A close-up of a sign

Description automatically generated

**INF8775 – Analyse et conception d’algorithmes**

**Groupe 01**

**TP1**

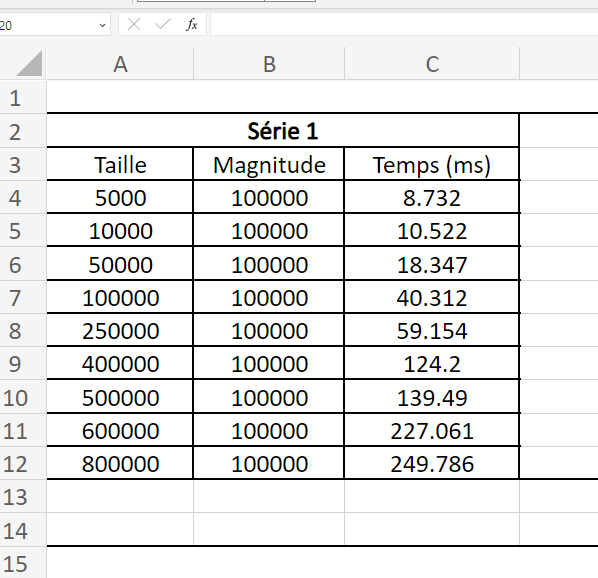
**2083768 - Dissou Hannan**

**2093771 - Tiomo Epongo, Einstein Franck**

**Soumis à: Pesant, Gilles**

**2023-10-05**

**Tableaux des résultats**



*Figure 1: Temps d’exécution tri par dénombrement Série 1*

A screenshot of a graph

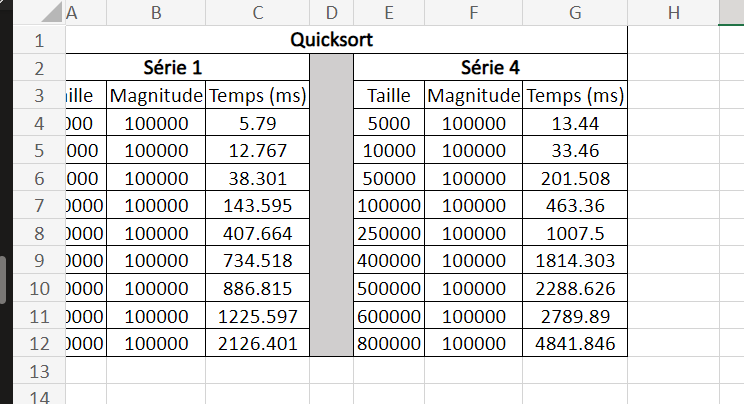
Description automatically generated

*Figure 2: Temps d’exécution tri par dénombrement Série 2*

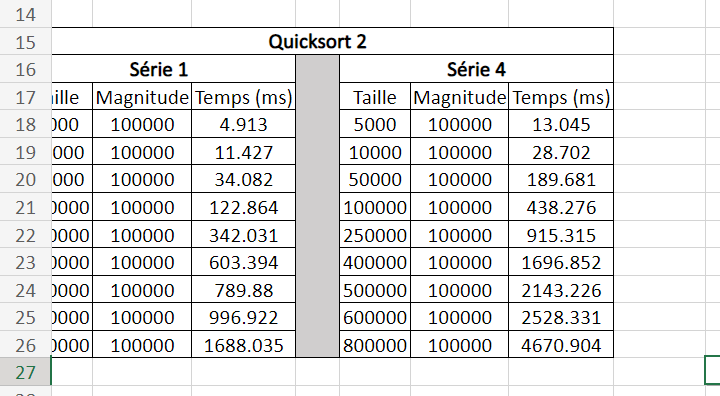
A screenshot of a spreadsheet

Description automatically generated

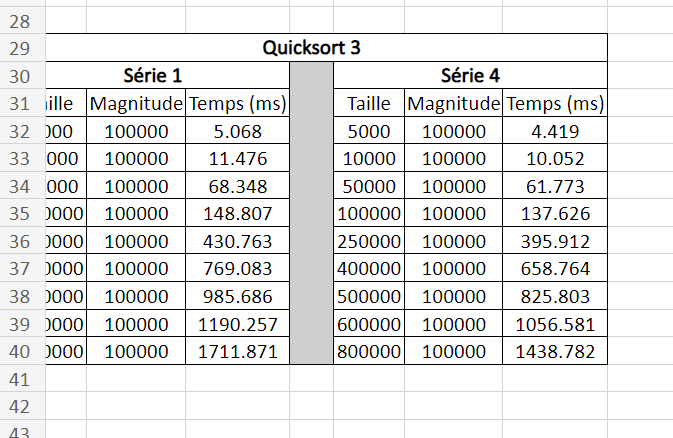
*Figure 3: Temps d’exécution tri par dénombrement Série 3*



