Разбор задачи «Подарок»

Задача заключалась в следующем: среди всех прямоугольных параллелепипедов с натуральными длинами сторон и площадью поверхности не более *n,* найти тот, объем которого максимален.

Задачу можно решать последовательными улучшениями.

O(n^3) — 20 баллов

Переберем все тройки натуральных чисел *a, b, c* $\leq$ *n.* Рассмотрим прямоугольный параллелепипед со сторонами (*a, b, c*). Площадь поверхности этого параллелепипеда есть *2ab+2ac+2bc = S.* Среди все таких, что $S \leq n$ найдем тот, объем которого максимален.

O(n^2) — 30 баллов

Зафиксируем две стороны *a* и *b.* Зададимся вопросом, какой может быть третья сторона. Из неравенства *2ab+2ac+2bc* $\leq n$ получаем, что $c \leq \frac{n-2ab}{2a+2b}$. Таким образом, достаточно рассмотреть только $c = \left\lfloor \frac{n-2ab}{2a+2b} \right\rfloor$. Перебирая все пары $a,b \leq n$ получаем решение за $O(n^2)$.

O(n log n) — 60 баллов

Для того, чтобы еще сильнее ускорить решение, заметим, что имеет смысл перебирать только такие стороны *a* и *b*, что *ab <= n.* Количество таких пар есть $\sum_{a=1}^{n} \frac{n}{a} = n \sum_{a=1}^{n} \frac{1}{a} = O(n \log n)$. Аналогичная идея применяется при оценке времени работы алгоритма Эратосфена.

O(n) — 70 баллов

Описанное выше рассуждение можно еще улучшить. Предположим, что $a \leq b \leq c$ — стороны параллелепипеда. Тогда $a, b \leq \sqrt{n/2}$. Действительно, если предположить обратное, то $b, c > \sqrt{n/2} \Rightarrow 2bc > n$. Исходя из этого наблюдения, достаточно перебрать O(*n*) пар значений *(a,b).*

O($\sqrt{n}$) — 100 баллов

Основная идея решения заключается в исследовании границ, в которых лежит оптимальный ответ. Кратко опишем ключевые моменты:

- Если стороны параллелепипеда можно было бы делать вещественными, то оптимальный ответ достигался бы при $a = b = c = \sqrt{n/6}$.
- Положим $a = \lfloor \sqrt{n/6} \rfloor$.
- Будем последовательно уменьшать a, пока оптимальный ответ при фиксированном *a* и вещественных *b* и *c* не станет хуже лучшего из найденных.
- Убедимся, что даже для *n* порядка 10^{13} потребуется перебрать не очень много значений *a.* Это легко можно сделать, проведя численный эксперимент. Аналитические же выкладки довольно громоздки и показывают, что достаточно перебрать порядка $\sqrt[4]{n}$ значений *a*.

- При фиксированном значении *a* оптимальный ответ достигается при $b=c=\sqrt{a^2+n}-a$.
- Положим $b=\left\lfloor\sqrt{a^2+n}-a\right\rfloor$.
- Будем последовательно уменьшать a, пока оптимальный ответ при фиксированных a, b и вещественном c не станет хуже лучшего из найденных.
- Снова убедимся, что требуется перебрать небольшое количество различных значений *b* (порядка $\sqrt[4]{n}$).
- Суммарное время работы программы есть O($\sqrt{n}$).

Существует также альтернативный подход к получению верного решения. Тем или иным способом догадавшись, что ответ надо искать вблизи $\sqrt{n/6}$, можно перебрать значения, отстоящие от этого значения не более чем на Δ. Из предыдущего решения следовало, что Δ следует задавать порядка $\sqrt[4]{n}$. Такое решение также имеет асимптотику O($n^{1/2}$).

Разбор задачи «ЮграНефтеТранс»

Задача заключалась в следующем: задано множество из n станций и m трубопроводов, соединяющих некоторые пары станций. Требуется выбрать множество из k станций, чтобы один из двух концов каждого трубопровода лежал в выбранном множестве. Если построить граф, в котором станции будут служить вершинами, а трубопроводы – рёбрами, то искомое множество будет являться *вершинным покрытием* в этом графе.

Рассмотрим различные варианты переборного решения этой задачи:

Вариант 1.

