TME2 : LU2IN006

PARTIE 1 : Gestion d'une bibliothèque avec une liste chainée de structures

Exercice 1 :

Q1.1) Les prototypes des fonctions de gestion d'un ouvrage, d'une bibliothèque ainsi que les structures `Livre` et `Biblio` sont dans le fichier header `biblioLC.h`

Q1.2) Voir le fichier code `biblioLC.c`

Q1.3) Voir le fichier header `entreeSortieLC.h`

Q1.4) Voir le fichier `main.c`

Q1.5) Voir le fichier Makefile

Q1.6) Voir les fonctions dans le fichier `biblioLC.c`

Q1.7) Voir la fonction dans le fichier `main.c` de prototype :

```
void menu();
```

Q1.8) Voir la fonction `int main(int argc, char** argv) ;` dans le fichier `main.c`

PARTIE 2 : Gestion d'une bibliothèque avec une table de hachage

Exercice 2 :

Q2.2) Voir la fonction dans le fichier `biblioH.c` de prototype :

```
int fonctionClef(char* auteur);
```

Q2.3) Voir les fonctions de prototype :

```
LivreH* creerlivre(int num,char* titre,char* auteur);
void libererlivre(LivreH* l) ;
BiblioH* creerbiblio(int m) ;
void libererbiblio(BiblioH* b) ;
```

dans le fichier code **`biblioH.c`**

Q2.4) Voir la fonction de hachage qui réalise l'opération $\mathrm{h(k)} = \lfloor \mathrm{m(kA} - \lfloor \mathrm{kA} \rfloor) \rfloor$ avec k qui est la clé associée à un livre et $A = \frac{\sqrt{5}-1}{2}$ dont le prototype est

`int fonctionHachage(int cle, int m);` dans le fichier **`biblioH.c`**

Q2.5) Voir la fonction dont le prototype est

`void inserer(BiblioH* b,int num,char* titre,char* auteur) ;` dans le fichier **`biblioH.c`**

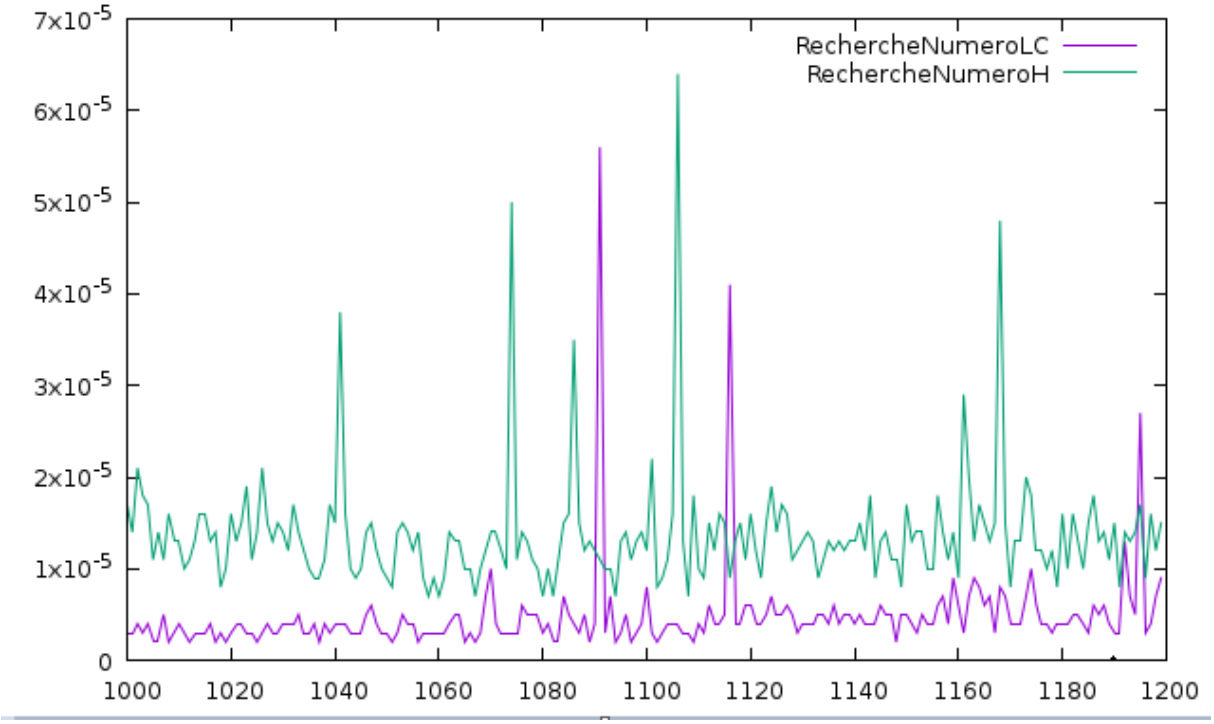
Q2.6) Toutes les fonctions adaptées à une table de hachage se situent dans le fichier **`biblioH.c`**

