

Хеш-таблицы

Множество (set): добавлять и удалять ключи, проверять принадлежность.

Словарь (dictionary, map) или ассоциативная таблица (associative table):
каждому ключу сопоставлены данные, ключи уникальны

Прямая адресация

Лог:

172.16.254.1



2886794753

T



2³²

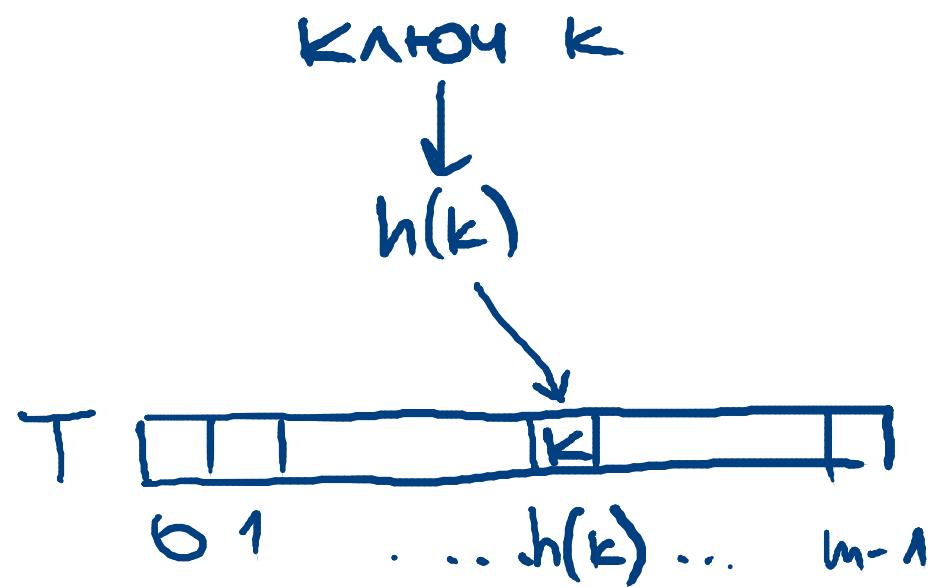
Все операции: O(1), но распределено

Хеш-функции

K - множество ячеек

$h: K \rightarrow \{0, \dots, m-1\}$ - хеш-функция

$h(k)$ - хеш-значение (код) ячейки k



Коллизия - совпадение хеш-значений:

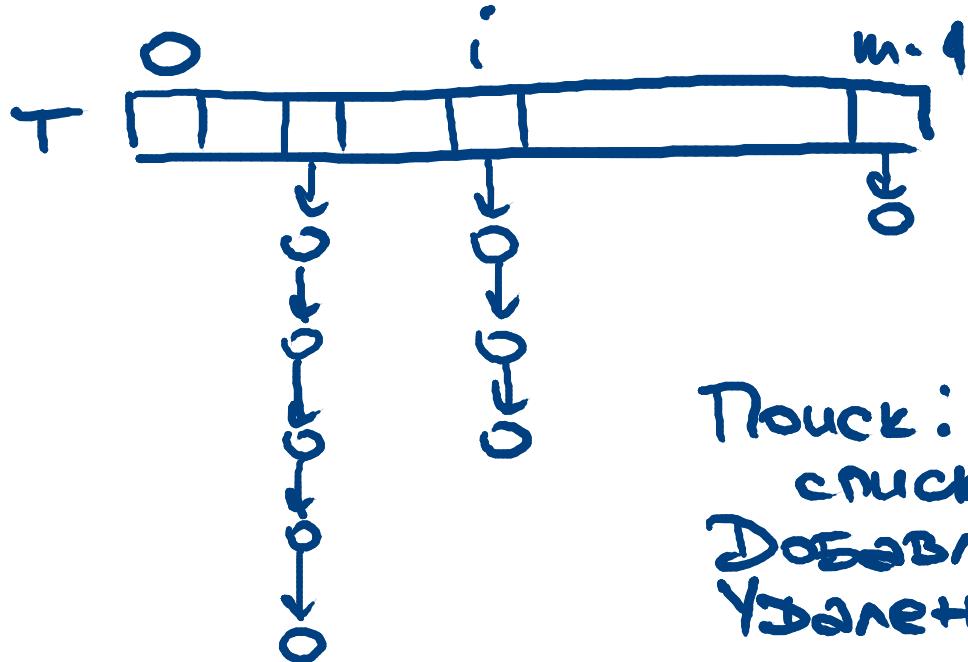
$$h(k_1) = h(k_2), k_1 \neq k_2$$

Если число элементов и больше m ,
то коллизии неизбежны.

