Практика по алгоритмам

Александр Мишунин, Алексей Давыдов*

Весна, 2015

*Составители сборника не являются авторами самих задач. Авторы не указаны в учебных целях.

1 Практика 1. Декартовы деревья

1.1 Практика

1. Мы хотим хранить пары $(a_i; b_i)$. Возможно ли по ключу a_i построить декартово дерево, а в каждой вершине поддерживать сумму всех b_i в поддереве? А можно поддерживать декартово дерево всех b_i в поддереве? За сколько будут работать операции с таким деревом?

2. (a) Нарисуйте все деревья, которые могут получится в результате операции *Merge(бамбук идущий влево-вниз, вершина)* и *Merge(бамбук идущий вправо-вниз, вершина).*

   (b) Что будет, если мы будем добавлять в декартово дерево элементы с одинаковыми ключами? Заметим, что поиск места разреза можно осуществлять двумя способами: $if(root.x > x)$ и $if(root.x \geq x)$.

3. Том Сойер красит забор. Делает он это таким образом: раз в 15 минут он подзывает очередного мальчишку, вручает тому ведро краски цвета c_i и отправляет его красить забор с жердочки l_i по жердочку r_i, при этом возможно, что эта часть забора была уже частично или полностью покрашена предыдущими ребятами, в таком случае ее все равно перекрашивают новым цветом, поверх старого. Периодически ему надо отвечать на запросы: какого цвета заданная жердочка.

   Вам дана последовательность из m запросов запросы могут быть двух видов:

   - Покрасить отрезок с l_i по r_i цветом c_i,
   - Ответить, какого цвета заданная жердочка.

   Препроцессинг: $O(n \log n)$, время на запрос: $O(\log n)$.

1.2 Домашнее задание

1. (Группа Давыдова)

   (a) Нарисуйте все деревья, которые могут получится в результате операции *Merge(дерево в форме λ, вершина).*

   (b) Что будет, если мы будем добавлять в декартово дерево элементы с одинаковыми ключами? Заметим, что поиск места разреза можно осуществлять двумя способами: $if(root.x > x)$ и $if(root.x \geq x)$.

2. Придумайте, как реализовать структуру данных, поддерживающую следующие операции на последовательности из n чисел:

   - замена i-го числа на x — $O(\log n)$,
   - сумма чисел с позиции l по позицию r — $O(\log n)$,
   - увеличиение каждого из чисел на позициях с l по позицию r на x — $O(\log n)$,
   - разворот подпоследовательности с позиции l по позицию r задом наперед — $O(\log n)$.

3. В одной очень большой и очень солидной компании в раздевалке есть вешалка для пальто. Она представляет собой n крючков, расположенных в ряд. Крючки пронумерованы натуральными числами от 1 до n слева направо.

   В компании очень сложный распорядок работ у сотрудников. В начале рабочего дня все сотрудники находятся не на работе, а вешалка в раздевалке пуста. В некоторые моменты времени сотрудники приходят на работу, а в некоторые уходят.

   Когда сотрудник приходит на работу, он вешает на один из свободных крючков свое пальто. Чтобы доставить своим коллегам как можно меньше неудобств, крючок, на который будет повешено

пальто, выбирается следующим образом. Сначала выбирается самый длинный отрезок из подряд идущих пустых крючков. Если таких отрезков несколько, то выбирается самый правый из них. После этого пальто вешается на крючок, расположенный в середине этого отрезка. Если в отрезке четное количество крючков, то пальто вешается на правый из двух срединных крючков.

Когда сотрудник уходит с работы — он забирает свое пальто. Так как все сотрудники в компании очень уважают друг друга, никто не трогает чужие пальто.

Время от времени директору этой очень солидной компании становится скучно и он отправляет свою секретаршу посмотреть сколько пальто висят на вешалке с i-го по j-ый крючок включительно. И эту прихоть приходится всегда выполнять — иначе директор начинает злиться и у него случается нервный срыв.

Чтобы не тратить слишком много времени на перемещение от кабинета директора до раздевалки и обратно, секретарша попросила вас написать программу, эмулирующую работу раздевалки компании.

Время обработки запросов директора, а также приходов и уходов сотррудников — логарифмическое.

4.* Придумайте, как реализовать структуру данных, поддерживающую следующие операции на последовательности из n чисел:

- Обмен местами соседних чисел на заданном отрезке четной длины (пример: $[1, 2, 3, 4, 5, 6], (2, 5) \to [1, 3, 2, 5, 4, 6]$),
- Вывод числа на заданной позиции.

Время работы — $O(\log n)$ на запрос.

2 Практика 2. Динамическое программирование

2.1 Практика

1. На билете есть $2n$ значный номер. Билет считается счастливым, если сумма первых n цифр совпадает с суммой последних n цифр. По заданому числу n требуется найти число счастливых n значных билетов за $O(n^2)$. Считать, что стандартные арифметические операции над числами выполняются за $O(1)$.

2. Найти максимальное по весу паросочетание на

   (a) дереве,

   (b) цикле,

   (c) связном графе из n вершин и n ребер

   за $O(n)$.

3. Найти максимальную возрастающую подпоследовательность, за $O(n \log n)$. Тут есть два разных варианта решения, подскази:

   (a) вспомните предыдущую пару,

   (b) в качестве состояния динамики — выберите минимальное число, на которое может заканчиватся возрастающая последовательность длины k.

4. Строка s является палиндромом, если она читается одинаково как слева направо, так и справа налево. Посчитать число подстрок-палиндромов за $O(n^2)$.

5. Дан n-угольник. Каждая диагональ треугольника, соединяющая вершины i и j имеет вес w_{ij}. Вес триангуляциии многоугольника есть сумма весов диагоналей, которые в ней проведены. Найти триангуляцию с минимальным весом за $O(n^3)$.

2.2 Домашнее задание

1. Строка s является палиндромом, если она читается одинаково как слева направо, так и справа налево. Требуется разбить строку s длины n на минимальное число палиндромов за $O(n^2)$.

2. Найти максимальное по весу паросочетание на кактусе за $O(n)$. Кактус — граф, в котором каждое ребро лежит не более чем на одном цикле.

3. Даны две последовательности длины n. Придумайте, как найти наидлиннейшую общую подпоследовательность этих последовательностей.

   (a) За $O(n^2)$,

   (b) За $O(n \log n)$, в случае, если в каждой из последовательностей все элементы различны.

4. Посчитайте, сколько существует перестановок из n элементов таких, что ни один элемент не стоит на своем месте, за $O(n^2)$ (Ответ нас интересует только по некоторому простому модулю, т. ч. арифметические операции работают $O(1)$).

3 Практика 3. Динамическое программирование - 2

3.1 Практика

1. Сколькими способами можно замостить доминошками клечатое поле:
   (a) $n \times 3$ за $O(n)$,
   (b) $n \times m$ за $(4^n m)$,
   (c) (*) $n \times m$ за $O(2^n nm)$.

   Ответ посчитать по модулю небольшого простого числа.

2. Пусть последовательность 64-битных чисел задана следующим образом: $a_1 = 1, a_2 = 1, a_i = 12345a_{i-1} + 6789a_{i-2}$, переполнение игнорируется. Как вычислить вычислить n-й член за время $O(\log n)$.

