ITIVITI

LET IT TRADE

Алгоритмы и структуры данных

Косяков Михаил Сергеевич
к.т.н., доцент кафедры ВТ

... и многие другие

https://vk.com/algoclass_2020

УНИВЕРСИТЕТ ИТМО

Мотивация

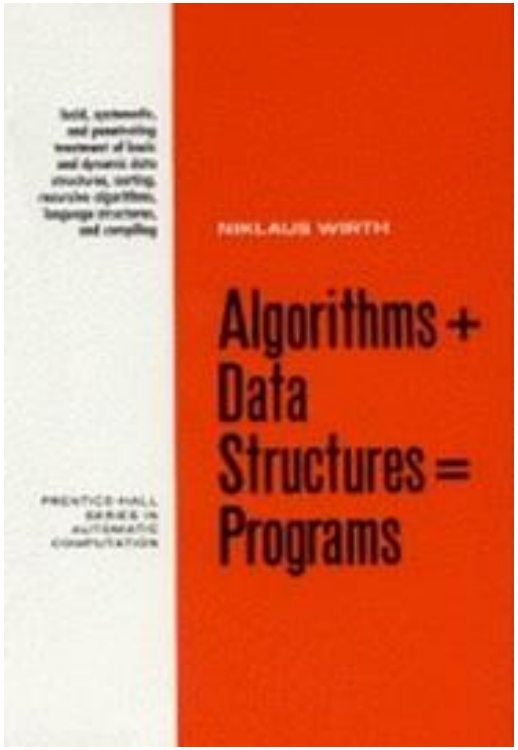
ITIVITI

LET IT TRADE

- Повышение уровня знаний в области Computer Science у выпускников технических специальностей ВУЗов
- Получение знаний из production - от компании занимающейся разработкой высокопроизводительного программного обеспечения
- Подготовка кадров среди студентов для дальнейшего сотрудничества в области разработки высокопроизводительного программного обеспечения

УНИВЕРСИТЕТ ИТМО

О чем речь?



«Алгоритмы + структуры данных = программы»

Никлаус Вирт (Niklaus Wirth)

Содержание курса

- Введение в теорию алгоритмов
- Алгоритмы сортировок
- Структуры данных
  - Линейные структуры
  - Бинарные деревья поиска
  - Хеши и хеш-функции
- Алгоритмы на графах
  - Обходы графов в ширину и глубину
  - Минимальные остовные деревья
  - Поиск кратчайших путей в графе

Практические занятия

ITIVITI

- Задачи по пройденным темам
- Автоматическая система проверки
- Базовые задачи: https://ipc.susu.ru/exercises-2.html?theme=L20
- Основные задачи: http://acm.timus.ru
- Регистрация: vtalgo20_## (первая буква имени и фамилия)
- 12 базовых задач
- 5 задач по каждой теме (в сумме 20 задач)
- Язык программирования: C / C++ (namespace std)
- $ clang-format [target file] или
- http://format.krzaq.cc/ LLVM format

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

Правила игры

- Защита только трех или пяти задач по теме
- Дедлайны сдачи задач (каникулы с 16.04.2020 по 18.04.2020)
  - Введение в алгоритмы: 13 марта 2020
  - Сортировка: 27 марта 2020
  - Структуры данных: 24 апреля 2020
  - Алгоритмы на графах: 22 мая 2020
- Решения необходимо прислать по e-mail (крайний срок 18:00 за день до защиты)
  - Словесное описание решения, код в приложении к сообщению
  - Имя файла с решением:
    <№группы>_<первая буква имени и фамилия>_<№задачи>
- E-mail: vtalgo@itiviti.com

Оценки

ITIVITI

- Оценка 4C: минимум три задачи по каждой теме (12 задач)
- Оценка 4B и допуск на экзамен: все 20 основных задач
- Оценка 5A: автоматом не ставится
- Каждый последующий пункт включает предыдущий
- На экзамене можно получить любую оценку
- Оценка 3D: все базовые задачи, определения, доказательство корректности, вывод формул, оценка сложности

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

Литература

- Алгоритмы и структуры данных:
  - Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ (2-е или 3-е издание)
  - Ахо А., Хопкрофт Д., Ульман Д. Структуры данных и алгоритмы
  - Седжвик Р. Алгоритмы на C++
  - Кнут Д. Искусство программирования (1, 2, 3-й тома)
- Язык программирования C / C++:
  - Страуструп Б. Язык программирования C++
  - Керниган Б., Ритчи Д. Язык C
  - Шилдт Г. Полный справочник по C++

