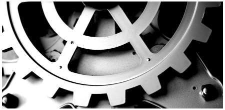
Algorithmique et structures de données
Calculs élémentaires de complexité



Gaël Mahé

slides : Elise Bonzon et Gaël Mahé

Université Paris Descartes

Licence 2

Calculs élémentaires de complexité

Calculs élémentaires de complexité

1. **Introduction**
2. Mesure de coûts
3. Applications

Pourquoi étudier la complexité des algorithmes ?

- Pour savoir si un algorithme est “efficace” ou non
- Pour pouvoir comparer deux algorithmes accomplissant la même tâche
- Pour chaque algorithme, on veut déterminer :
    - le temps d'execution
    - la place utilisée en mémoire
    - indépendemment de l'implémentation (langage choisi pour programmer, machine utilisée)
- On ne veut pas :
  “l'algorithme A, implémenté sur la machine M dans le langage L et exécuté sur la donnée D utilise k secondes de calcul et j bits de mémoire”
- On veut :
  “Quels que soient l'ordinateur et le langage utilisés, l'algorithme A_1 est **meilleur** *que l'algorithme A_2, pour des données de grandes tailles”*

Qu'est-ce que la complexité d'un algorithme ?

- Il s'agit de caractériser le comportement d'un algorithme sur l'ensemble D_n des données de taille n
- La complexité dépend
  - en général de la taille n des données
  - souvent aussi de la donnée $d \in D_n$ elle-même
- Plusieurs types de complexité :
  - En temps
  - En espace mémoire

Qu'est-ce que la complexité d'un algorithme ?

- Il s'agit de caractériser le comportement d'un algorithme sur l'ensemble D_n des données de taille n
- La complexité dépend
  - en général de la taille n des données
  - souvent aussi de la donnée $d \in D_n$ elle-même
- Plusieurs types de complexité :
  - En temps
  - En espace mémoire

Calculs élémentaires de complexité

Opérations significatives

- Complexité = nombre d'opérations significatives
- Par exemple, toute opération d'accès à un élément d'un vecteur :
  - Comparaison entre deux éléments d'un vecteur ($V(i) < V(j)$)
  - Affectation d'un élément du vecteur ($V(i) \leftarrow x$)
  - Échange de deux éléments d'un vecteur ($echange(V(i), V(j))$)
- Si plusieurs opérations significatives différentes sont choisies, elles doivent être décomptées séparément

Définitions et notations

- $\text{coût}_A(d)$: complexité de l'algorithme A sur la donnée $d \in D_n$ de taille n
- Complexité au **meilleur des cas** :

$$\text{coût } min_A(n) = \min_{d \in D_n} \{\text{coût}_A(d)\}$$

- Complexité au **pire des cas** :

$$\text{coût } max_A(n) = \max_{d \in D_n} \{\text{coût}_A(d)\}$$

- Complexité au **moyenne** :

$$\text{coût } moy_A(n) = \sum_{d \in D_n} \text{coût}_A(d) \times proba(d)$$

Complexité : ordres de grandeur (1)

- Complexité (moyenne, max, min) $=$ fonction de n

$\rightarrow$ Comment cette fonction croît quand la taille n des données croît ?

- On recherche **l'ordre de grandeur asymptotique**,
  i.e. le coût de l'algorithme quand n tend vers l'infini
  Par ex, pour n grand, $n+1 \approx n+2$.
- Ordres de grandeur de référence :

$$1 \leq \log_2 n \leq n \leq n \log_2 n \leq n^2 \leq n^3 \leq 2^n \ldots$$

$\rightarrow$ Plus la taille des données est grande, plus les écarts se creusent

- Les constantes multiplicatives n'ont qu'une importance secondaire :
  Ex : si $C_1(n) = 2n, \quad C_2(n) = 4n, \quad C_3(n) = n^2$,
  alors $C_1(n) < C_2(n) \, \forall\, n$ et $C_2(n) < C_3(n) \, \forall\, n > 4$