3. Красивые узоры. Дано поле $n \times 5$. Красим в черный и белый цвет. Узор красивый, если не содержит квадрата 2×2 одного цвета. Посчитать число красивых узоров за $O(\log n)$ по модулю небольшого простого числа.

4. Посчитать количество способов, которыми можно расставить на доске $n \times n$, так чтобы они не били друг друга:
   (a) Королей за $O(nFib(n)^2)$,
   (b) Коней за $O(n8^n)$

   по модулю небольшого простого числа.

3.2 Домашнее задание

1. Определим операцию тропического умножения над битовыми матрицами. Под $A \otimes B$ обозначим матрицу C, вычисляемую следующим образом: $C_{ij} = \min\limits_k(A_{ik} + B_{kj})$ (как и при классическом умножении, необходимо, что бы ширина матрицы A и высота матрицы B были равны). Под $A^{\otimes k}$ будем обозначать $A \otimes A \otimes \cdots \otimes A$ — k раз. Придумайте, как посчитать $A^{\otimes n}$ за (n^3), если диагональ матрицы A — нулевая.

2. Дана поседовательность из n чисел. Посчитайте для нее количество возрастающих подпоследовательностей длины k за $O(nk \log n)$.

3. Петя хочет раскрасить полоску из N клеточек в M цветов. Чтобы полоска получилась красивой, он решил воспользоваться таблицей сочетаемости цветов — это такая таблица $M \times M$ в каждой клетке которой написано, хорошо ли смотрятся данные цвета рядом (т.е. соприкосаясь по стороне клетки), или нет (очевидно, что эта таблица симметрична относительно главной диагонали). Помогите Пете посчитать, сколькими способами он может раскрасить полоску за $O(M^2N)$.

4. В новой игре “Closed Loops 7” игрокам предлагается клетчатая таблица N на M клеток. Ход состоит в том, что очередной игрок рисует цикл - замкнутую линию без самопересечений, идущую только по сторонам клеток. Каждый цикл можно нарисовать только один раз за всю игру (при этом, конечно, не запрещается рисовать циклы, пересекающиеся с уже нарисованными). Игроки ходят по очереди. Выигрывает тот, кто рисует последний возможный цикл. К примеру, если $N = 2$, $M = 1$, то циклов всего три и игрок, делающий третий ход, выигрывает.

   Вася позвал $K - 1$ друзей поиграть с ним. Чтобы произвести впечатление, он непременно хочет выиграть. Для этого ему нужно узнать, каким по счету игроком он должен быть, чтобы гарантированно одержать победу. Вася наслышан о ваших успехах в программировании, и за помощью он обратился именно к вам. Время работы: $O(M^{2M}N)$

4 Практика 4. RMQ и LCA

4.1 Практика

1. Дан массив a и бинарная функция f, возможно не ассоциативная. Придумайте, как отвечать на запросы: посчитать f на отрезке $a[l..r]$ за $O(1)$ с предпосчетом $O(n^2)$.

2. Придумайте, как отвечать на запросы RMQ за $O(\log^* n)$ с предподсчетом $O(n \log^* n)$.

3. Дано дерево T из n вершин. Расстоянием для вершин u и v дерева T назовем число ребер в пути от u до v; обозначим через $\texttt{dist}(u, v)$. Требуется уметь за $O(\log n)$ отвечать на запросы "Какого расстояние $\texttt{dist}(u, v)$ для заданных u и v?"

4. Вам дано дерево из одной вершины. Придумайте, как отвечать на следующие запросы:

   - Подвесить новую вершину к дереву — $O(1)$,
   - LCA двух вершин — $O(h)$,

   где h — высота дерева на данный момент.

5. Вам дано дерево из одной вершины. Придумайте, как отвечать на следующие запросы:

   - Подвесить новую вершину к дереву — $O(\log n)$,
   - LCA двух вершин — $O(\log n)$,

   где n — количество вершин в дереве на данный момент.

6. Дано дерево из одной вершины. Требуется уметь отвечать на следующие запросы за $O(\log n)$:

   - Подвесить новую вершину u к вершине дерева v и вернуть диаметр дерева.

   Диаметр дерева – это длина самого длинного пути в дереве.

7. Рассмотрим бинарную скошенную систему исчисления. На каждой позиции в скошенной записи числа может стоять цифра 0, 1 или 2. Число $a_k a_{k-1} \cdots a_2 a_1$ в скошенной системе переводится в десятичную по формуле $\sum_{i=1}^{k} a_i \cdot (2^i - 1)$.

   В скошенной системе исчисления есть два ограничения: цифра 2 может встречаться в записи не более одного раза; все цифры следующих меньших разрядов равны нулю. Пример первых чисел: 0, 1, 2, 10, 11, 12, 20, 100, 101 . . .

   (a) Докажите, что каждое неотрицательное целое число имеет ровно одну возможную запись в скошенной системе исчисления.

   (b) Придумайте, как увеличить число в скошенной системе на единицу за $O(1)$.

8. Рассмотрим структуру данных “скошенный список”. Для того, чтобы получить скошенный список длины n сперва запишем число n в скошенной системе счисления. Далее для каждого i смотрим в соответствующую позицию скошенной записи n и, если соответствующее число не ноль, рисуем столько полных двоичных деревьев высоты i. Пример скошенных списков длины 1, 2, 3, 4 - см.

   Придумайте как реализовать следующие операции со списком длины n:

   (a) Добавление элемента в начало списка за $O(1)$,

   (b) Доступ к i-му элементу за $O(logn)$,

   (c) Получить список из k последних элементов данного списка за $O(logn)$.

9. Вам дано дерево из одной вершины. Придумайте, как отвечать на следующие запросы:

- Подвесить новую вершину к дереву — $O(1)$,
- LCA двух вершин — $O(\log n)$,

где n — количество вершин в дереве на данный момент.

10. Дан лес подввешенных деревьев. Нужно отвечать на следующие запросы:
    (a) подвесить дерево с корнем v к вершине u другого дерева
    (b) отрезать поддерево с корнем в вершине v от ее дерева; в
    (c) проверить, в одном ли дереве лежат вершины u и v.

    Время: $O(\log n)$

11. Дан лес подввешенных деревьев. Нужно отвечать на следующие запросы:
    (a) подвесить дерево с корнем v к вершине u другого дерева
    (b) LCA вершин u и v.

    Время: $O(\log n)$

4.2 Домашнее задание

1. (a) (Группа Мишунина) Рассмотрим бинарную скошенную систему исчисления. На каждой позиции в скошенной записи числа может стоять цифра 0, 1 или 2. Число $a_k a_{k-1} \cdots a_2 a_1$ в скошенной системе переводится в десятичную по формуле $\sum_{i=1}^{k} a_i \cdot (2^i - 1)$.
   В скошенной системе исчисления есть два ограничения: цифра 2 может встречаться в записи не более одного раза; все цифры следующих меньших разрядов равны нулю. Пример первых чисел: 0, 1, 2, 10, 11, 12, 20, 100, 101 . . .
      i. Докажите, что каждое неотрицательное целое число имеет ровно одну возможную запись в скошенной системе исчисления.
      ii. Придумайте, как увеличить число в скошенной системе на единицу за $O(1)$.

