Санкт-Петербургский политехнический университет Петра Великого

Институт машиностроения, материала и транспорта

Высшая школа автоматизации и робототехники

Курсовая работа

Дисциплина: Объектно-ориентированное программирование

Тема: Дерево Фенвика (сумма, максимум, минимум)

Выполнил студент гр. 3331506/20102 Иванов Е.А.

Преподаватель Ананьевский М. С.

Санкт-Петербург

2025

### Оглавление

1. **Введение…………………………………………………………………. 1**
2. **Дерево Фенвика………………………………………………………… 2**
   1. Теоретические сведения**…………………………………………….. 3**
   2. Нахождение суммы **…………………………………………………. 4**
   3. Нахождение максимума **…………………………………………….. 5**
   4. Нахождение минимума**……………………………………………… 6**
   5. Анализ результатов**………………………………………………….. 7**
3. **Заключение……………………………………………………………... 8**
4. **Литература……………………………………………………………… 9**
5. **Приложение………………………………………………………….... 10**
6. **Введение**

Данная работа посвящена изучению и практическому применению структуры данных, известной как **дерево Фенвика**, также называемой **двоичным индексированным деревом (Binary Indexed Tree, BIT)**. Дерево Фенвика представляет собой элегантную и эффективную структуру данных, предназначенную для решения задач, связанных с вычислением префиксных сумм массива и выполнением операций обновления элементов массива.

**Постановка задачи:**

Зачастую в задачах программирования возникает необходимость в быстром вычислении суммы элементов массива в заданном диапазоне (от индекса l до индекса r), а также в возможности оперативно обновлять значения отдельных элементов массива. Наивный подход, подразумевающий простое суммирование элементов, обладает линейной временной сложностью O(n) для каждой операции запроса суммы, где n - размер массива. В случае, когда количество запросов на сумму и обновление велико, такой подход становится неэффективным. Целью данной работы является изучение альтернативной структуры данных, позволяющей существенно ускорить выполнение этих операций.

**Актуальность и значимость:**

Дерево Фенвика предоставляет логарифмическую сложность (O(log n)) как для операции вычисления префиксной суммы, так и для операции обновления элемента. Эта особенность делает его незаменимым инструментом в задачах, где важна скорость обработки большого количества запросов на суммирование и обновление элементов массива. В сравнении с более сложными структурами данных, такими как деревья отрезков, дерево Фенвика отличается простотой реализации и меньшим объемом занимаемой памяти.

**Область применения:**

Дерево Фенвика широко применяется в различных областях, таких как:

* **Олимпиадное программирование:** Решение задач, связанных с обработкой массивов, запросами сумм на отрезках и динамическим обновлением данных.
* **Обработка данных:** Анализ данных, требующий быстрого вычисления агрегированных показателей на основе изменяющихся данных.
* **Графика:** Обработка изображений и видео, где необходимо быстро выполнять суммирование пиксельных значений в определенных областях.
* **Финансовая аналитика:** Анализ финансовых данных, требующий оперативного расчета сумм транзакций за определенный период времени.
* **Игровое программирование:** Разработка игр, где требуется быстрое обновление и анализ игрового мира.

1. **Дерево Фенвика**
   1. Теоретические сведения

Алгоритм дерева Фенвика  это структура данных, дерево на массиве, которая обладает следующими свойствами:

* позволяет вычислять значение некоторой обратимой операции *F* на любом отрезке [*L*; *R*] за логарифмическое время;
* позволяет изменять значение любого элемента за *O(log N)*;
* требует памяти *O(N*);

Операция F может представлять собой различные операции, но наиболее распространены вычисление суммы или произведения элементов на заданном интервале. В некоторых случаях, при наличии ограничений и определенных модификаций, F может быть операцией поиска максимального или минимального значения на интервале, а также другими операциями.

