Практика по алгоритмам

Алексей Давыдов, Александр Мишунин, Михаил Слободкин*

Весна, 2017

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

1 Практика 1. Остовные деревья

1.1 Практика

1. Пусть дан взвешенный связный неорграф $G = \langle V, E \rangle$ с выделенной вершиной s. Все веса положительны и различны. Могут ли какое-либо минимальное покрывающее дерево в G и какое-либо дерево кратчайших путей из s не иметь ни одного общего ребра? Если да, приведите пример. Если нет, докажите, что такого не может быть.

2. В стране n аэропортов. Самолет может сделать перелет из аэропорта i в аэропорт j, израсходовав w_{ij} горючего. При этом $w_{ij} = w_{ji}$, и $w_{ii} = 0$. Требуется найти минимальный размер бака, позволяющий добраться самолету из любого города в любой, возможно с дозаправками. Решить за $\mathcal{O}(n^2)$.

3. Рассмотрим следующий алгоритм поиска минимального покрывающего дерева (алгоритм Борувки):

   - Пока в графе больше одной вершины:
     - Для каждой вершины найдем самое легкое инцедентное ей ребро и добавим его в множество S (одно и то же ребро может быть выбрано дважды).
     - Добавим все ребра из множества S в ответ.
     - Стянем граф по ребрам из S.

   Докажите, что такой алгоритм найдет минимальное покрывающее дерево в случае, если веса всех ребер в графе различны, при этом время работы будет $\mathcal{O}(E \log V)$.

   Придумайте, как модифицировать алгоритм, если возможны равные веса.

4. (a) Постройте пример для алгоритма Борувки, на котором он делает $\Theta(\log n)$ фаз.

   (b) Постройте пример для алгоритма Борувки, на котором он делает $\Theta(1)$ фаз.

5. Найти во взвешенном неорграфе такой цикл, что максимальный вес ребра этого цикла минимален. $\mathcal{O}((V + E) \log E)$.

6. Дан взвешенный связный неориентированный граф $G = \langle V, E \rangle$ и некоторое минимальное остовное дерево на нём. Пусть некоторого ребра $e \in E$ изменился вес. По графу, остовному дереву, ребру e и его новому весу найдите новое минимальное остовное дерево за $\mathcal{O}(V + E)$.

7. Пусть все ребра графа имеют различный вес. Докажите, что минимальное покрывающее дерево единственно.

8. Пусть даны взвешенный неорграф с неединственным минимальным остовным деревом и какой-то из его минимальных остовов. Найдите минимальный остов графа, отличный от данного. $\mathcal{O}(E \log V)$.

9. Проверить, что минимальное по весу остовное дерево единственно. $\mathcal{O}(E \log V)$.

10. Найдите за полиномиальное время второе по весу остовное дерево в неографе.

11. Дан взвешенный орграф, постройте ориентированное к корню остовное дерево с корнем в вершине 1 минимального веса.

1.2 Домашнее задание

1. По неориентированному графу и его минимальному остовному дереву найдите второе остовное дерево (т.е. самое легкое остовное дерево из несовпадающих с данным в условиях минимальным; деревья сравниваются как множества ребер).

   (a) $\mathcal{O}(V^2 + E)$.

   (b) $\mathcal{O}(E \log V)$.

2. Пусть дан связный взвешенный неорграф, будем рассматривать его ребра в порядке невозрастания веса и удалять текущее ребро, если связность графа при этом не нарушается. Докажите, что этот алгоритм находит минимальный остов, или придумайте контрпример.

3. Докажите, что максимальный вес ребра на пути между парой вершин в минимальном остове графа G не зависит от выбора конкретного минимального остова G.

4. Пусть в связном неорграфе с положительными весами выделено подмножество вершин, называемых терминалами. Дерево Штейнера это связный подграф исходного графа минимального веса, содержащий все терминалы.

   (a) Покажите, что на полном графе с неравенством треугольника данный алгоритм является 2-приближением для дерева Штейнера: удалим все вершины, кроме терминалов, и возьмем минимальный остов.

   (b) Придумайте 2-приближенный алгоритм для дерева Штейнера для произвольного связного неорграфа. $\mathcal{O}(V^3)$.

Дополнительные задачи

5. **Второе остовное дерево lvl. 2**
   Дан взвешенный связный неорграф G, вес его минимального остова равен w. Найдите за полиномиальное время минимальный по весу среди таких остовов G, вес которых строго превосходит w.

6. Есть взвешенный неорграф. Найти путь из s в t такой, что сумма трёх максимальных ребер на пути минимальна. $\mathcal{O}((n+m) \cdot poly(\log))$.

2 Практика 10. AVL деревья

2.1 Практика

1. Пусть вам даны два AVL-дерева T_1 и T_2. Придумайте, как построить AVL-дерево T, являющееся объединением деревьев T_1 и T_2 за время:
   (a) $\mathcal{O}(h(T_1) \times size(T_2))$
   (b) $\mathcal{O}(size(T_1) + size(T_2))$
2. Дано двоичное дерево, в котором AVL-свойство выполнено везде, кроме корня.
   (a) Пусть `root.l.h = root.r.h + 3`. Как восстановить AVL-свойство всюду за $\mathcal{O}(1)$?
   (b) Пусть `root.l.h = root.r.h + k`. Как восстановить AVL-свойство всюду за $\mathcal{O}(k)$?
3. Придумайте `merge` для AVL-деревьев за $\mathcal{O}(\log n)$.
4. Придумайте `split` для AVL-деревьев за $\mathcal{O}(\log^2 n)$.
5. Докажите, что придуманный `split` на самом деле работает за $\mathcal{O}(\log n)$.
6. Пусть дано AVL-дерево T, ключ k и число n. Придумайте алгоритм, который выведет n элементов, следующих за k, за время $\mathcal{O}(h(T) + n)$.
7. Уменьшите количество дополнительной информации в AVL-дереве до двух бит на вершину.
8. Придумайте структуру данных, позволяющую online за $\mathcal{O}(\log n)$ отвечать на следующие запросы:
   - добавить элемент x
   - удалить элемент x
   - найти i-ый по порядку элемент
   - найти порядок элемента x
9. Задача "вставка ключа": изначально все ячейки пусты, требуется online за $\mathcal{O}(\log n)$ обрабатывать запросы: 1) `Insert(i, x)` — вставить x в ячейку i. Если i-я ячейка занята, все ячейки, начиная с i-й, сдвигаются вправо; 2) Узнать элемент на позиции i
   Пример: $(5,1); (5,2); (5,3); (1,7); (1,8); (2,9) \to 8, 9, 7, 0, 3, 2, 1$.
10. Запросы online за $\mathcal{O}(\log n)$:
    - добавить пару $\langle x, y \rangle$
    - удалить пару $\langle x, y \rangle$
    - посчитать сумму y по всем парам таким, что $l \le x \le r$
11. Запросы online за $\mathcal{O}(\log n)$:
    - добавить пару $\langle x, y \rangle$
    - посчитать сумму y по всем парам таким, что $l \le x \le r$
    - посчитать сумму x по всем парам таким, что $l \le y \le r$
12. Запросы online за $\mathcal{O}(\log n)$:
    - `add(i, x)`
    - `del(i)`
    - `add(l, r, value)` — добавить *value* ко всем x, для которых $l \le i \le r$
    - `sum(l, r)` — сумма всех x, для которых $l \le i \le r$

2.2 Домашнее задание

1. Вершина дерева является *единственным ребенком*, если у ее родителя только один ребенок (корень единственным ребенком не является). Определим для дерева степень одиночества LR таким образом:

   $LR(T) =$ (Количество единственных детей в T)/(Количество вершин в T).

