 54

Slide 55



Slide 56

Exercices conseillés : 1.1 et 1.3

Slide 57

La prochaine fois : la pâte