*Figure 4: Temps d’exécution tri rapide 1 Série 1 et 4*



*Figure 5: Temps d’exécution tri rapide 2 Série 1 et 4*



*Figure 6: Temps d’exécution tri rapide 3 Série 1 et 4*

**Tests de puissance**

A graph with blue dots and numbers

Description automatically generated*Figure 7: Test de puissance du tri par dénombrement Série 1 et 3*

A graph with numbers and points

Description automatically generated

*Figure 8: Test de puissance du tri par dénombrement Série 2*

*A close-up of a graph

Description automatically generated*

*Figure 9: Test de puissance du tri rapide 1 Série 1*

*A graph with numbers and a number

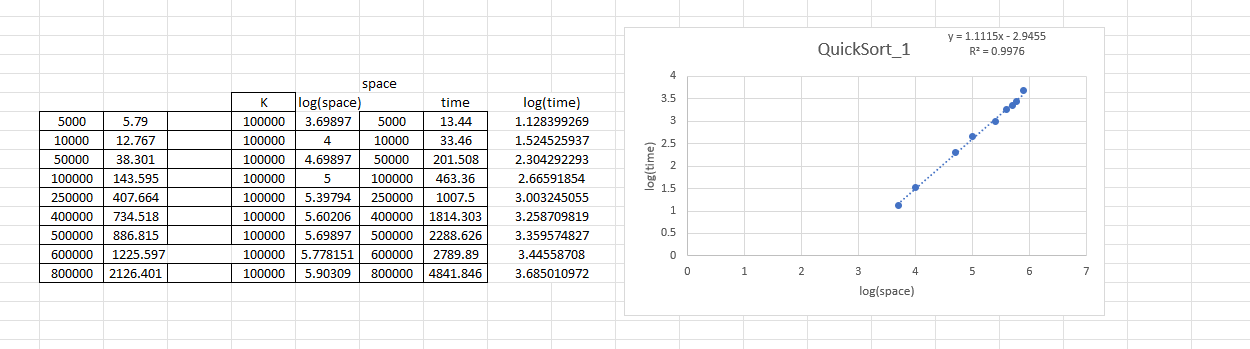
Description automatically generated with medium confidence*