   (b) (Группа Мишунина) Рассмотрим структуру данных “скошенный список”. Для того, чтобы получить скошенный список длины n сперва запишем число n в скошенной системе счисления. Далее для каждого i смотрим в соответствующую позицию скошенной записи n и, если соответствующее число не ноль, рисуем столько полных двоичных деревьев высоты i. Пример скошенных списков длины 1, 2, 3, 4, 5 - см. картинку.

Рис. 1: Лист [1]

Рис. 2: Лист [1 2]

Рис. 3: Лист [1 2 3]

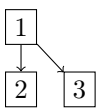


Рис. 4: Лист [1 2 3 4]

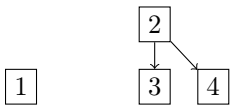
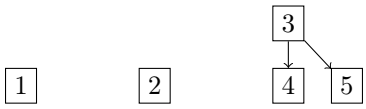


Рис. 5: Лист [1 2 3 4 5]



Придумайте как реализовать следующие операции со списком длины n:

i. Добавление элемента в начало списка за $O(1)$,

ii. Доступ к i-му элементу за $O(logn)$,

iii. Получить список из k последних элементов данного списка за $O(logn)$.

(c) Вам дано дерево из одной вершины. Придумайте, как отвечать на следующие запросы:

- Подвесить новую вершину к дереву — $O(1)$,
- LCA двух вершин — $O(\log n)$,

где n — количество вершин в дереве на данный момент.

2. Дан ориентированный граф, исходящая степень каждой вершины равна единице. Запросы: из вершины v сделать k шагов вперед.

   - Предподсчет: $O(n \log maxK)$, время на запрос: $O(\log k)$.
   - Предподсчет: $O(n \log n)$, время на запрос: $O(\log min(k, n))$.

3. Дано дерево, на вершинах которого могут быть пометки. Запросы: пометить вершину, снять пометку с вершины, число помеченных вершин в поддереве. Предподсчет $O(n)$, время на запрос $O(\log n)$.

4. (Группа Давыдова) Дан лес подвешенных деревьев. Нужно отвечать на следующие запросы:

   (a) подвесить дерево с корнем v к вершине u другого дерева

   (b) LCA вершин u и v.

   Время: $O(\log n)$

5. (Группа Мишунина) Дано дерево из одной вершины. Требуется уметь отвечать на следующие запросы за $O(\log n)$:

   - Подвесить новую вершину u к вершине дерева v и вернуть диаметр дерева.

   Диаметр дерева – это длина самого длинного пути в дереве.

5 Практика 5. Модулярная арифметика

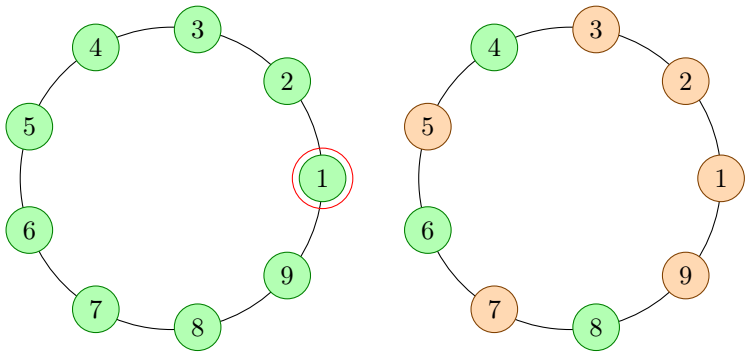
5.1 Практика

1. Дано число n, a и b. Известно, что $a^2 = b^2 \pmod n$, но $a \neq \pm b \pmod n$. Найдите нетривиальное разложение n на множители.

2. Дано множество целых чисел A и параметр b. Известно, что каждое число из A разлогается в произведение первых b простых множителей (т.е. $\forall a \in A\ a = \prod_{i=1}^{b} p_i^{\alpha_i}$). Требуется найти подмножество чисел A, которые в произведении дадут квадрат. Придумайте эффективный полиномиальный алгоритм, для поиска такого подмножества.

3. На кольцевой стоят n светофоров и хитро перемигиваются. Свет на каждом светофоре зажигается в таком порядке: зеленый-желтый-красный, снова зеленый и так по кругу. Все светофоры пронумерованны от 1 до n. Если светофор по номеру i решит поменять свой цвет то следующие, k_i светофоров с шагом a_i тоже поменяют свой цвет. Ниже приведен пример, когда переключается первый светофор: $k_1 = 6$, $a_1 = 2$.

   По начальному состоянию светофоров определите, возможен ли зеленый коридор. Т.е. такая ситуация, когда все светофоры переключены на зеленый. Решить за $O(n^3)$.



4. Для заданных n, k и простого p посчитайте за линейное время $\binom{n}{k} \bmod p$. Учтите, что p может быть меньше, чем n. Можно считать, что все операции в $\mathbb{Z}_p$ выполняются за $O(1)$

5. Есть множество $A \subseteq [n]$. Есть k ячеек, которые умеют хранить $O(\log n)$ бит информации. В каждый момент времени t ячеек могут оказаться недоступны. Требуется организовать такой способ хранения информации, чтобы в каждый момент времени можно было востановить множество A

   - используется менее $(t+1) \cdot |A|$ ячеек,
   - используется $t + |A|$ ячеек.

5.2 Домашнее задание

1. Оцените сложность построения решета Эратосфена на числах до n (число простых чисел меньших n можно оценить как $\frac{n}{\ln n}$.

2. Придумайте, как написать решето Эратосфена за линейное время (считаем, что арифметические операции с числами занимают $O(1)$). Подсказка: для каждого составного числа храните его минимальный простой делитель.

3. Назовем булеву матрицу $m \times n$ красивой, если у каждой ее клетки четное количество ненулевых соседей. Соседями клетки являются четыре клетки соседние с ней по стороне и она сама. Вас просят нарисовать красивую ненулевую матрицу заданного размера (или сообщить, что таких нет). Придумайте, как это сделать за полиномиальное время.

4. Для заданного n и простого p найти число таких $k \in [0, n]$, что $\binom{n}{k} \bmod p = 0$. Можно считать, что все операции по модулю p происходят за $O(1)$. Придумайте алгоритм, который работает

   - $O(n)$,
   - (*) $O(\text{poly}(\log n))$.

5. Дано $n = p \cdot q$, где p и q – простые. За $O(\sqrt{n})$ (полиномом от $\log n$ можно пренебречь) найдите все корни уравнения

   - $x^2 = x \pmod n$,
   - $x^2 = a \pmod n$ для заданного a.

6. (Группа Давыдова) Вам дано дерево из одной вершины. Придумайте, как отвечать на следующие запросы:

   - Подвесить новую вершину к дереву — $O(1)$,
   - LCA двух вершин — $O(\log n)$,

   где n — количество вершин в дереве на данный момент.

7. (Группа Давыдова) На кольцевой стоят n светофоров и хитро перемигиваются. Свет на каждом светофоре зажигается в таком порядке: зеленый-желтый-красный, снова зеленый и так по кругу. Все светофоры пронумерованны от 1 до n. Если светофор по номеру i решит поменять свой цвет то следующие, k_i светофоров с шагом a_i тоже поменяют свой цвет. Ниже приведен пример, когда переключается первый светофор: $k_1 = 6$, $a_1 = 2$.