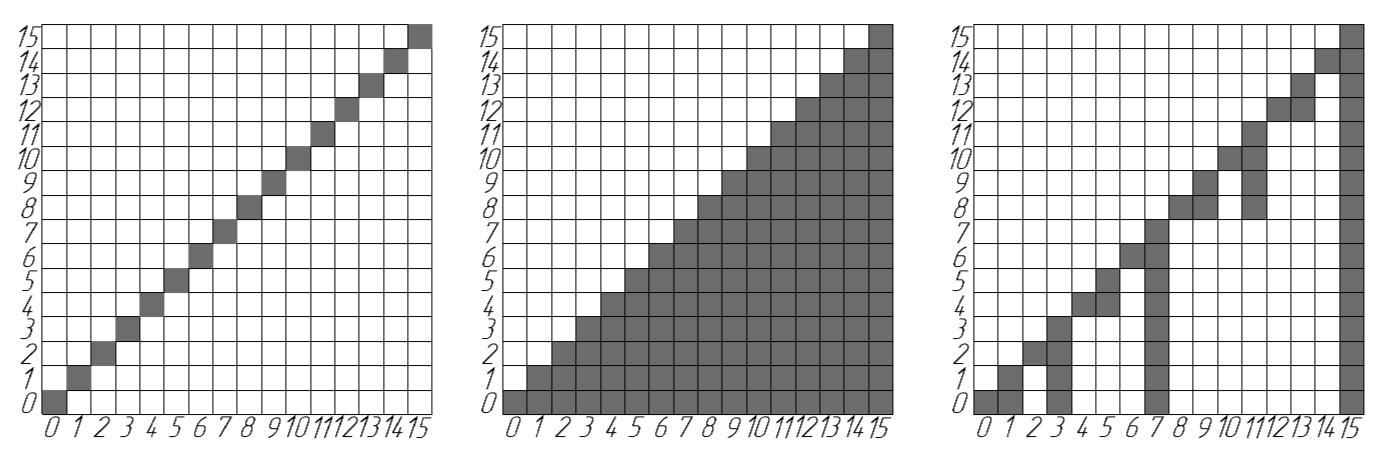
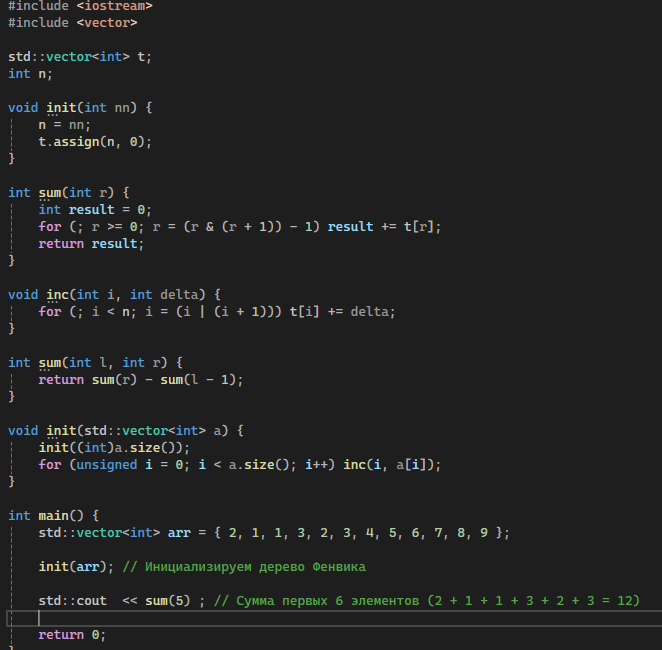


Рисунок 1 - Сравнение массивов и дерева Фенвика

* 1. Нахождение суммы



1.**int sum(int r)**: Вычисляет сумму исходного массива от индекса 0 до (включительно) индекса r.

2. int result = 0;: Инициализирует результат равным 0.

1. for (; r >= 0; r = (r & (r + 1)) - 1)Это основной цикл для вычисления префиксной суммы. Давайте разберём его:
   1. **r >= 0**: Цикл продолжается до тех пор, пока индекс r неотрицателен.
   2. **r = (r & (r + 1)) - 1**Это самая важная часть. Она переходит к следующему «родительскому» узлу в структуре дерева Фенвика. Эта операция очищает младший значащий бит . Например:

Если **r = 6** (двоичный 110), то r & (r + 1) = 110 & 111 = 110. Тогда 110 - 1 = 101, тo r становится 5.

Если **r = 5** (двоичный 101), то r & (r + 1) = 101 & 110 = 100. Тогда 100 - 1 = 111, тo r становится 3.

1. **void inc(int i, int delta)**: Увеличивает значение элемента с индексом i в исходном массиве на величину delta.

* **i = (i | (i + 1))**Это позволяет эффективно перейти к следующему «дочернему» узлу в структуре дерева Фенвика.
* Если i = 2 (двоичное 010), то i | (i + 1) = 010 | 011 = 011. Таким образом, i становится равным 3.
* Если i = 3 (двоичное 011), то i | (i + 1) = 011 | 100 = 111. Таким образом, i становится равным 7.

1. **void init(std::vector<int> a)**: Инициализирует дерево Фенвика, используя значения из существующего вектора a