*Figure 10: Test de puissance du tri rapide 2 Série 1*

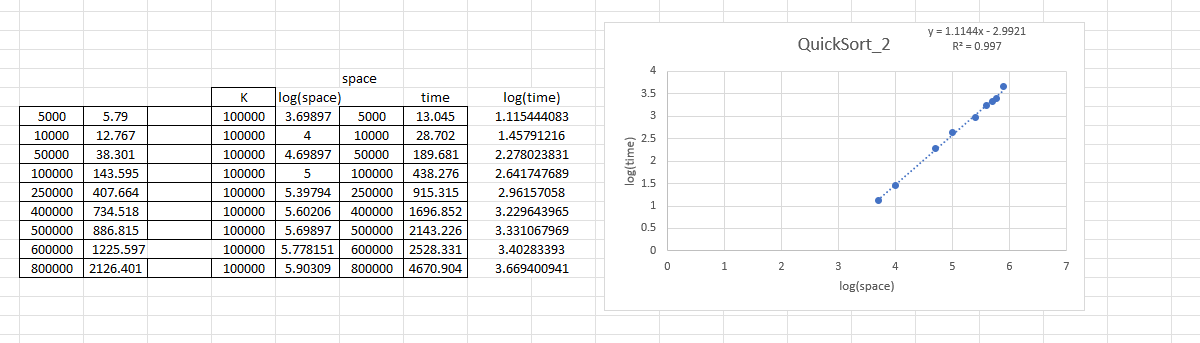
*A close-up of a graph

Description automatically generated*

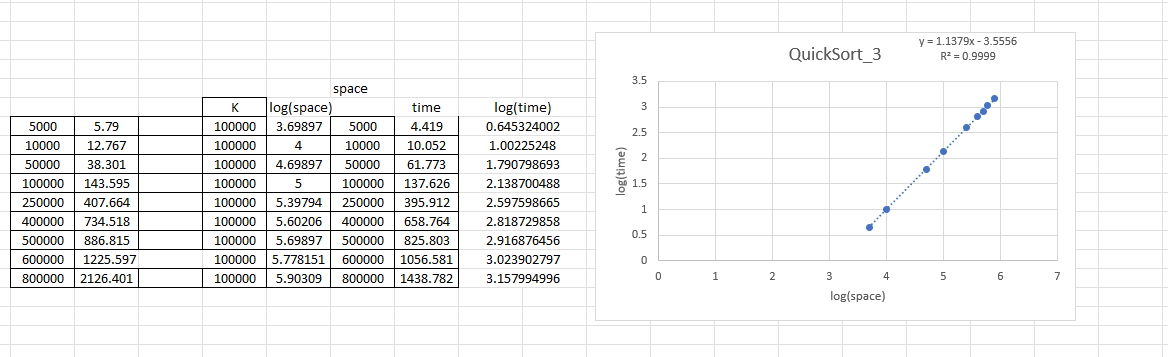
*Figure 11: Test de puissance du tri rapide 3 Série 1*