PARTIE 3 : Comparaison de deux structures

Exercice 3 :

Q3.1) Tous les tests de comparaison de calculs des différents algorithmes sont effectués dans le fichier code `vitesse.c` et exécuté dans le `main.c`

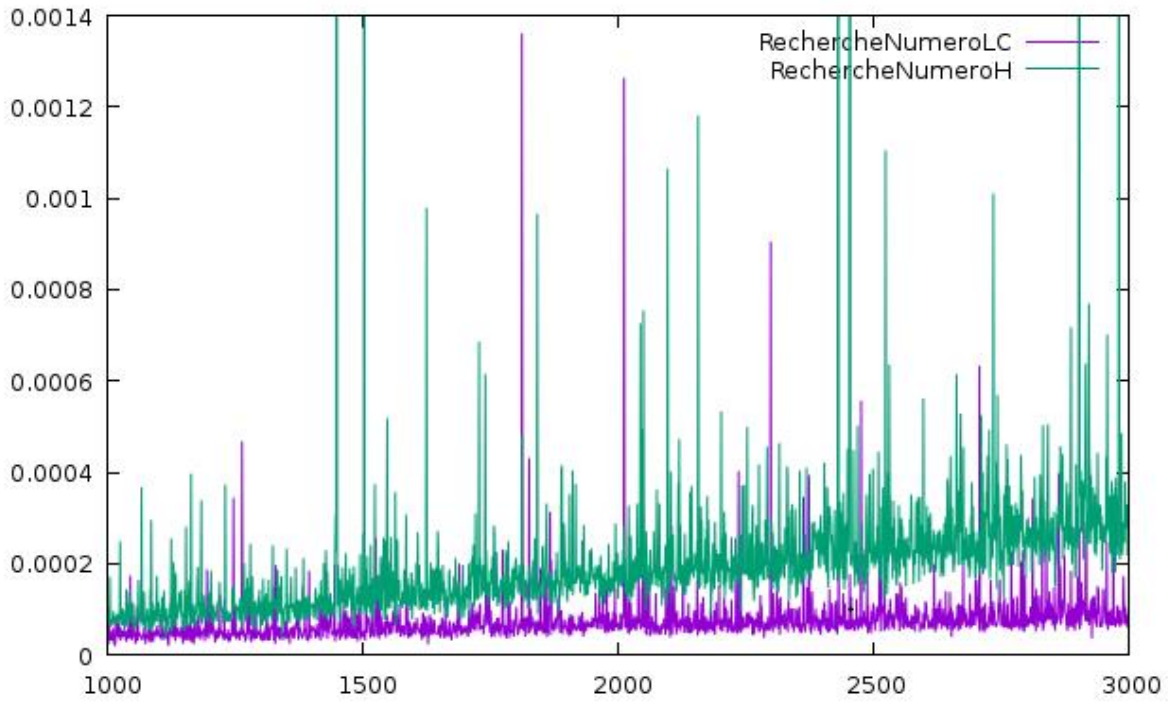
La fonction de recherche d'un ouvrage par son numéro lorsque le livre est présent est beaucoup plus efficace pour une structure de type liste plutôt qu'une table de hachage car la recherche s'effectue de manière linéaire et ne dépend pas d'une clé comme celle de la table de hachage. On constate la différence des temps de calcul à travers le graphe suivant :



- La courbe en violette correspond au temps de calcul de l'algorithme de recherche d'un ouvrage par son numéro sur une structure de liste chainée
- La courbe en verte correspond au temps de calcul de l'algorithme de recherche d'un ouvrage par son numéro sur une table de hachage.

