école dE TECHNOLOGIE SUPÉRIEURE

UNIVERSITÉ DU QUÉBEC

Rapport de laboratoire 2

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

# Introcuction

Cette première section contient une description des buts du laboratoire et des objectifs visés. Elle sert à introduire les sections qui suivent.

Ce laboratoire consiste à utiliser les algorithmes de tri créés dans le laboratoire 1 pour créer des algorithmes de recherches. Par la suite, les programmes de recherche seront testés et analysés pour comparer les résultats dans différentes situation pour créer une fonction de recherche optimale dans tous les cas.

# Les algotithmes de recherche

La deuxième section explique le principe de fonctionnement de chaque algorithme et donne son pseudocode, sans oublier de mentionner sa provenance. Si des modifications ont été apportées aux algorithmes suggérés, il faut les expliquer et les justifier. Cette section donne aussi un court rapport sur les difficultés et autres observations intéressantes rencontrées lors de l’implémentation.

## Recherche séquentielle

### Pseudo-code

Pour toutes les cases du tableau

Si c’est la valeur qu’on recherche

Retourner cette valeur.

### Explication

La recherche séquentielle est la plus simple. Elle consiste à tout simplement itérer a travers le tableau de valeurs jusqu’à ce qu’elle trouve la valeur recherché.

### Difficultés

Aucune difficulté à été rencontré lors de l’implémentation de cette fonction.

## Recherche binaire

### Pseudo-code

Tant que la valeur du milieu de la plage n’est pas la valeur recherchée

Si la valeur du milieu de la plage est plus petite que la valeur de la clé

Le début de la plage devient le milieu + 1

Sinon

La fin de la plage devient le milieu – 1

Le milieu devient la moyenne entre le début et la fin de la plage.

### Explication

Cet algorithme coupe constamment le problème en deux ce qui permet d’éliminer beaucoup de données raapidement. Cependant, pour utiliser cet algorithme, il est nécessaire de trier les données avant de rechercher la donnée. Son fonctionnement consiste à isoler la valeur entre une limite basse (le début) et une limite haute (la fin). À chaque itération, la fonction compare la clé avec la valeur du milieu entre le début et la fin puis retranche de la plage les nombre plus haut que le milieu si la clé est inférieur a celui-ci et vice versa pour le contraire.

### Difficultés

On a eu des difficulté avec nos IDE puisque un d’entre nous utilisait les normes de C++ 2011 tandis que l’autre utilisait ceux de 1998 donc lorsqu’on partagait le code ca ne fonctionnait que sur un seul ordinateur à la fois.

## Recherche par d’arbre binaire

### Pseudo-code

Tant que le résultat recherché n’est pas atteint

Si la clé est plus petite que la valeur du nœud

Descendre au nœud de gauche

Sinon

Descendre au nœud de droite

### Explication

Le problème avec l’arbre c’est qu’il doit être souvent rebalancé pour avoir la même profondeur sur toutes les branches de l’arbre. Le fonctionnement de cet algorithme est simplement de descendre dans l’hierachie de nombre utilisant deux simple règles qui ne changent jamais : si la clé est plus petite que la valeur du nœud, on descend au prochain nœud à gauche et si la clé est plus grande que la valeur du nœud, on descend vers le prochain nœud à droite jusqu’à ce qu’on retrouve la valeur demandée.

### Difficultés

Puisque un de nous deux était un peu moins familier avec la programmation orienté objet, on s,est inspiré d’un tutoriel en JAVA pour comprendre comment coder l’algorithme et nous avons eu beaucoup de problèmes avec les pointeurs NULL puisqu’en JAVA, les pointeurs n’existent pas.

## Recherche par table de hachage

### Pseudo-code

Garder le restant de la division entre la valeur et un nombre de cases.

Tant que la case n’est pas vide

Pointer vers la prochaine maille de la chaine.

Placer l’information à la suite des autres informations dans la case portant le restant de la division.

### Explication

Cette fonction de hachage divise simplement la clé par le nombre de case qu’il y a d’alloué et garde le restant de l’opération pour lui attribuer cette case. Si de l’information se trouve déjà dans la case, l’information devient une liste chainé. L’algorithme doit alors itérer à travers tous les mailles de la chaine jusqu’à ce qu’il en trouve une de libre pour y insérer la nouvelle information.

Par la suite lorsqu’on recherche de l’information à partir d’une clé il suffit de passer la clé dans la fonction de hachage et elle vous retournera la case dans laquelle l’information est stocké. Cependant il faudra peut etre fouiller un peu avant de trouver.

### Difficultés

Ayant un peu de difficulté à comprendre le fonctionnement d’une fonction pour table de hachage, nous nous sommes inspiré d’exemples sur internet. Beaucoup de temps aura été perdu ici mais c’était intéressant tout de même. Nous n’avons pas pris le temps d’optimiser la fonction de hachage.

# L’analyse théorique

Dans la troisième section, il faut fournir les détails de l’analyse théorique de chaque algorithme et justifie les conclusions de l’analyse. Cette section indique aussi les baromètres retenus (et non ceux qui ont été éliminés). Il faut distinguer clairement les formules exactes (ex. nombre d'exécutions, E) et les formules asymptotiques.

## Recherche séquentielle

### Barêmes et formule

L’ordre asymptotique de cette recherche est de N puisque le pire cas serait lorsque l’élément rechrché se trouve à la toute fin du tableau de données.

### Analyse

L’analyse de cet algorithme est assez, simple. Grandement inefficace lorsque le nombre de données est torp grand mais peut être plus rapide que d,autres lorsque le nombre d’éléments n’est pas très gros puisque les autres fonction nécessite du temps pour préparer les données avant de lancer la recherche.

## Recherche binaire

### Barêmes et formule

Sans compter l’algorithme qui tri les données, la formule asymptotique est N/2. Ce résultat est déjà beaucoup mieux que la recherche séquentielle mais peut-on trouver mieux.

### Analyse

## Recherche par d’arbre binaire

### Barêmes et formule

### Analyse

## Recherche par table de hachage

### Barêmes et formule

### Analyse

# Choix heuristique

Pour la recherche binaire, il faut expliquer dans cette section comment sont choisis les algorithmes de tri en tenant compte des caractéristiques des données. Pour la recherche « optimisée », il faut justifier les motifs du choix de la stratégie de recherche selon les caractéristiques des données et le nombre de requêtes.

## Recherche binaire

Le choix de l’algorithme de tri était clair grace au laboratoire 1. Lors de ce dernier laboratoire nous sommes arrivé

## Recherche optimisée

# Conclusion

Dans cette section, il faut présenter sa conclusion quant à l’atteinte des objectifs de départ ainsi que sur les avantages et les inconvénients de chaque méthode de recherche.

# Références

<https://www.youtube.com/watch?v=MkbxOZAJM_w>

<https://www.youtube.com/watch?v=M6lYob8STMI>

https://www.youtube.com/watch?v=m6n\_rozU8dA

Cette dernière section contient les références de vos sources. Dans le corps du rapport, vous devez également mettre un renvoi après chaque élément emprunté, ex. [1], [2]...