Задание 4

Коновалов Андрей, 074

1	2	3	4	5	6	7	8	9	Σ

Задача 1

(i) Решим рекуррентное соотношение

$$T(n) = T\left(\frac{n}{2}\right) + Cn, \;\; T(1) = C, \;\; n = 2^k$$

Проведем суммирование по дереву рекурсии

$$T(n) = \sum_{i=0}^{k} C\frac{n}{2^i} = \sum_{i=0}^{k} C2^{k-i} = \sum_{i=0}^{k} C2^i = C(2^{k+1} - 1)$$

Получаем ответ

$$T(2^k) = C(2^{k+1} - 1)$$

Задача 2

(i) Что бы перемножить две матрицы размера n алгоритмом I необходимо для каждого элементы матрицы результата сделать n произведений и $n-1$ сложение. Итоговое число арифметических операций:

$$T(n) = n^2 \cdot (n + (n-1)) = 2n^3 - n^2$$

(ii) Посчитаем отдельно число умножений $M(n)$, совершаемое во время перемножения двух матриц размера n алгоритмом II. При перемножении матриц размера n происходит 8 перемножений матриц размера $\frac{n}{2}$. При $n = 1$ происходит просто перемножение чисел. Получаем

$$M(n) = 8M\left(\frac{n}{2}\right), \;\; M(1) = 1$$

Явное выражение будет

$$M(n) = M(2^k) = 2^{3k} = n^3$$

Теперь отдельно посчитаем количество сложений $A(n)$ совершаемое во время перемножения двух матриц размера n. На каждом шаге алгоритма происходит 8 перемножений матриц размера $\frac{n}{2}$ и 4 сложения матриц того же размера. При $n = 1$ сложений не происходит. Получаем

$$A(n) = 8A\left(\frac{n}{2}\right) + 4\left(\frac{n}{2}\right)^2, \;\; A(1) = 0$$

Суммированием по дереву рекурсии получаем

$$A(n) = \sum_{i=0}^{\log_2 n - 1} 8^i \left(\frac{n}{2^i}\right)^2 + \sum A(1) = \sum_{i=0}^{\log_2 n - 1} 2^i n^2$$

Используя формулу для геометрической прогрессии получаем

$$A(n) = n^2 \frac{1 - 2^{\log_2 n}}{1 - 2} = n^2(n-1)$$

Суммарное число арифметический операций

$$T(n) = M(n) + A(n) = 2n^3 - n^2$$

(iii) При выполнении шага алгоритма Карацубы происходит 7 умножений и 18 сложений матриц вдвое меньшего размера. Аналогично *(ii)* запишем рекуррентное соотношение для количества умножений

$$M(n) = 7M\left(\frac{n}{2}\right), \;\; M(1) = 1$$

Разрешая получаем

$$M(n) = 7^{\log_2 n} = n^{\log_2 7}$$

Для количества сложений

$$A(n) = 7A\left(\frac{n}{2}\right) + 18\left(\frac{n}{2}\right)^2, \;\; A(1) = 0$$

Суммированием по дереву рекурсии получаем

$$A(n) = \sum_{i=0}^{\log_2 n - 1} 18 \cdot 7^i \left(\frac{n}{2^{i+1}}\right)^2 + \sum A(1) = \sum_{i=0}^{\log_2 n - 1} \frac{18}{4} n^2 \left(\frac{7}{4}\right)^i$$

Используя формулу для геометрической прогрессии получаем

$$A(n) = \frac{18}{4} n^2 \frac{1 - \left(\frac{7}{4}\right)^{\log_2 n}}{1 - \frac{7}{4}} = 6(n^2 - n^{\log_2 7})$$

Суммарное число арифметический операций

$$T(n) = M(n) + A(n) = 6n^2 - 5n^{\log_2 7}$$

(iv) Выражение для количества сложений $A(n)$ уже было получено в *(ii)*

$$A(n) = n^2(n-1)$$

Задача 3

(i) Пусть A не верно. Тогда существует корректный алгоритм поиска минимального элемента в массиве из n чисел, использующий только попарные сравнения, в котором существует реализуемый путь W от корня к листу, имеющий меньше, чем $(n-1)$ ребро.

Но, если верно B, то необходимо, что бы граф G соответствующий пути W был связен, а для этого необходимо, что бы он имел хотя бы $(n-1)$ ребро. Поскольку путь должен W имеет не меньше ребер, чем граф G получаем противоречие.

(ii) Допустим, что некоторый граф G не связен, а значит разбивается на по крайней мере две компоненты связности. Возьмем любые две M и N из них. Выполним алгоритм дважды: в первом случае уменьшим все числа из компоненты M на одно и то же число так, что бы минимум находился в M, во втором случае то же самое проделаем с числами из N, причем так, что бы минимум из N не совпадал с минимумом из M первого случая.

Результаты сравнений между числами из одной и той же компоненты связности не изменятся, а между компонентами сравнений не происходит. Соответственно в обоих случаях будет реализовываться тот путь, которому сопоставляется граф G. В обоих случаях алгоритм выдаст один и тот же ответ, а ответы должны быть разными. Противоречие.

Задача 4

Разобьем $\{a_{f(1)}, ..., a_{f(n)}\}$ на группы $\{b_1, ..., b_{\lceil \frac{n}{k} \rceil}\}$ по k элементов. Назовем аналогичное разбиение отсортированного массива $\{c_1, ..., c_{\lceil \frac{n}{k} \rceil}\}$. Заметим, что если число находится в группе c_i, то оно может находиться лишь в "соседних" группах b_{i-1}, b_i, b_{i+1}.

Теперь возьмем первые две группы b_1, b_2 и отсортируем их вместе за $O(2k \log 2k)$. Заметим, что первые k элементов полученного массива будут составлять c_1. Оставшиеся назовем d_2.

Далее возьмем d_2 и b_3 и отсортируем их вместе за $O(2k \log 2k)$. Будем идти двумя указателями по массиву c_1 и полученному, сравнивая элементы, пропуская те, что входят в c_1 (при перемешивании они могли оказаться в b_2, а значит и в d_2). После пропуска останется $\geq k$ элементов и первые k из них будут составлять c_2. Пропуск выполняется за $O(k)$. Оставшиеся назовем d_3.

Будем продолжать делать эту операцию и таким образом восстановим все группы c_i, а значит и отсортированный массив.

Оценим время выполнения

$$O(2k \log 2k) + \left(\left\lceil \frac{n}{k} \right\rceil - 1 \right) (O(2k \log 2k) + O(k)) = O(n \log k)$$