   (a) Покажите, что в любом ненулевом AVL-дереве T $LR(T) \leq \frac{1}{2}$.

   (b) Правда ли, что если $LR(T) \leq \frac{1}{2}$, то $h(T) \leq \log size(T)$?

   (c) Пусть в дереве T есть всего $\Theta(size(T))$ единственных детей и все они — листья. Правда ли, что для любого такого дерева $h(T) \leq \log size(T)$?

2. Покажите, что:

   (a) Добавление в AVL-дерево требует $\mathcal{O}(1)$ вращений.

   (b) Удаление из AVL-дерева может потребовать $\Omega(\log n)$ вращений.

3. Пусть в обычное несбалансированное бинарное дерево поиска добавляются различные элементы в порядке $x_1, x_2, \ldots, x_n$. Придумайте алгоритм, поддерживающий для этого дерева массив из отцов всех его вершин и реализующий добавление элемента в дерево вместе с обновлением массива отцов за $\mathcal{O}(\log n)$.

4. Придумайте, как реализовать структуру данных, поддерживающую непрерывную последовательность элементов со следующими операциями:

   - вставка элемента на i-ю позицию (если она занята, элементы, начиная с i-го, сдвигаются),
   - получение элемента, стоящего на i-й позиции,
   - развернуть кусок последовательности длины k, начинающийся с элемента i, задом наперед.

   Время обработки операций — $\mathcal{O}(\log n)$.

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

   - Обмен местами последовательных пар 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 — $\mathcal{O}(\log n)$ на запрос.

6. (группа Мишунина) Придумайте структуру данных на основе AVL дерева, позволяющую online за $\mathcal{O}(\log n)$ отвечать на следующие запросы:

   - `add(i, x)`
   - `del(i)`
   - `add(l, r, value)` — добавить по модулю 5 $value$ ко всем x, для которых $l \leq i \leq r$
   - `sum(l, r)` — сумма всех x, для которых $l \leq i \leq r$

Дополнительные задачи

7. Напишите на каком-нибудь строго типизированном функциональном языке реализацию добавления в AVL дерево так, чтобы AVL-инвариант гарантировался типизацией (вероятно, для этого потребуется язык с зависимыми типами).

8. Постройте тест, на котором недо-AVL-дерево, которое делает только малые вращения, делает $\Theta(n^2)$ операций после n запросов.

   Формально: изначально дерево пусто, нужно n раз вызвать `add(root,` x_i`)` для некоторой последовательности x_i, что суммарное время работы $\Theta(n^2)$.

   Либо докажите, что такого теста нет.

3 Практика 3. Splay ∧ Декартовы деревья

3.1 Практика

1. Покажите, что любые два корректные дерева поиска, построенные на одном и том же множестве ключей, можно получить друг из друга последовательностью поворотов.
2. Мальчик Петя взял пары (x_i, y_i) и построил на них декартово дерево. Приведите пример, на котором полученное дерево не является статически оптимальным для ключей x_i и их частот y_i.
3. Нарисуйте все Декартовы деревья, которые могут получится в результате операции `merge(бамбук идущий влево-вниз, вершина)` и `merge(бамбук идущий вправо-вниз, вершина)`.
4. Придумайте, как построить за линейное время Декартово дерево по отсортированному по ключу массиву пар.
5. Модифицируйте Декартово дерево так, чтобы `merge` с одним элементом работал за амортизированные $\mathcal{O}(1)$.
6. Пусть приоритеты случайны. Покажите, что наличие одинаковых ключей не портит оценку на матожидание высоты Декартова дерева для операции `insert`.
7. Попробуем улучшить константу у стандартного `insert` через `split` и `merge` следующим образом: сначала будем спускаться по девеву, пока не встретим узел с меньшим, чем у нового, приоритетом, и уже в это поддерево сделаем стандартный `insert`. В результате всегда получится корректное Декартово дерево.

   Пусть теперь приоритеты случайны, а ключи могут быть одинаковыми. Заметим, что при спуске у нас есть выбор, в какое поддерево идти при равенстве ключей. Влияет ли этот выбор на оценку матожидания высоты дерева?
8. Попробуем сделать Декартово дерево со случайными приоритетами персистентным. Сохранится ли хорошая оценка на высоту?
9. Попробуем сделать Splay дерево частично персистентным. Правда ли, что можно амортизационно оценить операции с деревом за $\mathcal{O}(\log n)$, а операцию возврата к предыдущему состоянию за $\mathcal{O}(1)$?
10. Дан невыпуклый несамопересекающийся многоугольник из n вершин.
    (a) Даны m точек. За $\mathcal{O}((m+n)\log n)$ для каждой точки определить, внутри она или снаружи.
    (b) Используя предподсчёт за $\mathcal{O}(n \log n)$ научиться в online по точке за $\mathcal{O}(\log n)$ определять, внутри она или снаружи.
11. Допустим, что мы хотим использовать декартово дерево как кучу (т.е. случайными будут не приоритеты, а ключи). Оцените время работы операций на такой куче. Можно ли слить две декартовы кучи за логарифмическое время?
12. Рассмотрим такую небинарную кучу:
    - *find-min* — вернуть верхний элемент
    - *merge* — сделать дерево с большим корнем самым левым ребенком дерева с меньшим корнем
    - *insert* — *merge* с деревом из одного элемента
    - *delete-min* — удалить корень и слить остальные элементы таким образом: идем слева направо, сливая детей попарно (1-2, 3-4, 5-6 и т. д.). Потом идем справа налево и сливаем все деревья в одно.

    Для каждого поддерева здесь выполняется свойство кучи. Докажите, что все операции можно амортизационно оценить как $\mathcal{O}(\log n)$

3.2 Домашнее задание

1. Пусть `splay`-дерево поддерживает множество S. Каждый элемент $x_i \in S$ был запрошен $p_i m$ раз, где m — общее число запросов. Гарантируется, что $0 < p_i \leq 1$ и $\sum_i p_i = 1$. Докажите, что `splay`-дерево обрабатывает все запросы за время $\mathcal{O}(m \cdot \left[1 + \sum_i p_i \cdot \log \frac{1}{p_i}\right])$.

2. Придумайте структуру данных, поддерживающую упорядоченный список S целых чисел, которая умеет отвечать на запросы:

   - `insert(x)` — вставить x в S, если его там не было.
   - `delete(x)` — удалить x из S, если он там был.
   - `S[k]` — вернуть k-тый по порядку элемент из S.
   - `max(l, r)` — найти $\max_{l \leq j < k \leq r} |S[j] - S[k]|$. Гарантируется $r - l \geq 1$.
   - `min(l, r)` — найти $\min_{l \leq j < k \leq r} |S[j] - S[k]|$. Гарантируется $r - l \geq 1$.

