AA-дерево

Шагиева Динара Ильназовна
группа 11-401

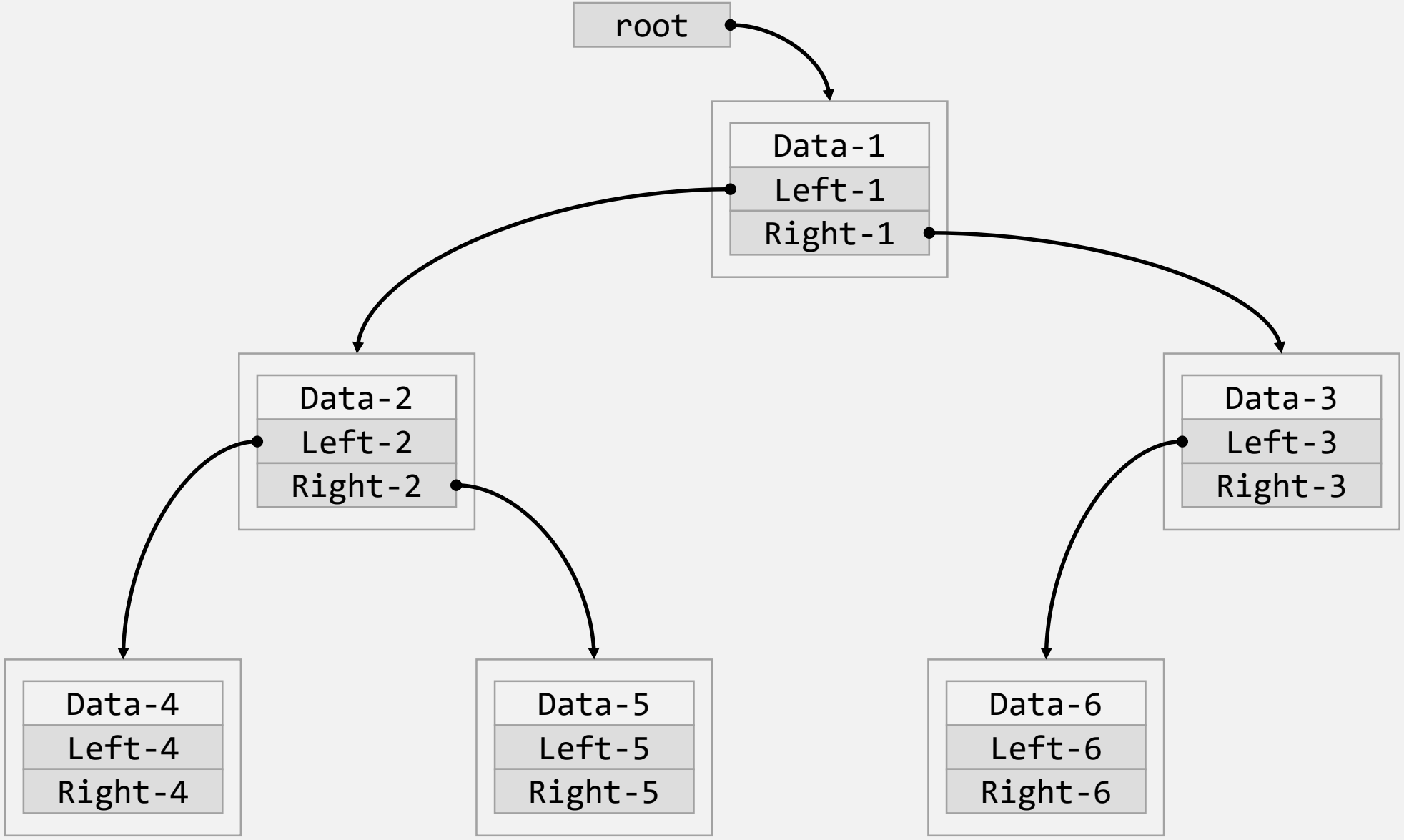
AA-дерево

(англ. AA-Tree)

- сбалансированное двоичное дерево поиска, которое является разновидностью красно-черного дерева с дополнительными ограничениями.