$\alpha = n/m$ - коэффициент заполнения

Первый способ разрешения

коллизий : цепочки (chaining)



Поиск: $O(l)$, где l -длина
списка

Добавление: $O(1)$
Удаление: $O(l)$

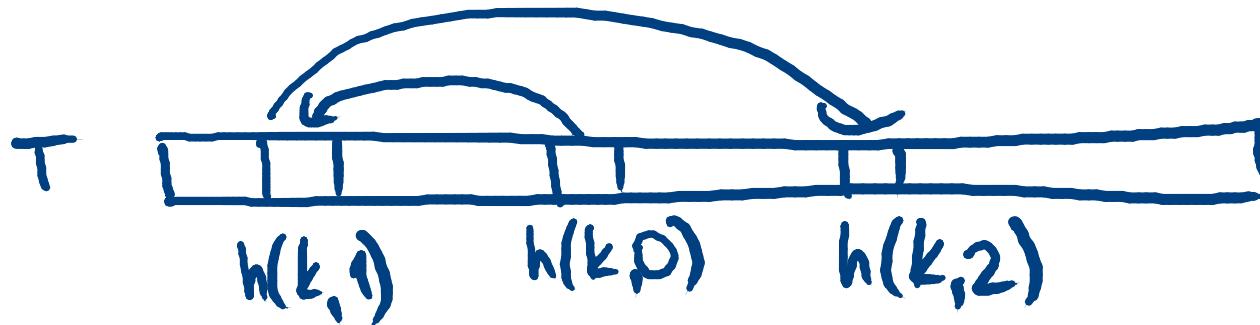
Хранятся не только ключи,
но и указатели

Второй способ: открытая адресация

$$\underline{n \leq m}$$

$$h: K \times \{0, \dots, m-1\} \rightarrow \{0, \dots, m-1\}$$

$\forall k \in K, h(k, 0), h(k, 1), \dots, h(k, m-1)$ —
перестановка $0, 1, \dots, m-1$



Пример: $h_0: K \rightarrow \{0, \dots, m-1\}$

$$h(k, j) = (h_0(k) + j) \bmod m$$

ПРИМЕРЫ ХЕШ-ФУНКЦИЙ

1. $K = [0, 1)$ $h(k) = \lfloor mk \rfloor$



2. $K = \mathbb{Z}$ $h(k) = k \bmod m$

$$h(k) = \lfloor m(ck \bmod 1) \rfloor$$

3. $K = \{\text{последовательности чисел}\}$

$$h(a_0, a_1, \dots, a_t) = (a_0 + a_1x + a_2x^2 + \dots + a_{t-1}x^{t-1}) \bmod m$$

x - фиксированное основание

Вероятностный анализ алгоритмов хеширования

Гипотеза простого равномерного хеширования:

$\eta h(k)$ выбирается случайно равномерно из $0, \dots, n-1$
 $h(k_1) \cup h(k_2)$ независимы для различных k_1, k_2

Теорема: в предположении гипотезы

- 1) средняя длина цепочки равна $d = \eta/m$;
- 2) среднее время поиска - $O(1+d)$.

Dok-bo:

1. k_1, \dots, k_n - клетки в таблице

$$X_{ij} = [h(k_i) = j], \quad E(X_{ij}) = \frac{1}{m}$$

Элементы j -й строкочки равны $\sum_i X_{ij}$

её мат. ожидание:

$$E(\sum_i X_{ij}) = \sum_i E(X_{ij}) = \frac{n}{m} = d$$

2. Безусловный поиск: просмотрели строкочку

Успешный поиск:

k_1, \dots, k_n - клетки таблицы, и предположим
дополнительно, что они добавлялись
именно в этот порядке

Покажем, что среднее (по хеш-значениям
и номерам клеткам) время поиска
нашего клетки - $O(1+\alpha)$.