ITIVITI

LET IT TRADE

Введение в теорию алгоритмов

УНИВЕРСИТЕТ ИТМО

Мотивация

ITIVITI

LET IT TRADE

- Представьте, что:
  - браузер загружает страницы и видео в 3-4 раза медленнее обычного
  - оплата по кредитной карте в магазине занимает 10-15 минут
  - чтобы набрать букву в СМС-ке смартфону требуется 5 секунд
  - и т.д. и т.п.
- Вот что может произойти если пользоваться неэффективными алгоритмами!

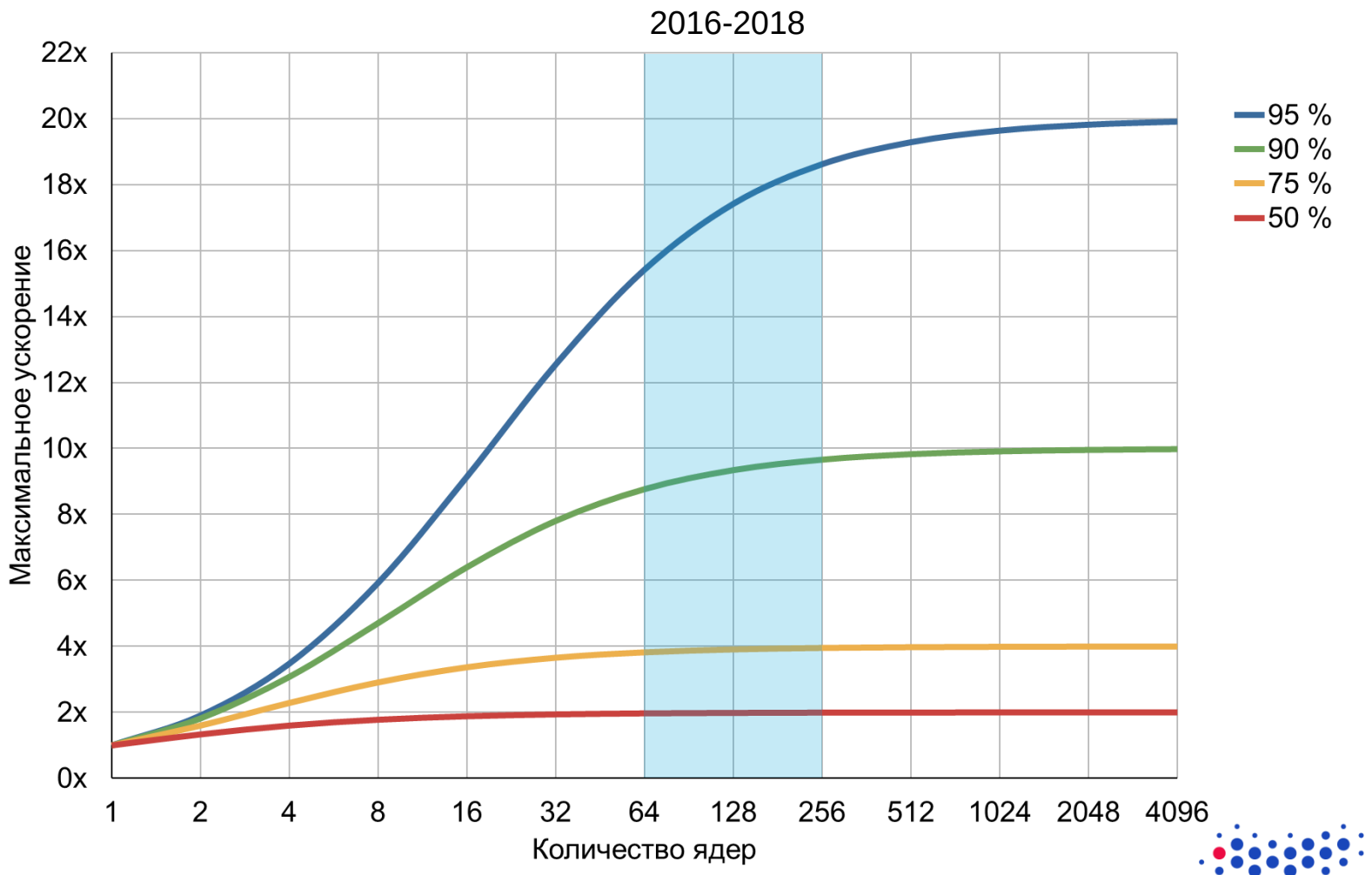
Мало мощности?

