# Вычислительная геометрия

Борис Золотов    Матвей Магин

9 июня 2022 г.

Летняя школа МКН СПбГУ

# Содержание

# О-нотация

# Идея $O$

Пусть некоторая программа принимает на вход данные. Она, скорее всего, работает тем дольше, чем больше данных поступило на вход. Мы хотим оценить время работы программы, сравнивая его с известными функциями $\mathbb{N} \to \mathbb{N}$.

# Определение из анализа

**Определение**

Функции $f, \ g : \ \mathbb{R} \to \mathbb{R}$. Говорят, что

$$f(x) = O(g(x)), \quad \text{если} \quad \exists\, C \, \forall\, x \ |f(x)| \leq C \cdot |g(x)|.$$

# Определение для нас

**Определение**

Время работы программы составляет $O(f(n))$, если $\exists\, C$: для входа *размера* $n$ количество операций составит не более $f(n) \cdot C$.

- Какие операции считаются в данном определении, обычно известно из контекста.
- Если вход состоит из *одного числа*, размер входа равен 1, вне зависимости от величины числа.

# Поиск наибольшего элемента в массиве

**Задача**

Дан массив из $n$ целых чисел, найти в нём наибольший элемент.

Отсмотрим все элементы по порядку, храня наибольший отсмотренный элемент. Мы обращаемся к каждому элементу по разу, поэтому $O(n)$. Быстрее, очевидно, нельзя.

[ 1 4 4 2 8 1 6 9 8 ]

max to date: 8

# Наличие элемента в отсортированном массиве

**Задача**

Дан массив $M$ из $n$ целых чисел, они упорядочены по возрастанию.
Дано число $k$ — проверить, есть ли оно в $M$.

Сравним $k$ и элемент в середине $M$ — назовём его $m$.
Если $m > k$, ищем в первой половине массива. Иначе во второй.

$$[\ 1\quad 2\quad 3\quad 5\quad 8\quad 13\quad 21\quad 34\quad 55\ ]$$

здесь? $\leftarrow$ | $\rightarrow$ или здесь?

# Как оценить время работы

Пусть $T(n)$ — время работы на массиве из $n$ элементов. Тогда

$$T(n) = 1 + T\left(\frac{n}{2}\right).$$

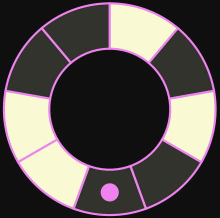
Отсюда

$$T(n) = O(\log_2(n)).$$

# Количество комнат на космическом корабле

**Задача**

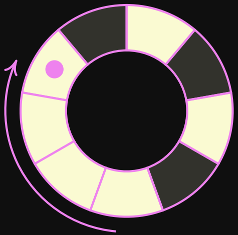
Космический корабль в форме кольца разделён на $n$ одинаковых комнат, в которых горит или не горит свет. Можно переходить между соседними комнатами, включать или выключать свет. Найти $n$.

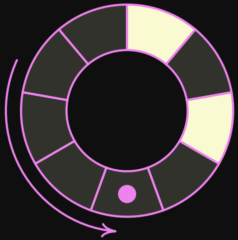


# Простое решение

Процедура проход($k$) — пройти на $k$ комнат вперёд, включая в каждой комнате свет (может быть что угодно).

Потом пройти на $k$ комнат назад, выключая в каждой комнате свет (должен быть включен).





# Оценка времени

Выполним проход(1), проход(2), проход(3), ...

В какой-то момент мы вернёмся, чтобы выключить свет в первой комнате, а он уже будет там выключен.
Значит, мы прошли полный круг.

$$2 \cdot (1 + 2 + \ldots + n) = O\left(n^2\right) \quad \text{переходов.}$$

# Решение за линейное время

Пусть $2^{k-1} \leq n < 2^k$.

Выполним $\text{проход}(1), \text{проход}(2), \ldots, \text{проход}\left(2^i\right), \ldots$

Каждый раз длина прохода увеличивается в два раза.
В какой-то момент на обратном пути мы придём
в комнату, свет в которой уже выключен.

$$2 \cdot \left(1 + 2 + 4 + \ldots + 2^k\right) \ = \ 2 \cdot \left(2^{k+1} - 1\right) \ \leq \ 8 \cdot n \ = \ O(n).$$

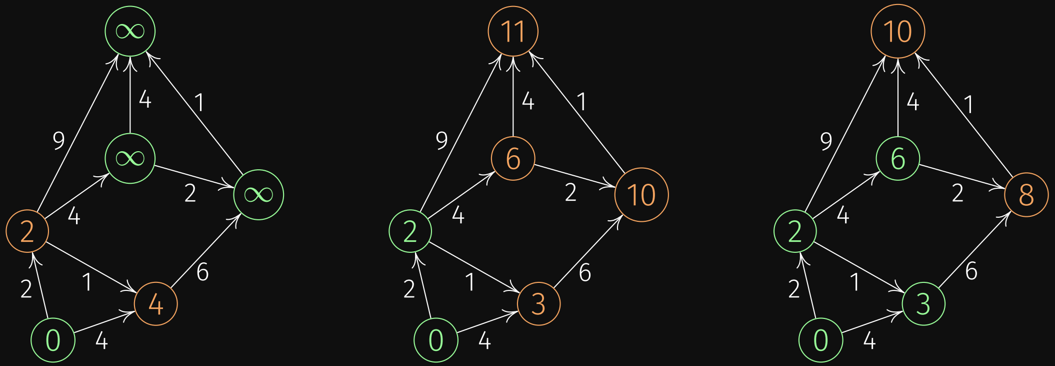
# Алгоритм Беллмана — Форда

**Задача**

Дан граф $G = (V, E)$, на рёбрах расставлены неотрицательные веса. Выбрана $v \in V$, найти наименьшие веса путей от $v$ до каждой из остальных вершин.

Изначально $d(v_i) = +\infty$ для всех $v_i$.

# Идея




Считаем, что все $d(v_i)$ обновляются одновременно. В реальности дело чуть лучше.

# Улучшение $d(v_i)$

**for** $i = 1$ to $|V| - 1$ **do**

  **for all** $e = (u_1, u_2) \in E$ **do**

    **if** $d(u_1) + \text{weight}(e) < d(u_2)$ **then**

      $d(u_2) \leftarrow d(u_1) + \text{weight}(e)$ ▷ Пытаемся с помощью ребра $e$ улучшить путь в $v_2$

    **end if**

  **end for** ▷ Нашли оптимальные пути длины $\leq i$

**end for**

# Асимптотика с несколькими параметрами

Время работы оценивается как $O(|V| \cdot |E|)$.

Мы, конечно, знаем, что $|V| \leq |E| + 1$ и $|E| \leq |V|^2$,
но зачастую нам нужна точная оценка
относительно обоих параметров входа.

# Замечания

- $\log_a n \ll n^k \ll a^n$
- $\log_a n \sim \log_b n$
- $\log_a (x^k) \sim \log_a x$