   По начальному состоянию светофоров определите, возможен ли зеленый коридор. Т.е. такая ситуация, когда все светофоры переключены на зеленый. Решить за $O(n^3)$.

6 Практика 6. RSA

6.1 Практика

1. Перед нами стоит задача: получить случайное простое число в диапозоне от 1 до 2^{128}. Решать ее будем так: выбираем случайное число и запускаем тест Миллера-Рабина. Этот метод не так плох как кажется, т.к. простые числа достаточно часты (среди чисел от 1 до n примерно $\frac{n}{\ln n}$ простых чисел). Известно, что любое составное число проваливает тест Миллера-Рабина с вероятностью $\frac{3}{4}$.

   (a) Сколько раз надо запустить тест Милера-Рабина для того, чтобы эту вероятность увеличить до 90%?

   (b) (Подсказка к предыдущей задаче) Если наступить на ногу динозавру он зарычит с вероятностью 100%, если наступить на ногу человеку — он зарычит с вероятностью 5%. Проведем мысленный эксперимент — вы наступили на ногу своего соседа и он зарычал. С какой вероятностью он динозавр? (Spoiler: динозавры вымерли)

2. Алена шифрует сообщение алгоритмом RSA (n, e, d). Однако алгоритм она помнила плохо и потому вместо $n = pq$, она взяла простое n. Дешифруйте сообщение m^e за $O(poly(\log n))$.

3. Иногда при шифровании RSA $(n = pq, e, d)$ возможно совпадение исходного и зашифрованного текста. Чему равно число таких совпадений?

4. В распоряжении взломщиков оказался волшебный оракул. Для любого открытого ключа (N, e) оракул может взломать 1% из возможных зашифрованных сообщений. Придумайте алгоритм, который взламывает любое сообщение со средним временем работы $O(poly(\log n))$.

5. Дано число $N = p \cdot q$ и $\phi(N)$. Факторизовать N за $O(poly(\log N))$.

6. RSA. Пусть есть N, e и d. Пусть $e = 3$. Разложить N на множители.

7. Алена отправила сообщение m, зашифрованное через RSA, трем людям. Для каждого человека определено свое $N_i = p_i \cdot q_i$, но везде одинаковое $e = 3$ Найдите сообщение Алены за $\texttt{poly}\log N$.

6.2 Домашнее задание

1. Дано $n = p \cdot q$, где p и q – простые. Найдите все корни уравнения

   - (группа Мишунина) $x^2 = x \pmod n$, за $O(polylog(n))$,
   - $x^2 = a \pmod n$ для заданного a, за $O(\sqrt{n}polylog(n))$.

2. (Группа Давыдова) Для заданного n и простого p найти число таких $k \in [0, n]$, что $\binom{n}{k} \bmod p = 0$. Можно считать, что все операции по модулю p происходят за $O(1)$. Придумайте алгоритм, который работает

   - $O(\text{poly}(\log n))$.

   Подсазка: докажите теорему Люка́: $C_n^m \equiv \prod_{i=0}^{k-1} C_{n_i}^{m_i} \pmod n$, где n_i и m_i — i-е цифры в представлении в p-ичной n и m системе счисления соответственно.

3. Алена отправила сообщение m, зашифрованное через RSA, двум людям. Для каждого человека определено свое e_i, но везде одинаковое $N = pq$, Оказалось, что e_i взаимно простые. Найдите сообщение Алены за $\texttt{poly}\log N$.

4. Прочитайте описание протокола обмена ключом Диффи–Хеллмана: `https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%94%D0%B8%D1%84%D1%84%D0%B8_%E2%80%94_%D0%A5%D0%B5%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD%D0%B0`.

   Придумайте, как обобщить его на n человек.

5. (a) Покажите, что зная $p, g, y_i \equiv g^{x_i} \pmod{n}$ мы можем узнать младший бит x_i.

   (b) Рассмотрим такой алгоритм:

   - Вычисляем младший бит x_i
   - Если бит единица — рассмотрим $y_i^{'} = y_i g^{-1}$, иначе $y_i^{'} = y_i$.
   - Вычисляем квадратный корень из $y_i^{'}$ и сводим задачу к задаче меньшего размера.

   Удастся ли таким алгоритмом взломать Диффи–Хеллмана за полиномиальное время?

7 Практика 7. Хеширование

7.1 Практика

1. Сколько различных функций из $\{0,1\}^n$ в $\{0,1\}^m$?

2. Допустим, что мы взяли случайную функцию $f : \{0,1\}^n \leftarrow \{0,1\}m$. Оцените вероятность:

   (a) $f(x) = y$ для фиксированных x и y,

   (b) $f(x) = f(y)$ для фиксированных x и y,

   (c) $f(x)$ отличается от $f(y)$ ровно в одном бите.

   Обратите внимание: и y фиксированные, а вероятность беретсся по функциям.

3. Вася написал 2^n случайных обратимых функций над $\{0,1\}^n$ и договорился с Петей, что будет использовать эти функции для шифрования их переписки. Шифр устроен так: Вася и Петя выбирают два номера функций (это их секретный ключ), затем применяет к сообщению первую функцию, а к ее результату — вторую.

   Женя подглядела случайные функции Васи и теперь пытается взломать его шифр.

   (a) Допустим, что Женя перехватила сообщение и соответствующий ему шифр. Придумайте, как Жене найти подходящий секретный ключ за $O(2^n T)$, где T — время вычисления случайной функции.

   (b) Сколько подходящих ключей найдет Женя в среднем?

   (c) Сколько пар нужно перехватить Жене для того, чтобы она могла найти секретный ключ с вероятностью $\frac{1}{2}$?

4. Пусть дана хэш-таблица размера n с хэш-функцией $h : K \to [n]$. На вход поступает n ключей. Будем предполагать, что хэш-функция отправляет каждый ключ в каждую ячейку независимо с равной вероятностью. Коллизии разрешаются с помощью односвязных списков, цепочек. Посчитаем максимальную длину цепочки.

   (a) Зафиксируем хэш-значение x. Доказать, что вероятность, что k ключей будут иметь хэш x составляет
   $$Q_k = \binom{n}{k} \cdot \left(\frac{1}{n}\right)^k \cdot \left(1 - \frac{1}{n}\right)^{n-k}.$$

   (b) Пусть P_k – вероятность максимальной цепочки иметь длину k. Доказать, что $P_k \le n \cdot Q_k$.

   (c) Вывести из Стрилинга, что $Q_k < (\frac{e}{k})^k$.

   (d) Показать, что для некоторого $c > 1$ верно $Q_k \le \frac{1}{n^3}$ при $k \ge c \cdot \frac{\log n}{\log \log n}$.

   (e) Доказать, что мат.ожидание длины цепочки не превосходит
   $$\mathbf{Pr}\left[M > c \cdot \frac{\log n}{\log \log n} \right] \cdot n + \mathbf{Pr}\left[M \le c \cdot \frac{\log n}{\log \log n} \right] \cdot c \cdot \frac{\log n}{\log \log n},$$
   где M – максимальная длина цепочки. Обратите внимание, что M – случайная величина. Вывести оценку сверху $O(\frac{\log n}{\log \log n})$.