- Рассмотрим задачу:
  - Время работы алгоритма 1 пропорционально функции $f = N^2$
  - Время работы алгоритма 2 пропорционально функции $f = N \log N$
- Есть два компьютера
  - Компьютер А выполняет 10^9 операций в секунду (частота ~ ГГц)
  - Компьютер Б выполняет 10^6 операций в секунду (частота ~ МГц)

Мало мощности?

- При $N = 10^8$ (всего 100 МБ):
- Компьютер А (быстрый) с помощью алгоритма 1 (медленный):
  - $10^8 \times 10^8 = 10^{16}$ операций
  - т.е. за $10^{16} / 10^9 = 10^7 \sim 2777.7$ часов или ~ 115.7 дней
- Компьютер Б (медленный) с помощью алгоритма 2 (быстрый):
  - $10^8 \times \log 10^8 = 10^8 \times 26.57$ операций
  - т.е. за $10^8 \times 26.57 / 10^6 = 2657$ секунд ~ 0.74 часа
- Очень медленный компьютер с помощью быстрого алгоритма решает задачу в **~ 3753** раза быстрее чем быстрый компьютер с медленным алгоритмом

Мало ядер? Закон Амдала



Вывод

ITIVITI

LET IT TRADE

Необходимо использовать ЭФФЕКТИВНЫЕ алгоритмы

Что значит эффективные?

УНИВЕРСИТЕТ ИТМО

Вычислительная сложность алгоритма

ITIVITI

- В каких единицах лучше измерять время работы алгоритма?
  - Важно сравнивать алгоритмы между собой
- От чего зависит время работы (реализации) алгоритма?
  - От *N* – мера размера задачи
  - От входных данных
  - От платформы исполнения
  - От языка программирования и компилятора
- Время выполнения ≈ число выполненных операций (строк кода)
- Что такое одна операция?