Перебор всех 2^n подмножеств множества вершин.

Такое решение набирает 10 баллов.

Вариант 2.

Перебор всех $C(n, k)$ подмножеств размера k. Проверку того, что выбранное множество является вершинным покрытием можно:

- За $O(m)$, перебором всех рёбер и проверкой того, что хотя бы один конец каждого ребра выбран.
- «На лету», во время перебора подмножеств. Достаточно заметить, что при добавлении вершины в покрытие количество рёбер, которое станет покрытым, равно количеству соседей добавленной вершины, не лежащих в покрытии.

Такое решение набирает 30 баллов.

Вариант 3.

Заметим, что истинно следующее утверждение:

Пусть v – вершина графа. Тогда в вершинном покрытии либо лежит вершина v, либо все соседи вершины v.

Будем повторять следующий процесс:

- Выберем вершину, не лежащую в покрытии.
- Добавим в покрытие либо её саму, либо всех её соседей.

Как только количество добавленных вершин станет равным k, проверим, получили ли мы вершинное покрытие. Если нет, то вернёмся на шаг назад и продолжим перебор.

Мы сделаем не более k шагов в глубину, а на каждом шаге будем перебирать 2 варианта. Сложность такого алгоритма – $O(2^k)$.

Это решение набирает 60 баллов.

Различные оптимизации

Заметим, что:

- Если у вершины не менее k соседей, то мы обязаны включить её в покрытие.
- Если у вершины один сосед, то мы можем добавить этого соседа в покрытие и исключить нашу вершину из рассмотрения.

Если провести описанные действия, то в графе останутся вершины со степенью не меньше одного и не больше k-1. Тогда описанный выше перебор будет работать за $O(1.6^k)$.

В процессе перебора выгодно сначала рассматривать вершины с максимальным количеством соседей. В этом случае, при добавлении всех соседей в покрытие, количество оставшихся вершин, которые нужно добавить в покрытие, резко сокращается. Если же в графе остались только вершины степени 2, то этот граф представляет собой несколько простых циклов, а значит, мы можем не перебирать два варианта, а всегда добавлять в покрытие саму вершину, а не её соседей.

Если воспользоваться описанной эвристикой, то сложность алгоритма будет составлять уже $O(1.466^k)$.

Это решение получает 100 баллов.

Разбор задачи «Москва - Ханты-Мансийск»

Задача заключалась в следующем: есть N человек, которые хотят улететь из Москвы в Ханты-Мансийск. Каждый день летает один самолет вместимостью K человек. У каждого человека есть множество дней, когда он может улететь. Это отрезок $[L_i\ R_i]$. Нужно придумать такое распределение людей по самолетам, что до Ханты-Мансийска долетит максимальное число людей. Среди людей есть участники РОИ, которых нужно перевезти обязательно (остальных людей будем называть обычными).

Без последнего условия (все люди одинаковы, участников РОИ не бывает) задачу можно решить любой из следующих жадностей за время O(NlogN):

Жадность 1:

- Перебираем отрезки в порядке возрастания правого конца (сортировка)
- Для каждого человека пытаемся выбрать самое левое свободное место (это делается деревом отрезков с операцией “взять самый левый ненулевой элемент на отрезке”)

Жадность 2:

- Перебираем самолеты по возрастанию времени вылета.
- Храним всех людей, которые могут улететь в текущий момент времени, но еще не улетели.
- Из всех хранящихся людей нужно выбрать K, которые сейчас полетят. Выберем тех, чьи отрезки закончатся раньше. Чтобы выбирать каждого человека за O(logN) используем кучу по ключу R.

Теперь перейдем к возможным решениям исходной задачи. Несколько полезных фактов:

- Мы можем как-то рассадить всех участников РОИ жадностью описанной выше. Для этого достаточно не обращать внимания на других людей. Будем обозначать эту рассадку P.
- Мы можем рассадить максимальное возможное людей, но тогда возможно не все участники РОИ улетят. Эту рассадку обозначим M.
- Больше чем в M мы людей точно не отправим. Если P существует, то всех участников РОИ отправить точно можно.

