Домашнее задание №4
Алгоритмы. 5 курс. Весенний семестр.

Горбунов Егор Алексеевич

16 марта 2016 г.

1 Мои решения

Задание №1

(a) Докажите, что любое 2-независимое семейство хэш-функций является универсальным.

(b) Докажите, что любое $k+1$-независимое семейство хэш-функций является k-независимым.

Решение:

(a) $Pr[h(x_1) = h(x_2)] = \sum_{y \in Y} Pr[h(x_1) = y \wedge h(x_2) = y] =^{\text{в силу 2-независимости}}= \sum_{y \in Y} \frac{1}{|Y|^2} = |Y|\frac{1}{|Y|^2} = \frac{1}{|Y|}$ Получили, что $Pr_{h \in \mathcal{H}}[h(x_1) = h(x_2)] \leqslant \frac{1}{|Y|}$, а значит 2-независимое семейство является универсальным. ∎

(b) Заметим, что можно $h(x_{k+1})$ может принимать значения из Y да и только, так что:

$$Pr\Big[\bigwedge_{i=1}^{k} h(x_i) = y_i\Big] = \sum_{y \in Y} Pr\Big[\bigwedge_{i=1}^{k} h(x_i) = y_i \wedge h(x_{k+1}) = y\Big] =^{\text{по } k+1\text{-независимости}}= \sum_{|Y|} \frac{1}{|Y|^{k+1}} =$$
$$= |Y|\frac{1}{|Y|^{k+1}} = \frac{1}{|Y|^k}$$

Получили, что хотели. ∎

Задание №2 Есть пара 2-независимых семейств хеш-функций $\mathcal{A} = \{f : A \to \mathbb{F}_2^n\}$ и $\mathcal{B} = \{g : B \to \mathbb{F}_2^n\}$. Постройте универсальное семейство хэш функций, которое:

(a) будет отправлять пары типа (A, B) в $\mathbb{F}_2^n$

(b) будет отправлять мультимножества их элементов типа A в $\mathbb{F}_2^n$

(c) будет отправлять множества их элементов типа A в $\mathbb{F}_2^n$

(d) будет отправлять списки их элементов типа A в $\mathbb{F}_2^n$

Решение:

(a) Рассмотрим такое семейство функций $\mathcal{C} = \{h((a,b)) = f(a) + g(b)\}$ (сложение в данном поле — это XOR). Вспомним свойства XOR, что если $c = a + d$, то $d = a + c$. Тогда:

$$
\begin{aligned}
Pr[h((a_1,b_1)) = h((a_2,b_2))] = Pr[f(a_1) + g(b_1) = f(a_2) + g(b_2)] =& \\
&= \sum_{v\in\mathbb{F}_2^n} P[f(a_1) + g(b_1) = v \wedge f(a_2) + g(b_2) = v] = \\
&= \sum_{v\in\mathbb{F}_2^n}\sum_{u\in\mathbb{F}_2^n} P[f(a_1) = u \wedge g(b_1) = u + v \wedge f(a_2) + g(b_2) = v] = \\
&= \sum_{v\in\mathbb{F}_2^n}\sum_{u\in\mathbb{F}_2^n}\sum_{e\in\mathbb{F}_2^n} P[f(a_1) = u \wedge g(b_1) = u + v \wedge f(a_2) = e \wedge g(b_2) = e + v] = \\
&=^{\text{в силу независимости } f \text{ и } g} = \\
&= \sum_{v,u,e\in\mathbb{F}_2^n} P[f(a_1) = u \wedge f(a_2) = e]P[g(b_1) = u + v \wedge g(b_2) = e + v] = \\
&=^{\text{в силу 2-независимости } \mathcal{A} \text{ и } \mathcal{B}} = \\
&= (2^n)^3 \cdot \frac{1}{2^{2n}2^{2n}} = \\
&= \frac{1}{2^n} = \frac{1}{|\mathbb{F}_2^n|}
\end{aligned}
$$

■

(b) Заметим, что семейство функций должно быть бесконечным семейство хэш-функций. Действительно, если семейство конечно, то по принципу Дирихле в силу бесконечности числа мультимножеств получится, что хотя бы для 2-х элементов совпадают хэши (хэшом будет конкатенация применения всех функций из семейства к объекту), а значит условие универсальности для этих объектов не выполняется.