$$X_{ij} = [h(k_i) = h(k_j)], E(X_{ii}) = 1, E(X_{ij}) = \frac{1}{m} (i \neq j)$$

время поиска k_i : $\chi_{i1} + \dots + \chi_{in}$
(т.к. клетки добавляются в начало цепочки)

среднее по всем ключам:

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^n E\left(\sum_{j=i}^n \chi_{ij}\right) &= \frac{1}{n} \sum_{i=1}^n \left(1 + \sum_{j=i+1}^n E(\chi_{ij})\right) = \frac{1}{n} \sum_{i=1}^n \left(1 + \sum_{j=i+1}^n \frac{1}{m}\right) = \\ &= 1 + \frac{1}{mn} \sum_{i=1}^n (n-i) = 1 + \frac{1}{mn} \left(n^2 - \frac{n(n+1)}{2}\right) = O(1+\alpha) \end{aligned}$$

□

Универсальное хеширование

Оп: семейство $\mathcal{F} = \{h : K \rightarrow \{0, \dots, m-1\}\}$

называется универсальным семейством

хеши-функцией, если $\forall k_1 \neq k_2 \in K$

$$\frac{|\{h \in \mathcal{F} : h(k_1) = h(k_2)\}|}{|\mathcal{F}|} \leq \frac{1}{m}$$

Теорема: среднее время поиска

для универсального хеширований $- O(1+d)$.

Dok-bo:

k_1, \dots, k_n - ячейки в таблице

$$\chi_{k,e} = [h(k) = h(e)]$$

время поиска ячейка e :

$$\chi_{k_1,e} + \chi_{k_2,e} + \dots + \chi_{k_n,e}$$

мат. ожидание:

$$\leq \frac{n-1}{m} + 1 \leq d+1$$

□

Идеально! И можно контролировать d .

Теорема: ניחז $K = \{0, \dots, t-1\}$, $p \in P$, $p \geq t$.

תור $h_{ab}(k) = ((ak+b) \bmod p) \bmod m$.

תודה $f_t = \{h_{ab} : 1 \leq a \leq p-1, 0 \leq b \leq p-1\}$

является универсальным семейством.

Док-во: фиксируем $k_1 \neq k_2$.

$$t_1 = (ak_1 + b) \bmod p, t_2 = (ak_2 + b) \bmod p$$

$t_1 \neq t_2$: если $t_1 = t_2$, то $a(k_1 - k_2) \bmod p$

Различные пары (a, b) дают различные пары (t_1, t_2) : то (t_1, t_2) однозначно восстанавливается (a, b)

$$t_1 - t_2 = a(k_1 - k_2) \bmod p \Rightarrow a = \overbrace{(t_1 - t_2) \cdot (k_1 - k_2)^{-1} \bmod p}^{\text{беск.}}$$

$$\begin{array}{l} a, b: \\ 1 \leq a \leq p-1, \\ 0 \leq b \leq p-1 \end{array}$$

беск.

$$\begin{array}{l} t_1, t_2: \\ 0 \leq t_1, t_2 \leq p-1 \\ t_1 \neq t_2 \end{array}$$

$$\Pr(h_{ab}(k_1) = h_{ab}(k_2)) = \Pr(t_1 = t_2 \bmod m) \leq \frac{p \cdot \frac{p-1}{m}}{p(p-1)} = \frac{1}{m}$$

$$t_1 - \text{фикс.}: \left\lceil \frac{p}{m} \right\rceil - 1 \leq \frac{p+m-1}{m} - 1 = \frac{p-1}{m}$$

□

Поиск образца в тексте

Вход: текст T и образец P .

Выход: все позиции i , такие что
 $0 \leq i \leq |T| - |P|$

$$T[i..i+|P|-1] = P.$$

Пример: $T = \underline{ab} \underline{a} bac \underline{aba}$ $P = aba$

Выход: 0, 2, 6

Простейшее решение

Попытаться приложить образец
для каждой позиции $0 \leq i \leq |T| - |P|$

Общее время работы:

