Алгоритмы

Михаил Каменщиков

Обо мне

- лид команды Рекомендаций в Авито
- проводил много собеседований по алгоритмам
- закончил мехмат НГУ и ШАД
- ~~slack~~mm: @makamenshchikov
- tg: @greenwo1f

Формат курса

- 10(11) занятий с теорией и практикой, длительностью **3** часа.
- Время - **среда, 18:00**
- Домашки на Stepik после каждого занятия (дедлайн - 2 недели)
- Теоретический квиз + контест в конце курса

Цели курса

- Научиться оценивать эффективность программ, находить узкие места
- Знать, как устроены основные структуры данных, когда их нужно использовать
- Понимать, что происходит внутри библиотек, баз данных и какие это накладывает ограничения
- Уметь решать несложные задачи (leetcode easy/medium)

Оценка

- За **домашки на Stepik** можно будет получить **60** баллов
- Для 3 домашек будет **Code Review на GitHub** - **15** баллов
- За **квиз по теории** можно получить **20** баллов
- За **контест** можно получить **30** баллов
- Всего можно набрать **125** баллов
- **40+** - зачет, **60+** - хорошо, **90+** - отлично. Для зачета необходимо набрать не менее **20** баллов за домашки.
- Топ-3 курса по баллам и топ-3 в контесте получат книгу Кормэна :)

Темы курса

- Введение, жадные алгоритмы
- Базовые структуры данных
- Рекурсия
- Сортировки
- Кучи
- Бонус про мировые константы
- Бинарные деревья поиска
- Хэш-таблицы
- Графы
- Динамическое программирование
- Строки

Правила

- Кто включает камеру - тот молодец :)
- Задавать вопросы можно (и нужно!) по ходу лекции, в чате или голосом. Если голосом, то лучше «поднять руку» в зуме.
- Заполнять после занятия форму фидбека. Это очень помогает делать занятия лучше.

Зачем датасаентистам и аналитикам знать алгоритмы?

Как построить рекомендации

- На Авито более **110 миллионов** активных объявлений и порядка **20 миллионов** пользователей в день
- В результате работы ML алгоритма у нас получается матрица размера 110млн x 20млн, в каждой ячейке которой лежит **score** - насколько объявление подходит пользователю.
- Что здесь не так?
- Количество элементов в этой матрице: $110 * 10^6 * 20 * 10^6 = 2.2 * 10^{15}$
- Как нам выбрать 100 подходящих объявлений для каждого пользователя?

Опрос в Zoom

План занятия

- Узнаем, как оценивать сложность алгоритмов
- Посмотрим, как сложность влияет на время работы
- Познакомимся с жадными алгоритмами на примере задач
- Напишем код и научимся честно измерять время выполнения
- Разберемся с системой сдачи домашек

Что такое алгоритм и его сложность

- Алгоритм - это некоторая [конечная] последовательность простых операций, преобразующая входные данные в результат
- Сложность алгоритма - функция зависимости количества «простых операций» от размера входных данных
- Что такое «простая операция»?
- А может ли сложность зависеть еще от самих данных?

Поиск элемента в массиве

- Дан массив со случайными элементами
- Нужно ответить на вопрос - есть ли в массиве элемент X?
- Пройдя по массиву один раз мы точно сможем дать ответ
- А что, если элементы массива - это большие объекты (например, тексты)

Перерыв

Хороший, плохой, средний

- Врем работы алгоритма обычно зависит от размера входных данных, но также может зависеть и от характера данных
- Например: при поиске элемента в массиве мы можем найти элемент сразу, или не найти вовсе
- А если нам нужно посчитать сумму массива - вариантов нет
- Оценка в среднем - мат. ожидание времени работы по всем возможным наборам входных данных

Асимптотическая оценка сложности

- $f(n) = O(g(n)) \iff \exists\ c, N_0 : f(n) \leq cg(n)\ \forall\ n > N_0$
- $f(n) = \Omega(g(n)) \iff g(n) = O(f(n))$
- $f(n) = \Theta(g(n)) \iff f(n) = O(g(n)); f(n) = \Omega(g(n))$