   Каждый запрос должен обрабатываться за $\mathcal{O}(\log |S|)$.

3. Пусть есть произвольное `splay`-дерево, построенное на ключах $1, 2, 3 \ldots n$. Покажите, что в результате последовательных вызовов `splay(1)`, `splay(2)`, $\ldots$ `splay(n)` получится бамбук.

4. Пусть приоритеты случайны, а ключи все разные. Найдите матожидание количества листьев в Декартовом дереве из n вершин (точно, а не асимптотически).

5. Пусть приоритеты случайны, а ключи все разные. Обозначим за x_k вершину Декартова дерева, содержащую k-тый по порядку ключ. Всего в дереве n вершин.

   (a) Пусть $1 \leq i \leq j \leq k \leq n$. Найдите вероятность того, что x_j — общий предок x_i и x_k.

   (b) Пусть $1 \leq i \leq k \leq n$. Найдите матожидание длины пути между x_i и x_k.

   Ответ достаточно представить в виде суммы ряда.

6. Т.к. обычно приоритеты в Декартовом дереве случайны, то их хранение выглядит избыточным. Попробуем их не хранить, а во время операции `merge` будем делать следующее: если в дереве A n_A элементов, а в дереве B n_B элементов, то с вероятностью $\frac{n_A}{n_A+n_B}$ в качестве корня будет выбран корень дерева A, а с вероятностью $\frac{n_B}{n_A+n_B}$ в качестве корня будет выбран корень дерева B (далее слияние происходит аналогично слиянию в Декартовом дереве).

   Оцените матожидание глубины вершины в таком дереве.

Дополнительные задачи

7. Придумайте аналог Декартова дерева со случайными приоритетами для хранения точек на плоскости с операциями `splitX`, `mergeX`, `splitY`, `mergeY`. Все операции должны работать за $o(n)$.

8. Докажите логарифмическую верхнюю оценку на матожидание `максимальной` глубины вершины в Декартовом дереве со случайными приоритетами.

9. `C-c-c-combo!`
   Напишите псевдокод для структуры, умеющей отвечать на запросы:

   - `add(i, x)`
   - `del(i)`
   - `add(l, r, value)` — добавить *value* ко всем x, для которых $l \leq i \leq r$

- `set(l, r, value)` — установить в *value* все x, для которых $l \le i \le r$
- `sum(l, r)` — сумма всех x, для которых $l \le i \le r$
- `reverse(l, r)` — изменить порядок всех x, для которых $l \le i \le r$, на обратный

Время работы всех запросов $\mathcal{O}(\log n)$, online.

4 Практика 100. Деревья и пути

4.1 Практика

1. Даны подвешенное дерево и его Эйлеров обход. Придумайте, как за $\mathcal{O}(\log n)$ обновить Эйлеров обход при переподвешивании дерева за другую вершину (для разных вариантов обхода).

2. Дано дерево, на вершинах которого могут быть пометки. Запросы: пометить вершину, снять пометку с вершины, число помеченных вершин в поддереве. $\langle \mathcal{O}(n), \mathcal{O}(\log n) \rangle$.

3. Дано дерево с весами на ребрах, нужно online обрабатывать запросы:

   - Изменить вес ребра e.
   - Вывести вес простого пути между данной парой вершин.

   $\mathcal{O}(n)$ на предобработку, $\mathcal{O}(\log n)$ на запрос.

4. Дерево с весами в вершинах. Нужно научиться online отвечать на запрос $count(a, b, x)$ — количество вершин на пути из a в b, у которых вес $\geq x$, если:

   (a) Веса не меняются. $\mathcal{O}(\log n)$.

   (b) Теперь веса меняются. Запрос за $\mathcal{O}(\log^2 n)$, изменение веса за $\mathcal{O}(\log^2 n)$.

5. Дан лес подвешенных деревьев. Нужно отвечать на следующие запросы:

   - Подвесить дерево с корнем v к вершине u другого дерева.
   - Отрезать поддерево с корнем в вершине v от ее дерева.
   - Проверить, в одном ли дереве лежат вершины u и v.

   Время: $\mathcal{O}(\log n)$.

6. Назовем ребро дерева тяжелым, если размер поддерева, его нижнего конца больше либо равен половине размера поддерева его верхнего конца. Покроем дерево путями следующим способом: из каждой вершины, в которую не входит ни одного тяжелого ребра будем идти вверх, пока не дойдем до корня, или пока не пройдем по легкому ребру. Такое покрытие называется «Heavy-Light-Decomposition». Докажите, что путь из любой вершины в корень пересекает не более чем $\log n$ путей из покрытия, где n — количество вершин.

7. Дано дерево, у каждой вершины есть вес. Запросы: изменить вес вершины; найти максимум на пути из u в v. $\langle \mathcal{O}(n \log n), \mathcal{O}(\log^2 n) \rangle$.

8. Дано дерево с весами на ребрах, надо эффективно обрабатывать запросы:

   (a) $\min(a, b)$ — минимум на пути из вершины a в вершину b.
   $set(e, x)$ — присвоение ребру e веса x.

   (b) $\min(a, b)$ — минимум на пути из вершины a в вершину b.
   $add(a, b, x)$ — прибавление числа x к весам всех ребер на пути из вершины a в вершину b.

9. Дан невыпуклый несамопересекающийся многоугольник из n вершин.

   (a) Даны m точек. За $\mathcal{O}((m+n) \log n)$ для каждой точки определить, внутри она или снаружи.

   (b) Используя предподсчёт за $\mathcal{O}(n \log n)$ научиться в online по точке за $\mathcal{O}(\log n)$ определять, внутри она или снаружи.

4.2 Домашнее задание

1. Дано дерево из n вершин, по которому бегают муравьи. Вам поступает m запросов вида a, b, что означает, что очередной муравей пробежал из вершины a в вершину b.

   (a) Выведите, сколько раз муравьи пробежали через каждое из ребер за $\mathcal{O}(n+m)$ (Можно считать, что мы умеем решать `LCA` за $\langle \mathcal{O}(n), \mathcal{O}(1) \rangle$).

   (b) А теперь вам онлайн поступают вопросы про каждое из ребер (эти запросы могут поступить в любой момент, в том числе до окончания муравьиных пробегов). Обработайте каждый из двух типов запросов за $\mathcal{O}(\log^2 n)$ с предподсчетом за $\mathcal{O}(n \log n)$.

   (пункты "a"и "b"здесь считаются независимыми задачами)

2. Дан взвешенный неорграф. Найдите максимальный по длине отрезок $[l, r]$ весов ($W_{\min} \le l \le r \le W_{\max}$) такой, что множество рёбер с весами из (l, r) не содержит циклов. $\mathcal{O}((E+V) \log V)$.

3. Дан невыпуклый несамопересекающийся многоугольник из n вершин.

   (a) Даны m точек. За $\mathcal{O}((m+n) \log n)$ для каждой точки определить, внутри она или снаружи.