Complexité : ordres de grandeur (2)

- Soient f et g fonctions de $\mathbb{N}$ dans $\mathbb{R}^+$
- f est **dominée asymptotiquement** par g ssi :
  $\exists\, c \in \mathbb{R}^+, \quad \exists\, n_0 \in \mathbb{N} \quad \text{tel que} \quad \forall\, n > n_0, f(n) \leq c.g(n)$
- Notation : $f = O(g)$
- Exemple : $4n = O(n^2)$, $4n = O(n)$
- f et g ont le **même ordre de grandeur asymptotique** ssi :
  $f = O(g) \quad \text{et} \quad g = O(f)$, *i.e.* :
  $\exists\, c, d \in \mathbb{R}^+, \quad \exists\, n_0 \in \mathbb{N} \quad \text{tel que} \quad \forall\, n > n_0, d.g(n) \leq f(n) \leq c.g(n)$
- Notation : $f = \Theta(g)$
- Exemple : $4n = \Theta(n)$ mais $4n \neq \Theta(n^2)$

Complexités raisonnables

Table: Si un temps t est nécessaire pour traiter un problème de taille n,

Pour une complexité de :	1	$\log_2(n)$	n	$n\log_2(n)$	n^2	n^3	2^n
Temps si $n \times 10$	t	$t + 3,32$	$10t$	$(10+\epsilon)t$	$100t$	$1000t$	t^{10}
Taille traitable si $t \times 10$	∞	n^{10}	$10n$	$(10-\epsilon)n$	$3,16n$	$2,15n$	$n + 3,32$

- Pour les données de grande taille,
  les algorithmes utilisables sont ceux qui s'exécutent en un temps
  - constant
  - logarithmique (Ex : recherche dichotomique)
  - linéaire (Ex : recherche séquentielle)
  - $n \log n$ (Ex : bons algorithmes de tri)
- Les algorithmes qui prennent un temps polynomial n^k ne sont utilisables que pour des données de petite taille ou pour $k \leq 3$
- Les algorithmes en temps exponentiel sont presque inutilisables

Calculs élémentaires de complexité

Recherche séquentielle dans un vecteur non trié

Recherche séquentielle dans un vecteur non trié

début

/* ENTRÉES: Un vecteur V de taille n, un élément x */
/* SORTIE: vrai si $x \in V$, faux sinon */
$i \leftarrow 1$
tant que $i < n$ *et* $V(i) \neq x$ **faire** $i \leftarrow i + 1$
/* sortie de boucle */
si $V(i) = x$ **alors** **retourner** vrai
sinon **retourner** faux

fin

Recherche séquentielle dans un vecteur non trié

- Opération significative ?

Recherche séquentielle dans un vecteur non trié

Recherche séquentielle dans un vecteur non trié

début

/* ENTRÉES: Un vecteur V de taille n, un élément x */

/* SORTIE: vrai si $x \in V$, faux sinon */

$i \leftarrow 1$

tant que $i < n$ *et* $V(i) \neq x$ **faire** $i \leftarrow i + 1$

/* sortie de boucle */

si $V(i) = x$ **alors** **retourner** vrai

sinon **retourner** faux

fin

Recherche séquentielle dans un vecteur non trié

- Opération significative : Comparaison de l'élément x à un élément de V

Recherche séquentielle dans un vecteur non trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V

Recherche séquentielle dans un vecteur non trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** ?

Recherche séquentielle dans un vecteur non trié

Recherche séquentielle dans un vecteur non trié

début

/* ENTRÉES: Un vecteur V de taille n, un élément x */
/* SORTIE: vrai si $x \in V$, faux sinon */
$i \leftarrow 1$
tant que $i < n$ *et* $V(i) \neq x$ **faire** $i \leftarrow i + 1$
/* sortie de boucle */
si $V(i) = x$ **alors** **retourner** vrai
sinon **retourner** faux

fin

Recherche séquentielle dans un vecteur non trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** : Si x est le premier élément de V, on fait **2 comparaisons** : $V(i) \neq x$ et $V(i) = x$
- Complexité dans le **pire des cas** ?