Рассматриваем мультимножество размера k и выбираем независимо $f_1, f_2, \ldots, f_k \in \mathcal{A}$ и построим семейство с хэш-функциями $h(\{a_1, a_2, \ldots, a_k\}) = \sum_{i=1}^k f_i(a_i)$, теперь нужно оценить вероятность:

$$\Pr\left[\sum f_i(a_i) = \sum f_i(b_i)\right] = \sum_y \Pr\left[\sum f_i(a_i) = \sum f_i(b_i) = y\right]$$

Задание №3 Используя задачу с практики построить семейство 2-независимых хэш-функций $\mathcal{H} = \{h_i : \mathbb{F}_2^n \to \mathbb{F}_2^k\}_i$. Докажите, что семейство является 2-независимым.

Решение: Семейство выглядит так: $\mathcal{H} = \{h(x) = Ax + b\}$. Параметрами семейства выступают матрица A ранга k и размера $k \times n$ над $\mathbb{F}_2$:

$$\begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_k \end{pmatrix}$$

И вектор b над $\mathbb{F}_2$ длины k. Теперь будем показывать, что это семейство является 2-независимым, т.е. покажем, что $\forall x_1, x_2 \in \mathbb{F}_2^n,\ x_1 \neq x_2$ и $\forall y_1, y_2 \in \mathbb{F}_2^k$, что:

$$\Pr_{h \in \mathcal{H}}\left[h(x_1) = y_1 \wedge h(x_2) = y_2\right] = \frac{1}{|\mathbb{F}_2^n|^2} = \frac{1}{2^{2n}}$$

Нам нужно найти вероятность такой матрицы A, что $y_1 = Ax_1 + b$ и $y_2 = Ax_2 + b$, т.е. $A(x_1 - x_2) = y_1 - y_2$ и вероятность вектора b. Пусть $y_1 - y_2 = (l_1, l_2, \ldots, l_k)^T$.

$$\begin{aligned}\Pr\left[Az = y_1 - y_2\right] &= \Pr\left[A_1 z = l_1\right]\Pr\left[A_2 z = l_2 | A_1\right] \cdot \ldots \cdot \Pr\left[A_k z = l_k | A_1, A_2, \ldots, A_{k-1}\right] \\ &= \text{в силу того, что у нас все элементы } A \text{ независимы} \\ &= \Pr\left[A_1 z = l_1\right]\Pr\left[A_k z = l_k\right] \cdot \ldots \cdot \Pr\left[A_k z = l_k\right]\end{aligned}$$

Заметим, что в каждой вероятности $A_i z$ равно либо 1 либо 0. Вероятности 0 и 1 в результате суммы и разности по модулю равны $\frac{1}{2}$, таким образом будет $\Pr[A_i z = 1 \wedge l_i = 1] + \Pr[A_i z = 0 \wedge l_i = 0] = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$ (тут в силу независимости раскрывается вероятность произведения в произведение вероятностей). Таким образом получили:

$$\Pr\left[Az = y_1 - y_2\right] = \prod_{i=1}^{n} \frac{1}{2} = \frac{1}{2^n}$$

Вероятность же получить b такой, что $b = Ax_1 - y_1$ у нас всегда $\frac{1}{2^n}$, т.к. все элементы вектора независимы, а таким образом $\Pr_{h \in \mathcal{H}}\left[h(x_1) = y_1 \wedge h(x_2) = y_2\right]$ получается произведением полученных вероятностей для b и A:

$$\Pr_{h \in \mathcal{H}}\left[h(x_1) = y_1 \wedge h(x_2) = y_2\right] = \frac{1}{2^n}\frac{1}{2^n} = \frac{1}{2^{2n}}$$

