Hash Map

Diedler Baptiste

November 3, 2023

1 Introduction

Une table de hachage est une structure de données qui associe des clés à des valeurs.
Elle peut être représenté de différentes façons:
Sous forme de collection contenant des couples clé valeur.
Sous forme de tableau indexé contenant une collection de valeurs.
...
A chaque table de hashage est associé une fonction de hashage qui à une valer donnée, va retourner la clé de celle-ci dans la collection.
Un tableau peut être concidéré comme une table de hashage avec des clés qui vont de 0 à N-1 avec N la taille du tableau. Si celui-ci est associé à une fonction de hashage.
Par la suite nous verrons que les talbes de hachage ont un gros défaut qui est le conflit.

2 Fonction de hashage

Le principe de la fonction de hashage est de passer une valeur vers une clé. Le but est d'avoir une clé différente pour chaque valeur du domaine.
C'est un fonction qui passe d'un élément du domaine (les valeurs) vers un élements du co-domaine (les clés)
Cette fonction est nommée Φ:

$$\begin{aligned} \Phi : \text{Domaine} &\to \text{CoDomaine} \\ n &\mapsto [0, N-1] \end{aligned}$$

$$\lfloor N \cdot \text{Dec}(n \cdot r) \rfloor \quad \text{avec } r = \frac{1+\sqrt{5}}{2}$$

Or si le nombre d'éléments dans le domaine est supérieur au nombre d'éléments dans le co-domaine alors la fonction ne peut pas être bijective.
Pour pallier à ce problème, nous allons nous penchés sur la structure de données pouvant contenir de tels informations.

3 Conflit

Le conflit est un des problèmes dans les tables de hashages. Celui-ci arrive quand au moins deux valeurs donnent ont la même clé par la fonction de hashage.
Pour pouvoir contrer cela, nous utliserons les spécifiacations des structures de données car nous nous rendons compte que la taille du co-domaine de la fonction de hashage ne peut pas dépasser la taille de la structure de données.
EN cas de conflit, il faut essayer de positionner la nouvelle valeur le plus loin possible de la clé qui lui ai attribué afin d'utiliser un maximum d'espace de la strucutre de données.

4 Structures de données

4.1 Tableau

Le tableau est une structure de donnée pouvant représenter une table de hashage. En utilisant la fonction de hashage comme ci-dessus, on se rend compte que le tableau permet de donner une structure pratique. Car le temps de recherche dans le tableau est en temps constant. Mais le problème est que en cas de conflit nous somme obligé d'attribuer une autre case libre à la valeur entrante dans le tableau.
Or celà veut dire que avec une clé une seule valeur peut être atteinte et que le talbeau a un nombre limité de valeur à sa taille.
Il en est de même pour le tableau ou la liste contenant le couple d'éléments clé valeur. Sauf que en plus la recherche est en temps linéaire et non constant. De plus nous ne pouvont pas agrandir la taille du tableau car cala serait contraignant en terme d'espace mémoire et en temps.

4.2 Tableau de liste

Afin de pallier à ce conflit, nous pouvont utiliser des listes pour stocker les valeurs de chaque clé. Cela veut dire que les cases du tableau vont pointer vers la première valeur qui leur est attribuée. Et que ces valeurs pointeront elles mêmes vers la valeur suivante.
Or, cela a ses limites. On se rend compte que si les listes sont plus grandes que deux, nous allons perdre du temps dans la recherche des valeurs. Et que donc il faudra bien au bou d'un moment agrandir la taille du tableau contenant les listes.

5 Code

Algorithm 1 Insertion d'un élément dans un tableau T

Require: n : entier
  $p \leftarrow \Phi(n)$
  if $T[p]$ est occupé **then**
    $d \leftarrow \ln_2(n)$
    while $T[p]$ est occupé **do**
      $p \leftarrow p + d$
    end while
  end if
  $T[p] \leftarrow n$