**

*Figure 12: Test de puissance du tri rapide 1 Série 4*

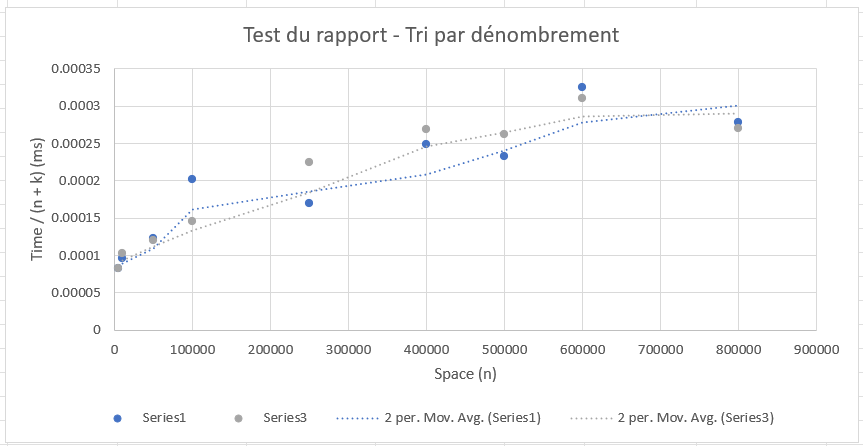
**

*Figure 13: Test de puissance du tri rapide 2 Série 4*

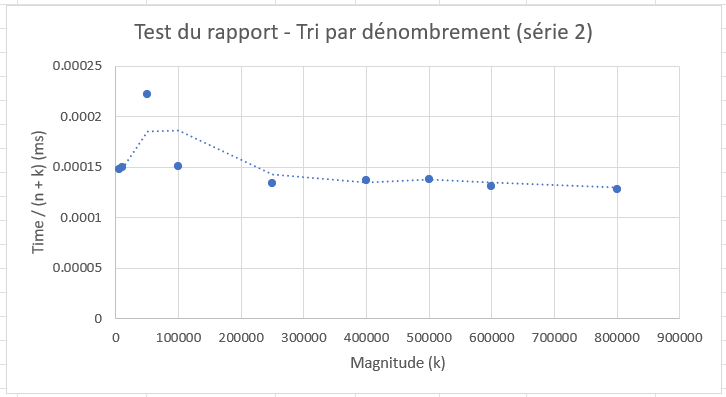
**

*Figure 14: Test de puissance du tri rapide 3 Série 4*

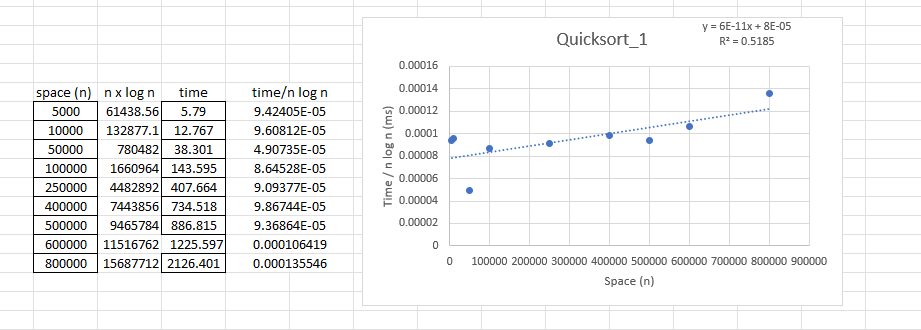
**Tests du rapport**



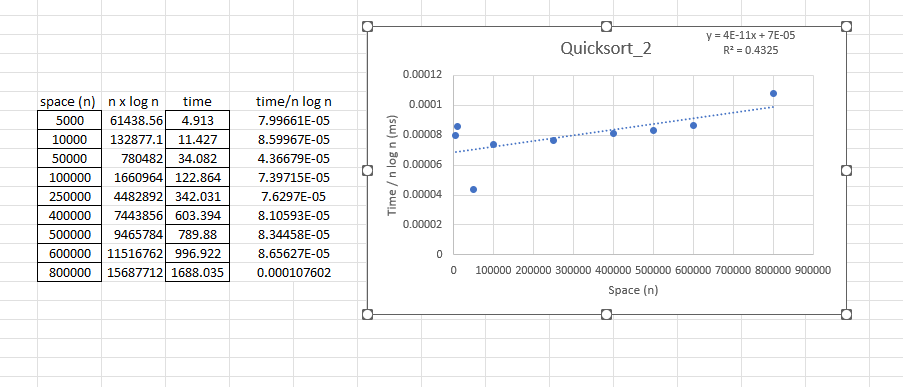
*Figure 15: Test du rapport du tri par dénombrement3 Série 1 et 3*

**

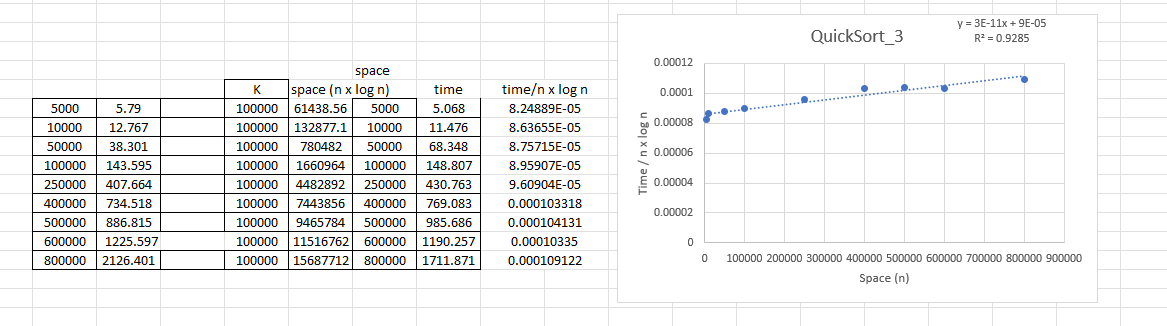
*Figure 16: Test de puissance du tri par dénombrement3 Série 2*

**

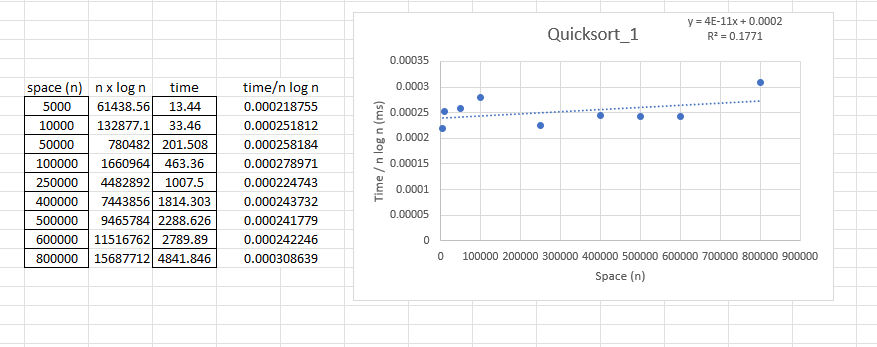
*Figure 17: Test du rapport tri rapide 1 Série 1*