   (b) Используя предподсчёт за $\mathcal{O}(n \log n)$ научиться в online по точке за $\mathcal{O}(\log n)$ определять, внутри она или снаружи.

   Подсказка: используйте сканирующую прямую.

4. Дан лес с положительными целочисленными весами на ребрах. Используя *Euler Tour Tree* научитесь обабатывать следующие запросы:

   - `link`(a, b, w) – провести ребро веса w между вершинами a и b (a и b гарантированно принадлежат разным деревьям)
   - `cut`(a, b) – удалить ребро
   - `sum`(a, b) – сумма весов рёбер на пути

   Все запросы нужно обрабатывать за $\mathcal{O}(\log n)$ online.

Дополнительные задачи

5. Придумайте модификацию Heavy-Light-Decomposition, поддерживающую добавлении новых листьев.

   (a) $\mathcal{O}(\sqrt{n})$ на добавление листа, $\mathcal{O}(\sqrt{n} + \log^2 n)$ на запрос `get`.

   (b) Добавление за $\mathcal{O}(\log^2 n)$. Подсказка: мы хотим быстро *split*-ить и *merge*-ить пути.

5 Практика 101. Хеширование

5.1 Практика

1. Сколько существует различных функций из $\{0,1\}^n$ в $\{0,1\}^m$?
2. Придумайте для строки длины n предподсчет за $\mathcal{O}(n)$, позволяющий:
   (a) Вычислить за $\mathcal{O}(1)$ полиномиальный хеш любой подстроки.
   (b) Лексикографически сравнить любые две подстроки за $\mathcal{O}(\log n)$.
3. Найдите все периоды строки. $\mathcal{O}(n)$.
4. (a) Найти число различных подстрок в строке. $\mathcal{O}(n^2)$.
   (b) Найти подстроку данной строки, встречающуюся максимальное число раз.
5. Найти k-й в лексикографическом порядке суффикс в строке.
   (a) $\mathcal{O}(n \log^2 n)$.
   (b) $\mathcal{O}(n \log n)$.
6. Определим строку Туэ-Морса:
   - $S_1 = 0$
   - $S_2 = 01$
   - $S_3 = 0110$
   - $S_n = S_{n-1}(\neg S_{n-1})$

   Заметим, что каждая строка является префиксом всех следующих. Обозначим как S_∞ строку из $\{0,1\}^\infty$, содержащую каждую S_i как префикс. Пусть требуется вычислить полиномиальный хеш от S_n для некоторого n. Докажите, что если результат полиномиального хеширования вычисляется по модулю 2^{64}, то вне зависимости от выбранной константы $hash(S_{12}) = hash(\neg S_{12})$.
7. Научиться искать образец в строке, если допустимо различие в один символ между образцом и найденной подстрокой. $\mathcal{O}(n \log m + m)$.
8. Найти наибольшую по длине строку, которая дважды без перекрытий встречается в заданной строке. $\mathcal{O}(n \log n)$.
9. (a) Найти количество подпалиндромов строки. $\mathcal{O}(n \log n)$.
   (b) Найти максимальный подпалиндром строки. $\mathcal{O}(n)$.
10. Дано начальное состояние в игре «Жизнь» в виде битовой матрицы $N \times N$. Нас интересует, сколько останеться живых клеточек на поле после большого количества итераций $T \gg N$. Для простоты можно считать N и T степенями двойки.
    (a) Определим функцию e, сопоставляющую тайлу размера $2^n \times 2^n$ кусок размера $2^{n-1} \times 2^{n-1}$ из его середины на 2^{n-2} шагов в будущем. Можно ли написать рекуррентное соотношение для e?
    (b) Придумайте, как с помошью хеширования решить нашу задачу быстрее, чем наивной симуляцией.

5.2 Домашнее задание

1. Семейство хэш-функций $\mathcal{H} = \{h : X \to Y\}$ называется универсальным, если:

$$\forall x_1, x_2 \in X, x_1 \neq x_2 : \mathop{\mathbf{Pr}}_{h \in \mathcal{H}} [\, h(x_1) = h(x_2) \,] \leq \frac{1}{|Y|}$$

   Семейство хэш-функций $\mathcal{H} = \{h : X \to Y\}$ называется k-независимым, если для любых различных $x_1, x_2, \cdots, x_k \in X$, для любых, возможно, совпадающих $y_1, y_2, \cdots, y_k \in Y$ выполняется:

$$\mathop{\mathbf{Pr}}_{h \in \mathcal{H}} \left[\bigwedge_{i=1}^{k} h(x_i) = y_i \right] = \frac{1}{|Y|^k}$$

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

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

2. Найдите за $\mathcal{O}(n \log n)$ максимальный общий подпалиндром двух строк.

3. Найдите в строке длины n за $\mathcal{O}(n)$ самую длинную подстроку, которая представима как степень некоторой строки длины k. Степенью строки называют её конкатенацию с собой несколько раз.

4. Будем называть два дерева изоморфными, если на каждой глубине первого дерева можно так перенумеровать вершины, что рёбра совпадут с рёбрами второго дерева.

   Пусть даны два корневых дерева, большое T и маленькое t. Требуется найти такую вершину x дерева T, что поддерево, индуцированное вершиной x, и t изоморфны.

   (a) При проверке на изоморфизм порядок детей важен, требуемое время работы $\mathcal{O}(n)$.

   (b) При проверке на изоморфизм порядок детей не важен, требуемое время работы $\mathcal{O}(n \log(n))$.

   Можно считать, что даны идеальные хеш-функции для списков из натуральных чисел.

Дополнительные задачи

5. Напишите на `Haskell` определение бесконечного списка `Int`-ов, представляющего собой последовательность Туэ-Морса (последовательность из задачи 6 практики) S_∞, такое, что:

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

6. Придумайте решение задачи 4 про деревья без использования хеш-функций. Требуемое время работы такое же.