Recherche séquentielle dans un vecteur non trié

Recherche séquentielle dans un vecteur non trié

début

/* ENTRÉES: Un vecteur V de taille n, un élément x */
/* SORTIE: vrai si $x \in V$, faux sinon */
$i \leftarrow 1$
tant que $i < n$ *et* $V(i) \neq x$ **faire** $i \leftarrow i + 1$
/* sortie de boucle */
si $V(i) = x$ **alors** **retourner** vrai
sinon **retourner** faux

fin

Recherche séquentielle dans un vecteur non trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** : Si x est le premier élément de V, on fait **2 comparaisons** : $V(i) \neq x$ et $V(i) = x$
- Complexité dans le **pire des cas** : Si $x \notin V$. $n + 1$ **comparaisons**
  - n comparaisons à $V(i)$ dans la boucle
  - 1 comparaison en fin de boucle

Recherche séquentielle dans un vecteur non trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** : Si x est le premier élément de V, on fait **2 comparaisons** : $V(i) \neq x$ et $V(i) = x$
- Complexité dans le **pire des cas** : Si $x \notin V$. $n + 1$ **comparaisons**
    - n comparaisons à $V(i)$ dans la boucle
    - 1 comparaison en fin de boucle
- Complexité **moyenne?**

Recherche séquentielle dans un vecteur non trié

Complexité **moyenne?**

- $C_{moy} = \Pr(x \in V) C_{moy}(x \in V) + \Pr(x \notin V) C_{moy}(x \notin V)$
- Soit $q = proba(x \in V)$, et donc $1 - q = proba(x \notin V)$
- $C_{moy} = q C_{moy}(x \in V) + (1 - q) C_{moy}(x \notin V)$

Recherche séquentielle dans un vecteur non trié

- Nombre de comparaisons si $x \in V$:
  - $C_{moy}(x \in V) = \sum_{j=1}^{n} \Pr(x = V(j)) C(x = V(j))$
  - On suppose que la place de x dans V est équiprobable. Donc $proba(x = V(j)) = \frac{1}{n}$
  - Si $x = V(j)$, combien de tests faut-il faire?

Recherche séquentielle dans un vecteur non trié

Recherche séquentielle dans un vecteur non trié

début

/* ENTRÉES: Un vecteur V de taille n, un élément x */
/* SORTIE: vrai si $x \in V$, faux sinon */
$i \leftarrow 1$
tant que $i < n$ *et* $V(i) \neq x$ **faire** $i \leftarrow i + 1$
/* sortie de boucle */
si $V(i) = x$ **alors** **retourner** vrai
sinon **retourner** faux

fin

Recherche séquentielle dans un vecteur non trié

Complexité **moyenne?**

- Nombre de comparaisons si $x \in V$:
  - $C_{moy}(x \in V) = \sum_{j=1}^{n} \Pr(x = V(j))C(x = V(j))$
  - On suppose que la place de x dans V est équiprobable. Donc $proba(x = V(j)) = \frac{1}{n}$
  - Si $x = V(j)$, il faut faire $1 + j$ comparaisons
  - Nombre moyen de comparaisons si $x \in V$:

$$\text{coût } moy(x \in V) = \frac{1}{n}\Sigma_{j=1}^{n}(1+j) = \frac{1}{n}(n + \frac{n(n+1)}{2}) = \frac{n+3}{2}$$

Recherche séquentielle dans un vecteur non trié

Complexité **moyenne?**

- Nombre de comparaisons si $x \in V$:
  - $C_{moy}(x \in V) = \sum_{j=1}^{n} \Pr(x = V(j)) C(x = V(j))$
  - On suppose que la place de x dans V est équiprobable. Donc $proba(x = V(j)) = \frac{1}{n}$
  - Si $x = V(j)$, il faut faire $1 + j$ comparaisons
  - Nombre moyen de comparaisons si $x \in V$:

$$\text{coût } moy(x \in V) = \frac{1}{n}\Sigma_{j=1}^{n}(1+j) = \frac{1}{n}(n + \frac{n(n+1)}{2}) = \frac{n+3}{2}$$