**Данную модификацию красно-черного дерева впервые предложил Арне Андерссон в 1993 году.*

Бинарное дерево



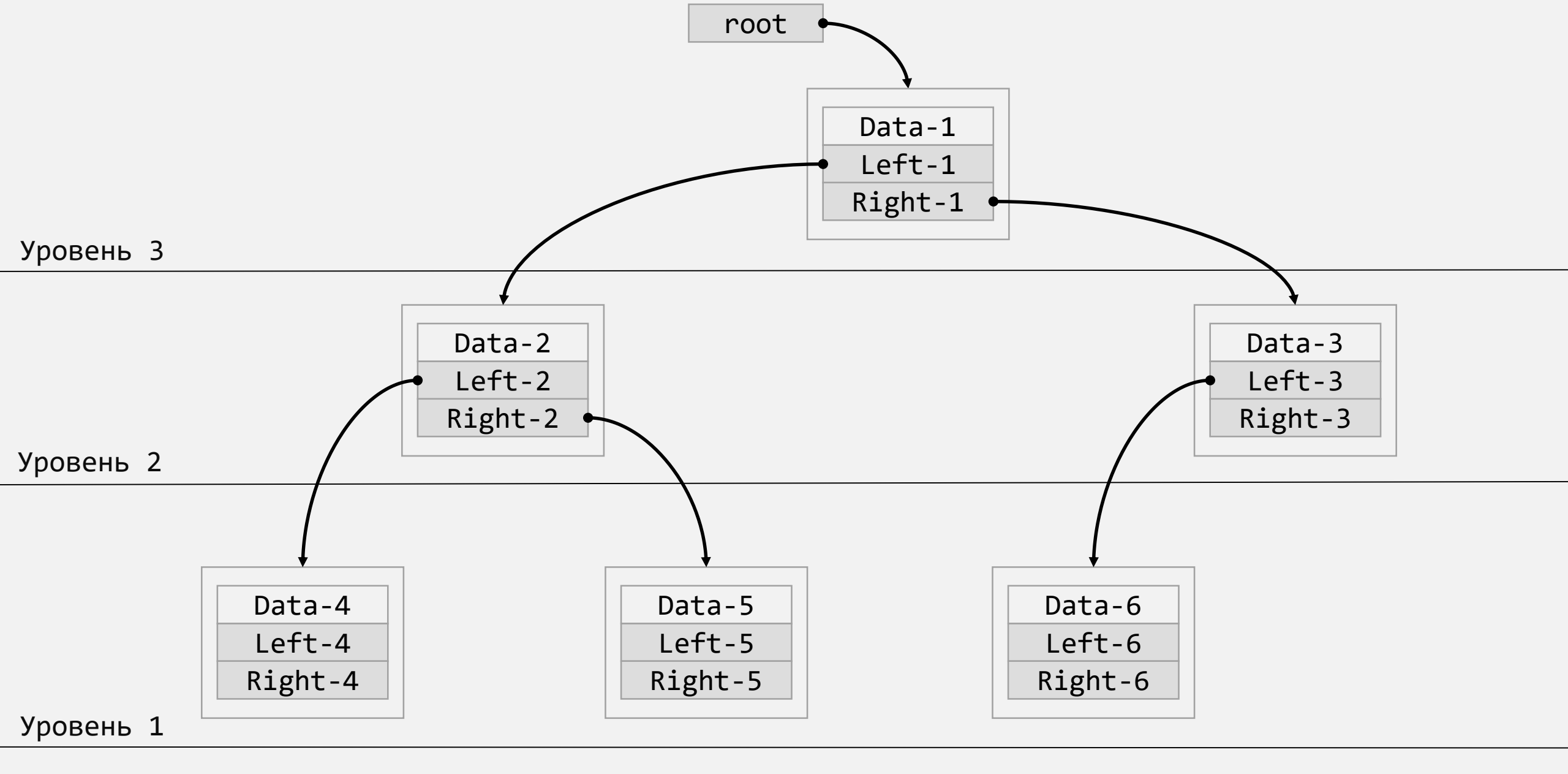
Узел (Node)

```
template< typename T >
struct Node {
  T data;
  int level;
  Node * left;
  Node * right;
};
```

Особенность АА-дерева – уровень вершины

Уровень вершины (англ. *level*) – вертикальная высота соответствующей вершины.

