Вопросы к экзамену алгоритмам
SPb ITMO, 5-й курс, 2-й семестр, 22 июня 2024

Общая информация

- Конспект доступен в телеграм-чате, ищется в истории слову «конспект». Есть две версии конспекта: от Саши Смаля (этот же курс, прошлый год), от ВШЭ ПМИ (3 семестра).
- Курсивом отмечены темы, разобранные на практиках.

Графы

1. BFS. Две версии: по слоям; с очередью. *Модификация для 1-k весов.*
2. Алгоритм Дейкстры. Реализации за $\mathcal{O}(n^2)$, $\mathcal{O}(m \log n)$, $\mathcal{O}(m \log_{m/n} n)$. d-ичная куча.
3. Алгоритм A*. Пример применения.
4. Пример сведения задачи к алгоритмам на графах: поменять двух шахматных коней местами за минимальное число ходов, кони ходят по очереди.
5. Кратчайшие пути в DAG. Решение за $\mathcal{O}(n+m)$ для отрицательных весов.
6. Форд-Беллман: версия динамикой $d[v,k]$; итеративная версия с $\mathcal{O}(V)$ памятью; с очередью.
7. Алгоритм Флойда. Отрицательные циклы: Флойдом, Форд-Беллманом.

RMQ и LCA

8. Формулировка задач RMQ, RSQ. Статическая версия. Динамическая версия.
9. Дерево отрезков. Какую задачу и за сколько решает?
10. Sparse Table (разреженные таблицы). Какую задачу и за сколько решает? Сравнение с Д.О.
11. *Улучшение Sparse Table до* $[\mathcal{O}(n \log^* n, \mathcal{O}(\log^* n))]$.
12. LCA. Двоичные подъёмы за $\mathcal{O}(\log n)$, isAncestor за $\mathcal{O}(1)$.
13. LCA. Эйлеров обход, сведение LCA к RMQ, LCA за $[\mathcal{O}(n \log n), \mathcal{O}(1)]$.
14. Эйлеров обход: функция от поддерева за $\mathcal{O}(\log n)$.
15. RMQ за $[\mathcal{O}(n), \mathcal{O}(1)]$ в RAM-w с помощью битовых масок и стека минимумов.

Деревья поиска

16. BST. Определение. `add/find` за $\mathcal{O}(h)$.
17. BST. Нижняя оценка на время добавления, сортировка деревом.
18. BST. Удаление за $\mathcal{O}(h)$. `next/prev` за $\mathcal{O}(h)$
19. BST. `next/prev` за $\mathcal{O}(h)$.
20. BST. `find` за $\mathcal{O}(1)$, `del` за $\mathcal{O}(1)$, `next/prev` за $\mathcal{O}(1)$.
21. AVL. Определение, оценка высоты.
22. AVL. Перебалансировка после `add`, single/double (малое/большое) вращение.
23. AVL. Оценка максимального числа вращений за одну операцию `add`.
24. BST. Хранение равных элементов. Хотя бы 2 варианта решения проблемы.
25. BST. xLR обход, восстановление дерева по xLR обходу за $\mathcal{O}(n)$.
26. BST. Сравнение BST с хеш-таблицей.
27. C++. `set<int>`, `map<int,int>`, `tree<int>`.
28. Treap. Определение. Построение за $\mathcal{O}(n \log n)$. Единственность.
29. Treap. Оценка глубины со ссылкой на quick-sort.
30. Treap. Базовые операции `split/merge` и `add/del` через них.
31. Treap. `add` через один спуск со `split`; `del` через один спуск с `merge`.

32. Неявный ключ и интерфейс `rope`, описание процедуры «вставка на k-ую позицию».
33. Неявный ключ и вращение массива, reverse на отрезке, массовые изменения на отрезке.
34. B-tree. Определение. Оценка высоты и времени работы. Добавление в дерево.
35. 2-3-tree. 2-3-4-tree. RB-tree и AA-tree: определения, биекция с изученными ранее деревьями.
36. Splay. Собственно дерево. `splay/add/del/split/merge`.
37. Splay. Оценка $\mathcal{O}(\log A - \log B)$ на время работы `splay`, оценка остальных операций.
38. Splay. Более тонкая оценка времени `splay`: $\mathcal{O}(\sum k_i \log \frac{m}{k_i})$. Доказательство.
39. Splay. Версия без хранения отца. Top-down implementation.
40. Персистентность: полная (full), частичная (partial); тривиальная версия персистентного массива; быстрая версия частично-персистентного-массива.
41. Full persistent BST. Path copying.
42. Full persistent массив, как дерево по неявному ключу. Full persistent CHM.