De manière générale, la liste chainée est mieux adaptée pour un algorithme de recherche de ce type car la complexité temporelle pire cas est en $O(n)$ contrairement à celui de la table de hachage qui est en $O(n^2)$.

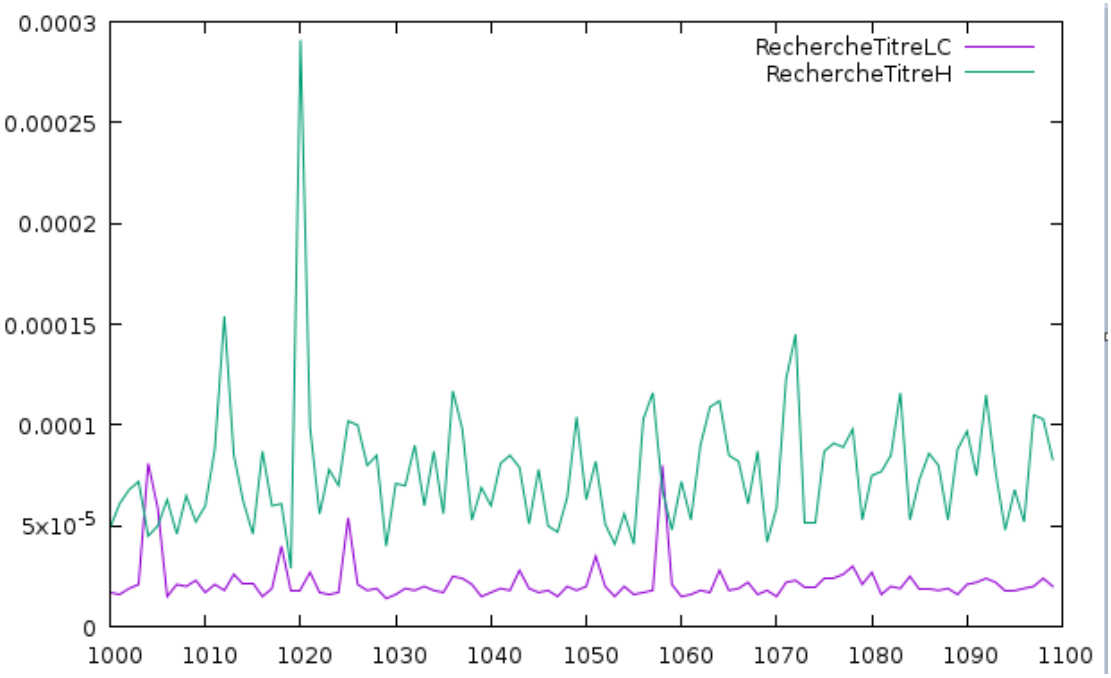
De même lorsque le livre n'est pas présent dans la bibliothèque de recherche, le premier algorithme sur les listes chainées reste plus efficace que la seconde. On le remarque sur le graphe suivant :



- La courbe en violette correspond à l'algorithme de recherche d'un ouvrage lorsque le livre n'est pas présent sur la structure de liste chainée.
- La courbe en verte correspond à l'algorithme de recherche d'un ouvrage lorsque le livre n'est pas présent sur la table de hachage.

On en conclut donc que la recherche d'un ouvrage est plus appropriée pour les listes chainées sur un grand échantillon.