AA-дерево


root
Data-1
Left-1
Right-1
Уровень 3
Data-2
Left-2
Right-2
Data-3
Left-3
Right-3
Уровень 2
Data-4
Left-4
Right-4
Data-5
Left-5
Right-5
Data-6
Left-6
Right-6
Уровень 1

Свойства

- Уровень каждого листа равен 1
- Уровень каждого левого ребенка ровно на один меньше, чем у его родителя
- Уровень каждого правого ребенка равен или на один меньше, чем у его родителя
- Уровень каждого правого внука строго меньше, чем у его прародителя
- Каждая вершина с уровнем больше 1 имеет двоих детей.

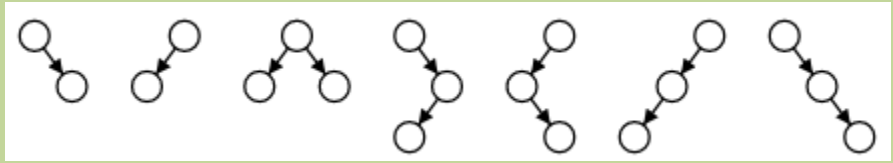
Преимущества

- Оценка на высоту деревьев соответствует оценке для красно-черного дерева - $2\log_2(N)$

- Все операции происходят за O(logN)

Преимущество над К-Ч деревом

- Для поддержки баланса красно-черного дерева необходимо обрабатывать 7 различных вариантов расположения вершин:



- В АА-дереве необходимо обрабатывать только два вида возможных расположений вершин. Мы должны проверить нет ли левой горизонтальной связи (на рис. слева) и нет ли двух последовательных правых горизонтальных связей (на рис. справа)

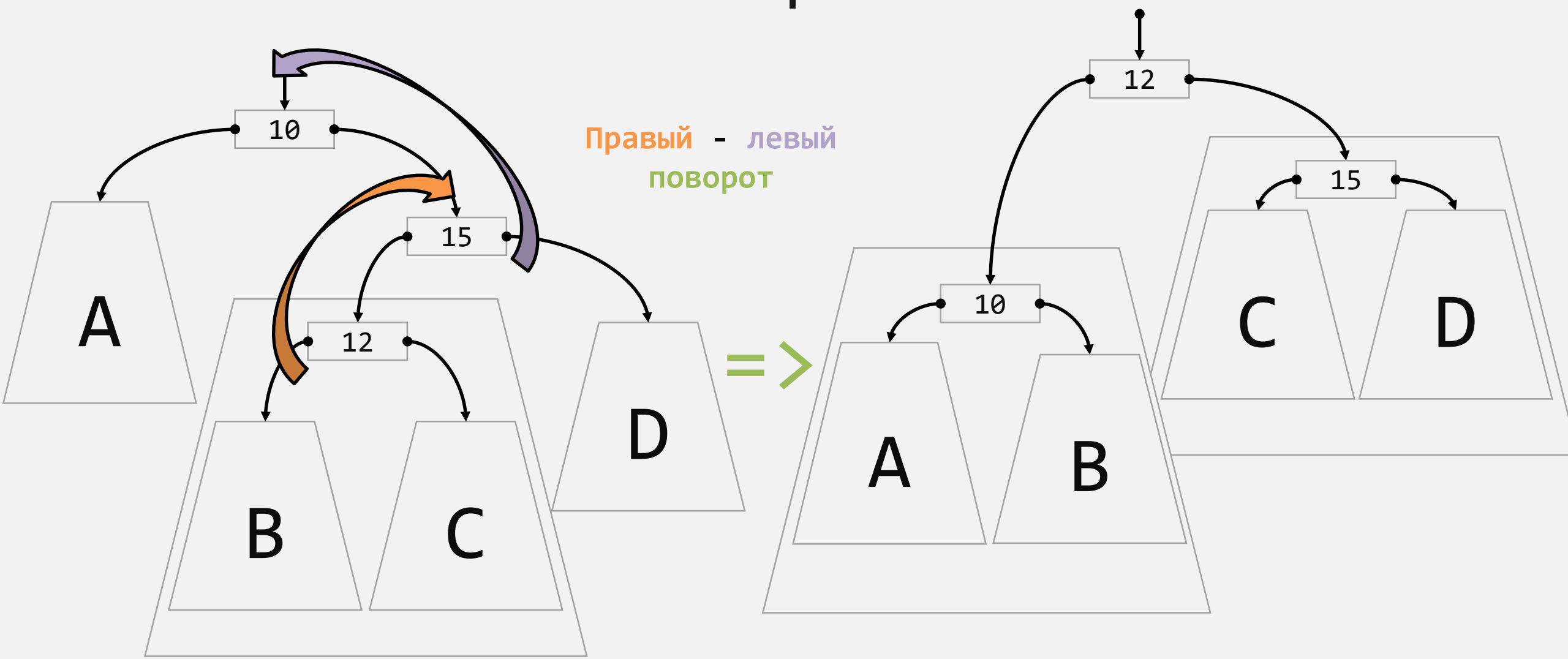
Недостаток

Так как в реализации вместо цвета обычно хранят «уровень» вершины, это может привести к дополнительным расходам по памяти

