

Билет 26. Разреженная таблица (Sparse Table)

Курс "Алгоритмы и структуры данных"

Билет 26. Разреженная таблица

Основная идея: Предподсчёт для мгновенных запросов на отрезках за $O(1)$!

Что такое разреженная таблица?

Разреженная таблица — структура данных, которая позволяет отвечать на запросы минимума/максимума на отрезке за $O(1)$ после предподсчёта за $O(n \log n)$.

Ограничения

- **Подходит для:** MIN, MAX, GCD, AND, OR
- **Не подходит для:** SUM, XOR (нужны другие структуры)
- **Статические данные:** Нет обновлений элементов

Принцип работы

Идея: Предподсчитать ответы для всех отрезков длины 2^k

Для каждого начала i и степени k храним:

$$st[k][i] = \min(arr[i], arr[i+1], \dots, arr[i+2^k-1])$$

Пример построения


Исходный массив: [1, 3, 2, 5, 4, 8, 6, 7]

Шаг 1: $k = 0$ (отрезки длины 1)

	0	1	2	3	4	5	6	7
k=0:	1	3	2	5	4	8	6	7

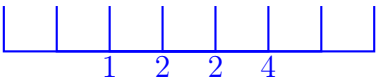
Шаг 2: $k = 1$ (отрезки длины 2)

k=1:	1	3	2	5	4	8	6	7
------	---	---	---	---	---	---	---	---



Шаг 3: $k = 2$ (отрезки длины 4)

k=2:	1	3	2	5	4	8	6	7
------	---	---	---	---	---	---	---	---



Формальное описание

Рекуррентная формула

$$st[k][i] = \min(st[k-1][i], st[k-1][i + 2^{k-1}])$$

Размеры таблицы

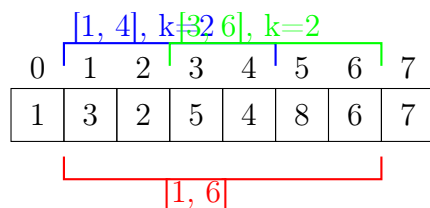
- Количество строк: $\lfloor \log_2 n \rfloor + 1$
- Количество столбцов: n

Ответ на запрос $[L, R]$

Алгоритм

1. Находим $k = \lfloor \log_2(R - L + 1) \rfloor$
2. Разбиваем отрезок $[L, R]$ на два перекрывающихся отрезка длины 2^k :
3. Ответ: $\min(st[k][L], st[k][R - 2^k + 1])$

Пример: Запрос минимума на $[1, 6]$



Вычисление:

- Длина отрезка: $6 - 1 + 1 = 6$
- $k = \lfloor \log_2 6 \rfloor = 2$
- $\min(st[2][1], st[2][6 - 4 + 1]) = \min(st[2][1], st[2][3])$
- $\min(2, 4) = 2$

Реализация на C++

```
#include <vector>
#include <cmath>
using namespace std;

class SparseTable {
private:
    vector<vector<int>>> st;
    vector<int> log_table;
```

```

public:
    SparseTable(vector<int>& arr) {
        int n = arr.size();
        int k = log2(n) + 1;

        st.resize(k, vector<int>(n));

        // Инициализация для k=0
        for (int i = 0; i < n; i++)
            st[0][i] = arr[i];

        // Построение таблицы
        for (int j = 1; j < k; j++) {
            for (int i = 0; i + (1 << j) <= n; i++) {
                st[j][i] = min(st[j-1][i], st[j-1][i + (1 << (j-1))]);
            }
        }

        // Предподсчёт логарифмов
        log_table.resize(n + 1);
        log_table[1] = 0;
        for (int i = 2; i <= n; i++)
            log_table[i] = log_table[i/2] + 1;
    }

    int query(int l, int r) {
        int len = r - l + 1;
        int k = log_table[len];
        return min(st[k][l], st[k][r - (1 << k) + 1]);
    }
};

```

Пример работы

Массив: [1, 3, 2, 5, 4, 8, 6, 7]

Построенная таблица:

k \ i	0	1	2	3	4	5	6	7
0	1	3	2	5	4	8	6	7
1	1	2	2	4	4	6	6	-
2	1	2	2	4	-	-	-	-
3	1	-	-	-	-	-	-	-

Запросы:

- query(0, 3): $\min(1, 2) = 1$
- query(2, 5): $\min(2, 4) = 2$
- query(4, 7): $\min(4, 6) = 4$
- query(1, 6): $\min(2, 4) = 2$

Анализ сложности

- **Предподсчёт:** $O(n \log n)$ времени и памяти
- **Запрос:** $O(1)$ времени
- **Обновление:** Не поддерживается (статические данные)