Хеширование

Использование ключей в качестве индекса в массиве обеспечивает высокую производительность операций поиска, добавления, удаления элементов, при этом предполагается, что ключи соответствуют диапазону индексов. Однако такими благоприятными свойствами обладают далеко не все ключи.

Хеширование – это способ хранения данных, при котором на основании значений ключей элементов вычисляется значения хеш-функции – индексы, которые используются для поиска, добавления, удаления элементов из хеш-таблиц (специализированных контейнеров, основанных на массивах или других аналогичных структурах).

Хеширование – преобразование ключа записей (данных) в индекс (адрес) их размещения в соответствующей структуре данных (хеш-таблице). Основан на использовании хеш-функций, т.е. функций осуществляющих преобразование. Основной задачей хеш-функций (и соответственно алгоритмов, лежащих в их основе), является получение из входных значений ключей, в общем случае имеющих произвольное распределение и диапазон значений, равномерно распределенных в заданном диапазоне значений. В основе базовых алгоритмов широко используются операции взятия остатка от деления на величины, характеризующие исходные данные и, например, на размер таблицы, а также генераторы псевдослучайных чисел.

Хеш-функции (Седжвик)

```
int HashInt(typeI i, int M) {return i % M;}
int HashInt’(typeI i, int M) {return (int)(0.616161*(float)i) % M;}
int HashFloat(typeF f, int M) {return ((f-a)/(b-a))*M;}/* [a;b] */
int Hash’(type v, int M) {return (16161*(unsigned)v) % M;}
int Hash’’(type v, int M) {return v & (M – 1);}

int HashString(char * s, int M) {
int h = 0, a = 127;
for(; *s!= ‘\0’; s++) h = (a * h + *s) % M;
return h;
}

int HashStringU(char * s, int M) {
int h = 0, a = 31415, b = 27183;
for(; *s!= ‘\0’; s++, a = a*b % (M-1)) h = (a * h + *s) % M;
return h;
}
```

Хеш-функции. Критерий хи-квадрат

$$\chi^2 = M/N * \Sigma(f_i - N/M)^2,$$

где M – размер хеш-таблицы, N – количество ключей, f_i – количество ключей с хеш-значением i, $i \in [0; M)$.

При N > cM значение χ^2 должно принадлежать диапазону $(M - M^{1/2}; M + M^{1/2})$ с вероятностью $1 - 1/c$.

Критерий считается достаточно жестким, поэтому на практике достаточно отбирать хеш-функции достаточно равномерно отображающие ключи в индексы.

Коллизии. Методы устранения

В случае, когда нескольким разным значениям ключей соответствует одно и то же хеш-значение (индекс) возникает коллизия.

Методы устранения коллизий:

1. раздельное связывание (метод цепочек);

2. методы с открытой адресацией (линейное зондирование, двойное хеширование, расширяемые или динамические хеш-таблицы).

Раздельное связывание

Для решение проблемы коллизий для каждого адреса хеш-таблицы строится связный список [необязательно список] элементов, ключи которых отображаются на этот адрес. Списки могут быть как упорядоченными, так и неупорядоченными.

Средняя длина списка N/M. Вероятность того, что эта величина отличается от N/M крайне мала. Вероятность нахождения в списке k элементов равна ($\alpha = N/M$) :

$$P = C_{N,k} * (\alpha/N)^k * (1 - \alpha/N)^{N-k} \approx (\alpha^k e^{-\alpha})/k!$$

Вероятность наличия в списке более $t\alpha$ элементов, меньше чем

$$P = (\alpha e/t)^t e^{-\alpha}$$

Среднее количество элементов, вставленных до первого совпадения равно $(\pi M/2)^{1/2}$, а среднее количество элементов вставленных прежде, чем в каждом списке окажется. по меньшей мере, по одном элементу равно MH_M.

Линейное зондирование

При коэффициенте загрузки $\alpha = N/M <$ можно для устранения коллизий использовать методы с открытой адресацией.

Линейное зондирование: при наличии конфликта (т.е. когда хеширование осуществляется в ту позицию, которая уже занята) осуществляется проверка следующей позиции в таблице (`i=(i+1)%M`). Такую проверку принято называть зондированием. В результате вставки элементов могут образоваться непрерывные группы занятых ячеек таблицы – кластеры, что естественно увеличивает время вставки и поиска элемента.

Среднее количество зондирований при попаданиях и промахах:

коэффициент загрузки α		1/2	2/3	3/4	9/10
попадания	$(1+1/(1-\alpha))/2$	1,5	2	2,5	5,5
промахи	$(1+1/(1-\alpha)^2)/2$	2,5	5	8,5	55,5

Линейное зондирование

При удалении из таблицы возникает вопрос: что делать с элементами справа? Два варианта:

1. Повторное хеширование всех элементов справа вплоть до пустой области.

2. Замена удаляемого элемента специальным, аналогичным, но не эквивалентным «пустому» элементу.

Двойное хеширование

При наличии конфликта номер очередной проверяемой позиции определяется не последовательным перебором, а с помощью вспомогательной хеш-функции `i = (i + k)%M`. Важно, чтобы *k* было отлично от нуля.

Среднее количество зондирований при попаданиях и промахах:

коэффициент загрузки α		1/2	2/3	3/4	9/10
попадания	$\ln(1/(1-\alpha))/\alpha$	1,4	1,6	1,8	2,6
промахи	$1/(1-\alpha)$	2	3	4	10

При удалении из таблицы возникает вопрос: что делать с элементами этой последовательности? Только один вариант: замена удаляемого элемента специальным, аналогичным, но не эквивалентным «пустому» элементу.