**Киевский национальный университет имени Тараса Шевченка**

**Факультет компьютерных наук и кибернетики**

**Алгоритмы и сложность**

**Лабораторный проект № 7**

**Обобщенный метод Рабина Карпа**

**Отчет**

**Подготовил:**

студент группы К-28

Шатохин Максим Сергеевич

**Киев-2018**

1. **Постановка задачи**

Обобщите метод Рабина-Карпа поиска образца в текстовой строке так, чтобы он позволил решить задачу поиска заданного образца размером m на m в символьном массиве размером n на n. Образец можно двигать по горизонтали и вертикали, но не обращать.

1. **Описание алгоритма**

Пусть есть текст Text[n][n] и шаблон s[m][m]. Для выполнения данной задачи будем следовать следующим образом. Будем искать в i-ой строке, где (0<=i<=n-1), 0-вую строку s Алгоритмом Рабина-Карпа. При нахождении ее – проверяем отрезки строки под ней все тем же алгоритмом Рабина-Карпа на соответствие.

1. **Алгоритм Рабина-Карпа**

Алгоритм начинается с подсчета hash(s[0..m−1]) и hash(p[0..m−1]), а также с подсчета pm, для ускорения ответов на запрос.

Для i∈[0..n−m] вычисляется hash(s[i..i+m−1]) и сравнивается с hash(p[0..m−1]). Если они оказались равны, то образец p скорее всего содержится в строке s начиная с позиции i, хотя возможны и ложные срабатывания алгоритма. Если требуется свести такие срабатывания к минимуму или исключить вовсе, то применяют сравнение некоторых символов из этих строк, которые выбраны случайным образом, или применяют явное сравнение строк, как в наивном алгоритме поиска подстроки в строке. В первом случае проверка произойдет быстрее, но вероятность ложного срабатывания, хоть и небольшая, останется. Во втором случае проверка займет время, равное длине образца, но полностью исключит возможность ложного срабатывания.

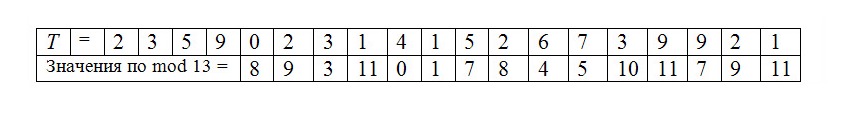
Если требуется найти индексы вхождения нескольких образцов, или сравнить две строки − выгоднее будет предпосчитать все степени p, а также хеши всех префиксов строки s.

1. **Пример**

Пусть алфавит D={0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, то есть каждый символ в алфавите есть d–ичная цифра, где d=│D│.

Пусть образец имеет вид W = 3 1 4 1 5

Вычисляем значения чисел из окна длины |W|=5 по mod q, q — простое число.

23590(mod 13)=8, 35902(mod 13)=9, 59023(mod 13)=9, …

k1=314157(mod 13) – вхождение образца,

k2=673997(mod 13) – холостое срабатывание.

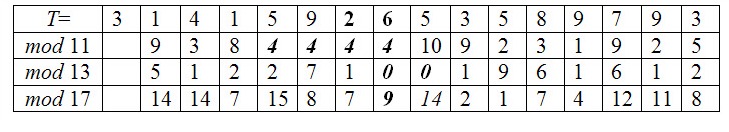
Из равенства ki= kj (mod q) не следует, что ki= kj (например, 31415=67399(mod 13), но это не значит, что 31415=67399). Если ki= kj (mod q), то ещё надо проверить, совпадают ли строки W[1…m] и T[s+1…s+m] на самом деле.

Если простое число q достаточно велико, то дополнительные затраты на анализ холостых срабатываний будут невелики.

В худшем случае время работы алгоритма РК — Θ((N-M+1)\*M), в среднем же он работает достаточно быстро – за время О(N+M).

Пример: Сколько холостых срабатываний k сделает алгоритм РК, если

q= 11, 13, 17. Пусть W={2 6}



26 mod 11=4 → k =3 холостых срабатывания,

26 mod 13=0 → k =1 холостое срабатывание,

26 mod 17=9 → k =0 холостых срабатываний.

Очевидно, что количество холостых срабатываний k является функцией от величины простого числа q (если функция обработки образца mod q) и, в общем случае, от вида функции для обработки образца W и текста Т.