7.2 Домашнее задание

1. В вашем распоряжении есть хешфункция, которая умеет отправлять объекты A и B равномерно в $\{0,1\}^{64}$. Придумайте функцию, которая

   (a) будет отправлять пары (A, B) равномерно в $\{0,1\}^{64}$,

   (b) будет отправлять списки из элементов A равномерно в $\{0,1\}^{64}$,

   (c) будет отправлять мультимножества из элементов A равномерно в $\{0,1\}^{64}$,

   (d) будет отправлять множества из элементов A равномерно в $\{0,1\}^{64}$.

2. Вам дано два корневых дерева (не бинарных), большое T и маленькое t. Требуется найти такую вершину x дерева T, что дерево индуцированное вершиной x и t совпадают.

   - При проверке на совпадение порядок детей важен,
   - При проверке на совпадение порядок детей не важен.

   Решайте в предположении, что у вас есть хорошие хешфункции для списков.

   Вопрос: подойдут ли здесь функции придуманные в предыдущей задаче?

3. Придумайте решение предыдущей задачи без использования хеш-функций.

4. 4c) и 4d) с практики.

5. (группа Давыдова)

   (a) Покажите, что зная $p, g, y_i \equiv g^{x_i} \pmod{p}$ мы можем узнать младший бит x_i.

   (b) Рассмотрим такой алгоритм:

   - Вычисляем младший бит x_i
   - Если бит единица — рассмотрим $y_i^{'} = y_i g^{-1}$, иначе $y_i^{'} = y_i$.
   - Вычисляем квадратный корень из $y_i^{'}$ и сводим задачу к задаче меньшего размера.

   Удастся ли таким алгоритмом взломать Диффи–Хеллмана за полиномиальное время?

41;297;0c

8 Практика 8. Семейства универсальных 2-независимых хеш-функций

8.1 Практика

Ниже есть несколько стандартных определений для семейства хэш-функций. $U(\mathcal{H})$ обозначает равномерное распределение на семействе хэш-функций $\mathcal{H}$. Обратите внимание, что вероятности считаются по случайно взятой хэш-функции.

- Семейство хэш-функций $\mathcal{H} = \{h_i : X \to Y\}_i$ называется универсальным, если для любых $x_1 \neq x_2 \in X$
$$\mathop{\mathbf{Pr}}_{h \leftarrow U(\mathcal{H})} [\ h(x_1) = h(x_2)\ ] \leq \frac{1}{|Y|}$$
.

- Семейство хэш-функций $\mathcal{H} = \{h_i : X \to Y\}_i$ называется 2-независимым, если для любых $x_1 \neq x_2 \in X, y_1, y_2 \in Y$
$$\mathop{\mathbf{Pr}}_{h \leftarrow U(\mathcal{H})} [\ h(x_1) = y_1\ \wedge\ h(x_2) = y_2\ ] = \frac{1}{|Y|^2}$$
.

- Семейство хэш-функций $\mathcal{H} = \{h_i : X \to Y\}_i$ называется k-независимым, если для любых различных $x_1, x_2, \cdots, x_k \in X$, для любых, возможно совпадающих $y_1, y_2, \cdots, y_k \in Y$
$$\mathop{\mathbf{Pr}}_{h \leftarrow U(\mathcal{H})} \left[\bigwedge_{i=1}^{k} h(x_i) = y_i \right] = \frac{1}{|Y|^k}$$
.

1. Докажите, что из любое 2-независимое семейство хэш-функций является универсальным. Приведите пример семейства, которое является универсальным, но не является 2-независимым.

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

3. Построим хэш-функцию $h : \mathbb{F}_2^n \to \mathbb{F}_2^k$ для $k < n$. Зафиксируем матрицу A ранга k над полем $\mathbb{F}_2$ и вектор $b \in \mathbb{F}_2^k$. Пусть
$$h(x) := A \cdot x + b.$$
Найдите I максимальной мощности такое, что $|h(I)| = 1$. Какова мощность I?

4. Используя функцию из задания 3, постройте 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.

5. Пусть есть семейство хэш-функций $\mathcal{H} = \{h_i : X \to [m]\}_i$. Придумайте семейство 2-независимых хэш-функций вида
   - $h_i : X \times X \to [m]$;
   - $h_i : X^k \to [m]$, где k – константа.

6. Определим хэш-функцию $h_{a_0,\ldots,a_{k-1}} := \sum_{i=0}^{k-1} a_i \cdot x^i \mod p$. Покажите, что семейство $\mathcal{H} = \{h_{a_0,\ldots,a_{k-1}} \mid a_i \in \mathbb{Z}_p \text{ при } i < k\}$ является k-независимым.

   Цитата с сайта e-maxx.ru

Один из лучших способов определить хэш-функцию от строки S следующий:
$h(S) = S[0] + S[1] * P + S[2] * P^2 + S[3] * P^3 + \cdots + S[N] * P^N$
где P - некоторое число.

Разумно выбирать для P простое число, примерно равное количеству символов во входном алфавите. Например, если строки предполаются состоящими только из маленьких латинских букв, то хорошим выбором будет $P = 31$. Если буквы могут быть и заглавными, и маленькими, то, например, можно $P = 53$.

Во всех кусках кода в этой статье будет использоваться $P = 31$.

Само значение хэша желательно хранить в самом большом числовом типе - int64, он же long long. Очевидно, что при длине строки порядка 20 символов уже будет происходить переполнение значение. Ключевой момент - что мы не обращаем внимание на эти переполнения, как бы беря хэш по модулю 2^{64}.

Такой хеш называется *полиномиальный*. Иногда на практике используется не 2^{64} а другое значение модуля.

7. Построим набор строк S_i по следующему принципу:

   - $S_1 = 0$
   - $S_2 = 01$
   - $S_3 = 0110$
   - $S_n = S_{n-1} + (notS_{n-1})$

   Допустим, что теперь мы хотим вычислить полиномиальный хеш от этой строки. Докажите, что если результат полиномиального хеширования берется по модулю 2^{64}, то вне зависимости от выбранной константы $hash(S_{10}) = hash(notS_{10})$.

8. Пусть теперь модуль, по которому берется многочлен в полиномиальном хешировании не фиксирован, а тоже выбирается случайно. Докажите, что и такое семейство полиномиальных хеш-функций не является универсальным 2-независимым. Подсказка: Покажите, что существует константа α, что для всех достаточно больших n найдется множество $I \subseteq A^n$ такое, что $|I| \geq |A|^{n-\alpha}$ и $|h(I)| = 1$ ($h(I) = \{h(i) | i \in I\}$). Предложите алгоритм построения такого множества.

8.2 Домашнее задание

1. (Группа Давыдова) Построим набор строк S_i по следующему принципу:

   - $S_1 = 0$
   - $S_2 = 01$
   - $S_3 = 0110$
   - $S_n = S_{n-1} + (notS_{n-1})$

   Допустим, что теперь мы хотим вычислить полиномиальный хеш от этой строки. Докажите, что если результат полиномиального хеширования берется по модулю 2^{64}, то вне зависимости от выбранной константы $hash(S_{10}) = hash(notS_{10})$.

2. Напишите на `Haskell` определение бесконечного списка `Int`-ов, представляющего собой последовательность S_∞ из задачи 7 практики, такое, что:

   - определение содержит не более 128 символов
   - можно использовать стандартные функции из модулей `Data.List` и `Data.Bits`
   - вычисление в нормальную форму `take n` от списка занимает время $\mathcal{O}(n)$.