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

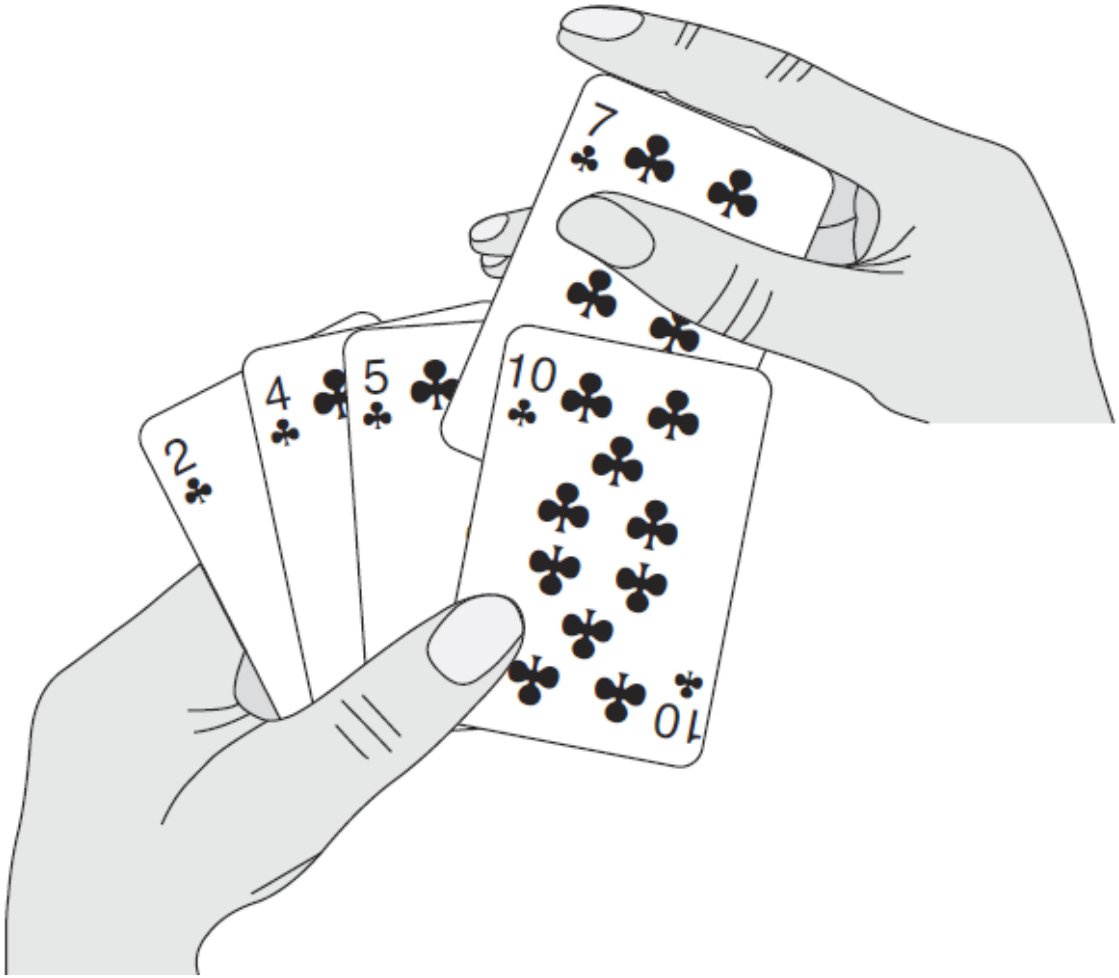
Вычислительная сложность алгоритма

ITIVITI

LET IT TRADE

- Какой из алгоритмов лучше?
  - $f_1(N) = 239 \times N^2 + 30 \times N + 566$
  - $f_2(N) = 10 \times N^3$
  - При маленьких N? При больших N?
- Постоянные множители будут варьироваться в зависимости от языка программирования, компилятора, архитектуры компьютера…
- $f(N)$ – это какое время?

Сортировка вставкой



Сортировка вставкой

```
1.void insertion_sort(int * a, int n)
2.{
3.     int i, j, t;
4.     for(i = 1; i < n; ++i) {
5.         t = a[i];
6.         j = i;
7.         while (j > 0 && a[j-1] > t) {
8.             a[j] = a[j-1];
9.             --j;
10.         }
11.         a[j] = t;
12.     }
13.}
```

Инвариант: в начале каждого цикла `for` подмассив `a[0..`*i*`-1]` состоит из отсортированных элементов изначального подмассива `a[0..`*i*`-1]`

Average case vs Worst case analysis

ITIVITI

- Среднее время работы
  - Дает среднее время выполнения при определенных предположениях о частоте появления того или иного входа
  - Требует знания предметной области
- Наихудшее время работы
  - Дает верхнюю оценку времени выполнения вне зависимости от входных данных
  - Не требует знания предметной области
  - Позволяет гарантировать время выполнения
  - Легче рассчитывается
  - Среднее часто ведет себя как наихудшее

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

Вычислительная сложность алгоритма

- Пренебрежем постоянными множителями, членами меньшего порядка и сфокусируемся на больших N
  - Математически много проще ☺
  - Непонятно как считать постоянные множители
  - На самом деле почти ничего не теряем!
  - На маленьких задачах и так все быстро (тут постоянные множители важны!)
  - Закон Мура <-> вычислительные потребности
- Время работы $f(N)$ пропорционально:
  - $1, \log N, N, N \log N, N^2, N^3, 2^N$
  - Насколько увеличится $f(N)$ при $N \to 2N$?

Асимптотическая эффективность алгоритма

ITIVITI

LET IT TRADE

- Лучший алгоритм ≈ наихудшее время работы алгоритма растет наиболее медленно с увеличением размера входной задачи (т.е. смотрим только на порядок роста)
  - Цель – линейный рост
- Какой из алгоритмов лучше?
- $f_1(N) = 239 \times N^2 + 30 \times N + 566$
- $f_2(N) = 10 \times N^3$
  - $f_1(30) = 216\,566$; $f_2(30) = 270\,000$
  - $f_1(100) = 2\,393\,566$; $f_2(100) = 10\,000\,000$
  - $f_1(1000) = ???$; $f_2(1000) = ???$
- Что лучше: N^{100} или 2^N?

