Третий этап республиканской олимпиады. Разбор задач.

Typ 1

Задача 1. Новогодние подарки

- Дано число x
- К нему применяется следующий алгоритм:
  - Если x < y, то остановиться
  - Иначе уменьшить число не более чем на $\left\lfloor \frac{x}{c_i} \right\rfloor$
- Какое наименьшее число можно получить?

Наивные решения

- Пусть n=1, т. е. нам не надо выбирать c_i
- Заметим, что нам почти всегда выгодно брать ровно $\left\lfloor \frac{x}{c_i} \right\rfloor$
- Нам выгодно брать меньше, только если мы таким образом придем ровно в y
- Далее симулируем процесс
- Такое решение набирает 48 баллов

Наивные решения

- При $n > 1$ из всех c_i всегда выгодно брать наименьшее
- Используя это замечание, получаем 71 балл

Полное решение

- Избавимся от симуляции «в лоб»
- Можно заметить, что после уменьшений может получиться одна из следующих ситуаций:
  - x < y сразу (тогда просто выводим x)
  - Мы дошли до y. Тогда выводим $y - \left\lfloor \frac{y}{c_i} \right\rfloor$
  - x стал меньше, чем c_i. Тогда $\left\lfloor \frac{x}{c_i} \right\rfloor = 0$, т. е. мы уже ничего не можем сделать

Полное решение

- Аккуратно учитываем эти все случаи
- Например, так
  - Если $x < y$ ==> выводим x
  - Если $y < c_i$ ==> выводим $c_i - 1$
  - Иначе ==> выводим $y - (y/c_i)$
- c_i, конечно, наименьшее из возможных
- Получаем 100 баллов :)
- Асимптотика – $O(n)$

Задача 2. Надежная сеть

- n компьютеров соединены между собой m кабелями
- Сигнал может ходить в обе стороны
- Кабели бывают двух типов: a и b

- Необходимо найти путь из любой вершины в любую другую, проходящий по каждой вершине не более одного раза, чтобы получившаяся строка пути была четной длины и ее первая половина равнялась второй

Задача 2. Решение (36 и 77)

- Граф представляет из себя цепочку
- n <= 2000 и m = n - 1
- Составим строку и в ней найдем ответ за O(n^2)

- Граф представляет из себя дерево
- n <= 2000 и m = n - 1
- Запустим поиск в глубину из каждой вершины и в ней найдем ответ за O(n^2)

Задача 2. Решение (100 баллов)

Начнем строить пути.

- a → ab
- b → ba
- ab → aba
- ba → bab
- aba → abab
- bab → baba

- Если из вершины выходят 2 ребра с одной буквой, то выводим ответ
- Иначе ищем пути из 4 ребер (abab или baba)

Задача 3. Столкновение галактик

- n галактик, в каждой из которой одна звезда
- звезда в i-й галактике с яркостью i
- запросы двух видов
  - объединить галактики, в которых находятся элементы x и y
  - найти медиану галактики x

Задача 3. Решение (26 баллов)

- $n <= 8000$
- Используем массив, храним медиану, при добавлении – пересчитываем ее за $O(n)$.
- Тогда время работы – $O(n^2 + q)$.

Задача 3. Решение (59 баллов)

- n, q <= 2*10^5, одна из сталкивающихся галактик состоит из одной звезды
- заведем 2 set'a и будем добавлять элементы по следующему правилу:
- Добавим элемент в множество, размер которого меньше, или же в первое множество
- Если максимум первого множества больше минимума второго, то обменяем эти элементы

Задача 3. Решение (59 баллов)

- Все элементы первого множества меньше элементов второго множества
- Размер первого множества больше либо равен размеру второго

  Тогда ответом будет являться максимальный элемент второго множества

- Для n <= 8000 будем для каждой галактики запоминать два множества и в любом порядке будем их объединять

Задача 3. Решение (100 баллов)

- Объединяем меньшее множество с большим
- Худший случай: n log n объединений
- Время работы $O(n(\log n)^2)$

Задача 4. Города на Венере

- Два игрока играют в слова по классическим правилам
- Игроки знают одинаковый набор слов
- Все слова состоят только из букв a, b, c
- Необходимо определить, кто затащит

Простые решения

- Пусть слова состоят только из букв a, b
- Тогда можно анализировать различные случаи
- Сам анализ оставляем в качестве упражнения :)
- Получаем 26 баллов за подзадачу

Простые решения

- Пишем динамическое программирование
- dp[mask][last] – выиграет ли текущий игрок, если уже назвали все слова в маске mask, а последнее названное слово – last
- Как пересчитывать?