Хеширование, хеши, хеш-таблицы

43. Определения: хеш-функция, хеш, коллизия (простой вопрос).
44. Хеш-таблица на списках (цепочки, закрытая). Оценка времени `add/find/del`.
45. Хеш-таблица с открытой адресацией (массив). Оценка времени `add/find`.
46. Хеш-таблица с открытой адресацией (массив). Любой корректный вариант `del`.
47. Пример хеш-функции $f(i) = i \bmod p,\ p \in [n, 2n)$ и простое. Обоснование хорошести f для списков. Контрпример для f для открытой адресации: n операций, $\Theta(n^2)$ времени.
48. Два способа исправления проблемы для открытой адресации:
пример хорошей хеш-функции; двойное хеширование.
49. Сравнение хеш-таблиц на списках и с открытой адресацией. Время. Память.
50. Удвоение (перехеширование) хеш-таблиц. Обоснование амортизированного времени $\mathcal{O}(1)$.
51. Хеш от множества. Операции над множеством `add/del/hash`.
52. Хеш строк (полиномиальные хеши). Построение за $\mathcal{O}(n)$, подсчёт хеша за $\mathcal{O}(1)$ без делений.
53. Pr[коллизии] для фиксированных строк $\frac{n}{m}$, для случайных строк $\frac{1}{m}$. Выбор пары $\langle x, m\rangle$ для полиномиального хеширования, обоснование. В чём проблема с $m = 10^9+7$? с $m = 10^{18}+3$.
54. Алгоритм Рабина-Карпа поиска подстроки в строке. RP и ZPP версии.
Оценка ошибки, выбор m, подойдёт ли $m = 10^9+7$?
55. Задача «количество различных подстрок». Решение хешами за $\mathcal{O}(n^2)$, оценка ошибки.
56. *Поиск общей подстроки двух строк за* $\mathcal{O}(n \log n)$.
57. *Вопроса не будет на экзамене.*
Построение теста против полиномиального хеширования против любой пары $\langle x, m\rangle$.
58. Универсальное семейство хеш-функций: определение, пример семейства с доказательством универсальности. Применение к хеш-таблицам, отличие от варианта «хеш-таблицы без универсального семейства».
59. Совершенное хеширование: определение, пример для $2^k \to k$, одноуровневая схема.
60. Совершенное хеширование: двухуровневая схема.
61. Фильтр-блюма. Сравнение с уже известными хеш-таблицами.
62. Хеш-таблица «кукушка». Сравнение с уже известными хеш-таблицами.

Алгебра и теория чисел

63. Гаусс: приведение матрицы к трапецевидной форме.
64. Гаусс: выделения базиса из данных векторов за $\mathcal{O}(nmk)$.
65. СЛАУ: построение пространства решений за $\mathcal{O}(nmk)$.

66. Гаусс: разложение вектора в треугольном базисе за $\mathcal{O}(nk)$,
разложение вектора в произвольном базисе за $\mathcal{O}(nk^2)$.
67. Ортогонализация Грамма-Шмидта за $\mathcal{O}(n^3)$. Разложение в ортогональном базисе за $\mathcal{O}(n^2)$.
68. Арифметика по модулю: умножение, возведение в степень, деление. Малая теоремы Ферма и теорема Эйлера для поиска обратного.
69. Расширенный Алгоритм Евклида для поиска обратного, сравнение с предыдущим способом.
70. Криптография: закрытый ключ; открытый ключ, RSA.
71. Генерация случайных больших простых чисел.
72. Проверка на простоту: $\mathcal{O}(n^{1/2})$; $\mathcal{O}(n^{1/2}/\log n)$; тест Ферма, числа Кармайкла и проверка за $\mathcal{O}(n^{1/3})$; тест Миллера-Рабина за $\mathcal{O}(\log n)$ арифметических операций.

