

# TD3 – Recherche dichotomique

Killian Reine

## Rappels de cours.

### Introduction au TD

La recherche dichotomique est un algorithme très efficace pour retrouver un élément dans un tableau trié. L'idée principale est de réduire l'espace de recherche de moitié à chaque étape, en comparant l'élément recherché avec l'élément du milieu.

Ce TD a pour objectif de :

- Comprendre le fonctionnement de la recherche dichotomique.
- Savoir la mettre en œuvre en pseudo-code.
- Étudier manuellement les étapes et opérations.
- Identifier le meilleur cas, le pire cas, et le nombre de comparaisons.

## Exercice 1

### Comprendre le principe

Soit le tableau trié :

$$T = [2, 4, 7, 10, 15, 20, 25]$$

On cherche  $x = 4$ .

#### 1 Quelle est la taille du tableau ?

Le tableau contient 7 éléments. La taille du tableau est donc  $n = 7$ .

#### 2 Quel est l'élément du milieu ?

Pour trouver l'élément du milieu, on calcule l'indice du milieu :

$$\text{milieu} = \left\lfloor \frac{\text{debut} + \text{fin}}{2} \right\rfloor = \left\lfloor \frac{0 + 6}{2} \right\rfloor = 3$$

L'élément du milieu est donc  $T[3] = 10$  (en indexant à partir de 0).

*Remarque :* Si on indexe à partir de 1, l'indice du milieu serait  $\lfloor \frac{1+7}{2} \rfloor = 4$  et l'élément du milieu serait  $T[4] = 10$ .

#### 3 Comparer $x$ avec l'élément du milieu. Quelle partie du tableau devons-nous conserver pour continuer la recherche ?

On compare  $x = 4$  avec l'élément du milieu  $T[3] = 10$  :

$$4 < 10$$

L'élément recherché est strictement inférieur à l'élément du milieu. Puisque le tableau est trié, l'élément recherché ne peut se trouver que dans la partie gauche du tableau.

On conserve donc la partie :  $T[0..2] = [2, 4, 7]$ .

#### 4 Répéter jusqu'à trouver l'élément.

Étape 2 :

- Intervalle :  $[0, 2]$
- Milieu :  $\lfloor \frac{0+2}{2} \rfloor = 1$
- Élément du milieu :  $T[1] = 4$
- Comparaison :  $4 = 4$  ✓

L'élément est trouvé à l'indice 1.

**Conclusion :** L'algorithme a trouvé l'élément  $x = 4$  en 2 étapes (comparaisons).

**Exercice 2****Écrire l'algorithme****1 Précisez les entrées et sorties.**— **Entrées :**

- un tableau  $T$  de taille  $n \geq 1$ , trié dans l'ordre croissant ;
- un élément  $x$  à rechercher, appartenant au même ensemble que les éléments de  $T$  ;
- deux indices debut et fin délimitant l'intervalle de recherche dans le tableau.

— **Sortie :**

- soit un indice  $i$  tel que  $T[i] = x$  si l'élément est présent dans l'intervalle  $[\text{debut}, \text{fin}]$  ;
- soit une valeur spéciale (par exemple  $-1$  ou Faux) indiquant que  $x$  n'appartient pas au tableau.

**2 Algorithme de la recherche dichotomique (version récursive).**

```

Fonction RecherchedichotomiqueProcédure RechercheDichotomique(T, x, debut, fin)
  si debut > fin alors
    retourner -1 // Élément non trouvé
  fin si

  milieu ← floor((debut + fin) / 2)

  si T[milieu] = x alors
    retourner milieu // Élément trouvé
  sinon si T[milieu] > x alors
    // Rechercher dans la partie gauche
    retourner RechercheDichotomique(T, x, debut, milieu - 1)
  sinon
    // Rechercher dans la partie droite
    retourner RechercheDichotomique(T, x, milieu + 1, fin)
  fin si
Fin Procédure

```

**Convention algorithmique :**

- Les indices du tableau commencent à 0.
- L'appel initial se fait avec  $\text{debut} = 0$  et  $\text{fin} = n - 1$  où  $n$  est la taille du tableau.
- L'algorithme est récursif : il s'appelle lui-même avec un intervalle réduit.

