Хеширование строк и других объектов, хеш–таблицы

Горденко М.К.

Хеширование

Хеширование — разбиение множества ключей (однозначно характеризующих элементы хранения и представленных, как правило, в виде текстовых строк или чисел) на непересекающиеся подмножества (наборы элементов), обладающие определенным свойством.

Идея хеширования состоит в использовании некоторой частичной информации, полученной из ключа, в качестве основы поиска, те вычисляется хеш-адрес h(K), который используется для поиска

Хеш-функция

Хеш-функция — это какая-то функция, сопоставляющая объектам какого-то множества числовые значения из ограниченного промежутка.

«Хорошая» хэш-функция:

- Быстро считается — за линейное от размера объекта время
- Имеет не очень большие значения (обычно влезающие в 64 бита)
- «Детерминированно-случайная» — если хеш может принимать n различных значений, то вероятность того, что хэши от двух случайных объектов совпадут, равна примерно 1/n.

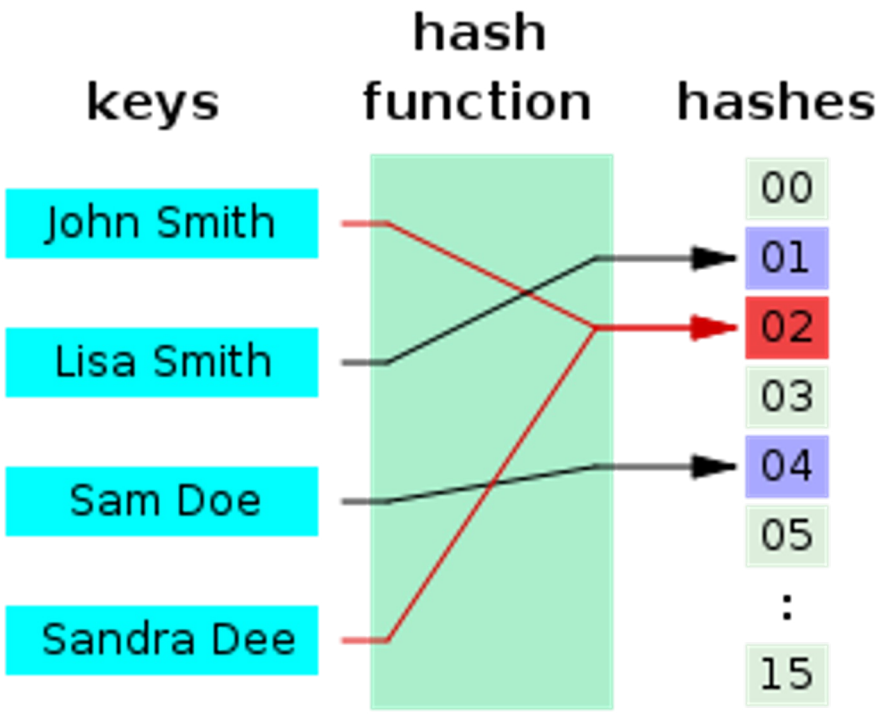
Хеш-функция

Хеш-функция должна сопоставлять некоторому объекту некоторое число (его хеш) и обладать следующими **свойствами**:

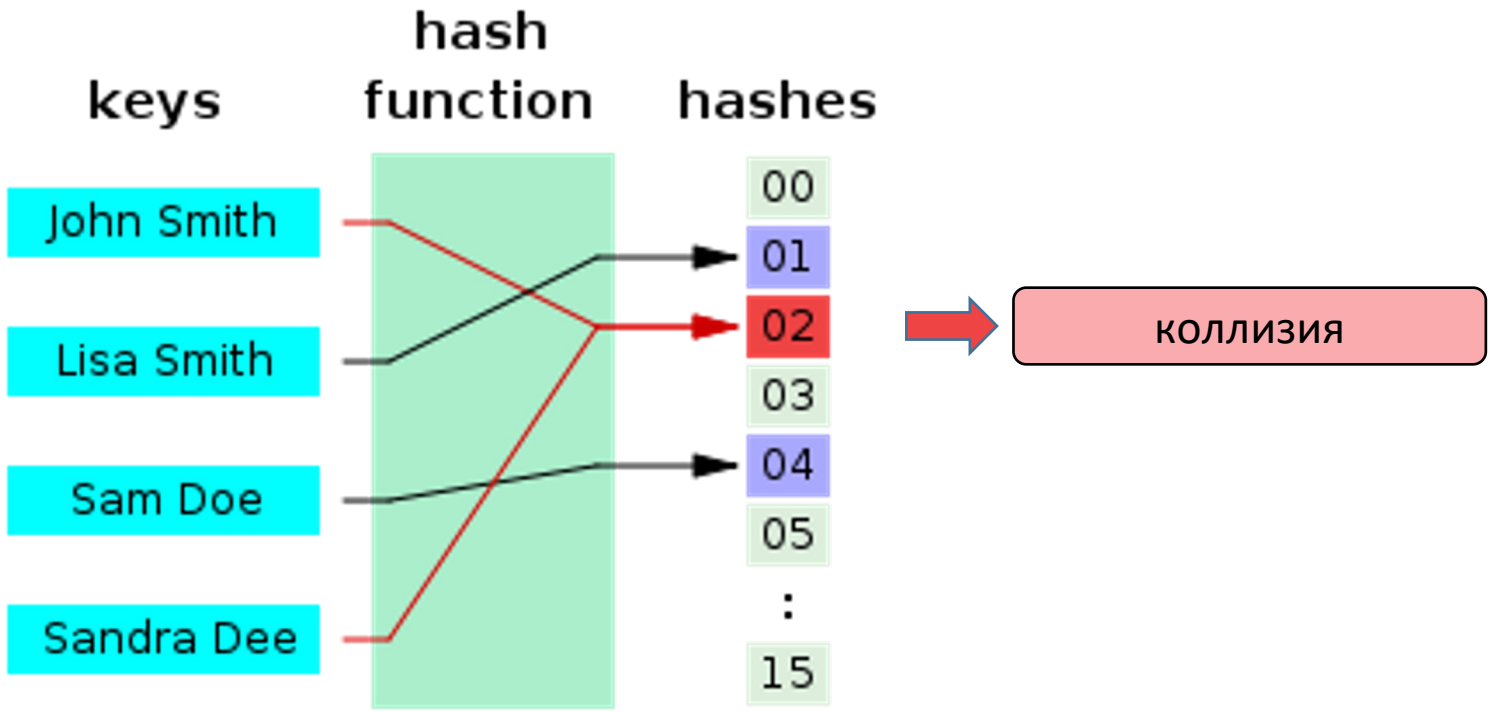
- Если два объекта совпадают, то их хеши равны.
- Если два хэша равны, то объекты совпадают с большой вероятностью.

Коллизией называется ситуация равенства двух хешей у несовпадающих объектов.

Схематичная работа хеш-функции



Схематичная работа хеш-функции



Хеш-таблица

Хеш-таблица - динамическая структура данных, реализующая интерфейс ассоциативного массива.

Она позволяет хранить пары типа <Ключ, Значение> и выполнять 3 операции:

- добавление новой пары
- поиск по ключу
- удаление по ключу

Хеш-таблица

Пусть есть хеш-таблица с хеш-функцией
h(S) = len(S) % 4.

ключ	значение

0	
1	
2	
3	

Хеш-таблица: добавление новой пары

Добавим значение ‘хеш’.

h(‘хеш’) = 3 % 4 = 3

ключ	значение

0	
1	
2	
3	хеш

Хеш-таблица: добавление новой пары

Добавим значение ‘утята’.

h(‘утята’) = 5 % 4 = 1

ключ	значение

0	
1	утята
2	
3	хеш

Хеш-таблица: добавление новой пары

Добавим значение ‘котёнок’.

h(‘котёнок’) = 7 % 4 = 3

Коллизия!