**

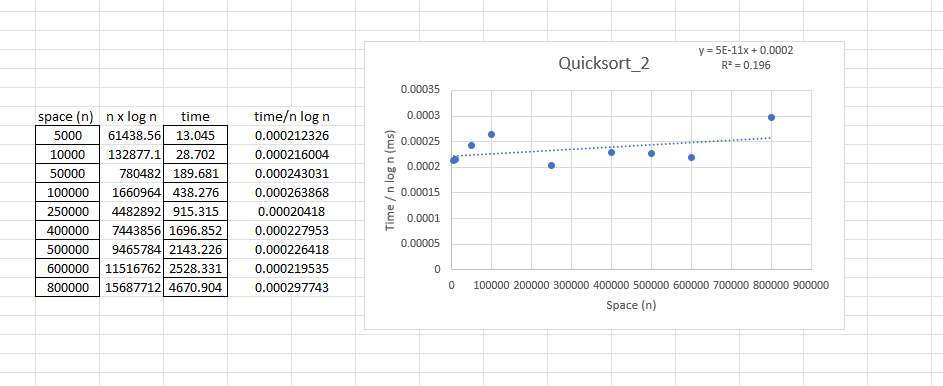
*Figure 18: Test du rapport tri rapide 2 Série 1*

**

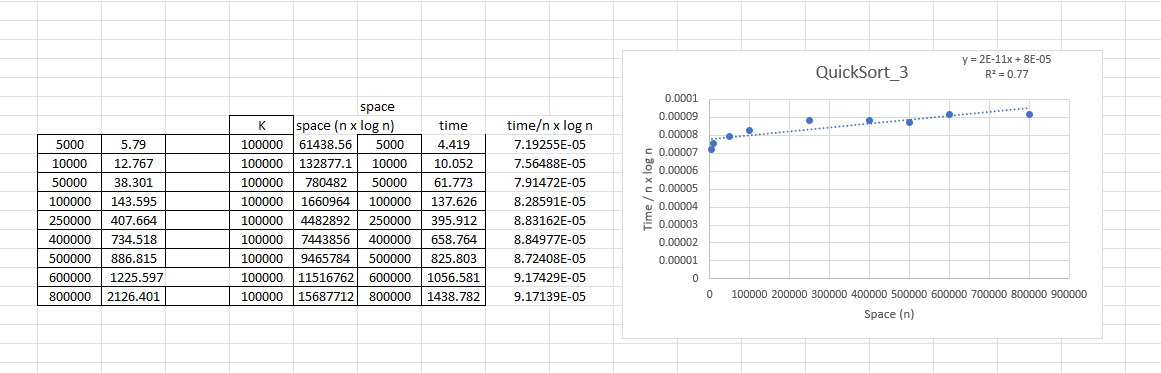
*Figure 19: Test du rapport tri rapide 3 Série 1*

**

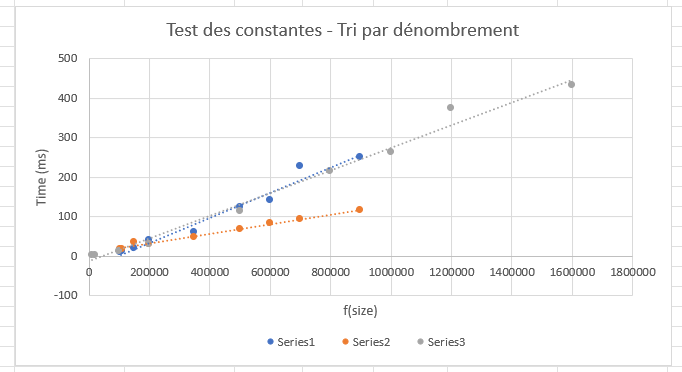
*Figure 20: Test du rapport tri rapide 1 Série 4*

**

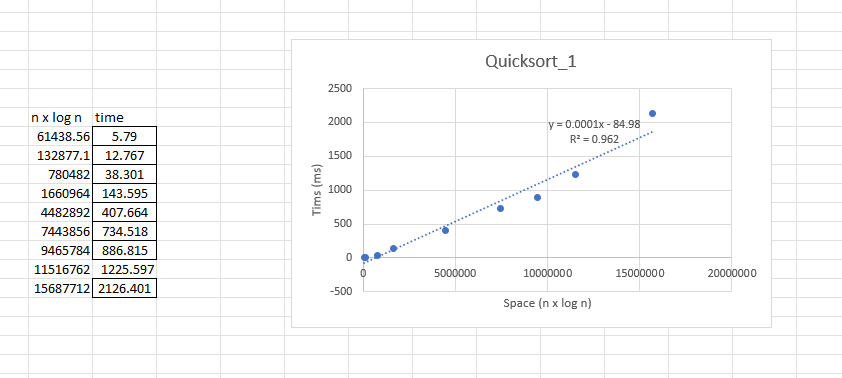
*Figure 21: Test du rapport tri rapide 2 Série 4*