Простые решения

- Если mask содержит все биты – текущий игрок проиграет
- Иначе перебираем все состояния, в которые мы можем перейти
- Для этого переберем, какое следующее слово мы возьмем
  - Учитываем, что оно должно начинаться на ту же букву, на которую заканчивается предыдущее, и еще не содержится в mask

Простые решения

- Если одно из состояний, в которое мы перейдем, проигрышное – то мы можем выиграть, придя в него
- Если мы можем перейти только в выигрышные состояния, то проигрываем
- Такое решение работает за $O(2^n \cdot n^2)$ и берет 19 баллов
- Как улучшить?

На пути к полному решению

- Заметим, что буквы посередине не важны
- Тогда остается только 9 типов слов: «aa», «ab», «ac», «ba», «bb», «bc», «ca», «cb», «cc».
- Тогда мы можем посчитать десятимерную (o_O) динамику
  - dp[aa][ab][ac][ba][bb][bc][ca][cb][cc][last]
  - Здесь храним количество слов данного типа и тип последнего слова
- Пересчитываем аналогично предыдущему случаю
- Это работает при $n \leq 40$

На пути к полному решению

- Представим граф из вершин «a», «b», «c» и слова в качестве ребер
  - Например, слово «ab» – ребро между a и b
- Тогда если есть ребра вида «ab» и «ba», то их можно удалить
- Аналогично удаляем пару ребер вида «aa»
- Почему так можно делать?

На пути к полному решению

- Пусть какой-то игрок выигрывает, не используя «ab» и «ba»
- Тогда он выигрывает и при добавлении этих ребер
- Если его соперник использует одно из этих ребер, то мы используем противоположное
- Иначе мы не используем эти ребра

На пути к полному решению

- После таких удалений остается не более трех ребер вида «aa», «bb», «cc»
- Также из ребер вида «ab» и «ba» останется только один тип ребер
- Таким образом, число измерений в динамике заметно сокращается
- Получаем dp[mask][ab][bc][ac][last] – $O(n^3)$ состояний
- Это 60 баллов

Последний рывок

- Еще уменьшим число состояний
- Рассмотрим два случая:
- Если ребра вида ab, bc, ac образуют цикл – разность в количестве взятий этих ребер не превосходит 1
- Если какое-либо из ребер не входит в цикл – его можно использовать не более одного раза

Последний рывок

- Таким образом, мы получаем всего O(n) состояний динамики
- Данное решение уже проходит тесты и берет полный балл
- Существуют альтернативные решения – например, с разбором случаев

Typ 2

Задача 1. Lines 2020

- n разноцветных шаров, расположенных в ряд
- i-й шар имеет цвет ci
- За один ход можно уничтожить какой-либо шар
- При этом если сосед или соседи этого шара имели такой же цвет, то они также уничтожатся, далее уничтожатся соседи соседей, имеющие такой же цвет, и так далее

- По правилам игры последовательность не может остаться пустой
- Требуется минимизировать сумму номеров цветов

Задача 1. Решение (100 баллов)

- Заметим, что нам не выгодна ситуация, когда после удаления шаров на их месте образуется последовательность шаров одинакового цвета большей длины
- Тогда найдем последовательность шаров одинакового цвета с минимальной суммой
- Сначала слева направо пройдемся до этой последовательности и удалим шары, а затем – справа налево

Задача 2. Радиотелескопы

- Дана последовательность из n чисел
- Необходимо оставить w < k чисел таким образом, чтобы значение

  $(a_1 \text{ xor } a_2) + (a_2 \text{ xor } a_3) + \ldots + (a_{w-1} \text{ xor } a_w)$

