Algorithmique
Hachage : Résolutions des collisions par chaînage

Rappels

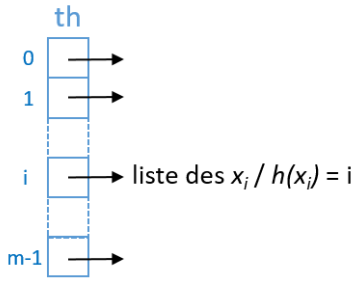
Pour toute clé x d'une collection d'éléments, la fonction de hachage h est une fonction uniforme telle que $h(x)$ (valeur de hachage primaire) donne l'indice, compris entre 0 et $m-1$, de x dans le tableau de hachage.

Si plusieurs clés ont la même valeur de hachage, on parle alors de *collision* (primaire).

Résolution des collisions par chaînage

Parmi les méthodes de résolution des collisions, il existe les méthodes dites **indirectes** : par chaînage. Les éléments en collision sont chaînés entre eux : soit à l'extérieur du tableau, le hachage avec chaînage séparé ; soit dans une zone de débordement du tableau de hachage, le hachage coalescent.

1 Chaînage séparé



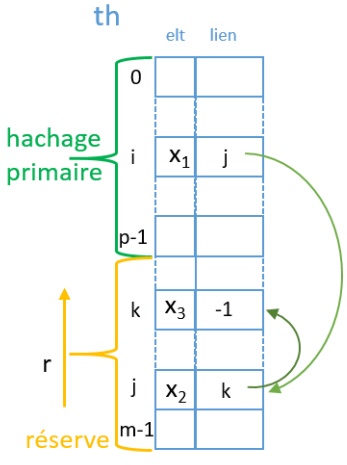
La méthode par chaînage séparé construit pour chaque valeur de hachage une liste des clés : chaque case du tableau de hachage contient la liste des éléments dont la valeur de hachage primaire correspond à l'indice de la case du tableau.

Recherche :
Une fois la valeur de hachage de la clé trouvée, il suffit d'effectuer une recherche (séquentielle) dans la liste correspondante.

Insertion :
Deux méthodes d'insertion, selon que l'on autorise la redondance.
- Sans redondance : l'élément n'est inséré que si la recherche est négative. Le coût de la recherche est compensé par le maintien de listes plus courtes.
- L'élément est directement inséré dans la liste correspondant à sa valeur de hachage. Le coût de la recherche a été économisé, mais cela peut créer des listes plus longues.

Suppression :
Également deux possibilités :
- La recherche s'arrête sur la première occurrence de l'élément qui est supprimée.
- La recherche se poursuit jusqu'à la fin de la liste pour supprimer toutes les occurrences.

2 Hachage coalescent



Avec réserve

Le tableau de hachage de taille $m = p + r$ contient deux zones :
- une zone de **hachage primaire** de p éléments ($h(x) \in [0, p-1]$),
- et une **zone de réserve** de r éléments permettant de gérer les collisions.

Les éléments en collisions sont chaînés à l'intérieur du tableau : chaque case contient en plus l'indice (lien de collision) de l'élément suivant (élément en collision), qui est dans la zone de réserve (utilisée en partant du bas).
Le problème est le suivant :
- Si la réserve est trop petite, elle se remplit trop vite et l'utilisation du tableau est incomplète.
- Si la réserve est trop grande, l'effet de dispersion est perdu : on se trouve avec des "listes" d'éléments en collisions de longueurs approchant n (nombre d'éléments).

Sans réserve

Une solution est de ne pas considérer de zone de réserve, d'utiliser m en valeur maximale de hachage primaire et de gérer les collisions comme s'il existait une zone de réserve (en partant de l'indice maximum et en remontant). C'est cette solution qui est retenue pour les méthodes décrites ci-dessous.

L'inconvénient : cela crée des **collisions secondaires**, collisions qui ne sont pas dues à la coïncidence des valeurs de hachage primaire. Lorsqu'une place i est utilisée comme réserve par un élément y tel que $h(y) \neq i$: il y aura collision avec le prochain x tel que $h(x) = i$. Dans ce cas, les listes de collisions de x et de y auront des éléments en commun.

Opérations [1]

Recherche

La recherche est similaire à celle avec chaînage séparé : Une fois la valeur de hachage de la clé trouvée, il suffit d'effectuer une recherche séquentielle dans la "liste" correspondante, en suivant les liens jusqu'à trouver l'élément ou arriver en fin de "liste".

Insertion

L'insertion de la valeur x commence par le calcul de sa valeur de hachage.
- Si la case correspondante est vide, alors l'élément y est inséré.
- Sinon on effectue une recherche de x en parcourant la liste de collisions, mais en gardant le dernier élément :
  - si x est présent, l'insertion ne se fait pas
  - sinon on remonte de la fin du tableau jusqu'à trouver une place vide où insérer x, cet élément est relié au dernier élément de la liste des collisions (dont il devient le "suivant").

Suppression

La suppression pose un problème à cause des collisions secondaires : on risque en supprimant du "chaînage" un élément de séparer des listes de collisions fusionnées.

La solution est donc de ne pas supprimer réellement un élément, mais de marquer la case qu'il occupe comme *libre* : chaque case pourra alors être soit *vide* (état initial), *occupée* ou *libre* (a été occupée puis libérée).

Dès lors, il faut adapter les méthodes vues ci-dessus [2].

Pour la **recherche**, il suffit juste de ne pas tenir compte des cases *libres* (on continue la recherche).

Lors du parcours de la liste de collisions dans l'**insertion**, on conserve l'éventuelle première place *libre* rencontrée :
- Si une telle place existe, alors on y insérera tout simplement l'élément.
- Sinon, on procède à la recherche d'une place en réserve comme pour l'insertion "classique".

1. Les algorithmes sont donnés en annexes
2. Dans les algorithmes en annexe, les différences sont en bleu.