*Figure 22 : Test du rapport tri rapide 3 Série 4*

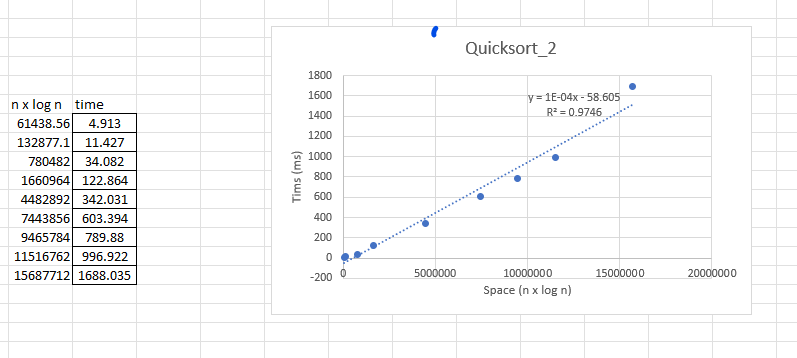
**Tests des constantes**



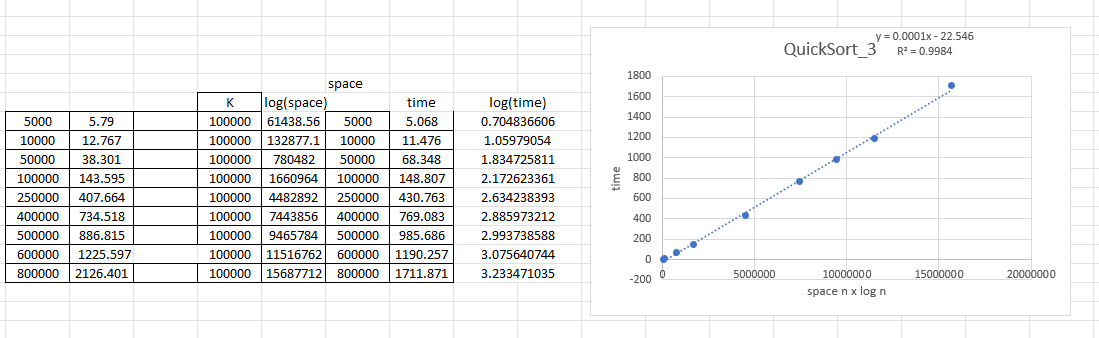
*Figure 23: Test des constantes du tri par dénombrement Série 1, 2 et 3*

**

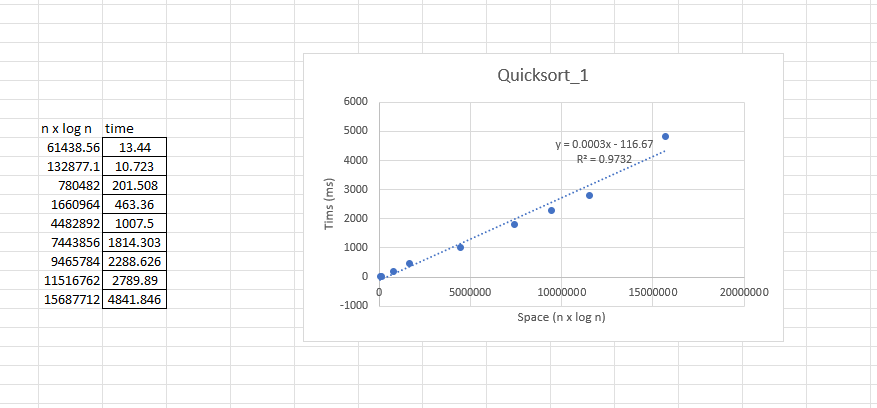
*Figure 24: Test des constantes du tri rapide 1 Série 1*