6 Практика 110. Хеширование 2

6.1 Практика

1. Пусть случайно и равновероятно выбрана функция $f : \mathbb{F}_2^n \to \mathbb{F}_2^m$. Найдите вероятность события
   (a) $f(x) = y$ для фиксированных x и y,
   (b) $f(x_1) = f(x_2)$ для фиксированных x_1 и x_2,
   (c) $f(x_1)$ отличается от $f(x_2)$ ровно в одном бите.
2. Является ли семейство из одной идеальной хеш-функции:
   (a) универсальным?
   (b) 2-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
$$h(x) := A \cdot x + b.$$
   (a) Найдите I максимальной мощности такое, что $|h(I)| = 1$. Какова мощность I?
   (b) Постройте семейство 2-независимых хэш-функций $\mathcal{H} = \{h_i : \mathbb{F}_2^n \to \mathbb{F}_2^k\}_i$. Докажите, что построенное семейство является 2-независимым.
4. В вашем распоряжении 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 (и докажите, что построено правильно) универсальное семейство хеш-функций, которое:
   (a) будет отправлять пары типа (A, B) в $\mathbb{F}_2^n$,
   (b) будет отправлять мультимножества из элементов типа A в $\mathbb{F}_2^n$,
5. В вашем распоряжении есть 2-независимое семейство хеш-функций, отправляющее векторы из R^2 в F. Придумайте, как построить универсальное семейство хеш-функций для наборов из n точек плоскости, которое:
   (a) будет считать сдвинутые наборы равными и иметь область значений $\mathbb{F}^{n-1}$,
   (b) кроме сдвига допускает и масштабирование (область значений $\mathbb{F}^{n-1}$),
   (c) также допускается поворот (область значений $\mathbb{F}^{n^2(n-1)}$)
6. Есть множество X и 2-независимое семейство хеш-функций $\mathcal{H} = \{f : X \to \mathbb{F}_2^n\}$. Определим похожесть двух множеств $A \subset X$ и $B \subset X$ как
$$d(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$
   Пусть $2^n \gg |A \cup B|^2$, заметим, что:
$$d(A, B) \simeq \mathop{\mathbf{Pr}}_{h \in \mathcal{H}} [\min h(A) = \min h(B)] .$$
   Случайно выбрав из $\mathcal{H}$ k хеш-функций, мы можем вероятностно оценить $d(A, B)$ с точностью порядка $\frac{1}{\sqrt{k}}$ за $\mathcal{O}(k(|A| + |B|))$ вычислений хеш-функций.
   (a) Докажите, что написанное выше правда (на самом деле не совсем).
   (b) Придумайте, как добиться того же результата с одной хеш-функцией и $\mathcal{O}(|A| + |B|)$ вычислений. Возможно, придется ввести дополнительные требования на семейство.
   (c) Придумайте реализацию предыдущего пункта за один проход, $\mathcal{O}(|A| + |B| + k \log k)$ времени и $\mathcal{O}(k)$ памяти (считая, что хеш вычисляется за $\mathcal{O}(1)$).

6.2 Домашнее задание

1. В вашем распоряжении 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 (и докажите, что построено правильно) 2-независимое семейство хеш-функций, которое:

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

   (b) будет отправлять списки из элементов типа A в $\mathbb{F}_2^n$.

   (c) будет отправлять мультимножества из элементов типа A в $\mathbb{F}_2^n$,

   (d) будет отправлять множества из элементов типа A в $\mathbb{F}_2^n$,

2. Пусть модуль m, по которому берется многочлен в полиномиальном хешировании не фиксирован, а выбирается случайно равновероятно из некоторого конечного набора простых чисел. Пусть константа x для полиномиального хеширования выбирается равновероятно из множества $\{1, 2, \ldots, m-1\}$. Докажите, что 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 $|h(I)| = 1$, где $h(I) = \{h(i) \,|\, i \in I\}$. Предложите алгоритм построения такого множества.

   Альтернативная подсказка: используйте принцип Дирихле

3. Равномерной называется хеш-функция, у которой у каждого значения одинаковое количество прообразов. Пусть даны две равномерные хеш-функции $f : A \to [n]$ и $g : B \to [n]$, где A, B — произвольные множества, а n — нечётное натуральное число. Постройте равномерную хеш-функцию с областью значений $[n]$ для:

   (a) Упорядоченных пар (A, B).

   (b) Неупорядоченных пар $\{A, B\}$. Возможны два варианта: $A = B$ и $A \cap B = \varnothing$.

Дополнительные задачи

4. Пусть дана w-разрядная RAM машина (т.е. машина, которая умеет производить арифметические операции и чтение/запись по адресу с w-битными словами за $\mathcal{O}(1)$). Для простоты можно считать w степенью двойки. Придумайте для такой машины предподсчет за $\mathcal{O}(w)$ операций, который позволит по w-битному числу вида 2^k восстанавливать k за $\mathcal{O}(1)$ операций. Ограничение на память — можно хранить O(w) слов.

7 Практика 111. Алгебра и криптография

7.1 Практика

В данной серии задач все алгоритмы должны работать полиномиальное время.

1. Дано число n, a и b. Известно, что $a^2 = b^2 (\mathrm{mod}\ n)$, но $a \neq \pm b\ (\mathrm{mod}\ n)$. Найдите нетривиальное разложение n на множители.

2. Дано множество натуральных чисел A и параметр b. Известно, что каждое число из A разлагается в произведение степеней первых b простых чисел (т.е. $\forall a \in A\ a = \prod_{i=1}^{b} p_i^{\alpha_i}$). Требуется найти непустое подмножество A, числа из которого которые в произведении дадут квадрат некоторого натурального числа. Придумайте полиномиальный от b и $|A|$ алгоритм, для поиска такого подмножества.

3. Дана матрица A размера $n \times m$ над конечным полем, $n \geq m$. Найдите матрицу B размера $m \times n$ такую, что $BA = I_{m \times m}$, или установите, что такой не существует.

4. Вам дана система линейных уравнений по модулю n. Решите данную систему при условии, что

   (a) $n = pq$, p, q — простые.
   (b) $n = p^k$, p — простое.
   (c) для произвольного n.

5. Есть множество $A \subseteq [n]$ и k ячеек, каждая из которых умеет хранить $\mathcal{O}(\log n)$ бит информации. В каждый момент времени максимум t из k ячеек могут оказаться недоступны (мы можем узнать про ячейку, доступна ли она, и, если доступна, прочитать данные). Требуется организовать такой способ хранения информации, чтобы в любой момент времени можно было востановить множество A.

   (a) $k = (t+1) \cdot |A|$.
   (b) $k = t + |A|$.

6. Допустим, случайно оказалось, что сообщение, шифруемое RSA, не взаимно просто с n. Сломается ли процедура шифрования/дешифрования? Чем плохо такое сообщение?

7. Аня решила послать приглашение на секретную вечеринку Боре, Ване и Гоше. Аня разослала им одинаковый текст приглашения M закодированный с помощью RSA. У Вани, Бори и Гоши выбраны различные n, но ключ e у всех одинаковый: $e = 3$. Придумайте, как Дима сможет узнать, где будет происходить секретная вечеринка, если ему доступны все три шифрованных приглашения и открытые ключи.

8. В d-мерном пространстве над $\mathbb{Q}$ даны точка и набор из k векторов — базис подпространства. Найдите расстояние от точки до подпространства.

9. 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 $\mathcal{O}(1)$.

10. Дан набор векторов с весами, оболочка векторов совпадает со всем пространством. Выбрать из данных векторов базис минимального суммарного веса.

11. Вы знаете, что последовательность $1, 1, 2, 3, 5, 8, 13, \ldots$ образована линейным рекуррентным соотношением с коэффициентами $1, 1$: $a_i = a_{i-1} + a_{i-2}$. Решим обратную задачу: дана последовательность длины n, найти минимальное k и k коэффициентов, задающие данную последовательность, как линейную рекуррентность.

12. Дана матрица A размера 2×2 над кольцом целых чисел. Придумайте алгоритм, представляющий A в виде $A = LDR$, где D — диагональная матрица, а L и R — обратимые.