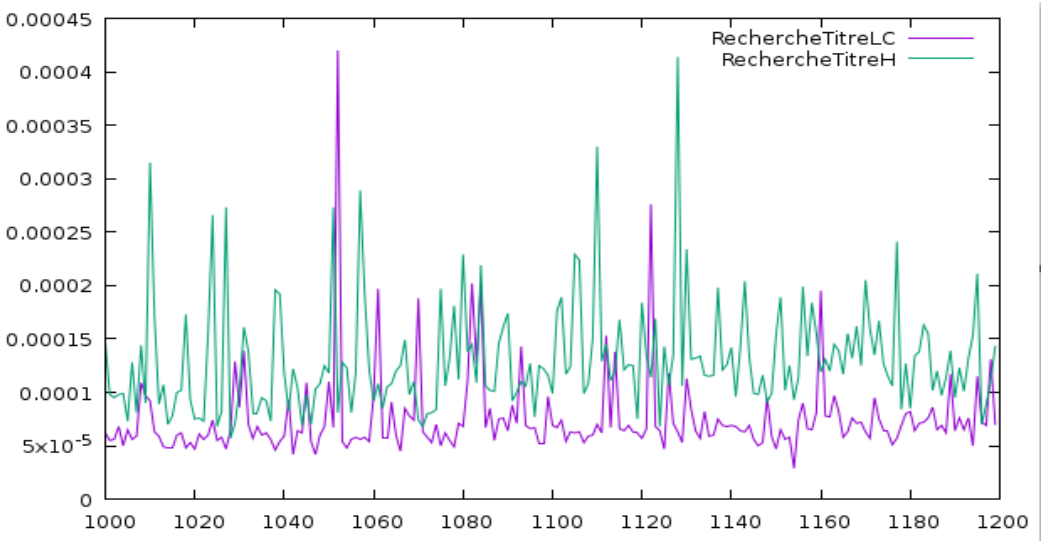
La fonction de recherche d'un ouvrage lorsque le livre est présent par son titre semble plus efficace également sur les listes chainées. En effet la complexité temporelle pire-cas sur les listes chainées est de $O(n)$ tandis que l'algorithme sur une table de hachage possède une complexité temporelle pire cas de $O(n^2)$. L'écart du temps de calcul est visible sur le graphe suivant :



- La courbe en violette correspond à l'algorithme de recherche par titre sur les listes chainées de complexité temporelle pire cas $O(n)$
- La courbe en verte correspond à l'algorithme de recherche par titre sur la table de hachage de complexité temporelle pire cas $O(n^2)$

On constate que l'algorithme de recherche par titre est plus rapide sur les listes chainées.

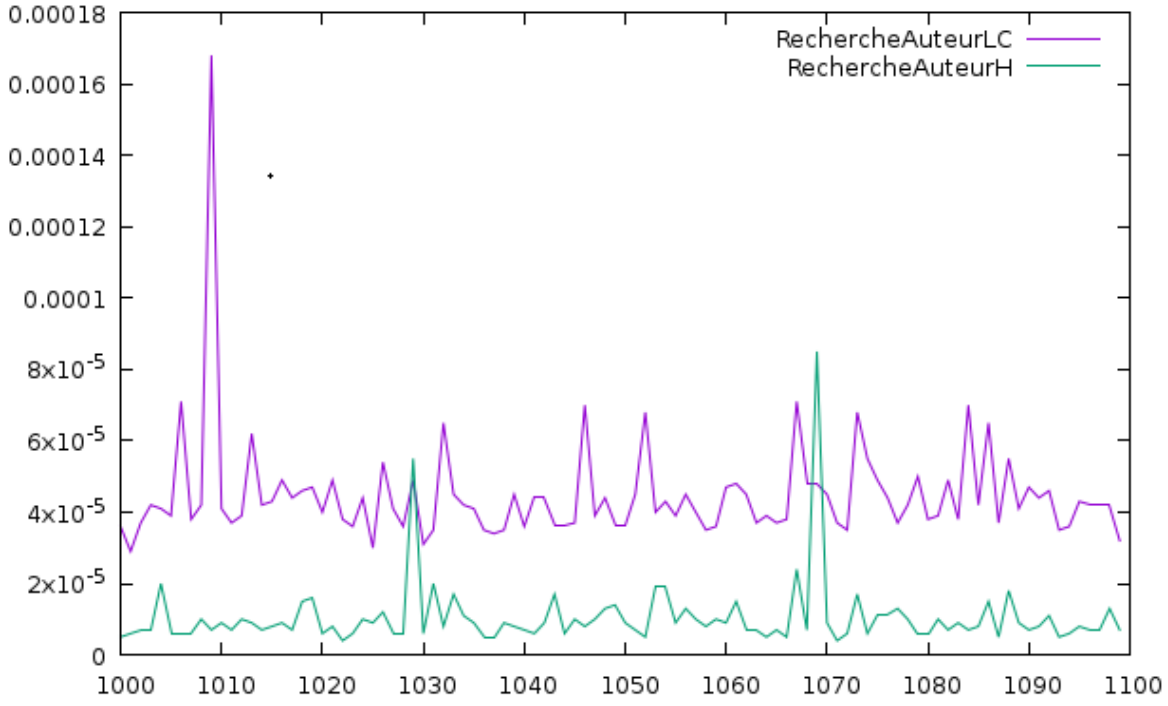
De même lorsque le livre n'est pas présent dans la bibliothèque la vitesse de calcul de l'algorithme sur les listes chainées reste plus appropriée que le second algorithme. On le voit sur le graphe ci-dessous :



- La courbe violette correspond au premier algorithme de recherche par titre sur les listes chainées n'est pas présent
- La courbe en verte correspond au second algorithme de recherche par titre sur les tables de hachage lorsque le livre n'est pas présent

Sur un grand échantillon on constate que l'algorithmes sur les listes chainées restent plus efficace que celui sur les tables de hachages.

