Оглавление

Глава 1. Теория

**Постановка задачи.**

Основными задачами данной работы являются:

1. Реализация точного поиска в тексте на основе хеширования.
2. Сравнение работы алгоритма с конкурентами.
3. Реализация неточного поиска в тексте с указанными процентом совпадения.
4. **Хеши. Основные понятия. Коллизии и способы их решения.**

Для начала введём основные понятия о том, что такое хэши.

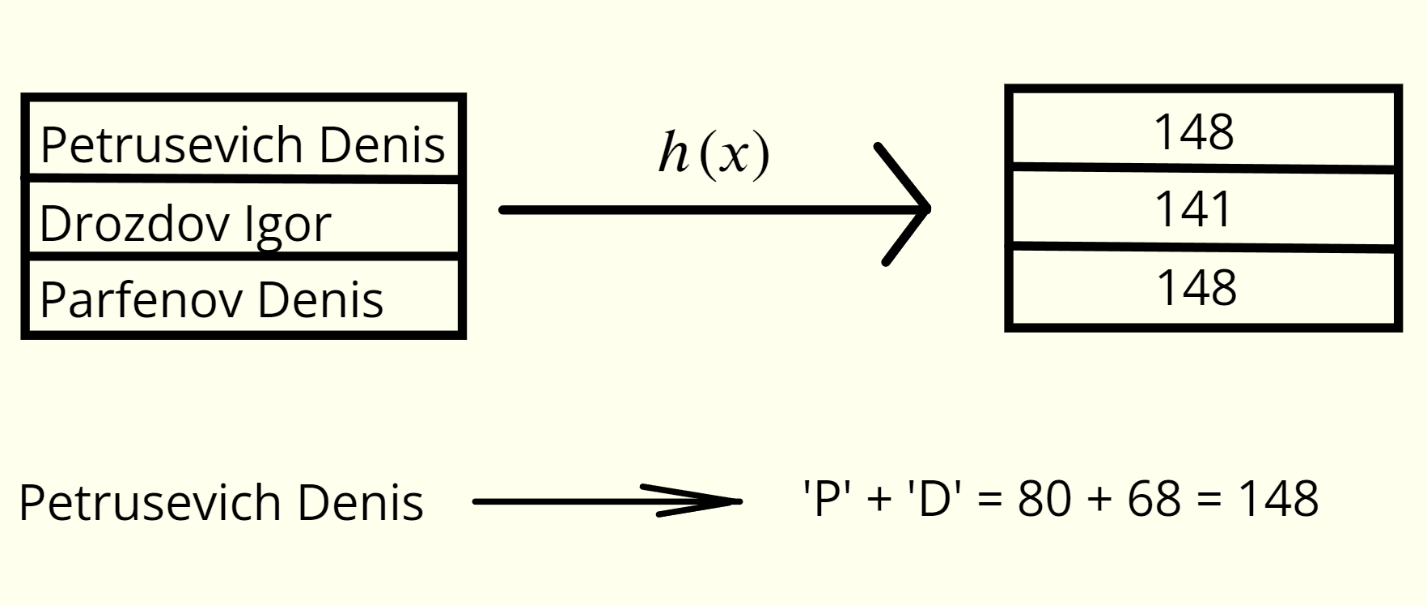
Хеш-функция — это некая функция, которая сопоставляет объектам какого-то множества числовые значения из промежутка.

Преобразование, производимое хеш-функцией, называется хэшированием. Результат хеширования и есть хеш.

Рассмотрим на небольшом примере:

Предположим, у нас есть список преподавателей потока КМБ-22. Нашей задачей является присвоение уникального номера каждому преподавателю. Придумаем простую хеш-функцию, которая берёт первую букву имени и первую букву фамилии, складывает их числовые значения в таблице ASCII (для удобства будем записывать имена и фамилии преподавателей на латинице) и выдаёт результат. Принцип и результат применения нашей хеш-функции можем увидеть на рисунке 1.

Рисунок 1. Принцип работы хеш-функции на примере списка преподавателей.



Диапазон значений, в которые наша хеш-функция обращает имена преподавателей нетрудно вычислить: самое маленькое значение будет , а самое большое .

Плюсы такой хеш-функции:

1. Несложная формула => не требует больших вычислений.
2. Легкая в придумывании и запоминании.

Минусы:

1. Маленький диапазон значений.
2. Высокая вероятность возникновения коллизий.

Коллизия — сопоставление двум объектам из входного множества одного значения из числового диапазона.

В нашем примере у нас возникла коллизия: преподаватели Petrusevich Denis и Parfenov Denis имеют одинаковое значение.

Как решить данную проблему?

1. Изменить хеш-функцию на “хорошую”.
2. Применить метод цепочек.
3. Использовать метод открытой адресации.

Рассмотрим решения проблемы поочерёдно.

1.

Что мы понимаем под “хорошей” хеш-функцией?

“Хорошая” хеш-функция — хеш-функция, которая является инъективным отображением набора данных на числовое множество, т.е. в которой отсутствуют коллизии.

Подобрать такую функцию можно только исходя из набора данных: на каких-то наборах данных эта хеш-функция себя ведёт хорошо, на каких-то плохо.

К сожалению, не существует идеальных хеш-функций. Если мы добавим большое количество данных, то коллизии обязательно возникнут. Это объясняется парадоксом дней рождений: в мультимножество нужно добавить случайных чисел от 1 до n, чтобы какие-то два совпали. Таким образом, получаем, что любую хеш-функцию можно сделать “плохой” просто добавив большое количество данных.

2.

Метод цепочек представляет из себя хранение списка ключей для каждого значения. То есть при возникновении коллизий, мы просто будем добавлять ключи в один список, а затем, при поиске, будем вычислять хеш и проходить по списку в поисках нужного нам значения.

Этот метод чаще всего применяется для разрешения коллизий.

3.

Метод открытой адресации предполагает, что при возникновении коллизий, мы будем искать следующее значение, которое ещё не занято и записывать туда наш ключ. Последовательность значений, которая будет просматриваться называется последовательностью проб. Очевидно, что если мы добавим n + 1 объект, то наша хеш-функция сломается.

**2. Алгоритм поиска с помощью хеширования.**

Теперь, познакомившись с основными понятиями, можно составить алгоритм, который будет делать поиск в тексте с помощью наших хешей.

В качестве основы будем использовать полиномиальное хеширование, т.е. наша хеш-функция будет представлять из себя многочлен.

Пусть у нас есть текст, состоящий из n символов. Тогда хеш-функция будет иметь вид:

Здесь q — некоторый большой модуль, который задаёт диапазон наших значений (поскольку взятие по модулю q даёт нам q различных элементов от 0 до q-1),

p — простой модуль, ,

— i-й символ исходного текста, i = 1, 2, …, n.

Но вычислять хеш каждый раз по такой формуле было бы бессмысленным. Поэтому мы заранее предпосчитаем хеши для префиксов длины 1, 2, …, n и сможем вычислять хеш любой подстроки нашего текста.

Разберём на примере:

Пусть дан текст “abcde”, мы хотим посчитать хеш для подстроки “cde”. Как нам это сделать с помощью предпосчитанных префиксов?

Искомый хеш:

Практика

Ыфв

Список литературы

ыфв