**

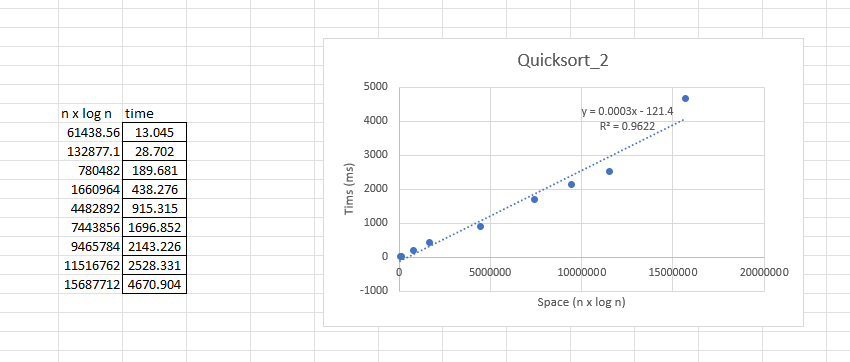
*Figure 25: Test des constantes du tri rapide 2 Série 1*

**

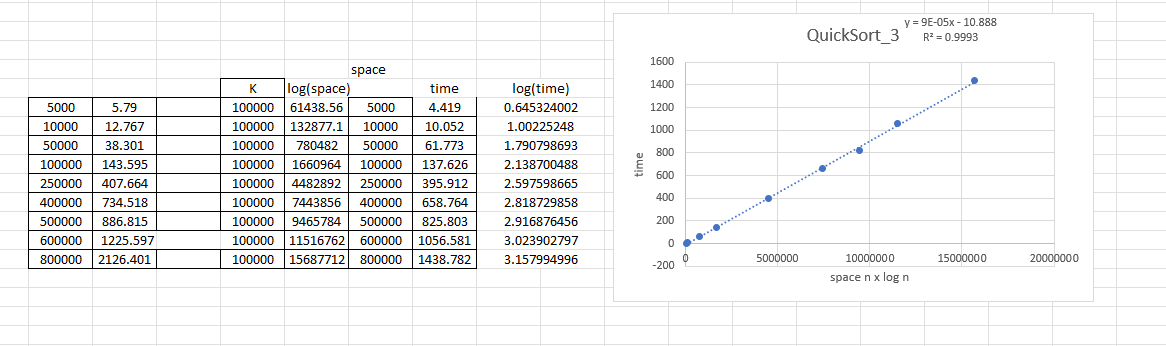
*Figure 26: Test des constantes du tri rapide 3 Série 1*

**

*Figure 27: Test des constantes du tri rapide 1 Série 4*

**

*Figure 28: Test des constantes du tri rapide 2 Série 4*

**

*Figure 29: Test des constantes du tri rapide 3 Série 4*

**Analyse du test de puissance**

Pour le tri par dénombrement de la Série 1 et la Série 2, on aperçoit que la courbe tracée par le test de puissance nous démontre une croissance super-polynomiale par rapport à la régression linéaire. En effet, le R2 de celle-ci linéaire est seulement de 0.945 ce qui signifie peu de corrélation linéaire et on peut y voir une certaine courbe. Le temps et l’espace (la magnitude pour la Série 2) ne sont donc pas linéairement dépendants et la complexité parait être plus grande que O(n).

Pour la Série 3, le R2 de la régression linéaire est de 0.99 et on peut y voir une droite passant près des points du nuage. On peut en déduire que la régression est polynomiale de degré m = 1.26 et que l’espace et le temps sont linéairement dépendant dans ce cas et que la complexité temporelle en est O(n).

Pour le tri rapide, les graphiques sont sensiblement semblables pour la Série 1 et la Série 4 pour les 3 algorithmes de tri rapide différents. Le R2 est autour de 0.99, ce qui semble prouver une régression linéaire polynomiale de degré m d’environ 1.15 et donc une dépendance linéaire entre le temps et l’espace pour les 3 algorithmes. La complexité temporelle semble alors être O(n).

**Consommation théorique asymptotique**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Meilleur cas | Moyen cas | Pire cas |
| Counting Sort | O(n) | O(n + k) | O(n + k) |
| Quicksort | O(n log n) | O(n log n) | O(n2) |
| Quicksort seuil expérimental | O(n log n) | O(n log n) | O(n2) |
| Quicksort pivot aléatoire | O(n log n) | O(n log n) | O(n2) |

**Analyse du test du rapport**

Pour le tri par dénombrement, la Série 1 et la Série 3 produisent des courbes semblables qui semble converger positivement vers une constante b d’environ 0.0003. Nous avons comme axe des ordonnées y/ f(x), y étant le temps et f(x) étant la taille + la magnitude (n + k). Avec cette convergence, nous pouvons en déduire que la complexité de cet algorithme semble être O(n +k).