Решение за $O(N^2 \log N)$, пока без доказательства: всех обычных людей будем по очереди пытаться добавить в P. За O(NlogN) той же жадностью, что и раньше, будем проверять, можно ли распределить по самолетам всех уже распределенных и еще одного нового. Полученный ответ оптимален. Но пока **не понятно, почему.**

Чтобы развивать мысль дальше заметим, что задачу можно читать так «Дан двудольный граф, **найдите максимальное паросочетание**, содержащее такие-то вершины». Двудольный граф это отрезки и места в самолетах. Каждому отрезку $[L_i\ R_i]$ соответствует $K*(R_i - L_i + 1)$ свободных мест. В этом графе O(NM) ребер, а максимальное паросочетание имеет размер не более N.

Паросочетание можно искать алгоритмом Куна (работает за O(VE)), о нем можно прочесть по ссылке (http://e-maxx.ru/algo/kuhn_matching). Коротко алгоритм можно описать так: по очереди перебираем вершины первой доли, не входящие в паросочетание. Если вершину можно добавить, добавляем (автоматически добавиться и еще одна из второй доли, а паросочетание как-то перестроится). Для нас самое важное, что множество

вершин в паросочетании только расширяется. Значит, если участников РОИ распределить по свободным местам жадно, а потом увеличить паросочетание алгоритмом Куна до максимального, мы получим верный ответ. В частности мы получили решение вида «применить в лоб общий вариант алгоритма Куна», работающее за O(VE) = **$O(N^2M)$**, и доказали корректность алгоритма за **$O(N^2 \log N)$** – это просто одна из возможных реализаций алгоритма Куна на нашем графе. Еще важно увидеть, что любое распределение (например, P и M) – это паросочетание, а любое паросочетание – это возможное распределение людей по свободным местам.

Решение за O(NlogN) (для понимания необходимо понимать алгоритм Куна!)

- Ищем паросочетания P и M жадностью за O(NlogN).
- Смотрим на симметрическую разность P и M. Она содержит дополняющие пути, их можно выделить поиском в глубину за O(N).
- Увеличив P этими путями за O(N), мы получим новое паросочетание Q, которое по размеру равно максимальному и содержит всех участников РОИ. Это и есть ответ.

Другое решение за O(NlogN):

- Первая фаза алгоритма берет всех участников РОИ и уменьшает им правые границы. Будем перебирать самолеты в порядке убывания времени и в очередной самолет сажать K участников РОИ с максимальными левыми концами отрезков. У участников, которых можно было отправить текущим рейсом, но в K максимальных не вошедших, уменьшим правую границу на 1. Пример: 5 участников [2,4] [1,4] [3,4] [2,3] [3,3], K = 2. Тогда [3,4] и [2,4] отправим в 4-й день, а [1,4] уменьшится до [1,3]. Выбирая в день 3 из [1,3] [2,3] и [3,3], оставляем [1,3], который сокращается до [1,2]. В итоге участники: [1,2] [2,3] [3,3] [2,4] [3,4].
- Вторая фаза – это модификация жадности, которая рассаживает максимальное число пассажиров, выбирая в очередной день K с минимальным правым концом (см. начало разбора). Единственное различие заключается в том, что если есть 2 человек с одинаковым правым концом отрезка, и один из них – участник РОИ, мы обязательно возьмем именно участника.
- **Почему все участники РОИ войдут в ответ?** Первая фаза гарантирует нам, что если в T-й день мы не отправили участников РОИ, отрезки кончаются позже T, мы еще как-нибудь сможем их отправить. Т.е. обязательно в T-й день нужно отправить только тех, кто позже уже улететь точно не сможет. Вторая фаза алгоритма как раз так и поступает.
- **Почему ответ имеет максимальный размер?** Мы применили ту же жадность, что и в самом начале, сделав уточнение в случае, в котором исходная жадность могла сделать любой выбор.

Разбор задачи «Ханты-Мансийск - Париж»

В данной задаче необходимо найти путь передачи сообщения от Поликарпа, проживающего в Ханты-Мансийске де Кодеру, проживающему в Париже.

Сначала необходимо заметить, что, если есть возможность передать сообщение от одного человека другому напрямую, то это лучше, чем передавать через посредников. Это верно, так как при передаче через посредников, найдутся два последовательных человека с меньшим или равным общим префиксом, чем у изначальных источника и получателя.

