*Département d'Informatique*

*INF1042, fiche td-tp séance 6*

*(Fondements mathématiques), Mai 2020*

*Coordonnées par :Professeur Maurice TCHUENTE ;*

*Dr Thomas MESSI NGUELÉ, Assistant Lecturer*

ETUDIANT : NOUTCHEU LIBERT JORAN
MATRICULE :19M2310

*July 10, 2020*

⋆ ⋆ Chapter un ⋆ ⋆

# probleme sur les Fondements mathématiques

## 1.1. Expliqu'ons l'algorithme du tri par bulles (bubble sort en anglais).

Le tri à bulles est un algorithme de tri. Il consiste à comparer répétitivement les éléments consécutifs d'un tableau, et à les permuter lorsqu'ils sont mal triés.

⊛ Sur un tableau de n éléments (numérotés de 1 à n), le principe du tri à bulles est le suivant :

∗ On parcourt le tableau à trier à l'envers, en comparant les éléments consécutifs deux-à-deux tout en faisant remonter vers le début du tableau les plus petits éléments.

∗ –La remontée est progressive : un élément remonte s'il est plus petit que son voisin de gauche.

∗ Lorsque l'on a fini de parcourir le tableau une première fois, on est sûr d'avoir placé le plus petit élément à la bonne place.

∗ On parcourt donc le nouveau tableau à l'envers pour placer le second plus petit élément, troisième plus petit, jusqu'au dernier.

### 1.1.1 Montrer que le nombre d'échanges est au plus égal au nombre de comparaisons

Le nombre de comparaisons "si Tab[ j-1 ] > Tab[ j ] alors" est une valeur qui ne dépend que de la longueur n de la liste (n est le nombre d'éléments du tableau), ce nombre est égal au nombre de fois que les itérations s'exécutent, le comptage montre que la boucle "pour i de n jusquà 1 faire" s'exécute n fois (donc une somme de n termes) et qu'à chaque fois la boucle "pour j de 2 jusquà i faire" exécute (i-2)+1 fois la comparaison "si Tab[ j-1 ] > Tab[ j ] alors".

La complexité en nombre de comparaisons est égale à la somme des n termes suivants (i = n, i = n-1,....)

C = (n-2)+1 + ([n-1]-2)+1 +.....+1+0 = (n-1)+(n-2)+...+1 = n(n-1)/2 (c'est la somme des n-1 premiers entiers). La complexité en nombre de comparaison est de de l'ordre de $n^2$, que l'on écrit O($n^2$).

Calculons par dénombrement le nombre d'échanges dans le pire des cas (complexité au pire = majorant du nombre d'échanges). Le cas le plus mauvais est celui où le tableau est déjà classé mais dans l'ordre inverse et donc chaque cellule doit être échangée, dans cette éventualité il y adonc autant d'échanges que de tests.

La complexité au pire en nombre d'échanges est de l'ordre de $n^2$, que l'on écrit O($n^2$).

### 1.1.2 Montrons que le nombre de comparaisons est au plus égal à n(n-1)/2

Choisissons comme opération élémentaire la comparaison de deux cellules du tableau.

Le nombre de comparaisons "si Tab[ j-1 ] > Tab[ j ] alors" est une valeur qui ne dépend que de la longueur n de la liste (n est le nombre d'éléments du tableau), ce nombre est égal au nombre de fois que les itérations s'exécutent, le comptage montre que la boucle "pour i de n jusquà 1 faire" s'exécute n fois (donc une somme de n termes) et qu'à chaque fois la boucle "pour j de 2 jusquà i faire" exécute (i-2)+1 fois la comparaison "si Tab[ j-1 ] > Tab[ j ] alors".

La complexité en nombre de comparaisons est égale à la somme des n termes suivants (i = n, i = n-1,....)

C = (n-2)+1 + ([n-1]-2)+1 +.....+1+0

= (n-1)+(n-2)+...+1 = n(n-1)/2 (c'est la somme des n-1 premiers entiers).

### 1.1.3 Montrons que dans le tri par bulles, le nombre d'échanges égale au nombre d'inversions.

Montrons que dans le tri par bulles, le nombre d'échanges égale au nombre d'inversions. Soit $t_1$ le tableau juste avant cet échange, $t_2$ le tableau juste après cet échange, et $i_0$ la valeur de la variable i au moment de l'échange. Un échange ne se produit que lorsque t[i] > t[i+1] et il s'agi d'un échange entre t[i] et t[i+ 1].Ainsi, dans $t_1$, on avait $t_1[i_0] < t_1[i_0+1]$ et $t_2$ est égal à $t_1$ dans lequel on a procédé à l'échange entre les éléments d'indices $i_0$ et $i_0 + 1$. Cet échange élimine exactement une inversion. En effet :

⋆ dans $t_1$, ($i_0$,$i_0$ +1) est une version, pas dans $t_2$

⋆ les autres inversions sont préservées :

- ∗ lorsque i < j et i,j tous deux différents de $i_0$ et $i_0 + 1$, (i,j) est une inversion dans $t_1$ si et seulement si c'est une inversion dans $t_2$;
- ∗ lorsque i < $i_0$ alors : (i, $i_0$) est une inversion dans $t_1$ si seulement si $(i, i_0 + 1)$ est une inversion dans $t_2$ et $(i, i_0 + 1)$ est une inversion dans $t_2$ ssi $(i, i_0)$ est une inversion dans $t_2$
- ∗ lorsque $i_0 + 1$ < j alors : $(I_0, j)$ est une inversion dans $t_1$ ssi $(i_0 + 1, j)$ est une inversion dans $t_2$ et $(i_0 + 1, j)$ est une inversion dans $t_1$ ssi $(i_0, j)$ est une inversion dans $t_2$;
- ∗ et ceci prend en considération toutes les inversions possibles dans $t_1$ et $t_2$.