ключ	значение

0	
1	утята
2	
3	хеш

Хеш-таблица: работа с коллизиями. Примитивный способ: замещение значения

Значение в месте коллизии затирается и записывается новое

ключ	значение

0	
1	утята
2	
3	котёнок

Хеш-таблица: борьба с коллизиями. Метод цепочек

Метод цепочек (открытое хеширование) — элементы с одинаковым хешем попадают в одну ячейку в виде связного списка.

Хеш-таблица: борьба с коллизиями. Открытая индексация

Открытая индексация (закрытое хеширование) – если ключ уже занят, то ищется другое свободное место в таблице. Существует несколько стратегий поиска места:

- **Последовательный** поиск. При попытке добавить элемент в занятую ячейку i начинаем последовательно просматривать ячейки i+1, i+2, i+3 и так далее, пока не найдём свободную ячейку.
- **Линейный** поиск. Выбираем шаг q. При попытке добавить элемент в занятую ячейку i начинаем последовательно просматривать ячейки i+(1·q), i+(2·q), i+(3·q) и так далее. По сути последовательный поиск - частный случай линейного, где q=1.
- **Квадратичный** поиск. Шаг q не фиксирован, а изменяется квадратично: q=1, 4, 9, 16... Соответственно при попытке добавить элемент в занятую ячейку i начинаем последовательно просматривать ячейки i+1, i+4, i+9.

Открытая индексация: последовательный поиск

i = h(S) = 3

i+1 = 0 (нумерация ячеек от 0 до 3) – свободная ячейка, вставляем значение сюда.

ключ	значение
0	котёнок
1	утята
2	
3	хеш

Открытая индексация: линейный поиск

i = h(S) = 3

Пусть q = 3. Тогда

i+q = (3 + 3) % 4 = 2 – свободная ячейка, вставляем значение сюда.

ключ	значение

0	
1	утята
2	котёнок
3	хеш

Открытая индексация: квадратичный поиск

i = h(S) = 3 - занято

q = 1. i+q^2 = (3 + 1) % 4 = 0 – свободная ячейка, вставляем значение сюда.

ключ	значение

0	котёнок
1	утята
2	
3	хеш

Открытая индексация: замечание

В отличие от хеширования с цепочками, при использовании этого метода может возникнуть ситуация, когда хеш-таблица окажется полностью заполненной, следовательно, будет невозможно добавлять в неё новые элементы. Так что при возникновении такой ситуации решением может быть динамическое увеличение размера хеш-таблицы, с одновременной её перестройкой.

Также может возникнуть ситуация когда шаг нашего поиска места зацикливается – свободные места ещё есть, однако они не находятся нашей стратегией. Решением проблемы может быть использование разных стратегий поиска или нескольких хеш-функций.

Хеш-таблица: поиск по ключу

При использовании методов простого замещения значений или метода цепочек поиск происходит просто по ключу.

Для открытой адресации поиск осуществляется аналогично добавлению: мы проверяем ячейку i и другие согласно выбранной стратегии, пока не найдём искомый элемент или свободную ячейку.

При поиске элемента может получится так, что мы дойдём до конца таблицы. Обычно поиск продолжается, пока мы не придём в ту ячейку, откуда начинался поиск.

Хеш-таблица: удаление по ключу

Удаление происходит за счёт работы поиска. Найти элементы и удалить их значения.

Хеш-таблица: оценка операций

Для способов реализации хеш-таблицы:

- С замещением значений – O(1)
- Метод цепочек – O(1)
- Открытая индексация – O(n)

Парадокс дней рождений

В группе, состоящей из 23 или более
человек, вероятность совпадения дней рождения (число и месяц)
хотя бы у двух людей превышает 50 %. Например, если
в классе 23 ученика или более, то более вероятно то, что у какой-
то пары одноклассников дни рождения придутся на один день,
чем то, что у каждого будет свой неповторимый день рождения.

Лемма

Вероятность коллизий при вставке в хеш-таблицу превышает 50%