Операции с многочленами и FFT

73. Базовые операции с многочленами: хранение, сложение, вычитание, умножение и деление на константу, умножение за $\mathcal{O}(nm)$, деление на $ax^k + b$ за $\mathcal{O}(n)$.
74. Умножение многочленов: алгоритм Карацубы за $\mathcal{O}(n^{1.585})$.
75. Связь умножения длинных чисел и умножения многочленов. Выбор системы счисления.
76. Комплексные числа: группа корней из единицы, умножение, деление, сопряжение.
77. Схема умножение через интерполяцию; рекурсивный FFT за $\mathcal{O}(n \log n)$; обратное FFT.
78. Обоснование корректности обратного FFT. Версия с `reverse`.
79. Точность операций. Выбор системы счисления 10^k для `double` и `long double`.
80. FFT над $\mathbb{Z}/p\mathbb{Z}$, использование для умножения над $\mathbb{Z}$.
81. Нерекурсивная реализация FFT: собственно FFT; обратная битовая запись;
предподсчёт корней; оптимальное хранение корней с учётом кеширования.
82. Перевод из одной системы счисления в другую за $\mathcal{O}(n \log^2 n)$.

Линейное программирование

83. Постановка задачи. Сведения между разными формами:
$Ax \leqslant b \leftrightarrow Ax = b$ и $x \to \max \leftrightarrow cx \to \min \leftrightarrow$ отсутствие минимизации.
84. Геометрический смысл: полиэдр, полупространства, выпуклость, вершины.
Добавление bounding-box. Решение перебором вершин полиэдра за $\mathcal{O}(2^{n+m}n^3)$.
85. Пример с $2n$ неравенствами и 2^n вершин полиэдра.
86. Симплекс метод: общая идея. Время работы симплекса в теории и на практике.
87. Нахождение начальной точки симплекс-методом.
88. Метод Эллипсоидов. *У нас не было математического определения Эллипсоида.*
89. Метод внутренней точки.
90. Двойственность: построение двойственной задачи;
теоремы о двойственности – слабая (с доказательством), сильная (только формулировка).
91. Примеры задач LP и ILP: паросочетание, вершинное покрытие, потоки, SAT.
92. TUM: определение; доказательство целочисленности решения; TUM для паросочетаний.

Потоки и паросочетания

93. Определения: поток, величина потока, разрез, величина разреза, циркуляция.
94. Связь потока и задачи «поиск k непересекающихся путей», декомпозиция потока за $\mathcal{O}(E^2)$.
95. Теорема и алгоритм Форда-Фалекрсона. Поиск потока за $\mathcal{O}(|f| \cdot E)$.
96. Поиск min разреза по max потоку за $\mathcal{O}(E)$.
97. Применение потока для поиска максимального паросочетания за $\mathcal{O}(VE)$.

98. Вершинные потоки и разрезы.
99. Алгоритм масштабирования потока за $\mathcal{O}(E^2 \log U)$ с доказательством времени работы.
100. Mincost поток: определение, поиск за $\mathcal{O}(|f| \cdot VE)$.
101. Mincost поток в случае наличия отрицательных циклов, алгоритм Клейна.
102. Формулировка задач max-flow и mincost-k-flow, как задач линейного программирования.

Строки

103. Префикс-функция. КМП для поиска s в t с $\mathcal{O}(s)$ доппамяти.
104. Бор. Способы хранения рёбер: массив, хеш-таблица, BST. Сравнение.
105. Бор. Сортировка строк за $\mathcal{O}(\sum |s_i| \log A)$. Поиск по словарю за $\mathcal{O}(|text| \cdot max|s_i|)$.
106. Ахо-Корасик полным автоматом.
Нахождение для каждого слова числа вхождений в текст за $\mathcal{O}(|text| + \sum |s_i|)$.
107. Суффиксное дерево за $\mathcal{O}(n^2)$, сжатый бор, поиск s в тексте суффдеревом за $\mathcal{O}(|s|)$.
108. Суффиксный массив за $\mathcal{O}(n \log n)$. Поиск строки s в тексте за $\mathcal{O}(|s| \cdot \log |text|)$.
109. Суффиксный массив $\leftrightarrow$ суффиксное дерево. В обе стороны за $\mathcal{O}(n)$.