Асимптотическая оценка сложности

- $f(n) = O(g(n))$ - если существуют константы c и N_0 такие, что $f(n) \leq cg(n)$ для всех $n > N_0$
- $f(n) = \Omega(g(n))$ - если $g(n) = O(f(n))$
- $f(n) = \Theta(g(n))$ - если $f(n) = O(g(n))$ и $g(n) = O(f(n))$

Асимптотическая оценка сложности

- $f(n) = O(g(n))$ - оценка сложности сверху [O-большое, Big-O]
- $f(n) = \Omega(g(n))$ - оценка сложности снизу [Омега, omega]
- $f(n) = \Theta(g(n))$ - точная оценка сложности [Тета, theta]

Как оценивать сложность

- Допустим, наш алгоритм совершает $10n^3 + 3n^2 + 4$ операций
- Смотрим только на самое быстрорастущее слагаемое и отбрасываем константы
- Получаем $10n^3 + 3n^2 + 4 = O(n^3)$
- Формально, для $N_0 = 1$ и $c = 20$ выполняется определение:
  $f(n) = O(g(n))$ - если существуют константы c и N_0 такие, что
  $f(n) \leq cg(n)$ для всех $N > N_0$

Скорость роста функций

- Любая степень $log(n)$ растет медленней любой степени n
- a^n растет быстрее, чем любая степень n
- как быть с $n!$?
- $n!$ растет быстрее чем a^n