$$O(|P| \cdot (|T| - |P|)) = O(|P| \cdot |T|)$$

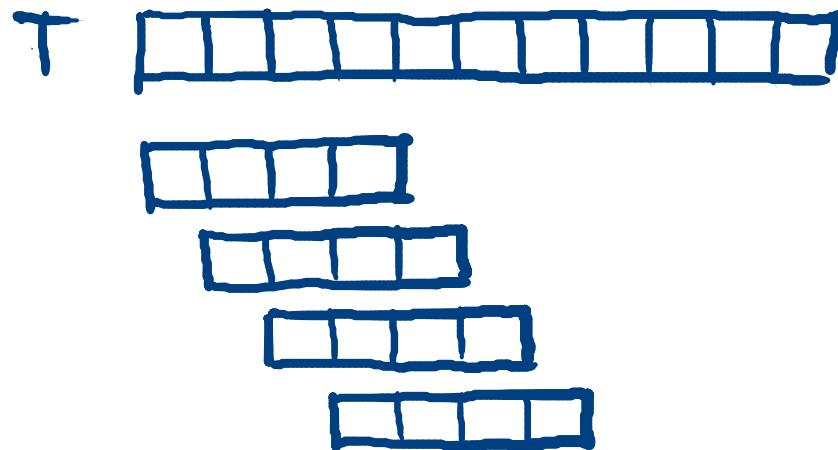
Оценка точна:

$T = aaaaaaaaaaaaaaaaaaaaaaa$

$P = aaaaab$

Основные идеи алгоритма

Рабина-Карпа

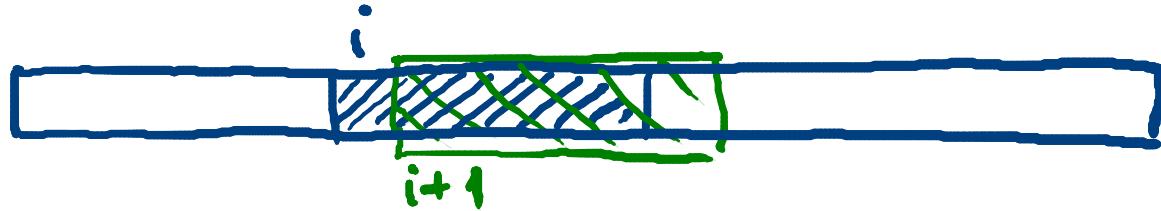


Вместо посимвольного сравнения строк R и $T[i..|P|+i-1]$ будем сравнивать их хеш-значения

Но ведь вычисление $h(T[i..|P|+i-1])$ всё равно займёт времени хотя бы $|P|$?

Да, но мы вычислим хеш-значения для всех i за $O(|T|)$!

Полиномиальная хеш-функция



$$T[i+1..i+|P|] = t_{i+1}, t_{i+2}, \dots, t_{i+|P|} .$$

$$T[i..i+|P|-1] = t_i, t_{i+1}, \dots, t_{i+|P|-1} .$$

$$h_{i+1} = t_{i+1} \cdot x^0 + t_{i+2} \cdot x^1 + t_{i+3} \cdot x^2 + \dots + t_{i+|P|-1} \cdot x^{|P|-2} + t_{i+|P|} \cdot x^{|P|-1} \pmod{p}$$

$$h_i = t_i \cdot x^0 + t_{i+1} \cdot x^1 + t_{i+2} \cdot x^2 + t_{i+3} \cdot x^3 + \dots + t_{i+|P|-1} \cdot x^{|P|-1} \pmod{p}$$

$$h_i = (h_{i+1} - t_{i+|P|} \cdot x^{|P|-1}) \cdot x + t_i \cdot x^0 \pmod{p}$$

Пересчит 38 корректируе время!

Алгоритм Рабина - Кара (P, T)

зарегистрировать большое $p \in \mathbb{P}$

выбрать случайное $1 \leq x \leq p-1$

вычислить и сохранить $(x^{(P)-1} \bmod p)$ и $h(P)$

идя справа налево по тексту T ,
вычислить и сохранить хеш-значения
всех окон размера $|P|$

для каждого окна W в T :

если $h(W) = h(P)$:

сравнить W и P посимвольно

Время работы

Эффективное наивного решения

При больших p среднее время

работы: $O(|T| + \text{occ} \cdot |P|)$, где

occ - число входящих P в T

Ложные срабатывания: если $W \neq P$, то

$$\Pr_x \{ h_x(W) = h_x(P) \} \leq \frac{|P|}{p}$$