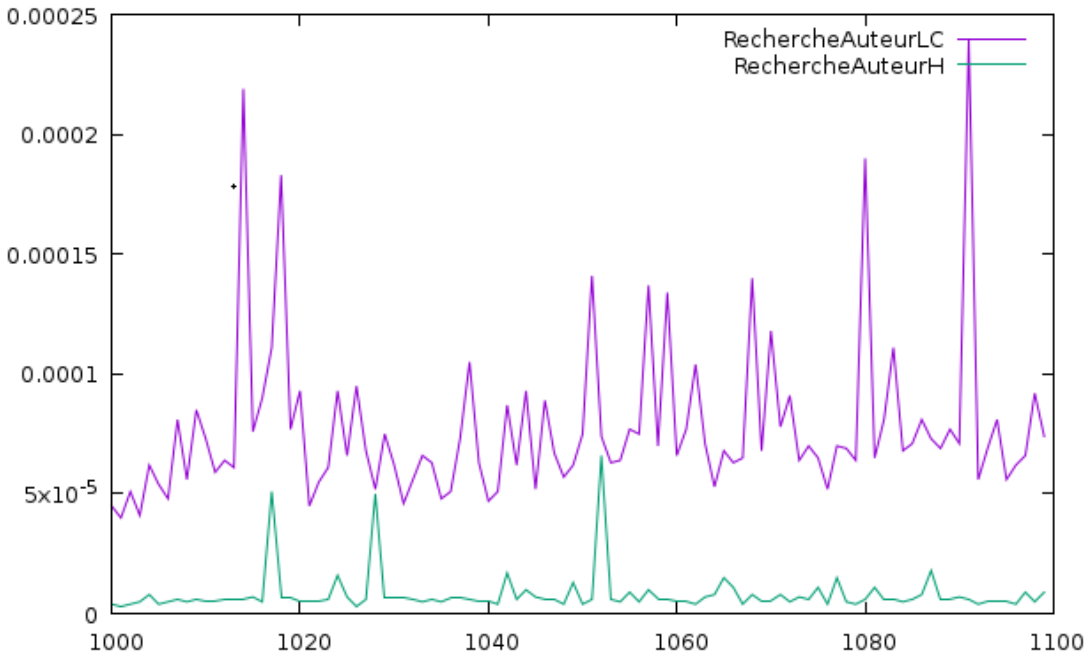
Le temp de calcul des fonctions de recherche de tous les ouvrages présents dans la bibliothèque par un même auteur selon la structure utilisées est différente. En effet cette fois l'algorithme de recherche sur les tables de hachage est plus efficace que celui sur les listes chainées par la recherche possède une complexité pire cas de $O(n)$ tandis que la recherche sur une table de hachage est en $O(1)$. On le constate sur le graphe ci-dessous :



- La courbe violette correspond à l'algorithme de recherche des ouvrages présent dans la bibliothèque par auteur sur une structure de liste chainée
- La courbe en verte correspond à l'algorithme de recherche des ouvrages présent dans la bibliothèque par auteur sur une table de hachage.