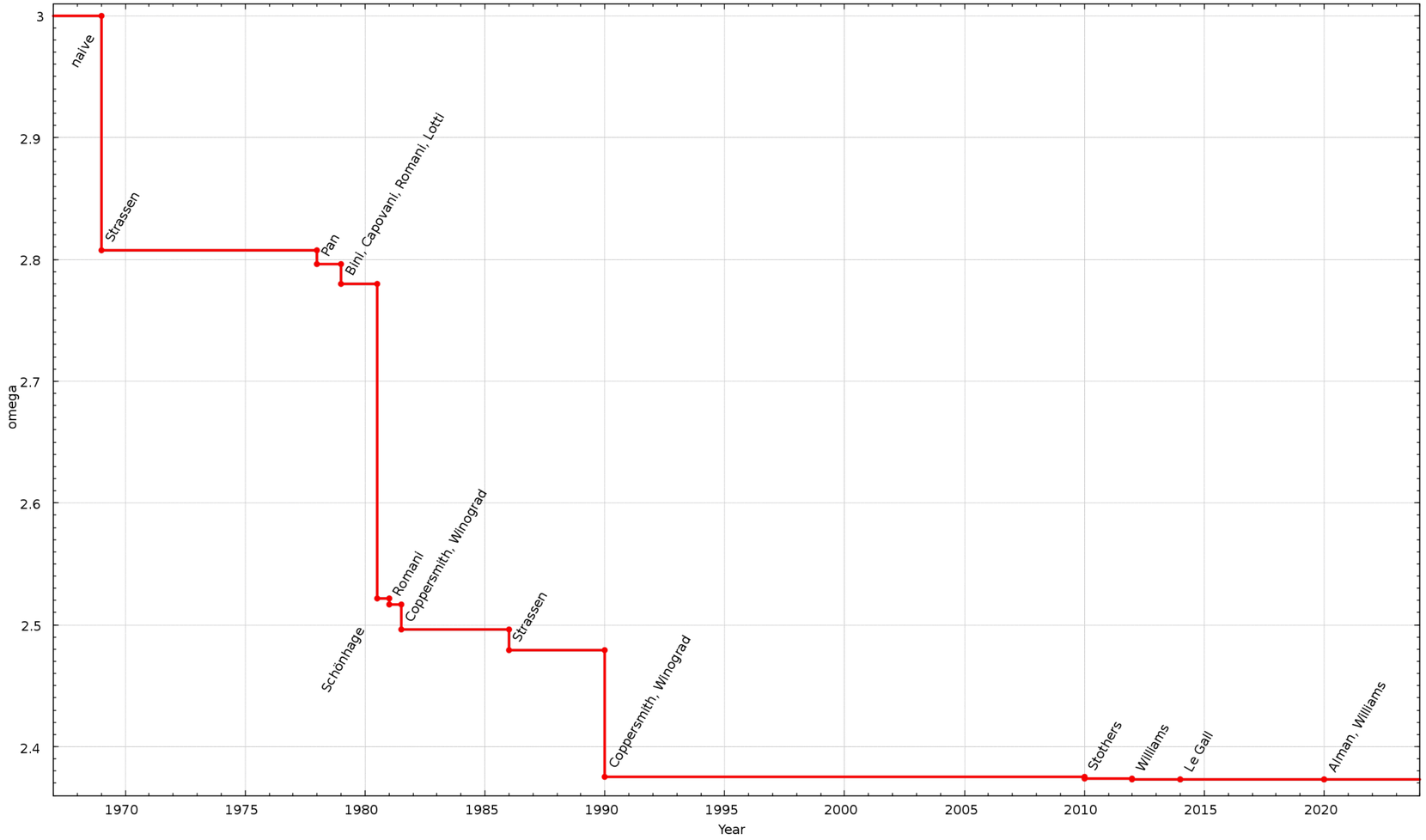
Типовые задачи

- $O(log(n))$ - бинарный поиск
- $O(\sqrt{n})$ - проверка числа на простоту
- $O(n)$ - линейный поиск, порядковые статистики
- $O(n * log(n))$ - сортировка слиянием, кучей
- $O(n^2)$ - сортировка пузырьком, вставками
- $O(a^n)$ - множество всех подмножеств
- $O(n!)$ - задача коммивояжера (полный перебор)

Проблемы теоретической оценки

- Не учитывает константы
- Сложность всех операций равна
- Нужно верить только **честному** бенчмарку на **реальных** данных.
- Пример - слияние сортированных массивов в питоне

Перемножение матриц



Время vs Память

- Задача - **посчитать количество простых чисел < N**
- Вариант 1: проверяем каждое число на простоту
- Вариант 2: решето Эратосфена

Время vs Память

- Задача - **посчитать количество простых чисел < N**
- Вариант 1: проверяем каждое число на простоту - $O(n^{1.5})$ время, $O(1)$ память
- Вариант 2: решето Эратосфена - $O(nloglog(n))$ время, $O(n)$ память

Доказательство: https://e-maxx.ru/algo/eratosthenes_sieve

Перерыв

Жадные алгоритмы

- Что такое жадный алгоритм?
- Алгоритм, который «слеп», на каждом шаге совершает некоторое «оптимальное» на данный момент действие
- Можно использовать для оптимизации, для некоторых задач они дают оптимальное решение.

Маршрут курьера

- Пример - хотим оптимизировать маршрут курьера
- Есть наша начальная позиция и N точек на карте, в которые надо доставить посылки и доступные временные интервалы
- Как будет выглядеть жадный алгоритм в таком случае?
- Даст ли жадный алгоритм оптимальный маршрут?
- Где это не работает?

Непрерывный рюкзак

- Задача - непрерывный рюкзак
- Нам нужно купить как можно больше качественной гречки, но мы можем унести только N кг, а в магазине дефицит и хорошей гречки почти не осталось.
- Пусть качество гречки характеризуется её ценой (дороже = лучше)
- Как нам максимизировать суммарную ценность?

Интервальное планирование

- Есть менеджер, его подчиненные хотят поставить ему встречу 1-1, каждый в своё время и с разной длительностью
- Менеджер одинаково хорошо относится ко всем сотрудникам, поэтому хочет провести встречи с как можно большим количеством
- Как ему составить расписание?

Повторение / Вопросы

- От чего зависит время работы программы?
- Чем отличаются O, Θ, Ω ?
- Как работают жадные алгоритмы?

Выводы

- Важно оценивать алгоритмы по времени и памяти, но нужно помнить про то, что оценка достаточно грубая
- жадные алгоритмы можно использовать, когда задача не требует «заглядывания вперёд»
- Скорость работы важна и может существенно влиять на бизнес

Как решать домашки

- Придумать самый простой в реализации алгоритм - он поможет тестировать более сложные (эффективные) алгоритмы
- Писать и проверять код **локально**
- Написать автотесты и прогонять на них решение