7.2 Домашнее задание

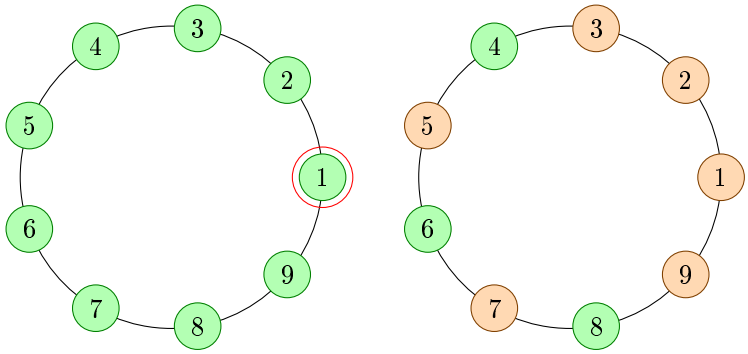
1. Дана матрица A размера $n \times m$. Найти такое представление $A = B \cdot C$, что B имеет размер $n \times k$ и k минимально. $\mathcal{O}(nm(n+m))$.

2. В распоряжении взломщиков оказался волшебный оракул, способный для любого открытого ключа (n, e) взломать 1% из возможных зашифрованных сообщений (т.е. детерминированная функция, которая за $\mathcal{O}(1)$ успешно сопоставляет набору (n, e, y) такое число m, что $y = m^e (\mod n)$, на случайно выбранной сотой части всех возможных входов). Придумайте алгоритм, который взламывает любое сообщение со средним временем работы $\mathcal{O}(poly(\log n))$.

3. Известны открытый ключ $(n, 3)$ и закрытый ключ (n, d) системы `RSA`, при этом n — произведение двух разных простых. Разложите n на множители. $\mathcal{O}(poly(\log n))$.

4. 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 пронумерованы 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 $\mathcal{O}(n^3)$.



5. У Винни-Пуха есть бесконечные запасы каждого из k видов горшочков мёда. Каждый вид горшочков характеризуется целым числом — сколько дней нужно Винни-Пуху, чтобы съесть весь мёд из одного такого горшка. Винни-Пух уже провёл серию экспериментов вида "взять несколько горшочков мёда, записать день недели, когда эксперимент начался, есть мёд, пока тот не закончится, записать день недели, когда эксперимент закончился" (в наборе может быть несколько горшочков одного вида, в неделе 7 дней). По понятным причинам Винни очень любит экспериментировать. А Кролик не любит мёд, но любит предсказывать будущее. Помогите Кролику определить, можно ли, зная результаты первых k экспериментов и набор горшков и день начала для $(k+1)$-го, предсказать результаты $(k+1)$-го эксперимента.

   (a) Каждый раз Винни берёт произвольные наборы горшков. $\mathcal{O}(k^3)$

   (b) Каждый раз Винни берёт горшки не более чем двух разных типов. $\mathcal{O}(k)$.

Дополнительные задачи

6. Рассмотрим матрицу $A \in \{0,1\}^{n \times m}$. Для произвольных i и j ($1 \le i \le n$, $1 \le j \le m$) разрешается поменять все значения в строке i и столбце j на противоположные (значение на пересечении

строки и столбца меняется). Требуется получить нулевую или единичную матрицу. Придумайте алгоритм за $o((nm)^3)$.

8 Практика 1000. FFT

8.1 Практика

1. Пусть вам известно $FFT([a_1, a_2, \cdots, a_n])$. Придумайте, как посчитать $FFT([a_2, a_3, \cdots, a_{n+1}])$ за линию.

2. Пусть корень из единицы вычисляется с точностью i знаков после запятой. Оцените, какое нужно выбрать i, чтобы при умножении с помощью `FFT` многочленов степени n с не превышающими k коэффициентами не возникало ошибок (ошибками, возникающими при сложении/умножении чисел, можно пренебречь).

3. За какое время можно посчитать 2^n в десятичной системе счисления, используя умножение на `FFT`?

4. (a) Пусть дан многочлен $p(x)$. Определим многочлен $R(p(x))$ как «разворот» $p(x)$: коэффициент при константе меняется с коэффициентом при $x^{deg(p(x))}$, коэффициент при x меняется с коэффициентом при $x^{deg(p(x))-1}$, и так далее. Докажите, что $R(p_1(x)p_2(x)) = R(p_1(x))R(p_2(x))$.

   (b) Пусть p_1 при делении на p_2 дает неполное частное q и остаток r. Докажите, что $R(p_1) \equiv R(p_2)R(q) \pmod{x^{deg(p_1)-deg(p_2)}}$.

   (c) Придумайте, как найти $R(p_2)^{-1} \mod x^k$ за $\mathcal{O}(n \log n)$.

   (d) Придумайте алгоритм, реализующий деление многочленов степени не более n с остатком за $\mathcal{O}(n \log n)$

5. Вам даны две последовательности, $\{a_i\}$ и $\{b_i\}$. Их сверткой называется последовательность $c_i = \sum\limits_{0 \le k \le i} a_k b_{i-k}$. Придумайте, как посчитать первые n членов свертки двух последовательностей за $\mathcal{O}(n \log n)$.

6. Даны 2 n-мерных вектора a и b. Посчитайте скалярные произведения a со всеми циклическими сдвигами b. $\mathcal{O}(n \log n)$.

7. Даны текст t и шаблон p над алфавитом размера k. $|t| \ge |p|$.

   (a) Для каждого из $|t| - |p| + 1$ наложений p на t узнать количество ошибок. $\mathcal{O}(k|t| \log |t|)$.

   (b) Предыдущий пункт с дополнительным условием: в тексте и в шаблоне допустимы символы “ ? ”, означающие совпадение с любым символом.

8. Дана чёрно-белая (без оттенков серого) картинка размера $N \times N$ и образец размера $K \times K, K \le N$. Найдите наилучшее вхождение образца в картинку за $\mathcal{O}(N^2(\log N))$. Наилучшим называется вхождение с минимальным суммарным количеством различных пикселей.

9. Эльф Леголас — пессимист. Начиная со своего сотого дня рождения каждому дню своей жизни он присваивает рейтинг уныния — действительное число из диапазона 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, происходящие 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 $\mathcal{O}(N \log N)$, где N — возраст Леголаса.

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

8.2 Домашнее задание

1. Найдите количество `AVL` деревьев высоты h из n вершин по простому модулю p за:

   (a) $\mathcal{O}(hn \log n)$.

   (b) $\mathcal{O}(hn_{\max})$. Здесь $n_{\max}$ — максимальное количество вершин в `AVL` дереве высоты h.

   Считайте, что в поле вычетов по модулю p вам известен примитивный корень из единицы достаточной степени.

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

3. Заданы картинка 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 $\mathcal{O}(n^2 \log n)$.

4. Даны строка s длины n и шаблон с вопросительными знаками. Найти подстроку s, точно подходящую под шаблон, за $\mathcal{O}(n \log n)$. Алфавит — **не** константа!

5. Даны строка длины n и шаблон со звездочками и вопросительными знаками. Проверьте, подходит ли строка целиком под шаблон за $\mathcal{O}(n\sqrt{n} \log n)$. Алфавит — **не** константа!

Дополнительные задачи

6. Перевод из системы счисления в другую быстрее квадрата.