УНИВЕРСИТЕТ ИТМО

Асимптотические обозначения

ITIVITI

LET IT TRADE

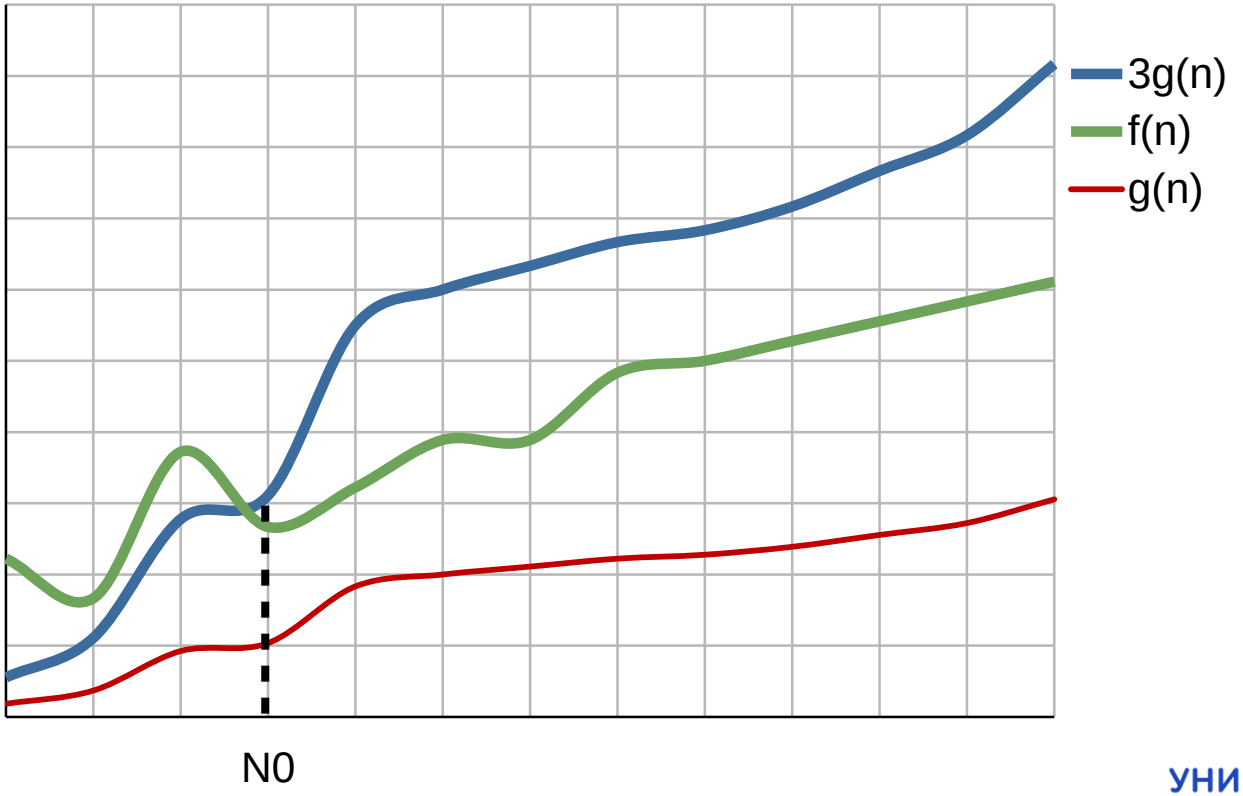
А как нам описать время выполнения безотносительно ко входным данным, а не только в наихудшем случае?

