Ecole Supérieure
d'Informatique et du Numérique
COLLEGE OF ENGINEERING & ARCHITECTURE

COMPTE RENDU

Algorithmique Avancée - TP1
3e année Cybersécurité - École Supérieure d'Informatique et du Numérique (ESIN)
Collège d'Ingénierie & d'Architecture (CIA)

Étudiant : HATHOUTI Mohammed taha

Filière : Cybersécurité

Année : 2025/2026

Enseignants : M.BAKHOUYA

Date : 27 septembre 2025

1 Rappel des objectifs du TP

Durant ce TP, nous avons étudié et comparé l'efficacité de différents algorithmes de recherche. Nous avons implémenté et analysé deux problèmes distincts :

1.1 Exercice 1 : Recherche du maximum dans un tableau

L'objectif était d'implémenter deux méthodes pour trouver le maximum dans un tableau de n éléments entiers :

— **Méthode incrémentale** : parcours séquentiel du tableau, complexité théorique O(n)
— **Méthode diviser pour régner** : division récursive du tableau, complexité théorique O(n)

1.2 Exercice 2 : Recherche binaire d'une valeur

L'objectif était d'implémenter la recherche binaire dans un tableau trié en utilisant :
— **Version itérative** : utilisation d'une boucle, complexité théorique O(log n)
— **Version récursive** : appels récursifs, complexité théorique O(log n)
Les expérimentations visaient à comparer les temps d'exécution réels avec les complexités théoriques respectives et à analyser les performances relatives de chaque approche.

2 Analyse expérimentale et résultats

2.1 Exercice 1 : Recherche du maximum

2.1.1 Première approche du problème

Nous avons tout d'abord implémenté les deux méthodes les plus courantes pour la recherche du maximum dans un tableau non trié.

Méthode incrémentale : Cette approche consiste à parcourir entièrement le tableau en vérifiant chacune de ses valeurs en maintenant le maximum courant :

— 100 000 éléments : 711 µs
— 500 000 éléments : 1 587 µs
— 1 000 000 éléments : 2 701 µs
— 5 000 000 éléments : 14 125 µs
— 100 000 000 éléments : 262 821 µs
— 500 000 000 éléments : 1 137 582 µs
— 1 000 000 000 éléments : 2 218 981 µs

On observe une croissance linéaire des temps d'exécution, cohérente avec la complexité théorique O(n). Cette méthode est très efficace en termes d'implémentation et de consommation mémoire.

Méthode diviser pour régner : Cette approche divise récursivement le tableau en deux parties et compare les maxima :

— 100 000 éléments : 1 113 µs
— 500 000 éléments : 7 138 µs
— 1 000 000 éléments : 11 620 µs
— 5 000 000 éléments : 48 965 µs
— 100 000 000 éléments : 987 315 µs

— 500 000 000 éléments : 4 833 437 µs
— 1 000 000 000 éléments : 9 814 589 µs

Bien que théoriquement en O(n), cette méthode présente des temps d'exécution systématiquement plus élevés que la méthode incrémentale tout en gardant une croissance bien linéaire, principalement à cause du coût des appels récursifs et de la gestion de la pile d'appels.

