

# Дерево отрезков

Лесников Юрий, seagest

## 1 Дерево отрезков

- Мы уже умеем отвечать на запросы вида: «найти сумму на отрезке», например, с помощью префиксных сумм.
- Хотим решать задачу «обновить значение элемента массива».
- Нам может помочь структура «Дерево отрезков».

**Лемма 1.1.** Пусть  $\mathbb{X}$  – абстрактное множество. Дерево отрезков способно за  $\mathcal{O}(\log n)$  получить на подотрезке результат любой операции  $\mathcal{B}: \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{X}$ , которая:

1. Ассоциативна:  $\forall a, b, c \in \mathbb{X} \quad \hookrightarrow \quad \mathcal{B}(\mathcal{B}(a, b), c) = \mathcal{B}(a, \mathcal{B}(b, c))$
2. Коммутативна:  $\forall a, b \in \mathbb{X} \quad \hookrightarrow \quad \mathcal{B}(a, b) = \mathcal{B}(b, a)$
3. Имеет нейтральный элемент:  $\exists 0 \in \mathbb{X} : \forall a \in \mathbb{X} \quad \hookrightarrow \quad \mathcal{B}(a, 0) = \mathcal{B}(0, a) = a$

**Замечание 1.1.** Задачи **RSQ/RMQ** (Range sum query/range minimum query), как правило, решаются с использованием этой структуры данных.

### 1.1 Структура

Рассмотрим для суммы:

- Корень содержит сумму всего массива.
- Левый ребенок – сумму первой половины элементов.
- Правый ребенок – сумму второй половины.
- Далее аналогично бьем пополам детей.

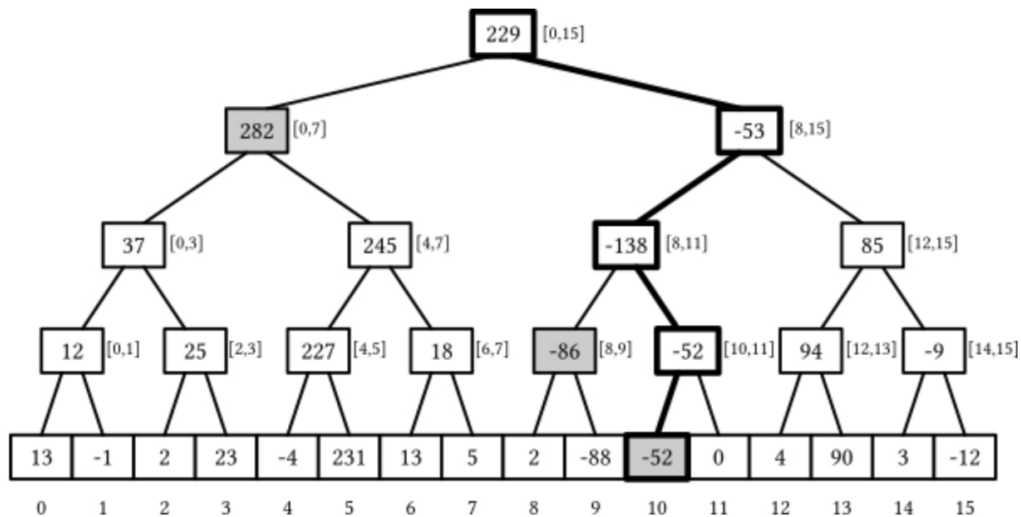


Рис. 1: Пример ДО на сумму

## 1.2 Построение

Пусть  $\mathbb{X}$  – абстрактное множество,  $\mathcal{B}: \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{X}$ , операция  $\mathcal{B}$  удовлетворяет всем трём условиям из леммы 1.1. Опишем рекурсивное построение дерева на массиве  $\mathcal{A}$  длины  $n$ :

1. Будем считать, что  $\log_2 n \in \mathbb{N}$ , иначе дозаполним массив  $\mathcal{A}$  до степени двойки нейтральными элементами.
2. Заведём массив  $\mathcal{T}$  длины  $2n - 1$ , последние  $n$  элементов которого будут элементами исходного массива.
3. Первые  $n-1$  элементов массива  $\mathcal{T}$  заполним следующим образом:  $\mathcal{T}[i] = \mathcal{B}(\mathcal{T}[2i+1], \mathcal{T}[2i+2])$  (заполнение от элемента с номером  $n-2$  и до 0 в цикле).

## 1.3 Ответ на запрос

Наша цель — получить результат операции на отрезке. Пусть мы находимся в узле, отвечающего за подотрезок  $[l, r]$ , а изначальный запрос был  $[L, R]$ .

1. Если  $[l, r] \cap [L, R] = \emptyset$ , то вернём нейтральный элемент.
2. Если  $[l, r] \subseteq [L, R]$ , то вернём значение в узле.
3. Иначе вернём результат операции с детьми.

