Yara KHAMIS

Iarina NISTOR

Rapport

Reformulation courte du sujet : Dans ce rapport, nous comparons les performances des listes chaînées et des tables de hachage pour la gestion de bibliothèques en langage C. Nous avons implémenté les fonctions de manipulation de bibliothèque d'abord en utilisant les listes chaînées, puis en utilisant les tables de hachage, afin de comparer leurs performances en termes de temps d'exécution.

Description des structures manipulées et de la conception globale du code :

Les structures de données utilisées dans ce projet sont les suivantes :

- **Livre** : Représente un livre avec des champs suivants :

num : Numéro unique du livre.

titre : Titre du livre.

auteur : Nom de l'auteur du livre.

suiv : Pointeur vers le livre suivant

- **Biblio** : Représente une bibliothèque de livres avec un pointeur vers la liste de livres et le nombre total de livres.

- **LivreH** : Représente un livre avec les mêmes champs que Livre, mais inclut également un champ supplémentaire :

clef : Une valeur utilisée pour indexer le livre dans la table de hachage, généralement générée à partir de l'auteur du livre.

- **BiblioH** : Représente une bibliothèque de livres avec les champs suivants :

nE : Nombre total de livres dans la bibliothèque.

m : Taille de la table de hachage utilisée pour stocker les livres.

T : Tableau de pointeurs vers des livres (LivreH), utilisé comme table de hachage pour accéder efficacement aux livres.

Le code est organisé en plusieurs fichiers :

•Liste Chainées :

- **biblio.h** : Déclare les structures Livre et Biblio, ainsi que les fonctions utilisées pour leur manipulation.

- **biblio.c** : Contient l'implémentation des fonctions déclarées dans biblio.h.

- **main.c** : Fichier principal contenant la fonction main() où le programme est exécuté. Il inclut des appels aux fonctions pour démontrer leur utilisation.

- **main_test.c** : Fichier pour tester les différentes fonctions de biblio.c (jeux d'essais).

- **entreeSortieLC.h** : Contient les déclarations des fonctions pour charger et enregistrer des livres.

- **entreeSortieLC.c** : Implémente les fonctions pour charger et enregistrer des livres à partir de fichiers.

•Table de Hachage:

- **biblioH.h** : Déclare les structures Livre et Biblio, ainsi que les fonctions utilisées pour leur manipulation.

- **biblioH.c** : Contient l'implémentation des fonctions déclarées dans biblio.h.

- **main1.c** : Fichier principal contenant la fonction main() où le programme est exécuté. Il inclut des appels aux fonctions pour démontrer leur utilisation.

- **main1_test.c** : Fichier pour tester les différentes fonctions de biblioH.c (jeux d'essais).

- **entreeSortieH.h** : Contient les déclarations des fonctions pour charger et enregistrer des livres.

- e**ntreeSortieH.c** : Implémente les fonctions pour charger et enregistrer des livres à partir de fichiers.

•Comparaison:

- **comparaison.c** : Teste les performances de recherche dans deux structures de données différentes : une liste chaînée et une table de hachage

- test_plot.c : Effectue une analyse comparative des performances entre deux structures de données : une liste chaînée et une table de hachage.

Algorithmes créés :

- Création de livres et de bibliothèques.

- Insertion d'un livre en tête de la bibliothèque.

- Recherche d'un livre par numéro, titre ou auteur.

- Suppression d'un livre de la bibliothèque.

- Fusion de deux bibliothèques en une seule.

- Recherche de doublons dans une bibliothèque.

- Fonction pour charger les n premières entrées à partir d'un fichier dans une bibliothèque.

- Fonction pour enregistrer tous les livres d'une bibliothèque dans un fichier.

Réponses aux questions :

Exercice 3 question 1 :

Pour les bibliothèques de petite taille, la liste chaînée peut être adéquate en raison de sa simplicité et de sa gestion plus simple.

Pour les bibliothèques de grande taille, la table de hachage est généralement préférable en raison de ses temps de recherche plus rapides.

Exercice 3 question 2 :

On s'attend à ce que les temps de recherche diminuent avec une taille de table de hachage plus grande, jusqu'à un certain point où les gains de performance deviennent marginaux. Au-delà de ce point, augmenter la taille de la table de hachage peut même conduire à une augmentation des temps de recherche en raison de la surcharge de gestion des collisions.

Exercice 3 question 4 :

Les courbes obtenues devraient montrer une augmentation continue des temps de calcul pour les deux structures de données à mesure que la taille de la bibliothèque augmente, mais la croissance devrait être plus lente pour la table de hachage par rapport à la liste chaînée, en raison de sa meilleure efficacité.

Jeux d'essais :

Pour valider le code, plusieurs jeux d'essais ont été utilisés, couvrant différentes opérations telles que l'insertion, la recherche et la suppression de livres, ainsi que la fusion de bibliothèques et la recherche de doublons. Les jeux d'essais ont été conçus pour tester les fonctionnalités du programme dans divers scénarios.