**conclusion :** nous venons de voire que tout échange au cours du tris a bulles élimine

exactement une inversion. le nombre d'échanges effectués au cours de ce tris sur $t$ est donc la différence entre le nombre d'inversions au départ,O(n), et le nombre d'inversions à fin du tri Mais le tableau final est trié et un tableau trié ne contient aucune inversion. Donc le nombre d' échange est simplement O(n) Le nombre d'échanges est égal au nombre d'inversions dans le tableau inital

### 1.1.4 *Montrons que Inv(i,p) ∈ [0, i-1]*

Pour illustrer ce principe, prenons la suite de nombres suivante : 6 0 3 5 1 4 2
Nous voulons trier ces valeurs par ordre croissant. Commençons par le commencement. Nous allons faire un premier passage.

0 6 3 5 1 4 2 // On compare 6 et 3 : on inverse
0 3 6 5 1 4 2 // On compare 6 et 5 : on inverse
0 3 5 6 1 4 2 // On compare 6 et 1 : on inverse
0 3 5 1 6 4 2 // On compare 6 et 4 : on inverse
0 3 5 1 4 6 2 // On compare 6 et 2 : on inverse
0 3 5 1 4 2 6 // Nous avons terminé notre premier passage en effectuant 5 échange
ici Inv(6,p) =5
∈[0 ,6-1] Nous allons donc refaire un passage mais en ométtant la dernière case.
0 3 5 1 4 2 6 // On compare 0 et 3 : on laisse
0 3 5 1 4 2 6 // On compare 3 et 5 : on laisse
0 3 5 1 4 2 6 // On compare 5 et 1 : on inverse
0 3 1 5 4 2 6 // On compare 5 et 4 : on inverse
0 3 1 4 5 2 6 // On compare 5 et 2 : on inverse
0 3 1 4 2 5 6 // Nous avons terminé notre passage avec 3 echange ici Inv(5,p) =3 ∈[0 ,5-1]

Nous allons donc refaire un passage mais en ométtant la dernière case.
0 3 1 4 2 5 6 // On compare 0 et 3 : on laisse
0 3 1 4 2 5 6 // On compare 3 et 1 : on inverse
0 1 3 4 2 5 6 // On compare 3 et 4 : on laisse
0 1 3 4 2 5 6 // On compare 4 et 2 : on inverse
0 1 3 2 4 5 6 // Nous avons terminé notre passage avec 2 echange ici Inv(3,p) =1 ∈[0 ,3-1] et Inv(4 ,p) =1 ∈[0 ,3-1]

Nous allons donc refaire un passage mais en ométtant les dernière case tries au tour précedant. car le tableau n'est pas encore tries
0 1 3 2 4 5 6 // On compare 0 et 1 : on laisse
0 1 3 2 4 5 6 // On compare 1 et 3 : on laisse
0 1 3 2 4 5 6 // On compare 3 et 2 : on inverse
0 1 2 3 4 5 6 // Nous avons terminé notre passage avec 1 echange celui de 2 et 3 ici Inv(2,p) =0 ∈[0 ,2-1] et Inv(3 ,p) =1 ∈[0 ,3-1]

| 0 1 2 3 4 5 6 // On compare 0 et 1 : On laisse car $0 < 1$ |
|---|

| 0 1 2 3 4 5 6 // On compare 1 et 2 : On laisse car $1 < 2$ |
|---|

| 0 1 2 3 4 5 6 // Nous avons terminé notre passage |
|---|

a la fin de ce tris on constate que inv(i,p) retourne le nombre de fois qu'un élément i de p est permuter et vue qu'on effectue un tris dans l'odre croisant

**conclusion :** inv(i,p) retourne le nombre de permutation p tel que : $a_i > a_j$ et i<j. il y'aéchange d element $a_i$ en position i avec l element $a_j$ en position j une fois cette element triés on reduit de une case de moins pour eviter de tries encore car on sait que de [0 à $a_{i-1}$] $a_i$ est le plus grand ce qui diminue le nombre de comparaison a intervale de [0, i-1] or dans le tris de bulles le nombre de comparaison est égal au nombre d'inversion donc Inv(i,p) $\in$ [0, i-1]

Chapter Deux

# ensemble et application exercice d'application

## 2.1. Décrire les structures de données

ici R = $(x_1 , y_1 , c_1 )$, ... , $(x_m , y_m , c_m )$ un ensemble de liste chainées ayant chaqu'un un tableau de trois liste chaine OU $x_m , y_m , c_m$ sont des tronçons de route represente les valeur des liste chainées et a chaque passage sur fusionera ses tableaux