3. Пусть мы можем потратить не более 100 байт памяти на предподсчет. Нужно научится по 64-битному числу вида 2^k восстанавливать k за $\mathcal{O}(1)$ арифметических операций и единственное чтение одного байта из предподсчитанных данных. Напишите псевдокод предподсчета и ответа на запрос.

4. Пусть имеется фильтр Блума длины m с k функциями. Фильтр Блума можно использовать для подсчета количества различных элементов в множестве. Известно, что матожидание количества бит, выставленных фильтром Блума $n(A)$ связанно с количеством различных элементов в множестве A следующей функцией:

$$|A| = -\frac{m \ln\left(1 - \frac{n(A)}{m}\right)}{k}$$

   Придумайте, как имея фильтры Блума для множества A и B оценить размер пересечения и размер объединения данных множеств.

5. Используя функцию из задания 3 с практики, постройте 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.

9 Практика 9. Преобразование Фурье

9.1 Практика

Примечание: в задачах с этой практики ϵ некоторое положительное число, много меньшее единицы.

1. Пусть есть полином $p(x) = a_0 + a_1 \cdot x^1 + \cdots + a_{n-1} \cdot x^{n-1}$. Известны все значения полинома в точках ω_n^i, где $\omega_n = e^{\frac{2 \cdot \pi \cdot i}{n}}$. На вход подается новое значение a_n. Требуется сдвинуть коэффициенты и посчитать значения для ω_n^i для полинома $p'(x) = a_1 + a_2 \cdot x^1 + \cdots + a_n \cdot x^{n-1}$.

2. Даны две циклические строки p и q. Длины обеих строк равны n. Строки состоят только из символов A, C, G и T. Требуется найти такой циклический сдвиг строки q, чтобы количество позиций, в которых она отличается от p было минимально за $O(n(\log n)^{1+\epsilon})$.

3. Даны две строки p и q. Длины обеих строк не превышают n. Строка q состоит только из символов a и b. Строка p может содержать a, b и "?". Здесь знак "?" выступает в качестве wildcard (т.е. под него можно подставить как a, так b). Найдите все вхождения строки p в строку q за $O(n(\log n)^{1+\epsilon})$.

4. Эльф Леголас — пессимист. Начиная со своего сотого дня рождения каждому дню своей жизни он присваивает рейтинг уныния — действительное число из диапозона от 0 до 1. Жизнь Леголаса была печальна, но насыщена — одни неприятности уступали место другим, и так из года в год. Однако последние k лет у Леголаса появилось стойкое ощущение Déjà vu, ему кажется что неприятности происходясщие с ним крайне напоминают один из предыдущих отрезков его страдальческой жизни, но вот какой именно — он вспомнить не может. Воспользовавшись дневником рейтинга уныния, помогите Леголасу найти такой период в его жизни, что среднеквадратичное отличие рейтинга уныния каждого дня от соответствующего дня последних k лет — минимально за $O(N^2(\log N)^{1+\epsilon})$, где N — возраст Леголаса.

5. Дана ч.б. (без оттенков серого) картинка размера $N \times N$ и образец, размера $K \times K, K \leq N$. Найдите наилучшее вхождение образца в картинку за $O(N^2(\log N)^{1+\epsilon})$. В наилучшем вхождении - минимально суммарное количество различных пикселей.

6. Дано n различных целых точек с координатами по модулю не превосходящими N. Из этих точек случайно выбирается три и на них строится треугольник. Найдите матожидание периметра этого треугольника за $O(N^2(\log N)^{1+\epsilon})$.

9.2 Домашнее задание

Далее мы считаем, что арифметические операции выполняются за $\mathcal{O}(1)$.

1. Даны два множества A и B целых чисел из отрезка $[0, 10 \cdot n]$. Известно, что размеры множеств не превышают n. Требуется найти множество

   $$C = \{a + b \mid a \in A, b \in B\}$$

   за время $\mathcal{O}(n \cdot \log n)$.

2. Дано уравнение: $x^n + y^n \equiv z^n \pmod{\text{m}}$. Найдите количество решений этого уравнения за $\mathcal{O}(m \log (n + m))$

3. По заданным комплексным z_i и положительным целым α_i посчитайте коэффициенты полинома $\prod_i (x - z_i)^{\alpha_i}$. Пусть $n = \sum \alpha_i$. Решите задачу за $\mathcal{O}(n \cdot \log n)$.

4. Матрица $A = (a_{i,j})$ размерности $n \times m$ является Теплицевой, если $a_{i,j} = a_{i+1,j+1}$ для всех $1 \leq i \leq n - 1$ и $1 \leq j \leq m - 1$.

   - Найдите компактное представление матрицы.

- Придумайте быстрый алгоритм для умножения Теплицевой матрицы на вектор за $\mathcal{O}((n+m)\cdot\log(n+m))$.

5. Придумайте, как свести вычисление `FFT` последовательности размера pn к p вычислениям `FFT` от последовательностей размера n и $\mathcal{O}(p^2n)$ дополнительных арифметических операций. Напишите псевдокод.

6. Заданы картина a и образец p в виде матриц вещественных чисел из $[0,1]$ размерами $n\times n$ и $k\times k$ соответственно ($n\geq k$). Требуется найти позицию (x,y), $0\leq x\leq n-k$, $0\leq y\leq n-k$, для которой:
$$\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}(p_{i,j}-a_{(y+i),(x+j)})^2\to\min$$
за время $\mathcal{O}(n^2\log n)$.

10 Практика 10. Линейное программирование

10.1 Практика

1. Пусть Вася умеет решать системы линейных неравенств из k уравнений от n неизвестных за время $LN(n,k)$. Васе дали задачу линейного программирования в стандартной форме с целочисленной матрицей A размера $n \times m$. Докажите, что он сумеет решить ее за $LN(n,k+1)n \log Mn$, где M — максимальное число, встречающееся в матрице , викторе или в минимизируемом функционале.

2. Рассмотрим такую игру с нулевой суммой: дана матрица A, первый игрок выбирает i, второй j. По итогам игры первый платит второму $A_{i,j}$ рублей. Стратегия игрока — это положительный вектор с покомпонентной суммой равной единице. Пользуясь стратегией a игрок выбирает i с вероятностью a_i. Составьте задачу линейного программирования для поиска стратегии минимизирующей максимальный проигрыш (максимум тут по стратегиям второго игрока). Постройте двойственную задачу. Докажите, что данная игра равновесна.

3. Рассмотрим задачу линейного программирования в стандартной форме с матрицей A. Докажите, что точка x является вершиной полиэдра системы тогда и только тогда, когда столбцы матрицы A индуцированные переменными, положительными в точке x — линейно независимы.

4. Матрица M тотально унимодулярна, если любой ее минор равен либо нолю, либо $+-1$. Докажите, что если матрица задачи линейного программирования в стандартной форме тотально унимодулярна, а вектор целый, то полиэдр данной задачи — целый.

5. Пусть полиэдр задачи линейного программирования целый. Докажите, что решение задачи целочисленного линейного программирования с данными ограничениями совпадает с решением задачи линейного программирования.