То есть исходный отрезок разбивается на дизъюнктное объединение подотрезков.

## 1.4 Время работы

**Теорема 1.1.** *Время ответа на запрос составляет  $\mathcal{O}(\log n)$ .*

*Доказательство.* Заметим, что на каждом уровне раскрываться вниз могут не более двух узлов, так как только крайние могут порождать дочерние запросы. Следовательно, время работы  $\mathcal{O}(\log n)$ .  $\square$

---

### Algorithm 1 Query Function

---

```
1: function query(int node, int a, int b)
2:    $l \leftarrow \text{tree}[\text{node}].\text{left}$ 
3:    $r \leftarrow \text{tree}[\text{node}].\text{right}$ 
4:   if  $[l, r] \cap [a, b] = \emptyset$  then
5:     return neutral
6:   end if
7:   if  $[l, r] \subseteq [a, b]$  then
8:     return tree[node].res
9:   end if
10: return  $\mathcal{B}(\text{query}(\text{node} \cdot 2 + 1, a, b), \text{query}(\text{node} \cdot 2 + 2, a, b))$ 
```

---

## 1.5 Обновление по индексу

Пусть необходимо обновить элемент с индексом  $i$ . Тогда:

1. Найдём в дереве лист, отвечающий за  $i$ -й элемент. Это  $(n - 1 + i)$ -й элемент.
2. Индекс родителя:  $\lfloor \frac{i-1}{2} \rfloor$
3. Обновляем значение в родителе через результаты на деревьях.
4. Повторяем, пока не обновим корень.

## 2 Групповые операции

**Определение 2.1.** Пусть  $\mathbb{X}$  – абстрактное множество,  $\mathcal{B}: \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{X}$  – операция запроса на отрезке. Операцией группового обновления будем называть операцию  $\mathcal{G}: \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{X}$ , которая применяется ко всем элементам на подотрезке. Условия на операцию  $\mathcal{G}$ :

1. Существование нейтрального элемента:  $\exists 0 \in \mathbb{X} : \forall a \in \mathbb{X} \quad \hookrightarrow \quad \mathcal{G}(a, 0) = \mathcal{G}(0, a) = a$
2. Ассоциативность:  $\forall a, b, c \in \mathbb{X} \quad \hookrightarrow \quad \mathcal{G}(\mathcal{G}(a, b), c) = \mathcal{G}(a, \mathcal{G}(b, c))$
3. Дистрибутивность:  $\forall a, b, c \in \mathbb{X} \quad \hookrightarrow \quad \mathcal{G}(\mathcal{B}(a, b), c) = \mathcal{B}(\mathcal{G}(a, c), \mathcal{G}(b, c))$

### 2.1 Проталкивание несогласованности

---

#### Algorithm 2 Push Function

---

```

1: function push(int node)
2:   tree[2 · node + 1].d ←  $\mathcal{G}(\text{tree}[2 \cdot \text{node} + 1].d, \text{tree}[\text{node}].d)$ 
3:   tree[2 · node + 2].d ←  $\mathcal{G}(\text{tree}[2 \cdot \text{node} + 2].d, \text{tree}[\text{node}].d)$ 
4:   tree[node].d ←  $\mathcal{G}_{\text{neutral}}$ 

```

---

### 2.2 Групповое обновление

---

#### Algorithm 3 Update Function

---

```

1: function update(int node, int a, int b, T val)
2:   l ← tree[node].left, r ← tree[node].right
3:   if [l, r] ⊆ [a, b] then
4:     tree[node].d ←  $\mathcal{G}(\text{tree}[\text{node}].d, \text{val})$ 
5:   return
6: end if
7:   push(node)
8:   update(2 · node + 1, a, b, val)
9:   update(2 · node + 2, a, b, val)
10:  tree[node].ans ←  $\mathcal{B}(\mathcal{G}(\text{tree}[2 \cdot \text{node} + 1].ans, \text{tree}[2 \cdot \text{node} + 1].d),$ 
11:     $\mathcal{G}(\text{tree}[2 \cdot \text{node} + 2].ans, \text{tree}[2 \cdot \text{node} + 2].d))$ 

```

---

### 2.3 Ответ на запрос

---

#### Algorithm 4 Query Function

---

```

1: function query(int node, int a, int b)
2:   {As in group update}
3:   ans ←  $\mathcal{B}(\text{query}(2 \cdot \text{node} + 1, a, b), \text{query}(2 \cdot \text{node} + 2, a, b))$ 
4:   tree[node].ans ←  $\mathcal{B}(\mathcal{G}(\text{tree}[2 \cdot \text{node} + 1].ans, \text{tree}[2 \cdot \text{node} + 1].d),$ 
5:      $\mathcal{G}(\text{tree}[2 \cdot \text{node} + 2].ans, \text{tree}[2 \cdot \text{node} + 2].d))$ 
6:
7: return ans

```

---