Таким образом, при передаче могут быть необходимы только люди, живущие в Дубае, Москве и Калининграде и всегда происходит ровно четыре передачи

В дальнейшем подсчет стоимостей одной передачи сообщения происходит самым простым способом – одним циклом находя наибольший общий префикс. Как n обозначается суммарное количество людей во всех городах.

Можно перебрать через каких людей в промежуточных городах передается сообщение, и найти минимальную стоимость. Это решение работает за $O(n^3)$ и получает 40 баллов.

Задачу можно сформулировать как задачу нахождения кратчайшего пути в графе, в котором вершинами являются люди, а ребра соответствуют передаче сообщения напрямую и стоят соответствующее число кредитов. Построение графа занимает $O(n^2)$ времени. Искать кратчайший путь в графе можно несколькими алгоритмами:

a) Алгоритм Флойда за $O(n^3)$ получает 40 баллов
b) Алгоритм Дейкстры за $O(n^2)$ получает 60 баллов
c) Можно заметить, что граф ациклический и искать кратчайший путь в нем стандартным алгоритмом за $O(n^2)$ и получить 60 баллов

Последний способ поиска кратчайшего пути наталкивает на идею решения данной задачи с использованием динамического программирования (ДП). Для каждого человека считается минимальная стоимость передачи ему сообщения. Так как выгодно только передавать сообщение в следующий часовой пояс (из Ханты-Мансийска в Дубаи, из Дубая в Москву и так далее), то эту стоимость можно считать в порядке этих часовых поясов. Для некоторого человека из текущего часового пояса можно перебрать кто из предыдущего часового пояса передаст ему сообщение и из всех таких способов выбрать самый дешевый. Это решение работает за $O(n^2)$ и получает 60 баллов.

Можно попытаться использовать это решение и при больших n – для того, чтобы уложиться во временной лимит можно в каждом городе при переходе к следующему сохранять только некоторое количество лучших (кому дешевле всего передать сообщение). Если оставлять 3000 человек, то такое решение получает 70 баллов.

Полные решения (100 баллов):

Используя аналогичную идею ДП, считать значения быстрее:

a) Предварительно отсортировать номера в каждом часовом поясе. Считая значение ДП для некоторого человека, перебирать длину совпадающего префикса его номера, находить (с помощью двоичного поиска) отрезок людей в предыдущем часовом поясе с таким префиксом номера. На этом отрезке находить минимальное значение ДП с использованием, например, дерева отрезков.
   Это решение работает за $O(n \log n)$.
b) Аналогично необходимо предварительно отсортировать номера в каждом часовом поясе (это можно сделать за $O(n)$ с использованием цифровой сортировки). Используя идею «двух указателей» последовательно считать значения ДП для людей из текущего

часового пояса, двигая второй указатель в предыдущем часовом поясе, пока номер человека в предыдущем часовом поясе лексикографически меньше номера человека из текущего часового пояса (для которого считается значение ДП). Для каждой длины совпадающего префикса номера текущего человека хранится минимальная стоимость передачи сообщения человеку из предыдущего часового пояса, имеющего номер с совпадающим префиксом. При перемещении указателя в предыдущем часовом поясе эти значения несложно пересчитываются. После прохода «двумя указателями» в одну сторону необходимо пройти указателями в другую сторону и выбрать из полученных способов минимальный по стоимости.

Это решение работает за O(n).

c) Используя идею meet-in-the-middle («встреча посередине») можно для каждого человека, проживающего в Москве посчитать минимальную стоимость передачи сообщения ему из Ханты-Мансийска через человека в Дубае, и в Париж через человека в Калининграде. Необходимо заметить, что передавать сообщение человеку A, проживающему в Москве надо через человека, проживающего в Дубае и имеющего наибольший совпадающий префикс или с Поликарпом, или с A.

Таким образом, такой подсчет можно реализовать методом «двух указателей» за O(n).