6. Рассмотрим задачу о маскимальном паросочетание в графе.

   - Сформулируйте эту задачу, как задачу целочисленного линейного программирования.
   - Докажите, что в случае двудольного графа полиэдр полученной задачи — целый. *Верно ли обратное?
   - Какая задача является двойственной к данной?
   - ** Пусть полиэдр не целый. Докажите, что если добавить к изначальным условиям такие: для любого нечетного подмножества вершин количество ребер индуцироанных этим подмножеством не превосходит его поливины, округленной вниз. То полиэдр получится целым.

7. Сведите к задаче линейного программирования задачу:

$$\min_{1 \le i \le p} \left[\sum_{j=1}^{n} c_{ij} x_j \right] \to \max$$

   При ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i \in [1 \dots m]$$

$$x_i \ge 0, i \in [1 \dots n]$$

8. Сведите к задаче линейного программирования задачу:

$$\sum_{i=1}^{p} \left| \sum_{j=1}^{n} c_{ij} x_j - d_i \right| \to \min$$

При ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i \in [1 \dots m]$$

$$x_i \geq 0, i \in [1 \dots n]$$

10.2 Домашнее задание

1. Пусть у нас задан орграф $G = (V, E)$ с двумя выделенными различными вершинами $s, t \in V$, для каждого ребра e которого задано вещественное неотрицательное число c_e — его пропускная способность.

   s-t потоком называется функция $f : E \to \mathbb{R}^+$ такая, что:

   $$\forall e \in E : 0 \leq f_e \leq c_e$$

   $$\forall v \in V \setminus \{s, t\} : \sum_{e=(u,v)} f_e - \sum_{e=(v,u)} f_e = 0$$

   Величиной потока называется:

   $$|f| = \sum_{e=(s,u)} f_e - \sum_{e=(u,s)} f_e$$

   s-t разрезом называется пара $(S, T = V \setminus S)$ подмножеств V такая, что $s \in S$ и $t \in T$. Весом разреза называется величина:

   $$\sum_{e=(u \in S, v \in T)} c_e$$

   - Сведите задачу нахождения максимального s-t потока к задаче линейного программирования.
   - Пусть теперь граф G будет `DAG`-ом, причем любая его вершина лежит на каком-то пути из s в t. Сведите задачу о минимальном s-t разрезе к задаче линейного программирования.

2. (группа Давыдова) Сведите к задаче линейного программирования задачу:

   $$\sum_{i=1}^{p} \left| \sum_{j=1}^{n} c_{ij} x_j - d_i \right| \to \min$$

   При ограничениях:

   $$\sum_{j=1}^{n} a_{ij} x_j = b_i, i \in [1 \dots m]$$

   $$x_i \geq 0, i \in [1 \dots n]$$

3. Пользуясь результатом предыдущей задачи, сведите задачу о минимальном s-t разрезе к задаче линейного программирования и покажите, что она дуальна задаче о максимальном потоке, для произвольного неориентированного графа. Считайте, что неориентированный граф это ориентированный граф, у которого для каждого ребра есть такое же в обратную сторону.

4. В произвольном орграфе сведите задачу о минимальном s-t разрезе к задаче линейного программирования и покажите, что она дуальна задаче о максимальном потоке.

5. Придумайте, как написать линейную программу, для поиска максимума такого функционала:

$$\sum_{i,j \in [1..n]} |c_{i,j}(x_i - x_j)|$$

при условиях:

$$Ax = b$$

$$\forall i : x_i > 0$$

6. (группа Давыдова) Рассмотрим задачу о маскимальном паросочетание в графе.
   - Сформулируйте эту задачу, как задачу целочисленного линейного программирования.
   - Докажите, что в случае двудольного графа полиэдр полученной задачи — целый. *Верно ли обратное?
   - Какая задача является двойственной к данной?

7. `Наследство с предыдущего ДЗ`
   По заданным комплексным z_i и положительным целым α_i посчитайте коэффициенты полинома $\prod_i (x - z_i)^{\alpha_i}$. Пусть $n = \sum \alpha_i$. Решите задачу за $\mathcal{O}(n \cdot (\log n)^2)$.

11 Практика 11. Потоки и разрезы

11.1 Практика

1. Дан взвешенный орграф. Веса неотрицательны. Найдите разрез между s и t минимальной стоимости.

2. Дан двудольный граф. Каждой вершине сопоставлено число a_v.

   - Выберите максимальное количество рёбер так, чтобы степени вершин были не более 1.
   - Выберите максимальное количество рёбер так, чтобы степени вершин были не более a_v.

3. Вам даны суммы элементов матрицы в каждом столбце и каждой строке. Необходимо восстановить матрицу, при условии, что она составлена из целых чисел от 0 до 100.

4. Дан неориентированный граф. Необходимо ориентировать его так, чтобы максимальная исходящая степень была минимальна.

   - $\mathcal{O}(E^2 \log V)$
   - $\mathcal{O}(E^2)$

5. По правилам футбольного турнира в каждом матче должна победить одна из команд, то есть, не бывает 'ничьих'. Вам дана матрица уже сыгранный матчей. Можно ли так доиграть турнир, чтобы каждая команда выиграла заданное число раз?

   - Каждая команда играет с каждой, для каждой команды известно, сколько игр она выиграла.
   - Для каждой команды задано, с какими командами проводятся матчи (то есть, не каждая команда играет с каждой). Итоговое количество очков задано не для всех команд.

6. Даны девочки, мальчики и собачки. Для каждой пары "мальчик, девочка" известно, хочет ли девочка дружить с мальчиком. Для каждой пары "собачка, девочка" известно, нравится ли собачка девочке. Нужно максимальному количеству девочек выделить по мальчику и собачке так, что:

   - Каждый мальчик не более чем с одной девочкой.
   - Каждая собачка не более чем у одной девочки.
   - Тройки гармоничны: девочка и хочет дружить с выбранным ей мальчиком, и собачка ей нравится.

7. Каждой вершине ориентированного графа сопоставлено число (не обязательно положительное) — её вес. Найдите замкнутое подмножество вершин максимальной суммарной стоимости вершин. Подмножество вершин называется замкнутым, если из него не исходят рёбра в другую часть графа.

8. Есть заказы и инструменты. Для каждого заказа известен список инструментов, который нужен, чтобы его выполнить. Каждый инструмент сделан умелыми японскими рабочими, поэтому бесконечно прочный, его можно один раз купить и много раз использовать. У каждого инструмента есть цена p_i. У каждого заказа есть прибыль, которую можно получить, выполнив заказ. Вы – бедный китайский рабочий. У вас изначально нет инструментов, но зато вы можете под нулевой процент в банке взять сколь угодно большой кредит, чтобы купить инструментов.

   - Вопрос: какую максимальную прибыль вы можете получить?
   - А теперь тот же вопрос, но ещё есть разные скидочные предложения!
     Скидка позволяет два инструмента i, j купить по специальной цене d: $\max(p_i, p_j) < d < p_i + p_j$. Каждый инструмент присутствует не более чем в одном скидочном предложении.

11.2 Домашнее задание

1. Дан граф и выделенные вершины s, t. Нужно проверить, правда ли существует единственный минимальный s-t разрез.

   - $\mathcal{O}(Poly(V, E))$
   - $\mathcal{O}(E)$ при условии, что нам уже известен максимальный поток (с доказательством).