2.1.2 Comparaison graphique

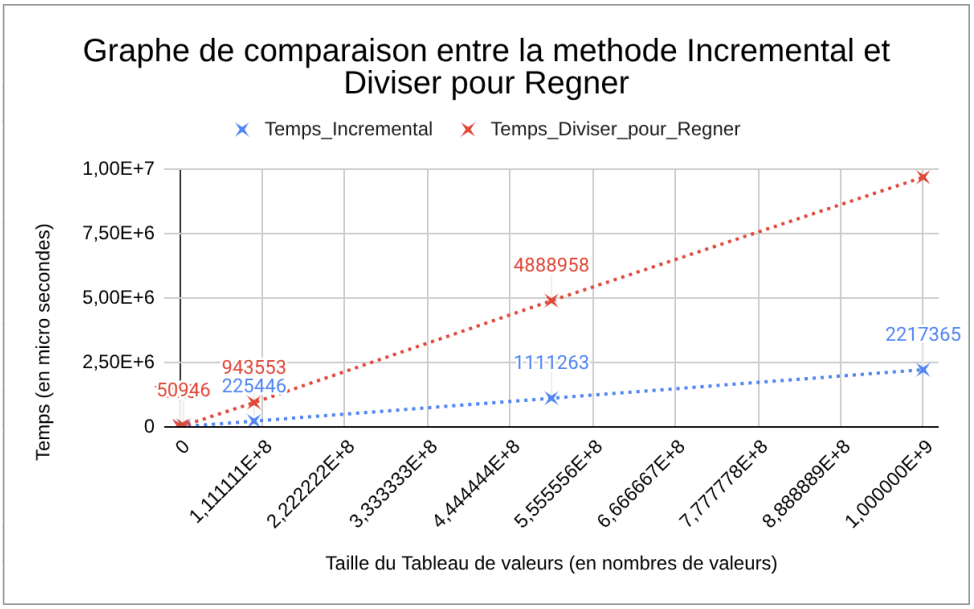

FIGURE 1 – Comparaison des temps d'exécution pour la recherche du maximum

Le graphique confirme la croissance linéaire pour les deux méthodes, avec un avantage net pour l'approche incrémentale.

2.2 Exercice 2 : Recherche binaire

2.2.1 Analyse des performances

Les deux implémentations de la recherche binaire ont été testées sur des tableaux triés de tailles croissantes (puissances de 2) :

Recherche binaire itérative :
— 1 048 576 éléments : 0.09 µs
— 2 097 152 éléments : 0.05 µs
— 4 194 304 éléments : 0.16 µs
— 8 388 608 éléments : 0.12 µs
— 16 777 216 éléments : 0.09 µs
— 33 554 432 éléments : 0.09 µs
— 67 108 864 éléments : 0.09 µs
— 134 217 728 éléments : 0.08 µs

Recherche binaire récursive :
— 1 048 576 éléments : 0.11 µs
— 2 097 152 éléments : 0.07 µs

— 4 194 304 éléments : 0.12 µs
— 8 388 608 éléments : 0.16 µs
— 16 777 216 éléments : 0.11 µs
— 33 554 432 éléments : 0.08 µs
— 67 108 864 éléments : 0.10 µs
— 134 217 728 éléments : 0.09 µs

2.2.2 Interprétation des résultats

Les résultats confirment bien la complexité O(log n) des deux méthodes. Les temps restent très stables (entre 0.05 et 0.16 µs) même quand on passe de 1 million à 134 millions d'éléments.

Les deux versions (itérative et récursive) donnent des performances quasiment identiques, ce qui correspond à la théorie. Les petites différences qu'on observe sont surtout dues aux limites de précision de clock(), qui n'est pas très précis pour mesurer des algorithmes aussi rapides que la recherche binaire.

Algorithme	**Complexité**	**Observation**	**Conformité**
Max. incrémental	O(n)	Linéaire	Oui
Max. diviser/régner	O(n)	Linéaire	Oui
Rech. bin. itérative	O(log n)	Constant	Oui
Rech. bin. récursive	O(log n)	Constant	Oui

TABLE 1 – Conformité théorie vs expérimentation

3 Conclusion

Les objectifs du TP ont été pleinement atteints. Les résultats expérimentaux confirment les complexités théoriques de tous les algorithmes étudiés :

3.1 Points clés observés

1. **Recherche du maximum** : La méthode incrémentale est plus rapide que la méthode diviser pour régner, même si elles ont toutes les deux une complexité O(n). Cela vient du fait que la méthode récursive fait beaucoup d'appels de fonctions qui prennent du temps.
2. **Recherche binaire** : Les deux méthodes (itérative et récursive) ont des performances similaires et respectent bien la complexité O(log n). Les temps restent constants même quand on augmente beaucoup la taille du tableau, ce qui montre que la recherche binaire est très efficace.

3.2 Enseignements

Ce TP montre qu'il est important de tester les algorithmes en pratique et pas seulement en théorie. Même si deux algorithmes ont la même complexité théorique (comme O(n) pour les deux méthodes de recherche du maximum), leurs performances réelles peuvent être très différentes. Pour la recherche binaire, les résultats confirment que les algorithmes

en O(log n) sont très efficaces même sur de gros volumes de données. Pour la recherche du maximum, on voit que parfois la méthode la plus simple (incrémentale) marche mieux que la plus complexe (diviser pour régner), même si elles ont théoriquement la même complexité.