- Nombre de comparaisons si $x \notin V$:
  - il faut parcourir tout le vecteur : $n + 1$ comparaisons :

$$\text{coût } moy(x \notin V) = n + 1$$

Recherche séquentielle dans un vecteur non trié

Complexité **moyenne?**

- $C_{moy} = qC_{moy}(x \in V) + (1-q)C_{moy}(x \notin V)$, avec $q = \Pr(x \in V)$
- Nombre moyen de comparaisons si $x \in V$:

$$\text{coût } moy(x \in V) = \frac{n+3}{2}$$

- Nombre de comparaisons si $x \notin V$:

$$\text{coût } moy(x \notin V) = n+1$$

- Complexité dans tous les cas :

$$\text{coût } moy(n) = q\frac{n+3}{2} + (1-q)(n+1) = (1-\frac{q}{2})n + 1 - \frac{q}{2}$$

- **Complexité de l'ordre de** $(1-\frac{q}{2})n$**.**
  Si $q = \frac{1}{2}$, coût $moy(n) \approx \frac{3}{4}n$.

Complexité : Recherche séquentielle dans un vecteur trié

Recherche séquentielle dans un vecteur trié

début
/* ENTRÉES: Un vecteur V de taille n, un élément x */
/* SORTIE: i si x apparait au rang i de V, 0 si $x \notin V$ */
si $x > V(n)$ **alors** **retourner** 0
sinon
$i \leftarrow 1$
tant que $V(i) < x$ **faire** $i \leftarrow i + 1$
/* sortie de boucle */
si $V(i) = x$ **alors** **retourner** i
sinon **retourner** 0
fin

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative ?

Complexité : Recherche séquentielle dans un vecteur trié

Recherche séquentielle dans un vecteur trié

début
  /* ENTRÉES: Un vecteur V de taille n, un élément x */
  /* SORTIE: i si x apparait au rang i de V, 0 si $x \notin V$ */
  si $x > V(n)$ **alors** **retourner** 0
  sinon
    $i \leftarrow 1$
    tant que $V(i) < x$ **faire** $i \leftarrow i + 1$
    /* sortie de boucle */
    si $V(i) = x$ **alors** **retourner** i
    sinon **retourner** 0
fin

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative : Comparaison de l'élément x à un élément de V

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative : Comparaison de l'élément x à un élément de V
- De quoi dépend le nombre de comparaisons ?

Complexité : Recherche séquentielle dans un vecteur trié

Recherche séquentielle dans un vecteur trié

début

  /* ENTRÉES: Un vecteur V de taille n, un élément x */

  /* SORTIE: i si x apparait au rang i de V, 0 si $x \notin V$ */

  si $x > V(n)$ **alors** **retourner** 0

  sinon

    $i \leftarrow 1$

    tant que $V(i) < x$ **faire** $i \leftarrow i + 1$

    /* sortie de boucle */

    si $V(i) = x$ **alors** **retourner** i

    sinon **retourner** 0

fin

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** ?

Complexité : Recherche séquentielle dans un vecteur trié

Recherche séquentielle dans un vecteur trié

début

/* ENTRÉES: Un vecteur V de taille n, un élément x */

/* SORTIE: i si x apparait au rang i de V, 0 si $x \notin V$ */

si $x > V(n)$ **alors** **retourner** 0

sinon

$i \leftarrow 1$

tant que $V(i) < x$ **faire** $i \leftarrow i + 1$

/* sortie de boucle */

si $V(i) = x$ **alors** **retourner** i

sinon **retourner** 0

fin

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** : Si $x > V(n)$, on ne fait **qu'une seule comparaison**
- Complexité dans le **pire des cas** ?

