ОСНОВЫ ПРОГРАММНОГО КОНСТРУИРОВАНИЯ

Лекция № 5

2 октября 2017 г.

ПОИСК ПОДСТРОКИ В СТРОКЕ

- Задача: дана строка — «стог сена» длины **N** (haystack). Определить, встречается ли в ней строка-«иголка» длины **M** (needle) и если да, то на какой позиции.

- Метод грубой силы: подставляем «иголку» к каждой возможной позиции в «стоге сена», покуда не совпадет или «стог» не кончится.
  Количество сравнений: **O((N−M) · M)**.

АЛГОРИТМ БОУЭРА-МУРА (МОДИФИЦИРОВАННЫЙ)

«стог сена» b c **a b c b c** b a a …

«иголка» **a a a a a**

- Явно нет смысла сдвигать «иголку» на одну позицию вправо.
- Хорошо бы как-то учесть частичное совпадение строк и не проверять все символы заново каждый раз.

СУТЬ АЛГОРИТМА

- Подставляем «иголку» к началу «стога» (**i = M−1**).
- Сравниваем символы с конца «иголки»: **h[k]** и **n[j]**, уменьшая **k** и **j** (изначально **k = i**, **j=M−1**).
- Если **j** дошло до начала «иголки», подстрока найдена!
- Если на каком-то шаге **h[k] ≠ n[j]**, то сдвигаем «иголку» на **d[h[i]]** вправо (**i += d[h[i]]**).

МАССИВ СДВИГОВ

d — хитрый массив, индексируемый символами **x** из конечного алфавита. Для каждого символа:

- если **x** отсутствует в **n**, то **d[x]** равно **M**;
- если **x** – не последний в **n**, то **d[x]** равно расстоянию от последнего вхождения **x** в **n** до конца **n**;
- если **x** – последний в **n**, то **d[x]** равно расстоянию от предпоследнего вхождения **x** в **n** до конца **n**.

```
n = "abcbc"
d['b'] = 1
d['c'] = 2
d['a'] = 4
d[...] = 5
```

АНАЛИЗ АЛГОРИТМА БОУЭРА-МУРА

- На практике работает очень хорошо, вплоть до: **O(N/M)**.

- В худшем случае (поиск "abbbb" в "bbbbbbbbbb"): **O((N–M)·M)**.

АЛГОРИТМ РАБИНА-КАРПА

Вместо сравнения подстрок будем использовать сравнение их хешей.

- Хеш-функция быстро преобразует произвольную строку в некоторое численное значение, причем равные строки преобразуются в равные значения.
- Например:

Основание исчисления R

$$\text{hash("abc")} = (97 \cdot 256^2 + 98 \cdot 256^1 + 99 \cdot 256^0) \% 997$$

Некое большое простое число Q

СУТЬ АЛГОРИТМА

- T_n — хеш «иголки».
- $T_{h,k}$ — хеш **M** символов «стога», начиная с позиции **k**.
- Перебираем **k = 0…(N–M)** и сравниваем T_n и $T_{h,k}$.
  - Если совпали, то проверяем посимвольно, и в случае совпадения – успех.
  - Иначе идем дальше.

ХИТРОСТЬ ВЫЧИСЛЕНИЯ ХЕШЕЙ В «СТОГЕ СЕНА»

- $T_{h,k} = R^{M-1} \cdot h[k] + R^{M-2} \cdot h[k+1] + \ldots + R^0 \cdot h[k+M-1]$

- $T_{h,k+1} = R^{M-1} \cdot h[k+1] + R^{M-2} \cdot h[k+2] + \ldots + R^0 \cdot h[k+M]$

- Если мы знаем $T_{h,k}$, то за константное время можно вычислить $T_{h,k+1}$:
  $T_{h,k+1} = (T_{h,k} - R^{M-1} \cdot h[k]) \cdot R + h[k+M]$.

АНАЛИЗ АЛГОРИТМА РАБИНА-КАРПА

- В среднем алгоритм работает за **O(N–M)**.
  - При достаточно большом простом **Q**, вероятность ложных совпадений **(1/Q)**.
- В худшем случае **O((N–M)·M)**.
- Отлично подходит для поиска нескольких «иголок» в одном «стоге».

АЛГОРИТМ КНУТА-МОРРИСА-ПРАТТА

- В худшем случае работает за **O(M+N)**.
  - Предварительно строит по «иголке» двумерный массив сдвигов.
  - Во время прохода по «стогу», никогда не идет назад.
- Оптимальный алгоритм, но непростой.

КОНЕЦ ПЯТОЙ ЛЕКЦИИ

Очень трудно найти в тёмной комнате чёрную кошку,
особенно, если её там нет!