Домашнее задание №8-9

Дедлайн: 16 апреля 2019 г., 23:00

Основные задачи

1. (2 + 2 балла)
   (i) Найдите произведение многочленов $A(x) = 2x+3$ и $B(x) = x^2-1$, используя рекурсивный $O(n \log n)$-алгоритм БПФ.
   (ii) Вычислите обратное ДПФ массива $A = [10, 3i\sqrt{2}+2+2i, 0, 3i\sqrt{2}+2-2i, -2, -3i\sqrt{2}+2+2i, 0, -3i\sqrt{2}+2-2i]$.

2. (1+2 балла)
   (i) Постройте $O(n \log n)$-БПФ-флгоритм для поиска подстрок с «джокерами».
   (ii) Покажите, как понизить трудоёмкость вашей процедуры до $O(n \log m)$.

3. (1 балл) Используя ДПФ, найдите решение системы линейных уравнений $Cx = b$, где C — циркулянтная матрица, порождённая столбцом $(1, 2, 4, 8)^\top$ и $b^\top = (16, 8, 4, 2)^\top$.

4. (1 + 2 + 3 + 2 балла)
   (i) Найдите примитивный корень восьмой степени в поле $\mathbb{Z}_{41}$.
   (ii) Вычислите ДПФ многочленов $A(x) = 2x + 3$ и $B(x) = x^2 - 1$ в поле $\mathbb{Z}_{41}$.
   (iii) Пусть A — матрица ДПФ длины n, ω_n — соответствующий первообразный корень степени n в поле $\mathbb{Z}_{41}$. Докажите, что $(A^{-1})_{i,j} \equiv n^{-1}(\omega_n^{-1})^{ji} \pmod{p}$.
   (iv) С помощью БПФ найдите произведение многочленов $A(x)$ и $B(x)$ из второго пункта в поле $\mathbb{Z}_{41}$.

5. Обсудим вопрос о минимальном числе T_{min} попарных сравнений, необходимых для нахождения минимального из n чисел. Для этой задачи алгоритм очевиден: нужно последовательно сравнивать числа, оставляя при каждом сравнении минимальное. Возникает правдоподобная гипотеза, что $T_{min} = n - 1$. Заметим, что даже в столь простой задаче ответ не очевиден, в частности, не проходит традиционный аргумент "по размеру входа", поскольку в $\frac{n}{2}$ сравнениях могут участвовать все числа, и речь фактически идет о том, какую часть информации о числах можно при сравнении передать. Рассмотрим два подхода к получению нижних оценок подобного рода.

   Первый подход связан с понятием разрешающего дерева для алгоритмов сортировки **[Кормен 1 §9.1], [Кормен 2 §8.1]**. Напомним, что произвольный алгоритм A сортировки массива из n чисел $\{a_1, \dots, a_n\}$ посредством попарных сравнений можно следующим образом изобразить в виде корневого двоичного дерева D_A. Каждая внутренняя вершина v дерева помечена некоторым сравнением $a_i?a_j$, а в паре выходящих из v ребер одно ребро имеет пометку $\leq$, а другое $\geq$. Листья D_A помечены соответствующими перестановками $\{\pi_1, \dots, \pi_n\}$, которые упорядочивают массив. Каждому конкретному входу $\{a_1, \dots, a_n\}$ отвечает его **реализация** — путь от корня к листу в D_A.

   Совершенно аналогично дается определение разрешающего дерева для задачи поиска минимального элемента, поиска медианы и т. д. (все сводится к изменению пометок листьев). Мы сохраним для этих «специализированных» деревьев обозначение D_A. На языке разрешающих деревьев утверждение о том, что $T_{min} = n - 1$, эквивалентно следующему: в любом корректном алгоритме поиска минимального элемента в массиве из n чисел, использующем только попарные сравнения, каждый реализуемый путь от корня к листу имеет не менее $(n - 1)$-го ребра.

   Назовем это **утверждением** $\mathcal{A}$.

Произвольному корректному алгоритму A нахождения минимума попарными сравнениями и произвольному реализуемому пути P в разрешающем дереве D_A отвечает (неориентированный) граф $G_A^P = (V, E)$ на n вершинах, в котором есть ребро $(v_i, v_j) \in E$, если и только если в пути P какая-то вершина имеет пометку $a_i?a_j$.

Утверждение $\mathcal{B}$. Для корректности алгоритма A нахождения минимума необходимо, чтобы граф G_A^P был связен.

(i) (1 балл) Докажите импликацию: $\mathcal{B} \Rightarrow \mathcal{A}$.

(ii) (1 балл) Докажите утверждение $\mathcal{B}$.