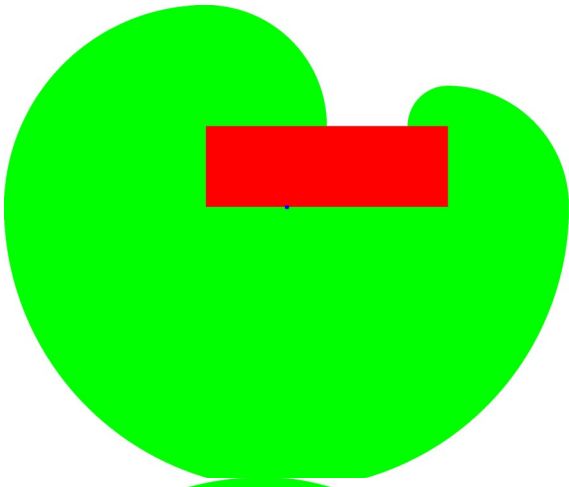
Разбор задачи «А олени — лучше»

Формулировка задачи:

- Задан выпуклый многоугольник и точка на его границе
- Требуется найти площадь области точек, до которых можно добраться из заданной точки не проходя через многоугольник.

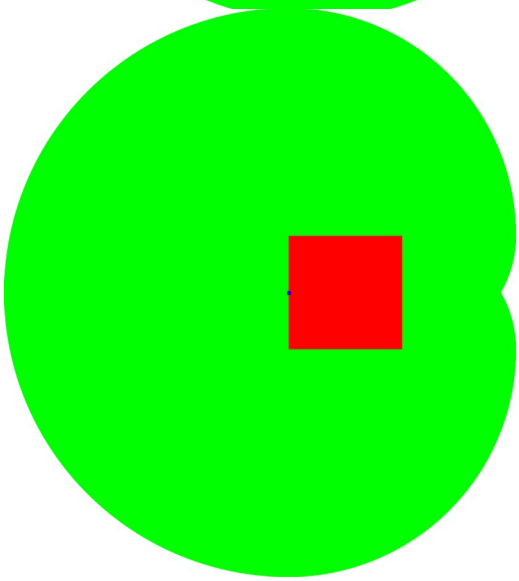
Решение для веревки не длиннее полупериметра:

- Область представляет собой объединение нескольких секторов окружностей разного радиуса.
- Если точка находится на стороне, то первый сектор это полуокружность радиуса равного длине веревки.
- Следующие сектора имеют меньший радиус. Радиус уменьшается по мере удаления от стартовой точки.



Вычисление площади сектора:

- Угол сектора равен углу между векторами последовательных вершин.
- В случае когда ограда является прямоугольником угол всегда равен 90 градусов. Поэтому площадь сектора равна четверти площади окружности.
- Площадь сектора окружности радиуса R с углом β равен R * R * β * 0.5

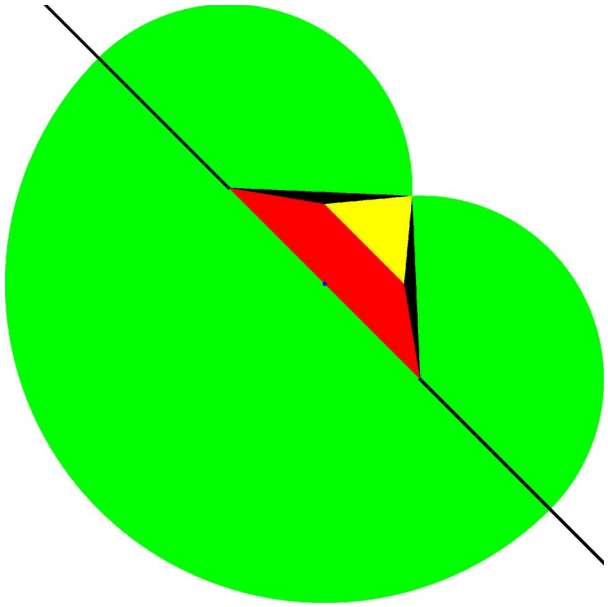


Решение для веревки длиннее полупериметра:

- Области покрытые слева и справа начинают пересекаться.
- Необходимо найти точку, когда они начнут пересекаться и позиции, в которых пересекаются сектора. Для этого переберем все пары секторов и найдем наиболее раннюю пару пересекающихся.

Ответ состоит из трех компонент:

- Сумма площадей секторов до момента пересечения.
- Сумма двух последних секторов, идущих до точки пересечения.
- Сумма треугольников с вершинами: точка пересечения и две последовательные точки многоугольника.



Для выбора ближайшего пересечения, самый простой способ — это посчитать искомую площадь при предположении каждого пересечения. Из полученных результатов выбрать минимум.