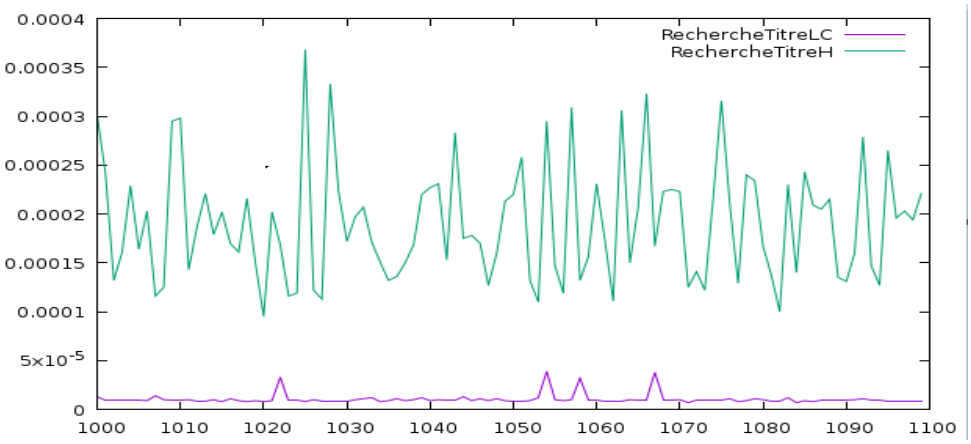
On constate que l'algorithme de recherche par auteur est plus rapide sur les tables de hachages.

De même lorsque le livre n'est pas présent dans la bibliothèque la table de hachage reste plus appropriée que la liste chainée. Voici le graphe de la comparaison :



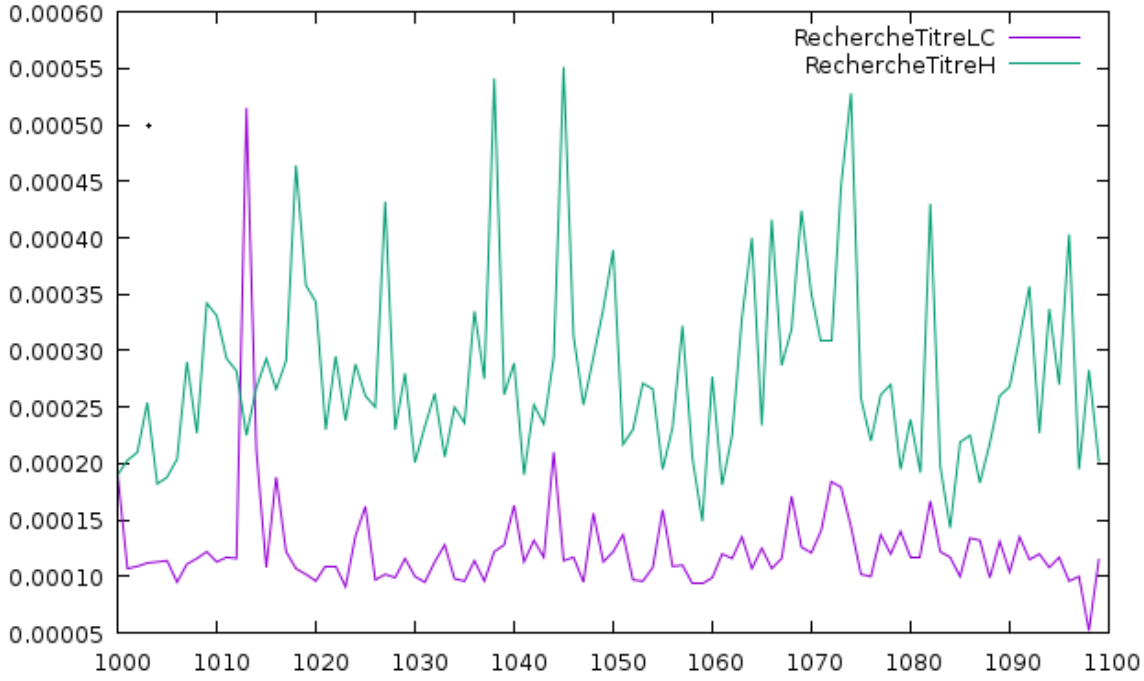
En conclusion, la table de hachage est donc plus adaptée à ce type de recherche car l'auteur d'un livre est utilisé par la fonction de hachage de la structure ce qui permet une recherche plus efficace et plus rapide que l'algorithme précédent.

Q3.2) En modifiant la taille de la table de hachage, on réduit les collisions possibles des clés ce qui tend à rendre plus performant les algorithmes de recherches sur les tables de hachage. Néanmoins dans les fonctions de recherche par titre et par numéro, la structure de table de hachage reste moins performante car on augmente la taille de la table de hachage donc on allonge le parcours de la boucle contrairement à celui qui utilise les listes chainées qui reste sur une courbe plutôt plate.