  было максимальным

Случай k = n

- Пусть k = n
- Тогда заметим, что нам выгодно всегда оставить все n чисел
- Почему?
- Требуется показать, что

  $(x \text{ xor } y) + (y \text{ xor } z) \geq x \text{ xor } z$

  - Это можно сделать побитово
- Тогда получим, что если мы удалили какой-то элемент, то его добавление не ухудшит ответ

Случай $k \neq n$

- Используем динамическое программирование
- dp[n][k] – максимальная сумма xor'ов, если мы рассмотрели первые n чисел, при этом мы оставили k чисел, одно из которых – a_n.
- Начальные значения: dp[n][1] = 0 для всех n
- Как пересчитывать?

Случай $k \neq n$

- Рассмотрим dp[n][k], $k \geq 2$
- Последний взятый элемент – a_n, переберем предыдущий
- Тогда получаем, что

  dp[n][k] = max(dp[w][k-1] + (a_w xor a_n))

  для всех w от 1 до n-1
- Сложность – $O(n^2 \cdot k)$

Полное решение

- Первое решение работает на подзадачу 4, второе решение – на подзадачи 1-3
- В совокупности оба решения дают полный балл по задаче

Задача 3. Добыча меди

- Даны n городов
- Требуется все города разбить на два района
- Полученные районы можно далее рекурсивно разбить на два района, полученные – еще на два района и т. д.
- Мы можем в любой момент остановить разбиение
- Получаем бинарное дерево

Задача 3. Добыча меди

- Стоимость вершины в дереве определяется таким образом:
  - для листовой вершины – среднее арифметическое по всем городам
  - для не-листовой вершины – среднее арифметическое по сыновьям
- Максимизировать стоимость корня