Complexité : Recherche séquentielle dans un vecteur trié

Recherche séquentielle dans un vecteur trié

début
/* ENTRÉES: Un vecteur V de taille n, un élément x */
/* SORTIE: i si x apparait au rang i de V, 0 si $x \notin V$ */
si $x > V(n)$ **alors** **retourner** 0
sinon
  $i \leftarrow 1$
  tant que $V(i) < x$ **faire** $i \leftarrow i + 1$
  /* sortie de boucle */
  si $V(i) = x$ **alors** **retourner** i
  sinon **retourner** 0
fin

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** : Si $x > V(n)$, on ne fait **qu'une seule comparaison**
- Complexité dans le **pire des cas** : Si $x = V(n)$ ou si $V(n-1) < x < V(n)$. $n+2$ **comparaisons**
  - 1 comparaison à $V(n)$ avant de rentrer dans la boucle
  - n comparaisons dans la boucle
  - 1 comparaison en fin de boucle

Complexité : Recherche séquentielle dans un vecteur trié

- Opération significative : Comparaison de l'élément x à un élément de V
- Le nombre de comparaisons dépend de x, de V et de n, la taille de V
- Complexité dans le **meilleur des cas** : Si $x > V(n)$, on ne fait **qu'une seule comparaison**
- Complexité dans le **pire des cas** : Si $x = V(n)$ ou si $V(n-1) < x < V(n)$. $n+2$ **comparaisons**
  - 1 comparaison à $V(n)$ avant de rentrer dans la boucle
  - n comparaisons dans la boucle
  - 1 comparaison en fin de boucle
- Complexité **moyenne?**

Complexité : Recherche séquentielle dans un vecteur trié

Complexité **moyenne?**

- $C_{moy} = \Pr(x \in V) C_{moy}(x \in V) + \Pr(x \notin V) C_{moy}(x \notin V)$
- Soit $q = proba(x \in V)$, et donc $1 - q = proba(x \notin V)$
- $C_{moy} = q C_{moy}(x \in V) + (1 - q) C_{moy}(x \notin V)$

Complexité : Recherche séquentielle dans un vecteur trié

Complexité **moyenne?**

- Nombre de comparaisons si $x \in V$:
  - $C_{moy}(x \in V) = \sum_{j=1}^{n} \Pr(x = V(j))C(x = V(j))$
  - On suppose que la place de x dans V est équiprobable (ne dépend pas de l'indice). Donc $proba(x = V(j)) = \frac{1}{n}$
  - Si $x = V(j)$, combien de tests faut-il faire?

Complexité : Recherche séquentielle dans un vecteur trié

Recherche séquentielle dans un vecteur trié

début

/* ENTRÉES: Un vecteur V de taille n, un élément x */

/* SORTIE: i si x apparait au rang i de V, 0 si $x \notin V$ */

si $x > V(n)$ **alors** **retourner** 0

sinon

$i \leftarrow 1$

tant que $V(i) < x$ **faire** $i \leftarrow i + 1$

/* sortie de boucle */

si $V(i) = x$ **alors** **retourner** i

sinon **retourner** 0

fin

Complexité : Recherche séquentielle dans un vecteur trié

Complexité **moyenne?**

- Nombre de comparaisons si $x \in V$:
  - $C_{moy}(x \in V) = \sum_{j=1}^{n} \Pr(x = V(j))C(x = V(j))$
  - On suppose que la place de x dans V est équiprobable (ne dépend pas de l'indice). Donc $proba(x = V(j)) = \frac{1}{n}$
  - Si $x = V(j)$, il faut faire $j + 2$ comparaisons
  - Nombre moyen de comparaisons si $x \in V$:

$$\text{coût } moy(x \in V) = \frac{1}{n}\Sigma_{j=1}^{n}(j+2) = \frac{1}{n}\left(\frac{n(n+1)}{2} + 2n\right) = \frac{n+5}{2}$$