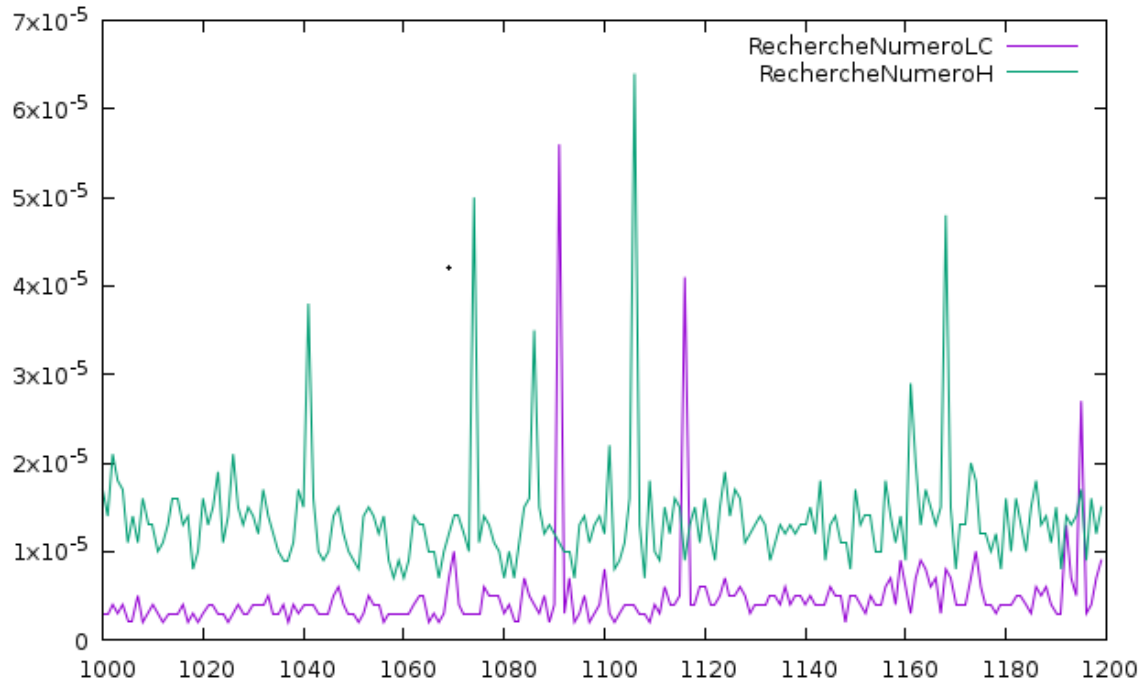


Le graphe ci-dessus correspond au test de comparaison sur l'algorithme de recherche des ouvrages présent par titre en augmentant la taille de la table de hachage. La structure de liste chainée reste plus performante.

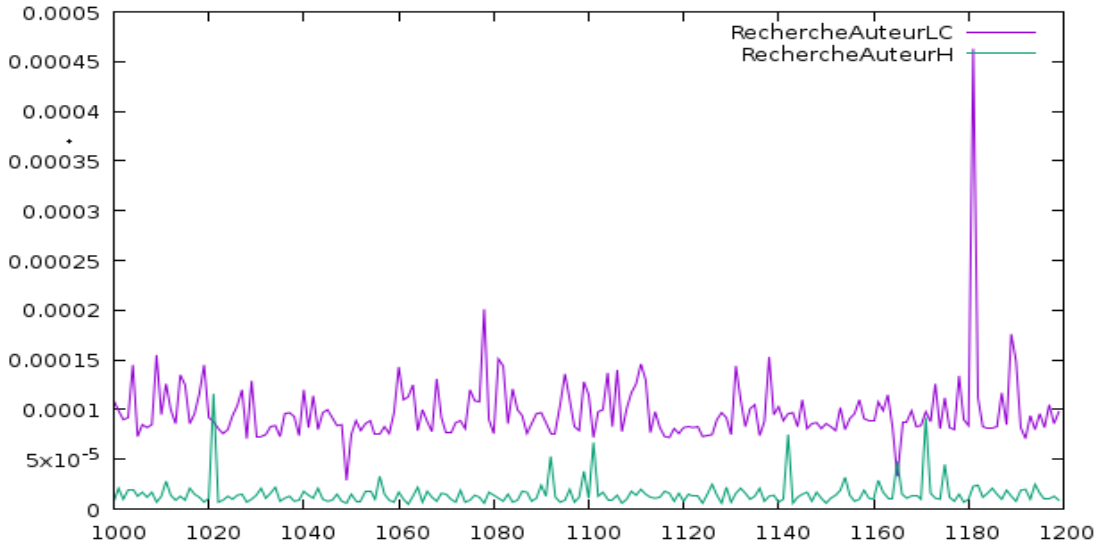
Lorsque le livre n'est pas présent dans la bibliothèque, le résultat global reste similaire à celui précédent, on le remarque dans le graphe ci-dessous :



