Rapport de TP : Développement d'un module de table de hachage en Python

Wolodia ZDETOVETZKY

03/2023

Contents

1 Introduction

L'utilisation de structures de données efficaces est un élément clé dans le développement d'applications performantes en informatique. Parmi les nombreuses structures de données existantes, les tables de hachage sont largement utilisées en raison de leur efficacité en termes de temps d'accès aux données. Dans cette étude, nous nous concentrerons sur le développement d'un module de table de hachage en Python. Les tables de hachage sont une structure de données qui permet de stocker et d'accéder rapidement à des valeurs à partir d'une clé. L'objectif principal des tables de hachage est de fournir une fonction de hachage efficace qui permet de calculer un index de tableau à partir d'une clé. Le processus de hachage consiste à appliquer une fonction de hachage à une clé pour calculer un index, qui est ensuite utilisé pour accéder à la valeur correspondante.

Les tables de hachage sont particulièrement adaptées pour stocker des données de grande taille et sont souvent utilisées dans les bases de données, les systèmes de gestion de fichiers et les applications de recherche. Les tables de hachage ont également l'avantage de pouvoir être utilisées pour résoudre des problèmes de recherche et de filtrage en temps constant.

Cependant, les tables de hachage présentent également certaines limites. Tout d'abord, les collisions peuvent se produire lorsqu'il y a deux clés différentes qui se retrouvent avec le même index. Cela peut ralentir les performances de la table de hachage. De plus, les tables de hachage ont une taille fixe, ce qui signifie qu'elles ne peuvent pas être agrandies une fois qu'elles sont créées.

Dans ce rapport, nous présentons le développement d'un module de table de hachage en Python qui offre un type abstrait Table. Le module se base sur une implémentation de table de hachage avec des collisions résolues par adressage fermé puis ouvert. La taille de la table est fixe et l'utilisateur peut choisir entre plusieurs fonctions de hachage et méthodes de rehachage. Le module offre également des méthodes d'insertion, de suppression, de recherche, d'union et d'intersection de tables ainsi qu'une méthode d'affichage.

Nous présentons aussi une analyse des performances de notre module de table de hachage en fonction de la taille de la table, du nombre d'insertions et de recherche d'une clé. Nous comparons les performances de notre module avec d'autres structures de données et implémentations.

Dans les sections suivantes, nous détaillons l'implémentation de notre module de table de hachage en Python, les tests de performances effectués ainsi que l'analyse de la complexité algorithmique de notre implémentation.

2 Présentation du sujet

Le sujet de ce rapport est le développement d'un module de table de hachage en Python. L'objectif est de créer un type abstrait Table en utilisant une implémentation de table de hachage, où la clé sera un entier et la taille de la table sera fixe.

Le module devra offrir plusieurs méthodes pour la manipulation des tables de hachage :

- `init(taille_table, fonction_hachage, type_rehachage)` - cette méthode initialisera la table de hachage en spécifiant la taille de la table, la fonction de hachage à utiliser et le type de rehachage (linéaire, quadratique ou double hachage).
- `insert(key, value)` - cette méthode insérera une paire clé-valeur dans la table.
- Si la clé existe déjà dans la table, la valeur associée sera mise à jour.
- `delete(key)` - cette méthode supprimera la paire clé-valeur associée à la clé spécifiée de la table.
- `exist(key)` - cette méthode renverra True si la clé spécifiée existe dans la table, False sinon.
- `value(key)` - cette méthode renverra la valeur associée à la clé spécifiée.
- `union(autre_table)` - cette méthode renverra une table résultant de l'union de la table actuelle avec une autre table spécifiée en argument.
- `intersection(autre_table)` - cette méthode renverra une table résultant de l'intersection de la table actuelle avec une autre table spécifiée en argument.
- `affichage()` - cette méthode affichera les éléments insérés dans la table ainsi que le nombre de rehachage nécessaire pour chaque insertion.

Dans un premier temps, la taille de la table sera fixe et les collisions seront gérées par adressage fermé, puis par adressage ouvert si nécessaire. La fonction de hachage sera donnée par l'utilisateur et le type de rehachage sera linéaire, quadratique ou double hachage.

Le module de table de hachage sera testé pour mesurer ses performances en termes de temps de recherche d'une clé en fonction du nombre d'éléments dans la table. Les tests seront effectués pour le rehachage linéaire et les résultats obtenus seront comparés à d'autres structures de données et implémentations existantes.