2. В неориентированном графе без кратных рёбер необходимо удалить минимальное число рёбер так, чтобы увеличилось количество компонент связности. $\mathcal{O}(V \cdot Flow)$. Оценить время работы того же алгоритма более точно как $\mathcal{O}(E^2)$.

3. Есть ориентированный граф с начальной и конечной вершинами. В начальной вершине есть K грузовиков. Грузовикам нужно попасть в конечную вершину. Время дискретно. За единицу времени каждый грузовик или стоит на месте, или перемещается в одну из соседних вершин. В любой вершине могут одновременно стоять несколько грузовиков. По любому из рёбер в каждый момент времени должен ехать не более чем один грузовик. Минимизируйте время, когда все грузовики окажутся в конечной вершине.

   - $\mathcal{O}(Poly(V, E, K))$
   - $\mathcal{O}(K(V + K)E)$

4. Есть n рабочих и m работ. И есть матрица умения: "какой рабочий какие работы умеет делать". Нужно максимально равномерно распределить работы между рабочими. То есть, каждой работе сопоставить рабочего, который умеет делать эту работу, а кроме того минимизировать $\max\limits_{i=1..n} k_i$, где k_i – количество работ, выданных i-му рабочему.

5. Из предыдущего домашнего задания мы знаем, что задача о минимальном s-t разрезе дуальна задаче о максимальном s-t потоке. Также, мы умеем находить в двудольном графе максимальное паросочетание с помощью максимального потока.

   - Какая задача дуальна задаче о максимальном паросочетании в двудольном графе?
   - Придумайте, как за полиномиальное время в двудольном графе найти максимальный по числу вершин подграф, являющийся полным двудольным графом.

12 Практика 12. Строки

12.1 Практика

Если задача решается хеш-функциями, поясните, какого порядка модуль стоит выбирать.

1. Дана строка s. Найти наидлиннейшую подстроку p, встречающуюся в s хотя бы два раза (возможно вхождения перекрываются). Время работы $O(n \log n)$, где n — длина строки s.

2. Дана картинка размера $m \times n$ и прямоугольный образец меньшего размера. Найти все вхождения образца в картинке за $O(m \cdot n)$.

3. Дана строка s длины n. Требуется посчитать количество её различных подстрок за (n^2).

4. Дана строка s и образец p. Проверить, что образец p входит в строку s. Допускаются вхождения с не более чем одной опечаткой (заменой символа). Время работы $O(|s| + |p|^2)$.

5. Дана строка s длины n. Требуется найти самое короткое её "сжатое"представление, т.е. найти такую строку t наименьшей длины, что s можно представить в виде конкатенации одной или нескольких копий t. Время работы: $O(n)$.

6. Найти все подпалиндромы заданной строки. Время работы $O(n \log n)$, где n — длина строки.

7. Дана строка s и образец p. Проверить, что образец p входит в строку s. Допускаются вхождения с не более чем одной опечаткой (заменой символа). Время работы $O(|s| + |p|)$.

8. Найти все подпалиндромы заданной строки. Время работы $O(n)$, где n — длина строки.

12.2 Домашнее задание

1. (группа Давыдова) Дана строка s длины n. Требуется найти самое короткое её "сжатое"представление, т.е. найти такую строку t наименьшей длины, что s можно представить в виде конкатенации одной или нескольких копий t. Время работы: $O(n)$.

2. (группа Давыдова) Найти все подпалиндромы заданной строки. Время работы $O(n \log n)$, где n — длина строки.

3. Найти подстроку в тексте. При сравнении строк, если несовпадений было не более двух, строки считаются равными. $\mathcal{O}(n)$.

4. За $\mathcal{O}(n)$ восстановить строку, если дана ее

   - Z-функция.
   - префикс-функция.

   Напишите псевдокод. Если строка не единственная, можно вывести любую.

5. Алгоритм кодирования LZSS. Дана строка s. Выписываем её слева направо. Пусть уже выписан префикс $[0, i)$. Можно или, потратив 1 доллар, записать в код строки s_i и выписать i-й символ, или, потратив 5 долларов, записать в код строки (j, len) и выписать сразу len символов. Здесь $j < i$, а $s[j : j+len) = s[i : i+len)$. Ваша задача — за $\mathcal{O}(n^2)$ выписать всю строку за минимальную стоимость.

6. Найти подстроку в тексте. При сравнении строк можно делать циклический сдвиг алфавита в одной из них.

   - $\mathcal{O}(n|\Sigma|)$

- $\mathcal{O}(n)$

7. Нужно за $\mathcal{O}(n \log n)$ найти максимальный общий подпалиндром.
8. Найти строку над алфавитом $\{0, 1\}$, в которой $\Omega(n^2)$ различных подстрок.

13 Практика 13. Суффиксные массивы

13.1 Практика

Тут $n \le SA(n) \le n \log n$ — время построение суффиксного массива и lcp для строки длины n

1. Найти наименьший циклический сдвиг строки за $\mathcal{O}(SA(n))$
2. Требуется за $\mathcal{O}(1)$ сравнивать подстроки строки. Разрешается произвести препроцессинг за $\mathcal{O}(SA(n))$.
3. Посчитать количество различных подстрок за $\mathcal{O}(SA(n))$
4. Найти самую длинную подстроку, входящую в заданную дважды:
   (a) вхождения могут пересекаться
   (b) вхождения не могут пересекаться

   за $\mathcal{O}(SA(n))$
5. Строка s называется максимальным повтором в t, если
   - s входит в t не менее двух раз.
   - Если r входит в t не менее двух раз, то s — не является собственной подстрокой r.

   Найти все максимальные повторы за $\mathcal{O}(SA(n))$,
6. Найти наибольшую общую подстроку двух строк за $\mathcal{O}(SA(n))$
7. Найти наибольшую общую подстроку k строк за $\mathcal{O}(kSA(n))$

13.2 Домашнее задание

1. Придумайте, как найти самый длинный рефрен — такую s, что $count(s) \cdot |s| \to \max$ за $O(SA(n))$
2. Придумайте, как реализовать, используя суффиксный массив, оптимальное сжатие `LZSS` за $\mathcal{O}(n \log n)$.
3. У вас были строка $s_0 s_1 \dots s_{n-1}$ длины n и ее суффиксный массив — массив позиций начал суффиксов, отсортированных в лексикографическом порядке. Символы строки — натуральные числа, не превосходящие n. Под покровом темноты подлые враги прокрались и стерли из массива все числа, делящиеся на 3, так, что даже дырок не осталось! Опишите алгоритм, восстанавливающий суффиксный массив за время $\mathcal{O}(n)$.
4. Постройте такую строку s, что её суффиксный массив совпадает с данным за $\mathcal{O}(n)$.
5. Найдите усредненное по всем парам суффиксов значение `LCP` строки длины n с помощью суффиксного массива за $\mathcal{O}(n)$.
6. Дан словарь слов суммарной длины L, постройте структуру, позволяющую отвечать на запросы типа `get(s, t)` — кол-во строк в словаре, которые начинаются с s, заканчиваются на t за время $\mathcal{O}(\log^p L + |s| + |t|)$ (p — константа). Время на построение — $\mathcal{O}(L \log^p L)$.