Основные методы

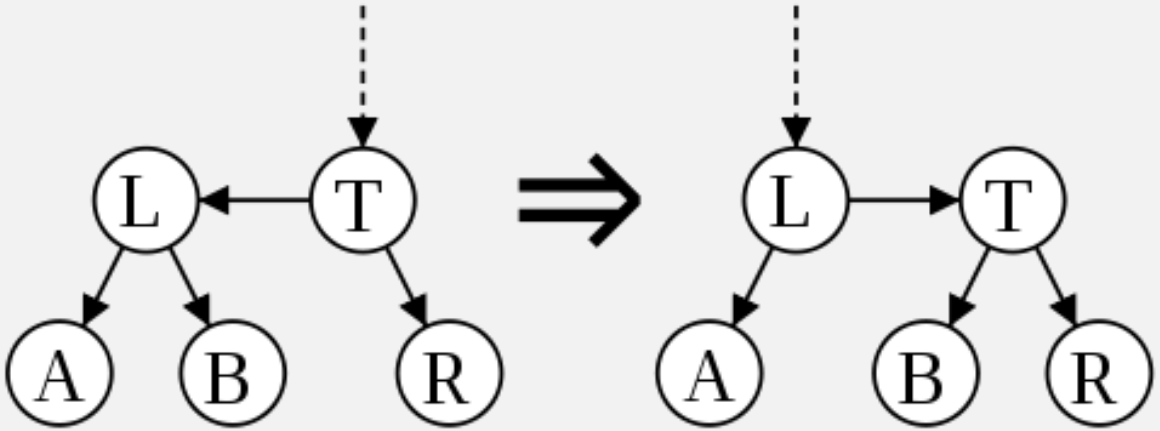
```
void insert(data) – добавляет новый элемент

void remove(data) – удаляет элемент

void clear() – полностью очищает дерево

void swap(first, second) – обменивает данные деревьев

bool contains(data) – проверяет наличие элемента

bool isEmpty() – проверяет пустоту дерева

size_t getSize() – возвращает размер (кол-во узлов) дерева
```

Повороты



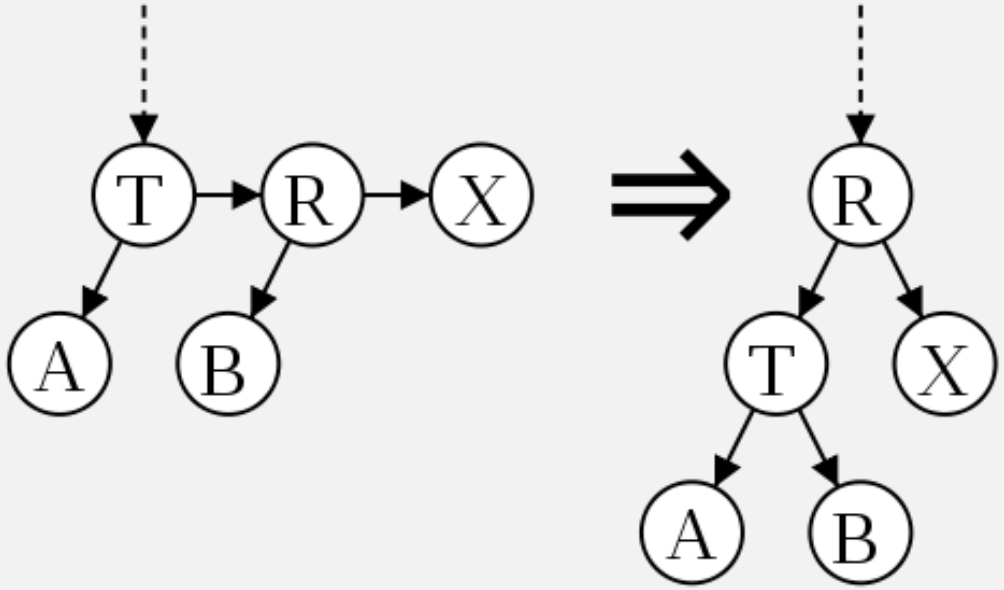
Балансировка

```
template <typename T>
Node * skew(Node * node)
{
  if (node != nullptr && node->left != nullptr)
    //Проверяем, одного ли уровня текущий узел и его левый сын
    if (node->left->level == node->level)
    {
      Node * leftChild = node->left;
      node->left = leftChild->right;
      leftChild->right = node;
      node = leftChild;
    }

  return node;
}
```