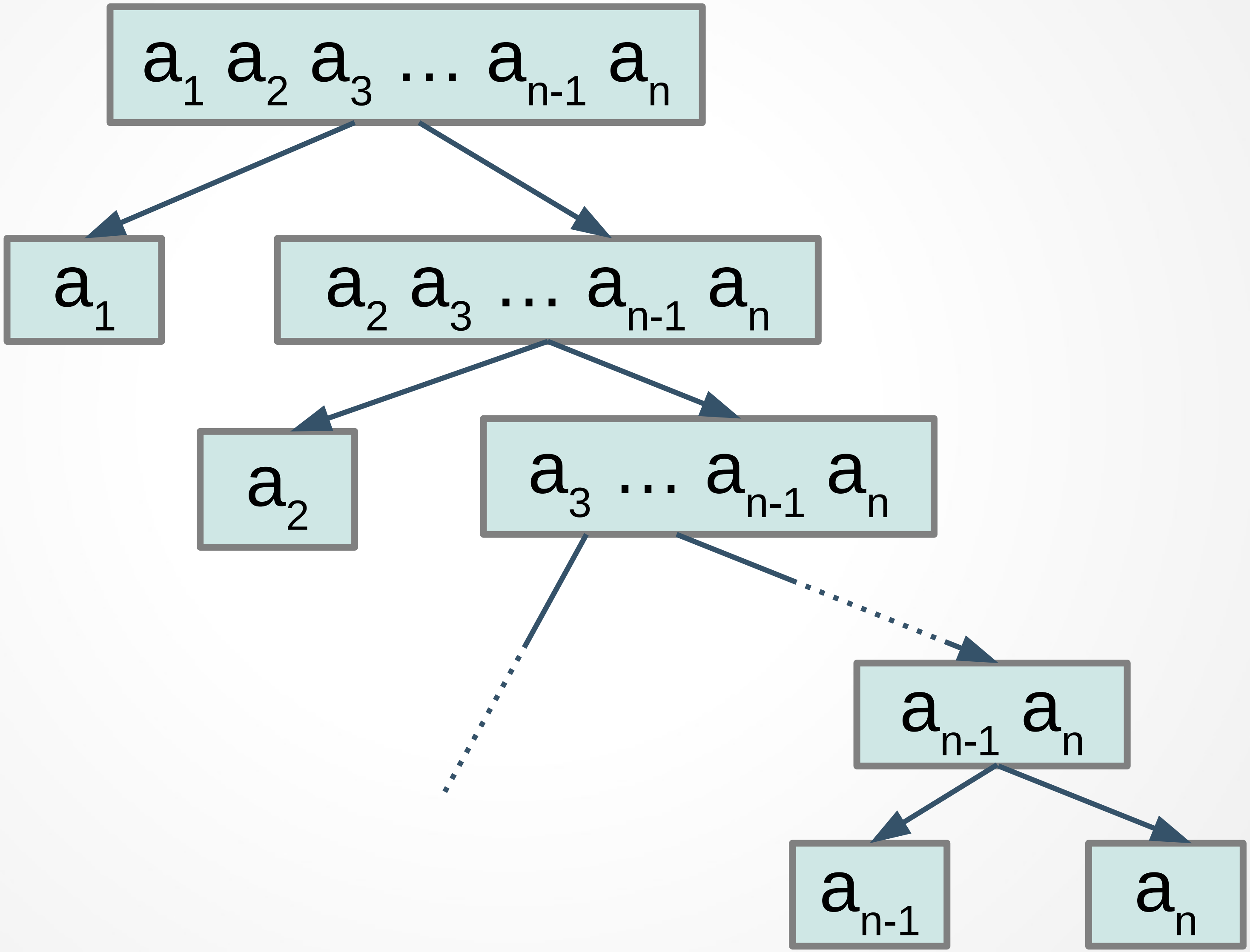
Оптимальная конструкция

- Как будет выглядеть оптимальное бинарное дерево в нашем случае?
- Отсортируем города по убыванию
- Пусть список городов в порядке убывания

  $a_1, a_2, \ldots, a_n$.

Оптимальная конструкция

- Тогда бинарное дерево выглядит так:



Простое решение

- Рассмотрим случай q=1
- Мы можем просто построить бинарное дерево и посчитать «в лоб» стоимость
- Предварительно надо отсортировать отрезок
- Решение берет 46 баллов
- Как улучшить?

Подсчет стоимости

- Давайте посмотрим, как считать стоимость в оптимальном случае
- Нетрудно заметить, что эта величина равна

  $1/2 * a_1 + 1/4 * a_2 + 1/8 * a_3 + \ldots + 1/2^n * a_n$

- Заметим, что какой-то вклад в результат вносят только где-то первые 60 элементов
- Остальные можно просто не учитывать

Полное решение

- Таким образом, решение сводится к поиска примерно 60 максимальных элементов на отрезке
- Это можно, например, сделать с помощью обычного дерева отрезков на максимум
- Для этого мы делаем следующее: находим максимум на отрезке, удаляем его, далее ищем следующий, удаляем его и т. д.
- После ответа на запрос удаленные элементы возвращаем назад

Полное решение

- Сложность – O(n*log n*60)
- Такое решение берет полный балл

Задача 4. Космическая реклама

- Строка длины n
- Можно заменить до k символов на любые другие
- Необходимо максимизировать количество палиндромов

Задача 4. Решение (17 баллов)

- Отошлем входные данные
- Получим 17 баллов

Задача 4. Решение (38 баллов)

- Заменим первые k букв на букву ‘a’
- Получим 38 баллов

Задача 4. Решение (100 баллов)

- Можно заметить, что в строке, состоящей из одинаковых букв, длины n, палиндромов больше, чем в двух строках длины n/2 из одинаковых букв.
- Поэтому будет выгодно заменять буквы так, чтобы в строке последовательно шло большое количество одинаковых букв.

Задача 4. Решение (100 баллов)

- Реализуем жадно
- Переберем позицию, с которой мы начнем заменять буквы
- Затем переберем букву, на которую мы будем менять буквы, не равные ей
- Выберем строку с максимальным количеством палиндромов и выведем ее
- Такое решение берет 100 баллов на данных тестах

Задача 4. Оптимальное решение

- Жюри не знает такого
- Однако решения выше достаточно для полного балла по задаче
- Возможно, существуют и другие подходы

Спасибо за внимание!

Вопросы?