Асимптотические обозначения для описания времени работы алгоритмов

УНИВЕРСИТЕТ ИТМО

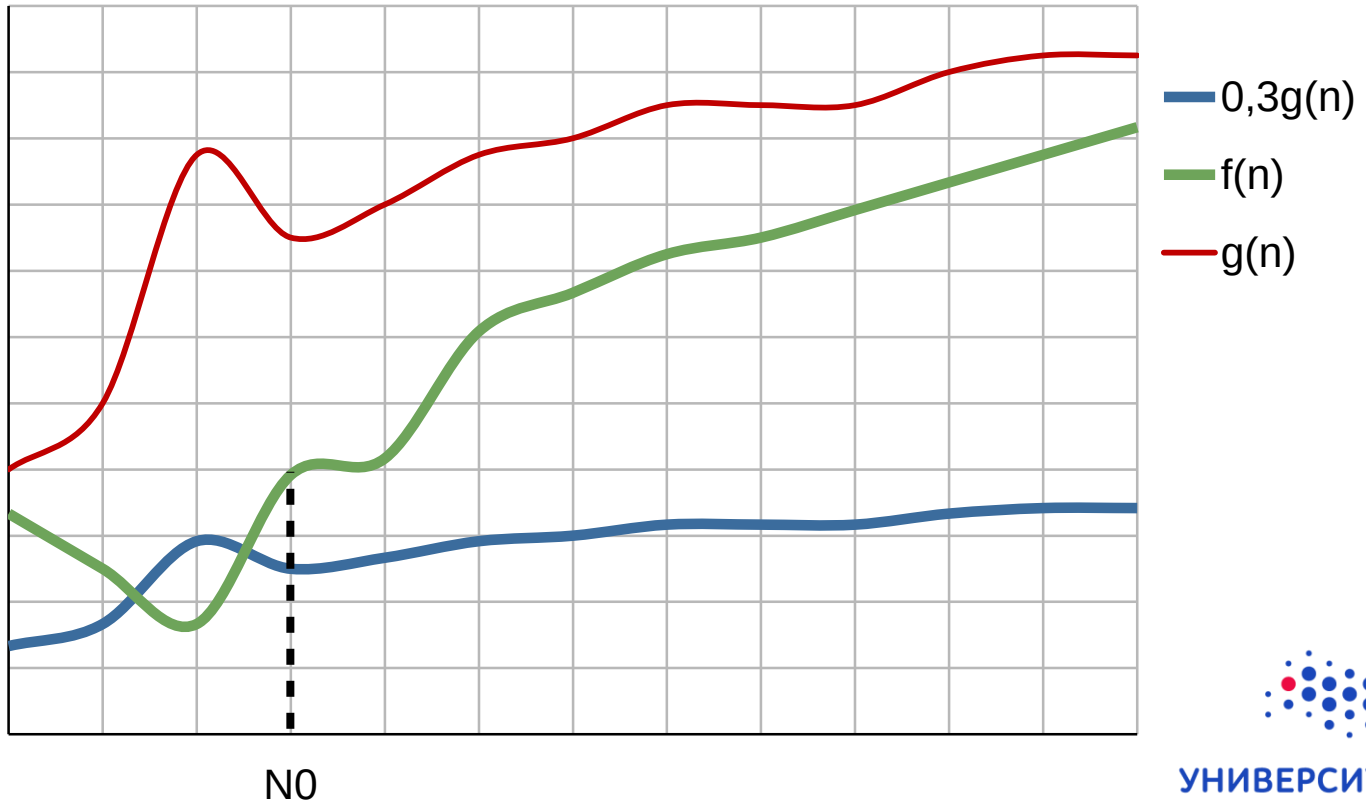
Асимптотические обозначения

- *O*-обозначения: асимптотическая верхняя граница
- Говорим, что $f(N) = O(g(N))$, если $\exists\ c$ и $N_0 > 0$ такие, что $0 \le f(N) \le cg(N)$ для всех $N \ge N_0$