$\blacksquare$

Задание №4 Есть хэш-таблица размера n с хэш-функцией $h : K \to [n]$. h равновероятно отправляет ключи в корзины. На вход поступает n ключей. Коллизии разрешаются цепочками. Показать:

(a) Зафиксируем $x \in [n]$. Доказать:

$$Q_k = \Pr\left[k \text{ ключей имеют хэш } x\right] = \binom{n}{k} \cdot \left(\frac{1}{n}\right)^k \cdot \left(1 - \frac{1}{n}\right)^{n-k}$$

(b) Пусть P_k — вероятность максимальной цепочки иметь длину k. Доказать, что $P_k \leqslant nQ_k$

(c) Показать, что $Q_k < (\frac{e}{k})^k$

(d) Показать, что для некоторого $c > 1$ верно $Q_k \leqslant \frac{1}{n^3}$ при $l \geqslant c\frac{\log n}{\log\log n}$

(e) Доказать, что матожидание длины максимальной цепочки не превосходит

$$\Pr\left[M > c\frac{\log n}{\log\log n}\right]n + \Pr\left[M \leqslant c\frac{\log n}{\log\log n}\right]c\frac{\log n}{\log\log n}$$

где M — максимальная длина цепочки. M — случайная величина. Вывести оценку сверху на это матожидание: $\mathcal{O}\left(\frac{\log n}{\log\log n}\right)$

Решение:

(a) У нас есть n элементов. Если мы зафиксируем какие-то k элементов, то ясно, что вероятность, что именно они будут иметь хэш x равна $\left(\frac{1}{n}\right)^k\left(1-\frac{1}{n}\right)^{n-k}$, т.к. вероятность, что у элемента хэш равен x равна $\frac{1}{n}$, а вероятность того, что не равен, соответственна $1-\frac{1}{n}$. А k элементов мы можем зафиксировать $\binom{n}{k}$ способами. Все такие выборы независимы, а значит нужно просуммировать $\binom{n}{k}$ раз $\left(\frac{1}{n}\right)^k\left(1-\frac{1}{n}\right)^{n-k}$ чтобы получить Q_k. ∎

(b) $P_k = \sum_{x\in[n]} P[$максимальная цепь имеет длину k и все элементы из неё имеют хэш $x]$. Ясно, что вероятность под суммой всяко меньше или равна Q_k (т.к. в Q_k не требуется максимальность), а значит: $P_k \leqslant \sum_{x\in[n]} Q_k = nQ_k$ ∎

(c)

$$Q_k = \frac{n!(n-1)^{n-k}}{k!(n-k)!n^k n^{n-k}} < \frac{n!(n)^{n-k}}{k!(n-k)!n^k n^{n-k}} = \frac{n!}{k!(n-k)!n^k} < \frac{n^k}{k!(n-k)!n^k} =$$
$$= \frac{1}{k!(n-k)!} < \frac{1}{k!} \approx^{\text{Стирлинг}} \left(\frac{e}{k}\right)^k \frac{1}{\sqrt{\pi k}} < \left(\frac{e}{k}\right)^k$$

∎

(d) Воспользуемся предыдущим пунктом.

$$Q_k < \left(\frac{e}{k}\right)^k = \left(\frac{e}{c\frac{\log n}{\log\log n}}\right)^{c\frac{\log n}{\log\log n}} = \left(e\frac{\log\log n}{c\log n}\right)^{c\frac{\log n}{\log\log n}} =$$
$$= \exp\left(\frac{c\log n}{\log\log n} + \frac{c\log n}{\log\log n}\left(\log\log\log n - \log c - \log\log n\right)\right) =$$
$$= \exp\left(-c\log n\left(1 - \frac{\log\log\log n - \log c + 1}{\log\log n}\right)\right)$$