7. Интерполяция в произвольных точках быстрее квадрата.

9 Практика 1001. Линейное программирование

9.1 Практика

1. Будем называть *стандартной формой* задачи линейного программирования ограничения $Ax \leq b; x \geq 0$, и *канонической формой* ограничения $Ax = b; x \geq 0$. Приведите задачу в стандартной форме к канонической и задачу в канонической форме к стандартной.

2. Сведите к задаче линейного программирования следующую задачу:

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

   При ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i \in [1 \dots m]; \quad x_i \geq 0, i \in [1 \dots n]$$

3. Рассмотрим такую игру с нулевой суммой: дана матрица A размера $n \times m$, первый игрок выбирает $i \in [1 \dots n]$, второй независимо $j \in [1 \dots m]$. Выигрыш первого игрока составляет $A_{i,j}$, а второго — $-A_{i,j}$. Игроки заинтересованы в максимизации матожидания своего выигрыша.

   (a) Сведите поиск оптимальной стратегии в этой игре к задаче линейного программирования.

   (b) Постройте двойственную задачу.

   (c) Покажите, что в данной игре существует равновесие по Нэшу. Стратегии a и b равновесны по Нэшу, если никому не выгодно отказываться от своей стратегии (если первый игрок сменит стратегию a на c, а второй продолжит придерживаться стратегии b, то результат первого не улучшится, аналогично для второго).

4. Рассмотрим задачу линейного программирования в канонической 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.

5. Матрица M тотально унимодулярна, если любой ее минор равен либо нулю, либо ± 1. Докажите, что если матрица задачи линейного программирования в канонической форме тотально унимодулярна, а вектор целый, то полиэдр данной задачи — целый.

6. Пусть полиэдр задачи линейного программирования целый. Докажите, что решение задачи целочисленного линейного программирования (максимум $c^T \cdot x$) с данными ограничениями совпадает с решением задачи линейного программирования.

7. 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 $LN(n, m+1)n \log(Mn)$, где M — максимальное число, встречающееся в матрице A, векторе или в минимизируемом функционале.

8. Рассмотрим задачу о максимальном паросочетании в графе.

   - Сформулируйте эту задачу, как задачу целочисленного линейного программирования.
   - Покажите, что в общем случае полиэдр данной задачи может быть не целым.
   - 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 U количество индуцированных этим подмножеством ребер не превосходит $\lfloor |U|/2 \rfloor$”, то полиэдр получится целым.

9.2 Домашнее задание

1. 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:

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

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

2. Приведите пример несовместной задачи линейного программирования, двойственная к которой так же несовместна.

3. 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:

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

   (a) Сведите задачу нахождения максимального s-t потока к задаче линейного программирования.

   (b) 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.

   (c) 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. Подсказка: назначьте каждой вершине v число ϕ_v, добавьте в целевую функцию от каждого ребра (u, v) добавку вида $g(\phi_v - \phi_u)$, и преобразуйте полученную задачу в задачу линейного программирования.

Дополнительные задачи

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:

$$Ax = b$$

$$\forall i : x_i > 0$$

10 Практика 1010. Потоки и разрезы

10.1 Практика

1. Дан двудольный граф. Каждой вершине сопоставлено число a_v.
   (a) Выберите максимальное количество рёбер так, чтобы степени вершин были не более 1.
   (b) Выберите максимальное количество рёбер так, чтобы степени вершин были не более a_v.

2. 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.

3. Дан неориентированный граф. Необходимо ориентировать его так, чтобы максимальная исходящая степень была минимальна.
   (a) $\mathcal{O}(E^2 \log V)$
   (b) $\mathcal{O}(E^2)$

4. 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.

5. Рассмотрим ориентированный граф. За одно действие можно удалить все входящие в вершину i ребра за стоимость $a_i \geq 0$, или все исходящие из вершины i рёбра за стоимость $b_i \geq 0$. Необходимо удалить все рёбра графа за минимальную стоимость.

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

7. 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.
   (a) $\mathcal{O}(poly(V, E, K))$
   (b) $\mathcal{O}(K(V + K)E)$

8. Какое максимальное количество уголков можно разместить на шахматной доске $n \times m$ с дырками? Уголком называется фигура, состоящая из трех клеток: центральная клетка черного цвета и две соседних с ней белых клетки со смежными сторонам.

9. Найдите ориентированный граф с целочисленными пропускными способностями, на которых детерминированный алгоритм Форда-Фалкерсона на основе `DFS` с фиксированным порядком просмотра рёбер в каждой вершине, **пропускающий максимум по дополняющему пути**, работает за экспоненту от V.

10.2 Домашнее задание

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

   (a) $\mathcal{O}(poly(V, E))$

   (b) $\mathcal{O}(E)$ 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.

   (a) $\mathcal{O}(V \cdot Flow)$.

   (b) $\mathcal{O}(E^2)$.

3. 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. $\mathcal{O}(nm^2)$.

4. Разбейте вершины ориентированного графа на циклы. Т.е. каждая вершина должна быть покрыта ровно одним циклом. Либо скажите, что это невозможно. $\mathcal{O}(EV)$.

Дополнительные задачи

5. 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.

   (a) Вопрос: какую максимальную прибыль вы можете получить?

   (b) 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.

6. Дана укладка планарного графа. Вершинам сопоставлены точки на плоскости, рёбра – отрезки между вершинами, рёбра не пересекаются. У рёбер есть пропускные способности. Граф неориентированный. Даны две вершины s и t, лежащие на одной грани. Задача: за $\mathcal{O}(Dijkstra)$ найти величину максимального потока из s в t.

7. Какое максимальное количество уголков можно разместить на шахматной доске $n \times m$ с дырками? Уголком называется фигура, состоящая из трех клеток: центральная клетка черного цвета и две соседних с ней белых клетки со смежными сторонам.

8. Найдите ориентированный граф с целочисленными пропускными способностями, на которых детерменированный алгоритм Форда-Фалкерсона на основе `DFS` с фиксированным порядком просмотра рёбер в каждой вершине, **пропускающий максимум по дополняющему пути**, работает за экспоненту от V.

