**Лекция 5**

**Алгоритмы хеширования данных**

На прошлой лекции мы рассматривали алгоритмы хеширования. Напомню

Что такое хеширование – преобразование относительно больших объемов информации в меньший объем без возможности обратного преобразования. Это не совсем сжатие информации, которое мы будем рассматривать дальше. Забегая на вперед, сжатие — это тоже хеширование, но с возможностью обратного преобразования.

Если у нас есть какие-то длинные кусочки информации, которые нам не очень нужны в длинном виде, мы применяем хеширование

То есть преобразуем

Иногда рассматривают это с точки зрения хеш-таблиц. Самый простой пример

Есть группа студентов. Студентов обычно идентифицируют по номеру зачётки, но он очень длинный и работать с ним не удобно, поэтому можно его преобразовать в другой номер, например номер журнала. Но мы за это заплатим тем, что студенты разных групп могут иметь одинаковые номера в журнале.

Так вот хеширование это алгоритм преобразования длинных слов в короткие.

Простой пример: Мы берем и преобразуем информацию при помощи хеш-функции. Мы берем слово, или как его называют ключ. Простая хеш-функция: h(Key) = Key % 1000

Key1=496843;

Key2=557843;

H(Key1)=843;

H(Key2)=843;

У нас получились одинаковые результаты, это называется коллизия.

При хешировании возможна коллизия.

Для того, чтобы коллизии не возникали, существует алгоритм разрешения коллизий (collision resolution).

Самый простой способ – использовать полученные ключи в качестве индекса элемента в массиве.

Для разрешения коллизий существует несколько подходов

Первый из них – метод открытой адресации

Если h(k) = I, а i-я позиция в хеш-таблице занята, то вычисляется новая позиция по формуле

I = h(k) + G(i);

Для определения функции G(i) используется:

- линейное зонирование (linear probing) – G(i) = n \* I,

- Квадратичное зондирование (quadratic probing) – G(i) = n\* i^2,

- Повторное хеширование (rehashing), или двойное хеширование (double hashing). Если I = rh(h(k)).

Поиск в хеш-таблице

1. Аргумент поиска ArgSearch хешируется в индекс I = h(ArgSearch)
2. Проверяется i-ая позиция таблицы:

- если ArgSearch совпадает с ключом i-ой записи, то поиск результативен (искомая запись имеет индекс i), конец поиска

- если совпадения не произошло, то переход к п. 3

- или i-ая позиция пуста, поиск безрезультативен, конец поиска

1. Выполняется повторное хеширование, т.е. проверяется позиция с индексом
2. I = rh(i) = rh(h(ArgSearch)

Метод цепочек

Метод …

Метод свертки

Вывод: Хеширование позволяет преобразовать некие длинные слова в более короткие. Если эти числа используется в качестве индекса массива, то мы можем уменьшить длину массива. Это позволяет ускорить поиск по массиву, но получаем вероятность возникновения коллизии.

Типичные функции:

Умножение на константу

Деление на целое число

Метод середины квадрата

Метод суммирования исходных чисел

Алгоритмы словарного сжатия данных

ё