école dE TECHNOLOGIE SUPÉRIEURE

UNIVERSITÉ DU QUÉBEC

Rapport de laboratoire 1

PRÉSENTÉ À

SAbeur Lafi

Dans le cadre du cours :

ALGORITHMES

ELE440-01

par

Julien lemay (LEMJ16059303)

Alexandre Lessard (LESA30099400)

montréal, le 2 octobre 2015

cid:image002.jpg@01CCE021.010CA410 Julien Lemay et Alexandre Lessard, 2015

Sommaire

[Introduction 1](#_Toc431509799)

[1. Tri par insertion 2](#_Toc431509800)

[1.1. Algorithme 2](#_Toc431509801)

[1.2. Analyse théorique 2](#_Toc431509802)

[1.3. Analyse expérimentale 2](#_Toc431509803)

[2. Tri par fusion 3](#_Toc431509804)

[2.1. Algorithme 3](#_Toc431509805)

[2.2. Analyse théorique 3](#_Toc431509806)

[2.3. Analyse expérimentale 3](#_Toc431509807)

[3. Tri par pigeonnier 4](#_Toc431509808)

[3.1. Algorithme 4](#_Toc431509809)

[3.2. Analyse théorique 4](#_Toc431509810)

[3.3. Analyse expérimentale 4](#_Toc431509811)

[4. Tri par tas 5](#_Toc431509812)

[4.1. Algorithme 5](#_Toc431509813)

[4.2. Analyse théorique 5](#_Toc431509814)

[4.3. Analyse expérimentale 5](#_Toc431509815)

[5. Tri rapide 6](#_Toc431509816)

[5.1. Algorithme 6](#_Toc431509817)

[5.2. Analyse théorique 6](#_Toc431509818)

[5.3. Analyse expérimentale 6](#_Toc431509819)

[6. Tri par base 7](#_Toc431509820)

[6.1. Algorithme 7](#_Toc431509821)

[6.2. Analyse théorique 7](#_Toc431509822)

[6.3. Analyse expérimentale 7](#_Toc431509823)

[Conclusion 8](#_Toc431509824)

# Introduction

Le but de ce laboratoire est d’étudier le comportement et la performance de plusieurs algorithmes de tri selon différentes configurations.

Les objectifs de ce laboratoire sont l’implémentation des différents algorithmes en langage de programmation C, C++ ou Java ainsi que d’utiliser l’analyse asymptomatique afin d’analyser les performances des algorithmes de tri.

Ce rapport comporte trois grandes parties pour chacun des 6 algorithmes. La première partie contient les pseudo-codes qui forment le squelette de nos fonctions de tri. La seconde contient l’analyse théorique où l’on discute de l’efficacité des algorithmes se basant sur leurs formules asymptotiques et baromètres. La troisième section contient l’analyse expérimentale des 6 méthodes de tri c'est-à-dire une description de l’algorithme de test, les résultats obtenus par les 6 fonctions et les conclusions qu’on peut en tirer.

# Tri par insertion

## Algorithme

Voici l’algorithme de tri par insertion tiré de la page Wikipedia de cet algorithme1:

**procédure** tri\_insertion(tableau T, entier n)

**pour** i de 1 à n-1

x ← T[i]

j ← i

**tant que** j > 0 et T[j - 1] > x

T[j] ← T[j - 1]

j ← j - 1

**fin tant que**

T[j] ← x

**fin pour**

**fin procédure**

Le tri par insertion fonctionne un peu comme on trie les cartes dans nos mains. On prend chaque carte (valeur). En allant de gauche à droite, une valeur à la fois, on vérifie si la valeur précédente est plus petite. Si oui, on la déplace devant cette valeur et on continue tant qu’il n’y a pas de valeurs plus petites (ou plus de valeurs) à gauche. Une fois la première valeur faite, on passe à la deuxième et ainsi de suite. Rendu à la fin, le « paquet » de valeurs est maintenant trié en ordre croissant.

## Analyse théorique

Nos baromètres sont :

Le baromètre le plus important à surveiller est .

Dans le meilleurs des cas, le tableau est en ordre et on ne rentre qu’une fois dans la boucle « tant que » de la ligne 5 à chaque « pour » à la ligne 2. Dans le pire des cas, le tableau est en désordre total et on rentre N fois dans la boucle « tant que » de la ligne 5 à chaque « pour » à la ligne 2.

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Min/Max |  |  |

Donc, pour le tri par insertion :

## Analyse expérimentale

L’analyse expérimentale a été effectuée selon le protocole indiqué dans les instructions du laboratoire 1.

Lorsqu’on observe les performances selon N qui est le nombre de données, on remarque facilement que l’asymptote supérieure suit une courbe qui est de l’ordre de . L’asymptote inférieure, un peu plus dur à discerner, est quant à lui de l’ordre de . L’ordre moyen est aussi de l’ordre de .

