

## Алгоритмы во внешней памяти

Алгоритмы во внешней памяти отличаются тем, что размер оперативной памяти ограничен, а объем данных, с которыми нужно работать, очень большой. В связи с этим приходится считывать и записывать данные на жёсткий диск. Так как обращение к ячейке памяти на жёстком диске — долгая операция, то в алгоритмах во внешней памяти для оценки времени работы используется только количество операций записи и считывания с жёсткого диска, даже если после этого в оперативной памяти производится экспоненциальное число операций.

**Определение.** Размер оперативной памяти в байтах обозначается за  $M$ .

**Определение.** С жёсткого диска за одну операцию можно считать целый блок непрерывных последовательных данных. Размер этого блока в байтах обозначается за  $B$ .

**Определение.** Асимптотикой алгоритма во внешней памяти считается количество операций считывания и записи данных на жёсткий диск.

**Задача 1.** Предложите алгоритм вычисления суммы всех чисел в массиве размера  $N$  байт, который хранится во внешней памяти, за  $\lceil \frac{N}{B} \rceil$  операций.

**Определение.** Время прохода по массиву обозначается  $Scan(N) = \lceil \frac{N}{B} \rceil$ .

**Задача 2.** Предложите алгоритм разворота массива размера  $N$  за  $\mathcal{O}(Scan(N))$ . Можно ли сделать это inplace?

**Задача 3.** Предложите алгоритм слияния двух отсортированных массивов суммарного размера  $N$  в один за  $\mathcal{O}(Scan(N))$ , чтобы это было:

- a) не inplace;
- \*\*\*b) inplace.

При каком минимальном  $M$  это возможно?

**Задача 4.** Предложите алгоритм сортировки массива размера  $N$ , за:

- a)  $\mathcal{O}\left(\frac{N}{B} \log_2 N\right)$ ;
- b)  $\mathcal{O}\left(\frac{N}{B} \log_2 \frac{N}{B}\right)$ ;
- c)  $\mathcal{O}\left(\frac{N}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$ .

**Определение.** Время сортировки массива обозначается за  $\mathcal{O}(Sort(N)) = \mathcal{O}\left(\frac{N}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$ .

**Задача 5.** *Join*: Дано 2 множества пар (ключ, значение). Требуется объединить их в тройки вида (ключ, значение1, значение2) за  $\mathcal{O}(Sort(N))$ .

**Задача 6.** *List Ranking*: Дан список размера  $N$  во внешней памяти, т.е. для каждого элемента задан указатель на следующий. Требуется для каждого элемента определить его ранг — расстояние до конца списка, т.е. кол-во элементов после него в списке, и отсортировать элементы по возрастанию ранга за асимптотику:

- a)  $\mathcal{O}(Sort(N) \log N)$ ;
- \*b)  $\mathcal{O}(Sort(N))$ ;

**Подсказка 1.** Обратите внимание, что время работы не обязательно детерминировано.

**Подсказка 2.** «Идея свести к задаче сортировки по рангам»

**Задача 7.** Предложите способ реализовать стек во внешней памяти, в котором операции добавления, поиска и удаления последнего элемента работают за:

- a)  $\mathcal{O}(1)$ ;
- b)  $\mathcal{O}\left(\frac{1}{B}\right)$ .

**Задача 8.** Предложите способ реализовать список во внешней памяти, в котором операции добавления элемента по указателю, удаления элемента по указателю и поиска указателя на  $k$ -й элемент соответственно работают за:

- a)  $\mathcal{O}(1)$ ,  $\mathcal{O}(1)$ ,  $\mathcal{O}(k)$ ;
- b)  $\mathcal{O}(1)$ ,  $\mathcal{O}(1)$ ,  $\mathcal{O}\left(\frac{k}{B}\right)$  с возможной инвалидацией указателей;
- c)  $\mathcal{O}(1)$ ,  $\mathcal{O}(1)$ ,  $\mathcal{O}\left(\frac{k}{B}\right)$  без инвалидации указателей.

**Задача 9.** *B-Tree*: Предложите способ реализовать online set во внешней памяти, в котором операции добавления, удаления и поиска работают в онлайн за:

- a)  $\mathcal{O}(\log_2 N)$ ;
- b)  $\mathcal{O}(\log_B N)$ ;
- b)  $\mathcal{O}(\log_B N)$ , при этом амортизированное число операций записи равно  $1 + \mathcal{O}\left(\frac{1}{B}\right)$ .

**Задача 10.** Предложите способ реализовать offline set во внешней памяти, в котором операции добавления, удаления и поиска работают за  $\mathcal{O}\left(\frac{1}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$ .

**Задача 11.** *Heap*: Предложите способ реализовать онлайн кучу во внешней памяти, в которой операции добавления, удаления минимума и поиска минимума работают амортизировано за  $\mathcal{O}\left(\frac{1}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$

— — —

**Задача 12.** Дан ациклический ориентированный топологически отсортированный граф вычислений некоторой функции, у которого есть несколько начальных вершин, входящие рёбра в вершину означают то, из каких вершин и значений в них надо вычислить значение в текущей вершине. Требуется посчитать значения во всех вершинах за  $\mathcal{O}\left(\frac{N}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$ , где  $N$  — число рёбер.

**Задача 13.** Дан граф. Требуется раскрасить его в  $d+1$  цвет, где  $d$  — максимальная степень вершины, за асимптотику  $\mathcal{O}\left(\frac{N}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$ , где  $N$  — число рёбер.

**Задача 14.** Дан граф, где  $V$  вершин и  $E$  рёбер. Требуется запустить на нём BFS, т.е. для каждой вершины узнать расстояние до первой за время:

- a) Меньше чем  $\mathcal{O}(E)$ ;
- b)  $\mathcal{O}(Scan(E) \cdot maxd + Sort(E))$ , где  $maxd$  — максимальное расстояние;
- c)  $\mathcal{O}(Sort(E) + V)$ ;
- \*\*d)  $\mathcal{O}\left(Sort(E) + \frac{Scan(E)}{\sqrt{\frac{E}{VB}}} + V \cdot \sqrt{\frac{E}{VB}}\right) = \mathcal{O}\left(Sort(E) + \sqrt{\frac{VE}{B}}\right)$ .

**Задача 15.** Дано дерево на  $N$  вершинах, дерево задано массивом предков. Предложите способ построить его эйлеров обход за время  $\mathcal{O}(Sort(N))$

**Задача 16.** Дан граф из  $V$  вершин и  $E$  рёбер. Требуется разделить его на компоненты связности за время:

- a) Меньше чем  $\mathcal{O}(E)$ ;
- b)  $\mathcal{O}(Sort(E) \log V)$ ;
- c)  $\mathcal{O}\left(Sort(E) \log \frac{VB}{E}\right)$ .

**Задача 17.** Дан взвешенный граф из  $V$  вершин и  $E$  рёбер. Требуется построить в нём остовное дерево за время:

- a) Меньше чем  $\mathcal{O}(E)$ ;
- b)  $\mathcal{O}(Sort(E) \log V)$ .