STRUCTURES DE DONNÉES ET ALGORITHMES · CHAPITRE 7

Retrouver une valeur à partir d'une clé

Définissez clé, valeur, hachage et collision. Construisez un comptage par dictionnaire et distinguez garanties moyennes et pire cas.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Relier clés, hachage, collisions et recherche associative.
- Une clé désigne une entrée associative ; elle ne désigne pas nécessairement une position de tableau.
- Une collision se résout en distinguant les clés, sans perdre leur valeur.

Prérequis. Avoir étudié les chapitres DSA précédents ; connaître collections, parcours et coûts asymptotiques.

Partir d'un besoin

Pour compter les occurrences de chaque code produit, parcourir toute une liste de codes déjà vus à chaque ajout répète du travail. Une table associative permet de demander directement le compteur associé à un code.

Définir la structure et les conventions

Une **table associative** relie une clé unique à une valeur. Une **table de hachage** réalise ce contrat en calculant un entier à partir de la clé, puis un emplacement à partir de cet entier. Une **collision** survient lorsque deux clés visent le même emplacement ; elle ne doit pas confondre les clés.

Une résolution par **chaînage** conserve plusieurs entrées dans un même compartiment ; une résolution par **adressage ouvert** cherche une autre case suivant une règle. Dans les deux cas, l'égalité des clés départage les collisions. Le **facteur de charge** mesure ici le nombre d'entrées rapporté au nombre de compartiments ou de cases.

Python utilise un dictionnaire, écrit avec des accolades : counts[code] désigne la valeur associée à code. `get(code, 0)` donne 0 si la clé manque. MATLAB utilise `containers.Map`, disponible aussi dans les versions antérieures à dictionary ; isKey teste la présence avant une lecture. Le type des clés est char (texte), celui des valeurs double (nombres). Les accolades de codes en MATLAB créent une collection de textes ; codes{index} en extrait un texte.

Suivre les états

Comptage par clé

Code reçu	Compteur de A	Compteur de B
A	1	Absent
B	1	1
A	2	1

Comparer les réalisations

Les compteurs finaux valent 2 et 1. Une clé n'est pas une position numérique. Modifier la valeur de A ne change pas celle de B. Aucun ordre de parcours des clés n'est nécessaire au résultat.

Pseudo-code

```
// example.pseudo
// count_codes
// A missing key has count 0.
// For each code, replace its count by count + 1.
// Input: A, B, A. Output: A -> 2; B -> 1.
```

Python

```python
# count_codes.py
codes = ["A", "B", "A"]
counts = {}
for code in codes:
    counts[code] = counts.get(code, 0) + 1
print(counts["A"], counts["B"])
```

MATLAB

```matlab
% count_codes.m
codes = {'A', 'B', 'A'};
counts = containers.Map('KeyType', 'char', 'ValueType', 'double');
for index = 1:numel(codes)
    code = codes{index};
    if isKey(counts, code)
        counts(code) = counts(code) + 1;
    else
        counts(code) = 1;
    end
end
disp([counts('A'), counts('B')]);
```

Justifier et compter

Sous des hypothèses de hachage bien réparti, charge maîtrisée et clés de taille bornée, une recherche ou mise à jour est O(1) en moyenne ; le pire cas théorique peut être O(n). Le comptage Python est alors O(n) en temps attendu et O(k) en mémoire pour k clés distinctes. Ce modèle de table de hachage ne constitue pas une garantie de complexité contractuelle pour containers.Map. Calculer le hachage ou comparer un texte de longueur variable a aussi un coût.

Prévoir puis expérimenter

Remplacez les codes par A, A, A, puis par A, B, C. Le nombre total d'occurrences vaut toujours la longueur d'entrée, mais le nombre de clés stockées change.

Exercice corrigé

▼ Pourquoi ne peut-on pas utiliser le hachage comme preuve que deux clés sont égales ?

Deux clés différentes peuvent avoir le même hachage. La structure doit vérifier aussi leur égalité et résoudre les collisions sans perdre d'entrée.

À retenir

- Une clé désigne une entrée associative ; elle ne désigne pas nécessairement une position de tableau.
- Une collision se résout en distinguant les clés, sans perdre leur valeur.
- Le temps constant moyen suppose un hachage et une charge adaptés ; ce n'est pas une borne constante universelle.

Version web : https://www.clearcode.fr/courses/programming/dsa/associative/hash-tables