11 Практика 1011. Строки

11.1 Практика

1. Найдите число различных подстрок строки.
   (a) Z-функцией за $\mathcal{O}(n^2)$.
   (b) По известному суффиксному массиву с `lcp` за $\mathcal{O}(n)$.
2. Научиться искать за $\mathcal{O}(n+m)$ образец в строке, если между образцом и найденной подстрокой допустимо различие:
   (a) в один символ.
   (b) в два символа.
3. Найти образец в строке, если допустимо в образце применять к алфавиту перестановку. $\mathcal{O}(n+m)$.
4. Построить
   (a) суффмассив по суффдереву
   (b) суффдерево по суффмассиву + `lcp`

   за линейное время.
5. Дан массив a длины n из m-битных чисел. Найдите пару $a_i, a_j : a_i \oplus a_j = \max$. $\mathcal{O}(nm)$.
6. Найдите наибольшую общую подстроку k строк суммарной длины n за $\mathcal{O}(n)$.
   (a) Суффиксным массивом.
   (b) Суффиксным деревом.
7. Найти самую длинную подстроку, входящую в заданную дважды:
   (a) вхождения могут пересекаться.
   (b) вхождения не могут пересекаться.
8. Дан словарь слов суммарной длины L. За время $\mathcal{O}(L)$ определите, существует ли бесконечная строка, не содержащая ни одно словарное слово как подстроку.
9. Даны словарь и текст. Нужно уметь обновлять ответ `online` при добавлении символов в конец текста.
   (a) Пересчитайте суммарное число вхождений слов из словаря в текст за $\mathcal{O}(1)$.
   (b) Поддерживайте множество всех вхождений слов из словаря в текст. Пересчёт за время $\mathcal{O}(1+|\Delta A|)$, где ΔA – приращение ответа после добавления очередного символа.
10. Постройте строчку над минимальным алфавитом, у которой суффиксный массив совпадает с данным. $\mathcal{O}(n)$.
11. 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 $\mathcal{O}(n)$.
12. Дан набор строк s_i. Для каждой s_i найдите min по длине подстроку, которая не встречается в других. $\mathcal{O}(n)$.

11.2 Домашнее задание

1. Найти подстроку в тексте. При сравнении строк можно делать циклический сдвиг алфавита в одной из них. $\mathcal{O}(n+m)$, алфавит — не константа.
2. Для каждого префикса строки найти количество его префиксов, равных его суффиксу. $\mathcal{O}(n)$.
3. Дан массив a длины n и число k. Элементы массива и k — m-битные числа. За время $\mathcal{O}(nm)$:
   (a) посчитайте количество таких пар индексов массива, что строка длины m, являющаяся побитовым `xor` элементов по этим индексам, $\geq k$.
   (b) посчитайте количество подмассивов a, побитовый `xor` всех чисел из которых $\geq k$.
   (c) найдите подмассив a, побитовый `xor` всех чисел из которого максимален.
4. За $\mathcal{O}(n)$ построить строку с заданной:
   (a) Z-функцией.
   (b) префикс-функцией.
5. Дан набор строк s_i суммарной длины n. Для каждой s_i найдите min по длине подстроку, которая не встречается в других. $\mathcal{O}(n)$.
6. Даны k строк суммарной длины n. Найдите p-ю лексикографически общую их подстроку за $\mathcal{O}(n)$.

Дополнительные задачи

7. За одну секунду в конец изначально пустого текста дописывается случайная буква (равномерное распределение). Какое матожидание времени T, когда первый раз s станет подстрокой выписанного текста?
8. Постройте строчку над минимальным алфавитом, у которой суффиксный массив (без попарных `lcp`) совпадает с данным. $\mathcal{O}(n)$.

12 Практика 1100. NP-complete

12.1 Практика

1. Приведите пример NP-трудной, но не NP-полной задачи.
2. Докажите, что задача 2SAT лежит в классе P.
3. Докажите, что следующие задачи NP-полны:
   (a) CIRCUIT-SAT: По данной булевой схеме проверить, выполнима ли она.
   (b) 3SAT: По данной булевой формуле в КНФ-виде, у которой каждый дизъюнкт содержит не более чем три литерала, проверить, выполнима ли она.
   (c) INDEPENDENT SET: По данному графу проверить, есть ли в нём независимое множество размера k.
   (d) VERTEX COVER: По данному графу проверить, есть ли в нём вершинное покрытие размера k.
   (e) CLIQUE: По данному графу проверить, есть ли в нём полный подграф (клика) размера k.
   (f) Целочисленное линейное программирование.
   (g) HITTING SET. Дан набор множеств $S = \{S_1, S_2, \cdots, S_3\}$ и число k. Требуется выяснить, есть ли такое множество H, что его размер не превышает k, и его пересечение с каждым множеством из набора не пусто.
4. Рассмотрим задачу о независимом множестве. Докажите, что если существует алгоритм, позволяющий за полиномиальное время ответить на вопрос: правда ли, что в заданном графе существует независимое множества размера n, то
   (a) Существует полиномиальный алгоритм, который строит независимое множество размера $\geq n$.
   (b) Существует полиномиальный алгоритм, который строит максимальное независимое множество.
5. Рассмотрим следующую игру: Дан ориентированный граф, каждому ребру которого сопоставлено целое число. На одной из его вершин стоит фишка. Два игрока ходят по очереди, каждый в свой ход может передвинуть фишку по ребру и заплатить противнику противнику сумму, соответствующую этому ребру. Будем считать, что первый игрок выигрывает, если существует стратегия, которая позволяет ему бесконечно обогатиться. Докажите, что такая задача лежит в $NP \cap coNP$.
6. Рассмотрим задачу EXACT 4SAT. Это такой SAT, где каждая дизъюнкция состоит ровно из 4 литералов. Докажите, что эта задача NP-полна.
7. Рассмотрим задачу 3SAT. Пусть каждый литерал входит в формулу не более чем 2 раза. Докажите, что даже такая задача NP-полна. Докажите, что, если каждый литерал входит в формулу не более чем один раз, то задача решается полиномиально.
8. Хорновская формула — формула вида $(a_1 \wedge a_2 \wedge \cdot \wedge a_n \to b)$. Все вхождения литералов в Хорновскую формулу — положительны. Докажите, что задача HORNSAT (задача о выполнимости конъюнкции конечного набора Хорновских формул) лежит в классе P.

12.2 Домашнее задание

Дополнительные задачи

1. Рассмотрим такую игру: Дан направленный взвешенный граф, на нем стоит фишка. Игроки ходят по очереди, каждый имеет право передвинуть фишку по ребру. Передвигая фишку игрок платит противнику сумму, написанную на этом ребре. Будем считать, что первый игрок выигрывает, если существует стратегия, которая позволяет ему бесконечно обогатиться. Докажите, что такая задача лежит в $NP \cap coNP$.

2. Дан набор векторов с весами, оболочка векторов совпадает со всем пространством. Выбрать из данных векторов базис минимального суммарного веса.

3. Классифицируйте рёбра заданного графа: какие обязательно лежат в максимальном паросочетании, какие могут лежать, и какие точно не лежат.

   (a) $\mathcal{O}(E^2)$

   (b) $\mathcal{O}(VE)$

   (c) $\mathcal{O}(E)$ по данному максимальному паросочетанию

4. Два игрока по очереди делают ходы на двудольном графе: передвигают фишку в одну из смежных вершин и удалают ребро. Проигрывает тот, кто не может сделать хода. Определите, кто выиграет при оптимальной игре.

5. Приведите полиномиальный алгоритм для нахождения чётности количества совершенных паросочетаний в двудольном графе с равным размером долей.

6. Определите четность числа совершенных паросочетаний в произвольном графе.

7. За одну секунду в конец изначально пустого текста дописывается случайная буква (равномерное распределение). Какое матожидание времени T, когда первый раз s станет подстрокой выписанного текста?

8. Постройте строчку над минимальным алфавитом, у которой суффиксный массив (без попарных `lcp`) совпадает с данным. $\mathcal{O}(n)$.

9. Покажите, что `P` $\neq$ `NP`.