Le résultat est similaire pour l'algorithme de recherche par numéro, on le remarque à travers le graphe ci-dessous :

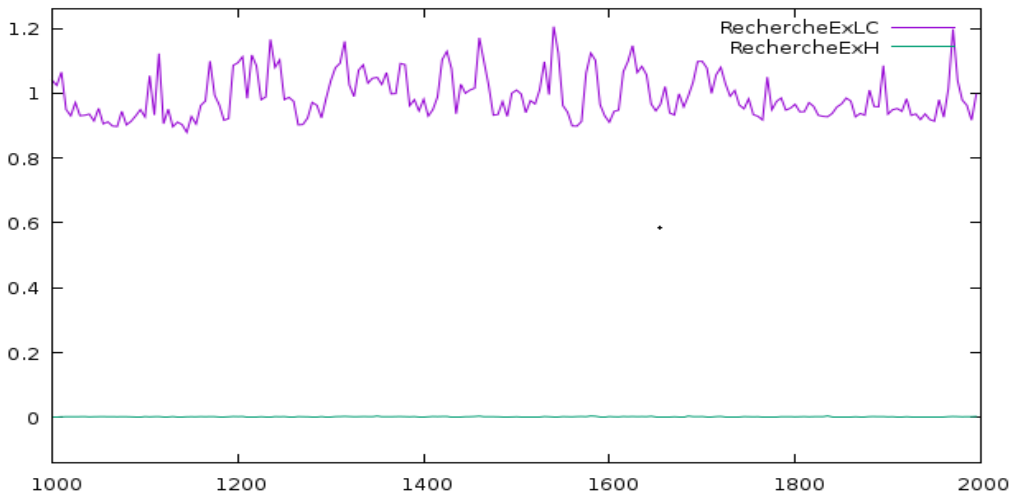


Cependant la fonction de recherche des ouvrages par auteur est plus performante pour la structure de table de hachage en augmentant la taille de celui-ci. En effet, en évitant au maximum les collisions, on réduit le temps de parcours dans l'indice de la case du tableau à parcourir, ceci est remarquable dans le graphe ci-dessous :



En conclusion, plus la taille de la table de hachage est grande, plus on évite les collisions et moins on aura à parcourir le tableau lors de la recherche ce qui permet d'améliorer les performances. La vitesse de calcul dans une structure comme celle-ci est dépendante de la taille de la table de hachage dans lequel on restreint la recherche.

Q3.3) En testant et en comparant les deux fonctions de recherches des livres en plusieurs exemplaires, on constate que l'algorithme sur les tables de hachages reste plus performant que celui sur les listes chainées. Voici le graphe associé à ce test :



- La courbe en violette correspond à l'algorithme de recherche des ouvrages en plusieurs exemplaires sur les listes chainées
- La courbe en verte correspond au second algorithme de recherche des ouvrages en plusieurs exemplaires sur les tables de hachages.

L'algorithme de recherche des ouvrages en plusieurs exemplaires est plus efficace dans la structure de table de hachage que celle sur les listes chainées.

Q3.4) La complexité temporelle pire cas de l'algorithme de recherche de plusieurs exemplaires sur les listes chainées est en $O(n^2)$ tandis que celle sur la table de hachage est en $O(n^3)$, on s'attend à ce que l'algorithme de recherche sur les tables de hachages soit moins performante que celle sur les listes chainées or sur le graphe on voit que la courbe verte est bien en dessous de la violette ce qui montre que la table de hachage est plus approprié pour ce genre de recherche. Ceci s'explique par l'usage d'une clé associé à chaque livre, puisque les auteurs d'un même livre auront une mémé clé associé pour tous les livres du même auteur seront parcouru à la suite dans un même tableau contrairement à la liste chainée qui nécessitera un parcours linéaire jusqu'au derniers élément.