(iii) (2 балла) Теперь дадим другое доказательство этой нижней оценки. Отметим, что само доказательство будет иллюстрацией методов **аммортизационного анализа** для получения нижних оценок.

Для этого запишем шаги алгоритма в формате конфигураций (a, b, c, d), где a элементов пока не сравнивались, b элементов были больше во всех сравнениях, c элементов были меньше во всех сравнениях, d были и больше, и меньше в сравнениях, т. е. начальная конфигурация такова: $Init = (n, 0, 0, 0)$. Введем "потенциальную функцию", определенную на конфигурациях: $f[(a, b, c, d)] = a + c$. Мы оценим трудоемкость алгоритма, просто поделив "разность потенциалов" между начальной и конечной конфигурациями на максимальное изменения потенциала за один шаг алгоритма.

Покажите, что при любом сравнении потенциал $f(\cdot)$ может уменьшиться не больше, чем на единицу, и что отсюда вытекает, что число шагов любого такого алгоритма не меньше $n - 1$.

6. Покажем, что любой алгоритм нахождения медианы массива из n элементов посредством попарных сравнений имеет сложность $T(n) = \frac{3n}{2} - O(\log n)$.

(i) (2 балла) Покажите, что любое разрешающее дерево поиска медианы позволяет также восстановить индексы всех элементов, бОльших медианы, и всех элементов, мЕньших медианы.

Из этой задачи вытекает, что нахождение медианы эквивалентно с виду более сложной задаче: найти медиану и массив L элементов, бОльших ее $(\frac{n}{2} - 1)$ элементов.

(ii) (3 балла) Покажите, что любое разрешающее дерево для медианы содержит путь от корня к листу длины $\frac{3n}{2} - O(\log n)$.

Комментарий. Можно использовать два соображения. Во-первых, если из дерева T для медианы выкинуть все сравнения, в которых участвуют элементы L, то получится дерево T_L поиска максимума (в нем максимум — это медиана). А из предыдущей задачи следует, что T_L должно иметь $\geq 2^{\frac{n}{2}-1}$ листьев. Во-вторых, массив L может быть произвольным, а отсюда можно получить оценку снизу на число листьев (и на высоту) T.

Наилучшие известные современные оценки: $(2 + \varepsilon)n \leq T(n) \leq 2.95n$.

7. (1 балл) Покажите, что если в любой модификации алгоритме Quicksort в качестве барьерного элемента использовать медиану текущего массива, причем искать ее посредством стандартного линейного алгоритма, то его сложность по наихудшему случаю станет $O(n \log n)$.

8. Пусть x — двоичное дерева поиска; обозначим $size[x]$ число ключей в поддереве с вершиной x. Выделим для $size[\cdot]$ поле в каждой вершине дерева. Пусть α — число и $1/2 \leq \alpha < 1$. Будем говорить, что вершина x дерева, не являющаяся листом, *α-сбалансирована*, если $size[[left[x]] \leq \alpha size[x]$ и $size[[right[x]] \leq \alpha size[x]$ ($left$ и $right$ — это указатели на левого и, соответственно, правого потомка x в двоичном дереве). Дерево называется *α-сбалансированным*, если все его внутренние вершины α-сбалансированы.

(i) (2 балла) Покажите из определения, что для *любой* вершины x 1/2-сбалансированного дерева выполнено:

$$size[[left[x]] - size[[right[x]] \in \{-1, 0, +1\}$$

.

(ii) (2 балла) **[Кормен 1, задача 18.3 (б)]или [Кормен 2, задача17.3 (б)]** Покажите, что поиск элемента в α-сбалансированном двоичном дереве с n вершинами выполняется за $O(\log n)$.

Дополнительные задачи

1. (3 балла) Пусть дано множество различных чисел $A \subseteq \{1, \ldots, m\}$. Рассмотрим множество $A + A$, образованное суммами пар элементов A. Докажите или опровергните существование процедур построения $A + A$, имеющих субквадратичную трудоёмкость $o(n^2)$.

2. (4 + 4 балла)

   (i) Дано n ключей и n замков. Все ключи и все замки различны между собой, а каждый ключ подходит к единственному замку. Ключи (и замочные скважины) упорядочены по величине, но визуально отличия неразличимы. На каждом шаге можно попытаться вставить конкретный ключ в конкретный замок и заключить, что он подходит или больше, или меньше искомого. Постройте вероятностный алгоритм подбора ключей, требующий в среднем $o(n^2)$ шагов.

   Комментарий. Очевидно, что прямой перебор подходящих пар ключей и замков требует квадратичного числа шагов. Удивительным кажется то, что для этой задачи построен детерминированный $o(n^2)$-алгоритм. Он очень хитрый.

   (ii) Покажите, что любая детерминированная процедура подбора ключей требует $\Omega(n \log n)$ шагов.