Разбор задачи «Земледелие 2.0»

Данная задача предполагает множество различных решений. Самые элементарные из них (для удобства оценки будем считать, что $n=\max(w, h)$):

$O(n^{10})$ – перебрать участок, который следует удобрить (он представляет собой прямоугольник, всего $O(n^4)$ вариантов), перебрать итоговый участок посадки (он также представляет собой прямоугольник, $O(n^4)$ вариантов), далее проверить корректность (возможность совершить посадку) за $O(n^2)$. Такое решение, представляющее собой перевод условия на язык программирования, набирало 10 баллов.

$O(n^6)$ (первое решение, простое) – перебрать фрагмент для посадки (больш*и*й прямоугольник, $O(n^4)$ вариантов). Осталось для каждого проверить, можно ли удобрить какой-то подпрямоугольник так, чтобы весь фрагмент состоял из единиц. Это делается за $O(n^2)$ – найдем нули с самыми большими и маленькими номерами строк и столбцов (всего 4 числа), это границы искомого подпрямоугольника.

$O(n^6)$ (второе решение) – перебрать ход (меньш*и*й прямоугольник, $O(n^4)$ вариантов), затем найти оптимальный удобренный участок за $O(n^2)$. Вторая часть решения является классической задачей, которая решается следующим образом. Зафиксируем нижнюю границу, затем будем перебирать вертикальные столбцы слева направо, храня столбцы в стеке. При добавлении очередного столбца из стека следует удалить все столбцы, большие текущего, проверяя прямоугольник, ограниченный удаляемым и добавляемым столбцами. Высоты столбцов можно предварительно подсчитать методом динамического программирования. Такое решение даже при самой неоптимальной реализации работало для w, h до 40 и набирало 30 баллов. Добавление отсечений или технических оптимизаций могло незначительно улучшить результат.

$O(n^4)$ – зафиксировать границы прямоугольника-ответа ($O(n^4)$ вариантов), затем проверить, что неудобренные клетки внутри образуют прямоугольник. Для того, чтобы эта проверка выполнялась за $O(1)$, нужно при переходе от одного прямоугольника к другому пересчитывать количество неудобренных клеток внутри, а также позиции крайних таких клеток. Такое решение работало для w, h порядка 100 и набирало около 50 баллов в зависимости от реализации. Также к нему можно добавить отсечения, главное из которых – при фиксированных трёх границах прерывать перебор, если теоретически получаемый ответ меньше текущего (отсечение по ответу).

$O(n^3)$ – зафиксировать верхнюю и нижнюю границы ответа ($O(n^2)$ вариантов). Заметим, что теперь все столбцы, содержащие неудобренные клетки, должны содержать их на одних и тех же позициях, а также быть расположенными подряд. Поэтому можно перебрать левую и правую границу ответа методом двух указателей за $O(n)$. Такое решение работало для w и h до 300 и набирало 60 баллов.

Следующие решения являются значительными продвижениями к верному и набирают от 80 баллов.

$O(n^2 \times \log n)$ – развитие предыдущего решения. Зафиксируем нижнюю границу ($O(n)$ вариантов). Теперь рассмотрим группы полосок с одинаковыми позициями нулей. При движении верхней границы эти группы будут разделяться (в случае, когда позиции нулей перестали совпадать), либо исчезать (когда группа нулей в столбце перестаёт быть непрерывной). Всего таких событий $O(n)$, так как они происходят не более одного раза для каждого столбца и не более одного раза для каждой пары соседних столбцов. Можно поддерживать дерево, либо другую структуру данных, поддерживающую эти операции за $O(\log n)$,

$O(n^2 \times \alpha(n))$ – развитие предыдущего решения. Будем двигать верхнюю границу в обратном направлении. В таком случае события разделения заменяются событиями объединения, поэтому можно использовать систему непересекающихся множеств.

Следующее решение является корректным и предполагалось к реализации участниками.

$O(n^2)$ – альтернативное развитие решения со сложностью $O(n^2 \times \log n)$. Можно заметить, что объединения происходят только между столбцами, находящимися рядом. Заменим столбцы группами столбцов, хранящихся в списке. Объединять рядом стоящие группы столбцов можно за O(1), откуда и получается верное решение. Это решение получало 100 баллов.