Lorsqu’on observe les performances selon R qui est l’intervalle de données, on remarque que peut importe l’intervalle de donnée, le nombre d’instructions est indépendant de l’intervalle de données.

Lorsqu’on observe les performances selon D qui est le degré de désordre des données, on remarque que le nombre d’instruction est proportionnel au degré de désordre des données.

# Tri par fusion

## Algorithme

Voici l’algorithme de tri par fusion tiré du livre *Introduction à l’algorithmique*2:

**FUSION**(A, p, q, r)

n1 ← q − p + 1

n2 ← r − q

**créer tableaux** L[1 . . n1 + 1] et R[1 . . n2 + 1]

**pour** i ← 1 à n1

**faire** L[i] ← A[p + i − 1]

**pour** j ← 1 à n2

**faire** R[j] ← A[q + j]

L[n1 + 1]←

R[n2 + 1]←

i ← 1

j ← 1

**pour** k ← p à r

**faire** si L[i] <= R[j]

**alors** A[k] ← L[i]

i ← i + 1

**sinon** A[k] ← R[j]

j ← j + 1

**TRI-FUSION**(A, p, r)

**si** p < r alors

q ← (p + r)/2

**TRI-FUSION**(A, p, q)

**TRI-FUSION**(A, q + 1, r)

**FUSION**(A, p, q, r)

Cet algorithme fonction par la stratégie « *Divide and conquer »*. À chaque itération, l’algorithme sépare le nombre de données en deux et renvoie chaque moitié en récursivité. Une fois chaque donnée séparée, les données sont ordonnées par paquets de 2 blocs à la fois. Ces blocs grossissent passant de taille 1 à la moitié de la taille total. À la fin, les deux derniers blocs sont réunis ensembles (et trier) et trier une dernière fois. Le tri est maintenant terminé.

## Analyse théorique

Si on analyse *Tri-Fusion(A,p,r)*, on trouve que c’est une fonction récursive. Si le nombre de données est en bas de 2, le nombre d’opérations effectuées est proche de 1. Si le nombre de données est plus grand ou égal à deux, les données sont séparés en deux moitié et renvoyer récursivement. Une fois, que la récursivité est terminée, elle passe dans une fonction de fusion. Cet algorithme suit donc le modèle suivant :

Trouvons l’ordre de :

Les baromètres de la fonction sont :

\*\*Attention, N est toujours au minimum à 2.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Analyse expérimentale

# Tri par pigeonnier

## Algorithme

**procédure** tri\_pigeonnier(Tableau de données, Nombre de données, Valeur maximum)

**Pour** la valeur maximal d’un nombre dans le tableau

Initialiser cette valeur à zéro.

**Pour** le nombre de données dans le tableau

Placer la valeur dans son pigeonnier respectif

**Pour** tous les chiffres du pigeonnier

**Tant** qu’il y a des chiffres dans le pigeonnier

Placer la valeur directement après la dernière valeur placée dans le tableau trié.

Incrémenter de 1 le chiffre à remplacer dans le tableau à retourner.

**fin procédure**

## Analyse théorique

## Analyse expérimentale

# Tri par tas

## Algorithme

## Analyse théorique

## Analyse expérimentale

# Tri rapide

## Algorithme

Voici l’algorithme de tri par rapide tiré de la page Wikipedia de cet algorithme5:

**partitionner**(tableau T, entier premier, entier dernier, pivot)

{

**échanger** T[pivot] et T[dernier]

j :=premier

**pour** i de premier à dernier - 1

{

si T[i] <= T[dernier] alors

échanger T[i] et T[j]

j := j + 1

}

échanger T[dernier] et T[j]

**renvoyer** j

}

**tri\_rapide**(tableau T, entier premier, entier dernier)

**début**

**si** premier < dernier **alors**

pivot := choix\_pivot(T, premier, dernier)

pivot := partitionner(T, premier, dernier, pivot)

tri\_rapide(T, premier, pivot-1)

tri\_rapide(T, pivot+1, dernier)

**fin si**

**fin**

## Analyse théorique

## Analyse expérimentale

# Tri par base

## Algorithme

## Analyse théorique

## Analyse expérimentale

# Conclusion

# Annexe

# Références

1 :Wikipedia. Tri par insertion,[En ligne], https://fr.wikipedia.org/wiki/Tri\_par\_insertion (Consulté le 1er octobre 2015).

2 : Cormen,T., Leiserson,C., Rivest, R., Stein, C.(2001). Introduction à l’algorithmique(2e ed.). MIT : DUNOD.

5 :Wikipedia. Tri rapide,[En ligne], https://fr.wikipedia.org/wiki/Tri\_rapide   
(Consulté le 1er octobre 2015).