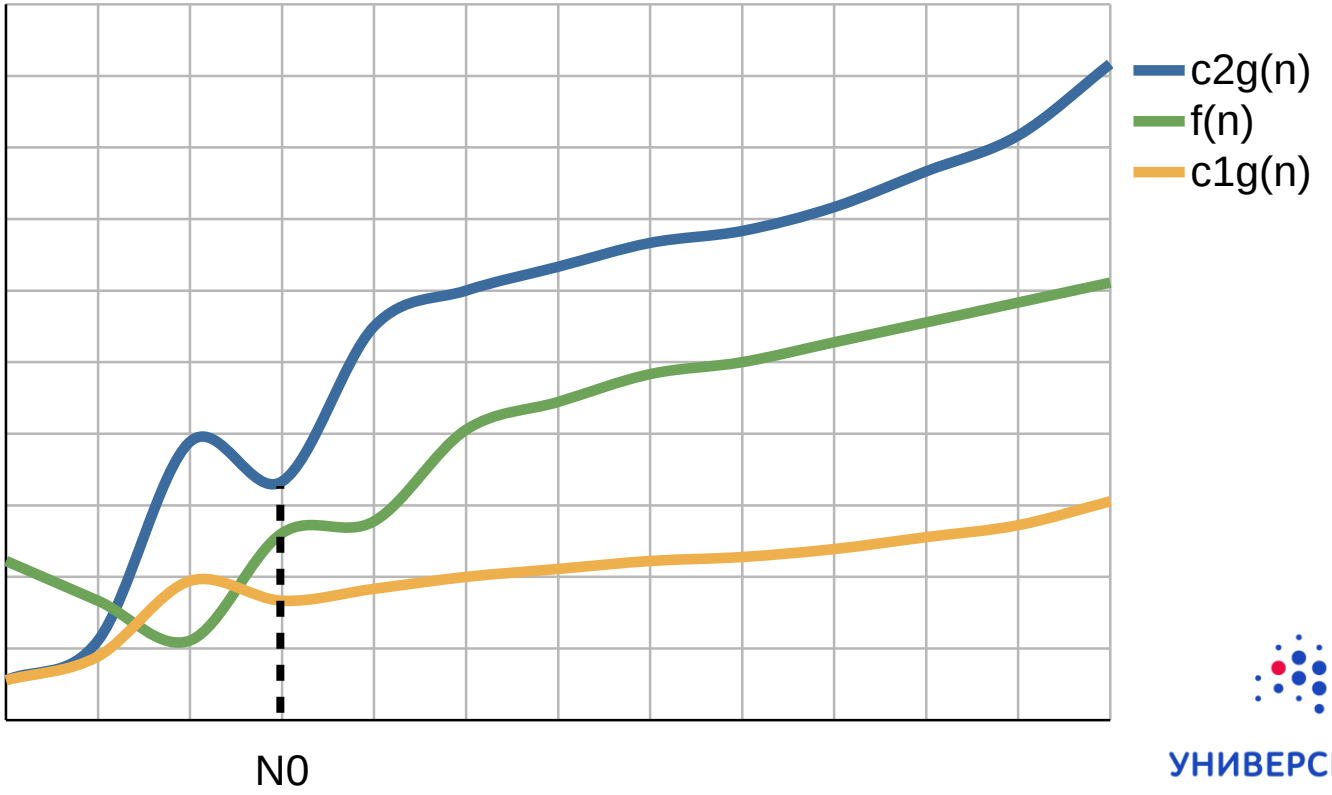
Асимптотические обозначения

- Ω-обозначения: асимптотическая нижняя граница
- Говорим, что $f(N) = \Omega(g(N))$, если $\exists\ c$ и $N_0 > 0$ такие, что $0 \le cg(N) \le f(N)$ для всех $N \ge N_0$



Асимптотические обозначения

- Θ-обозначения: асимптотически точная оценка
- Говорим, что $f(N) = \Theta(g(N))$, если $\exists$ c_1, c_2 и $N_0 > 0$ такие, что $0 \le c_1 g(N) \le f(N) \le c_2 g(N)$ для всех $N \ge N_0$



Асимптотические обозначения

- Пусть есть функция $f(N) = 239 \times N^2 + 30 \times N + 566$
- Что верно?
  - $f(N) = O(N)$
  - $f(N) = O(N^2)$
  - $f(N) = \Omega(N)$
  - $f(N) = \Theta(N^2)$
  - $f(N) = O(N^3)$
- Как можно выразить постоянную функцию?

Сортировка вставкой

```
1.void insertion_sort(int * a, int n)
2.{
3.     int i, j, t;
4.     for(i = 1; i < n; ++i) {
5.         t = a[i];
6.         j = i;
7.         while (j > 0 && a[j-1] > t) {
8.             a[j] = a[j-1];
9.             --j;
10.         }
11.         a[j] = t;
12.     }
13.}
```

$T(N) = O(N^2)$ – верхняя граница для наихудшего случая

$T(N) = \Omega(N)$ – нижняя граница для наилучшего случая

Верно ли, что $\Omega(N^2)$ - время работы в наихудшем случае?

Итоги

- Полином N^x растет быстрее N^y при $x > y$
- Любая экспонента растет быстрее любого полинома
- Любой полином растет быстрее любого полилогарифма
- На сладкое:
  - Асимптотика арифметической прогрессии
  - Асимптотика геометрической прогрессии
  - $T(N) = T(N/2) + T(N/2) + 1$

ITIVITI

LET IT TRADE

Спасибо за внимание!

УНИВЕРСИТЕТ ИТМО