Ясно, что найдётся такое c, т.к. то, что внутри скобки в экспоненте стремиться к 1 при стремлении n к бесконечности, такое, что: $Q_k < e^{-3\log n} = \frac{1}{n^3}$ ∎

(e) Распишем матожидание длины максимальной цепочки:

$$E = \sum_{k=1}^{n} kP_k = \left(P_1 + P_2 2 + \ldots + P_{c\frac{\log n}{\log\log n}} c\frac{\log n}{\log\log n}\right) + \left(P_{c\frac{\log n}{\log\log n}+1}\left(c\frac{\log n}{\log\log n}+1\right) + \ldots P_n n\right) \leqslant$$
$$\leqslant c\frac{\log n}{\log\log n}\left(P_1 + \ldots P_{c\frac{\log n}{\log\log n}}\right) + n\left(P_{c\frac{\log n}{\log\log n}+1} + \ldots P_n\right) =$$

= а это и есть то, что нужно, т.к. длина макс. цепи перебирается

$$= Pr\left[M > c\frac{\log n}{\log\log n}\right] n + \Pr\left[M \leqslant c\frac{\log n}{\log\log n}\right] c\frac{\log n}{\log\log n}$$

Для доказательство оценки сверху нужно просто воспользоваться пунктом c, d и (первый переход был получен выше):

$$E = \left(P_1 + P_2 2 + \ldots + P_{c\frac{\log n}{\log\log n}} c\frac{\log n}{\log\log n}\right) + \left(P_{c\frac{\log n}{\log\log n}+1}\left(c\frac{\log n}{\log\log n}+1\right) + \ldots P_n n\right) \leqslant$$
$$\leqslant \left(P_1 + P_2 2 + \ldots + P_{c\frac{\log n}{\log\log n}} c\frac{\log n}{\log\log n}\right) + n\left(Q_{c\frac{\log n}{\log\log n}+1}\left(c\frac{\log n}{\log\log n}+1\right) + \ldots Q_n n\right) \leqslant$$
$$\leqslant \left(P_1 + P_2 2 + \ldots + P_{c\frac{\log n}{\log\log n}} c\frac{\log n}{\log\log n}\right) + n\left(\frac{1}{n^3}\left(c\frac{\log n}{\log\log n}+1\right) + \ldots \frac{1}{n^3}n\right) =$$
$$= \mathcal{O}\left(\Pr\left[M \leqslant c\frac{\log n}{\log\log n}\right] c\frac{\log n}{\log\log n}\right) \overset{\text{Вероятность} \leqslant 1}{=} \mathcal{O}\left(\frac{\log n}{\log\log n}\right)$$

■

Задание №7 Про восстановление k по входному числе 2^k за $\mathcal{O}(1)$.

Решение: Нужно найти такое число, которое даёт различные остатки от деления на все 2^k, где k пробегает от 0 до 63. Это число будет небольшим (вроде как подойдёт 67). В силу того, что числа маленькие и $k < 64$ всё поместится в 100 байт. ■

Задание №8 Про изоморфизм деревьев.

Решение:

(a) Если порядок важен, то можно делать так: для дерева T найти все вершины, высота поддеревьев которых равна высоте t. Далее для каждой такой вершины построить строчку так: разметить все вершины в порядке обхода в глубину, а потом сделать эйлеров обход и получить список чисел. Так же сделать для дерева t. Теперь просто хэшами сравниваем эти списки.

(b) Если порядок не важен, то так же отбираем по нужной высоте вершины, как и в прошлой задаче, а потом в каждом поддереве каждой вершины начинаем рассматривать вершины с самых нижних по уровням. Только теперь нужно сравнивать каждый уровень с соответ. уровнем t хэшами не на списках, а на множествах :)

Задание №9 Про изоморфизм деревьев без хэшей

Решение: См. предыдущую, только сравниваем без хэшей, а за линию.