Complexité : Recherche séquentielle dans un vecteur trié

Complexité **moyenne?**

- Nombre de comparaisons si $x \in V$:
    - $C_{moy}(x \in V) = \sum_{j=1}^{n} \Pr(x = V(j)) C(x = V(j))$
    - On suppose que la place de x dans V est équiprobable (ne dépend pas de l'indice). Donc $proba(x = V(j)) = \frac{1}{n}$
    - Si $x = V(j)$, il faut faire $j + 2$ comparaisons
    - Nombre moyen de comparaisons si $x \in V$:

$$\text{coût } moy(x \in V) = \frac{1}{n} \Sigma_{j=1}^{n} (j+2) = \frac{1}{n} \left(\frac{n(n+1)}{2} + 2n \right) = \frac{n+5}{2}$$

- Nombre de comparaisons si $x \notin V$:
    - $n + 1$ cas possibles ($x < V(1)$; $V(1) < x < V(2)$; ...; $x > V(n)$)
    - Il faut faire 1 test si $x > V(n)$; et $j + 2$ tests si $V(j-1) < x < V(j)$
    - Nombre moyen de comparaisons si $x \notin V$:

$$\text{coût } moy(x \notin V) = \frac{1 + \Sigma_{j=1}^{n} (j+2)}{n+1} = \frac{1}{n+1} \left(1 + \frac{n(n+1)}{2} + 2n \right) \approx_{n \to \infty} \frac{n}{2} + 2$$

Complexité : Recherche séquentielle dans un vecteur trié

Complexité **moyenne?**

- $C_{moy} = qC_{moy}(x \in V) + (1-q)C_{moy}(x \notin V)$, avec $q = \Pr(x \in V)$
- Nombre de comparaisons si $x \in V$:

$$\text{coût } moy(x \in V) = \frac{n+5}{2}$$

- Nombre de comparaisons si $x \notin V$:

$$\text{coût } moy(x \notin V) \approx_{n \to \infty} \frac{n}{2} + 2$$

- Complexité dans tous les cas :

$$\text{coût } moy(n) = q\frac{n+5}{2} + (1-q)\frac{n+4}{2} = \frac{n}{2} + 2(1+q)$$

- **Complexité de l'ordre de** $\frac{n}{2}$

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit $L_k =$ longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit L_k = longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit $L_k =$ longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit L_k = longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit L_k = longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit $L_k =$ longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit $L_k =$ longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?

Recherche dichotomique de la 1ère occurrence : algorithme

Algorithme 93 : Recherche dichotomique d'une 1ère occurrence dans un vecteur trié

début

  /* ENTRÉES: Un vecteur V de taille n, un élément x */

  /* SORTIE: i si x apparait au rang i de V, 0 si $x \notin V$ */

  $inf \leftarrow 1$, $sup \leftarrow n$, $i \leftarrow 0$

  tant que $inf < sup$ **faire**

    $med \leftarrow (inf + sup)$ **div** 2

    si $x \leq V(med)$ **alors** $sup \leftarrow med$

    sinon $inf \leftarrow med + 1$

  si $V(inf) = x$ **alors**

    retourner inf

  sinon

    retourner 0

fin

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit L_k = longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?
1 comparaison : $x \leq V(med)$

Conclusion : complexité de l'ordre de $\log_2 n$ comparaisons
NB : nous ferons un calcul plus précis en TD

Recherche dichotomique de la 1ère occurrence dans un vecteur trié

Principe de l'algorithme:
à chaque itération, division par deux de l'intervalle de recherche.

- Soit L_k = longueur de l'intervalle de recherche à l'itération k
- A chaque itération, $L_{k+1} \simeq L_k/2$
- Donc $L_p \simeq L_0/2^p$, avec p : nombre d'itérations
- *i.e.* $1 \simeq n/2^p$
- Donc $2^p \simeq n$, *i.e.* $p \simeq \log_2(n)$

A chaque itération, opérations significatives ?
1 comparaison : $x \leq V(med)$

Conclusion : complexité de l'ordre de $\log_2 n$ comparaisons
NB : nous ferons un calcul plus précis en TD