**Rapport de labo 4**

Groupe 2

Pontarolo Stefano, Baltensperger Daniel, Mbayo Guilain

1. But

Ce labo a pour but de tester les différentes fonctions de tri vue en cours, c’est-à-dire le tri à bulle, le tri par sélection et le tri par insertion.

1. Déroulement

Pour atteindre cet objectif, il nous était demandé de créer au moins 25 vecteurs de taille 10, 100, 1000 et 10000 contenants des valeurs (unsigned) aléatoire ainsi qu’un tableau déjà trié et un tableau trié dans l’ordre inverse de chacune des tailles précédemment citées.

Nous devions ensuite les trier de chacune des trois manières tout en comptant le nombre d’opérations effectuées et le temps que cela a pris. Grâce à ces données, nous avons pu calculer l’écart type et la moyenne du nombre d’opération ainsi que le temps nécessaire pour chaque taille de vecteur et chaque méthode de tri.

Ces résultats permettaient de voir très visuellement quel algorithme était le plus efficace en termes de nombre d’opération et de temps nécessaire en fonction du nombre de données contenue dans les vecteurs.

Nous avons également pu observer les mêmes données pour les cas extrême, c’est-à-dire un tableau déjà trié dans l’ordre croissant et un tableau trié dans l’ordre décroissant.

1. Résultats

Nous avons pu observer les résultats suivants (bien entendu les valeurs varient un peu d’une fois à l’autre comme les nombres contenus dans les vecteurs sont aléatoires) :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 10 entiers | 100 entiers | 1000 entiers | 10000 entiers |
| Tri à bulle | Moyenne de temps | 0 | 0.03988 | 4.94684 | 539.008 |
| Ecart type | 45.6399 | 592.289 | 11604.5 | 6218.11 |
| Moyenne du nombre d’opérations | 222 | 24820 | 2.5026 \* 106 | 7.80332 \* 107 |
| Nb d’opérations vecteur croissant | 34 | 304 | 3004 | 30004 |
| Nb d’opérations vecteur décroissant | 349 | 34954 | 3499504 | 349995004 |
| Tri par insertion | Moyenne de temps | 0 | 0.03988 | 1.43612 | 149.894 |
| Ecart type | 26.2298 | 451.465 | 11374.8 | 10216.5 |
| Moyenne du nombre d’opérations | 134 | 10527 | 1.01076 \* 106 | 9.98916 \* 107 |
| Nb d’opérations vecteur croissant | 56 | 596 | 5996 | 59996 |
| Nb d’opérations vecteur décroissant | 236 | 20396 | 2003996 | 200039996 |
| Tri par sélection | Moyenne de temps | 0 | 0.03988 | 2.63296 | 265.251 |
| Ecart type | 2 | 18.412 | 118.08 | 11459.1 |
| Moyenne du nombre d’opérations | 256 | 20797 | 2.01028 \* 106 | 2.83269 \* 107 |
| Nb d’opérations vecteur croissant | 247 | 20497 | 2004997 | 200049997 |
| Nb d’opérations vecteur décroissant | 272 | 22997 | 2254997 | 225049997 |

*En vert : nb le plus petit par rapport aux autres méthodes de tri.*

*En rouge : nb le plus grand par rapport aux autres méthodes de tri.*

1. Observations

Nous avons observé que plus les tableaux sont grands, plus ils nécessitent d’opérations et plus ils prennent de temps. Les résultats augmentent de manières quasi exponentielles.

Ces résultats ne correspondent pas exactement à la théorie vue en cours. En effet, le tri par insertion se révèle plus efficaces que le tri par sélections dans presque tous les cas. Un seul cas le montre comme défavorable, la moyenne du nombre d’opération sur les vecteurs aléatoires de taille 10000. En revanche, en termes de temps, le tri par insertion reste plus efficace lorsqu’un grand nombre de données sont traitées.

On voit que dans le cas favorable, la méthode du tri à bulle (qui nécessite le moins d’opérations pour « checker » le tableau en dehors de sa réorganisation) est tout le temps plus efficace. En revanche, il est mole moins efficaces lorsqu’ il faut déplacer un grand nombre de valeurs dans le vecteur.

Au contraire, dans le cas défavorable, le tri par insertion se montre comme le plus efficace, il n’a donc pas besoin d’autant d’opérations que les deux autres pour déplacer les valeurs au bon emplacement dans le vecteur.