**Exercice 3****Simulation et étude des cas**

Pour le tableau :

$$T = [3, 8, 12, 18, 25, 30, 42, 50]$$

**1 Chercher  $x = 25$ .**

Étape	debut	fin	milieu	T[milieu]	Comparaison
1	0	7	3	18	$25 > 18 \rightarrow$ droite
2	4	7	5	30	$25 < 30 \rightarrow$ gauche
3	4	4	4	25	$25 = 25$ ✓

**Résultat :** L'élément 25 est trouvé à l'indice 4 en 3 comparaisons.

**2 Chercher  $x = 7$ .**

Étape	debut	fin	milieu	T[milieu]	Comparaison
1	0	7	3	18	$7 < 18 \rightarrow$ gauche
2	0	2	1	8	$7 < 8 \rightarrow$ gauche
3	0	0	0	3	$7 > 3 \rightarrow$ droite
4	1	0	-	-	debut > fin

**Résultat :** L'élément 7 n'est pas trouvé (retourne  $-1$ ). Il a fallu 3 comparaisons pour déterminer son absence.

**Exercice 4****Analyse de complexité****1 Pour un tableau de taille  $N$ , combien de comparaisons au meilleur cas ?**

Le meilleur cas se produit lorsque l'élément recherché se trouve exactement au milieu du tableau dès la première itération.

Nombre de comparaisons dans le meilleur cas :  $\boxed{1}$

**2 Pour un tableau de taille  $N$ , combien de comparaisons au pire cas ?**

Le pire cas se produit lorsque l'élément recherché se trouve à l'une des extrémités du tableau, ou qu'il n'est pas présent.

À chaque étape, la taille de l'intervalle de recherche est divisée par 2.

Le nombre maximal d'étapes nécessaires pour réduire un tableau de taille  $N$  à un tableau de taille 1 est donné par :

$$k = \lceil \log_2(N) \rceil + 1$$

Nombre de comparaisons dans le pire cas :  $\boxed{\lceil \log_2(N) \rceil + 1}$

Exemples :

- Pour  $N = 8$  :  $\lceil \log_2(8) \rceil + 1 = 3 + 1 = 4$  comparaisons
- Pour  $N = 16$  :  $\lceil \log_2(16) \rceil + 1 = 4 + 1 = 5$  comparaisons
- Pour  $N = 1000$  :  $\lceil \log_2(1000) \rceil + 1 \approx 10 + 1 = 11$  comparaisons

**3 Pourquoi la recherche dichotomique est plus efficace qu'une recherche séquentielle pour de grands tableaux ?**

Comparaison des deux algorithmes :

Algorithme	Meilleur cas	Pire cas
Recherche séquentielle	1	$N$
Recherche dichotomique	1	$\lceil \log_2(N) \rceil + 1$

**Avantages de la recherche dichotomique :**

- **Croissance logarithmique** : Le nombre de comparaisons croît très lentement avec la taille du tableau. Par exemple, pour un tableau de 1 000 000 d'éléments :
  - Recherche séquentielle : jusqu'à 1 000 000 comparaisons
  - Recherche dichotomique : environ 20 comparaisons
- **Efficacité pour les grandes données** : Plus le tableau est grand, plus l'avantage de la recherche dichotomique est important.
- **Principe de division** : À chaque étape, on élimine la moitié des éléments restants, ce qui permet de converger très rapidement vers la solution.

**Limitation** : La recherche dichotomique nécessite que le tableau soit **trié au préalable**. Si le tableau n'est pas trié, il faut d'abord le trier, ce qui a un coût en temps.

**Conclusion** : Pour des tableaux triés de grande taille, la recherche dichotomique est nettement plus efficace qu'une recherche séquentielle.