La Série 2 elle semble converger vers une constante b de 0.00015, mais il est difficile de savoir si elle converge négativement ou positivement. Dans le cas subséquent, on en déduirait la même hypothèse que pour les autres séries.

Pour le tri rapide, les 3 algorithmes autant pour la Série 1 et la Série 4 produisent des courbes similaires. Elles convergent toutes vers une constante positive b. On peut ainsi en déduire du fait que l’axe des ordonnées y/f(x), y étant le temps et f(x) étant la taille log(taille), que la complexité des 3 algorithme de tri rapide est O(n log n).

**Analyse du test des constantes**

Pour le tri par dénombrement dans les Séries 1, 2 et 3, avec comme axes des abscisses f(x) étant la taille + la magnitude (n + k). nous avons obtenu 3 régressions linéaires assez fiables. De ce fait, nous pouvons déduire que la complexité cet algorithme est O(n +k).

Pour ce qui en est des algorithmes de tri rapide, les algorithmes 1 et 2 semblent former des droites qui dont la régression linéaire ne parait pas totalement fiable R2 de 0.96, mais forment tout de même des droites. L’algorithme 3 avec pivot aléatoire lui, possède une régression linéaire totalement fiable avec un R2 de 0.99. Nous pouvons en déduire que la complexité temporelle de ces algorithmes semblent être O de la fonction f(x) des abscisses, donc O(n log n).

**Impact du seuil de récursivité**

Le seuil de récursivité joue un grand rôle dans la performance des algorithmes de tri rapide présentés dans ce laboratoire. En effet, ce seuil représente la taille minimale nécessaire pour un tableau afin que l’algorithme y soit appliqué. Peu importe la taille initiale du tableau sur lequel l’algorithme de tri rapide sera appliqué, le meilleur seuil sera toujours le même. Ceci est expliqué par la complexité asymptotique des algorithmes utilisés. Pour un tri par insertion, elle est de O(n2) et pour le tri rapide elle est de O(n log n). Pour des petites listes, le tri par insertion a une meilleure performance que le tri rapide étant donné que sa complexité algorithmique est plus lente que celle du tri rapide. De plus, ce dernier utilise beaucoup de mémoire puisqu’il divise constamment la liste passée en paramètre. Le tri par insertion ne possède pas ce problème donc sa gestion de mémoire est meilleure. C’est pour cela qu’un seuil de récursivité est utilisé pour passer du tri rapide au tri par insertion lorsque la taille de la liste passée en paramètre atteint une assez petite taille.

**Utilisation des algorithmes**

Avec ces résultats des différents tests, nous sommes venus à certaines conclusions.

Pour le tri par dénombrement, il serait préférable de l’utiliser pour les tableaux possédant une petite magnitude. En effet, comme la complexité temporelle dépend de la taille du tableau additionnée à la magnitude, si le tableau possède une grande taille, en minimisant la taille, le temps d’exécution en devient moindre.

Pour le tri rapide 1, donc le tri avec seuil de récursivité à 1, il est préférable de l’utiliser pour des tableaux de grandes tailles peu importe la magnitude. En effet, l’efficacité de cet l’algorithme avec un seuil de récursivité à 1 devient similaire à celle du tri par insertion lorsque la taille des partitions des tableaux à trier est petite. Une grande taille de tableau à trier est alors préférable.

Pour ce qui en est du tri rapide 2, avec le seuil de récursivité déterminé expérimentalement, il serait préférable de l’utiliser dans des cas où nous savons avec quelles données ou tailles possibles de tableaux et données nous allons travailler avec. En effet, nous trouvons le seuil de récursivité optimal de façon empirique pour un certain « dataset », alors nous évitons de nous retrouver dans une situation où notre seuil est non optimal pour ce « dataset » particulier et qu’il devient plus efficace d’utiliser le tri par insertion qui est O(n2).

Pour finir, l’algorithme de tri rapide, avec un pivot choisi de façon aléatoire, il serait préférable de l’utiliser sur des tableaux de toutes tailles et si on veut diminuer nos chances de rencontrer le pire cas et donc de posséder une complexité moyenne dans la plupart des cas. En effet, avec le choix d’un pivot aléatoire à chaque partition réduit les chances de choisir constamment le pire pivot, on conserve alors une complexité moyenne O(n log n).