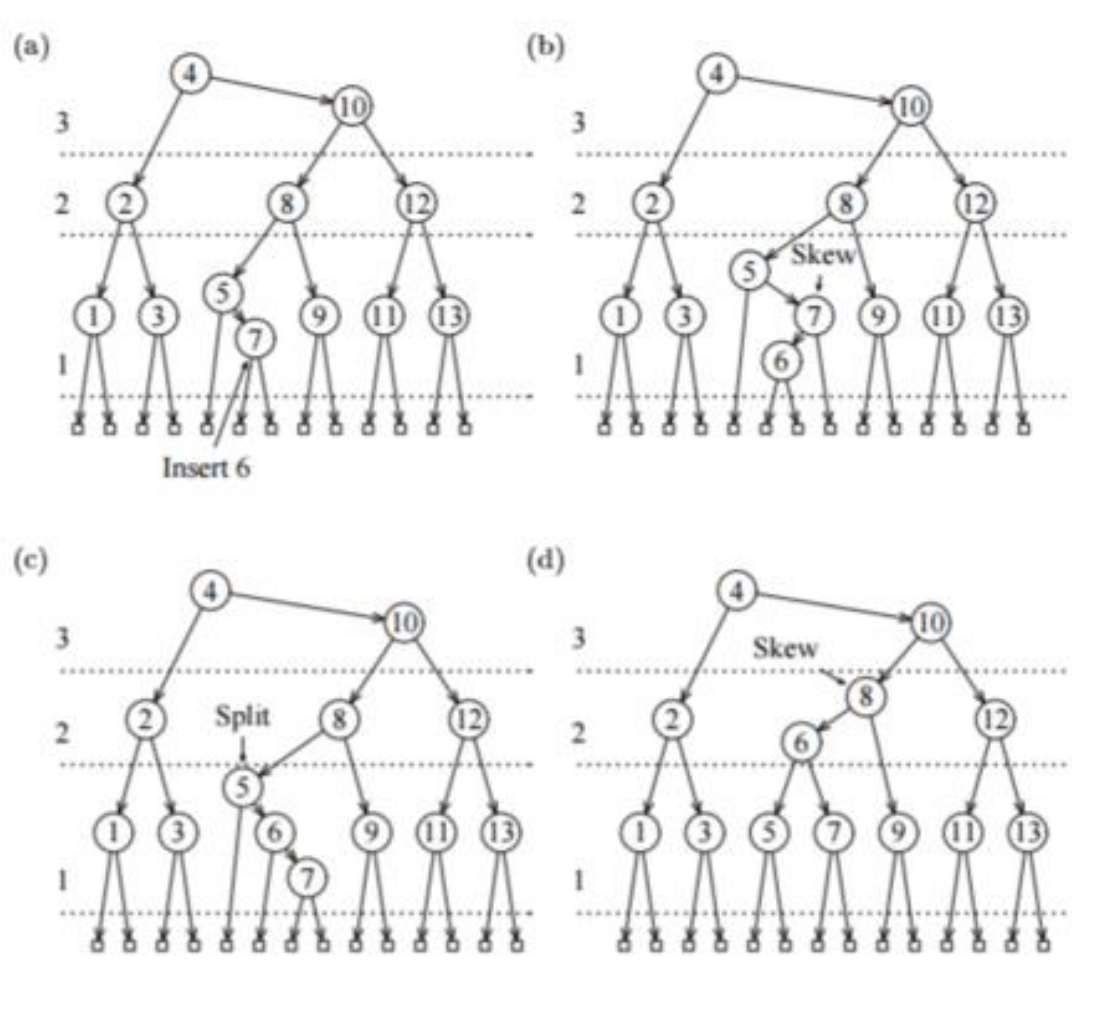


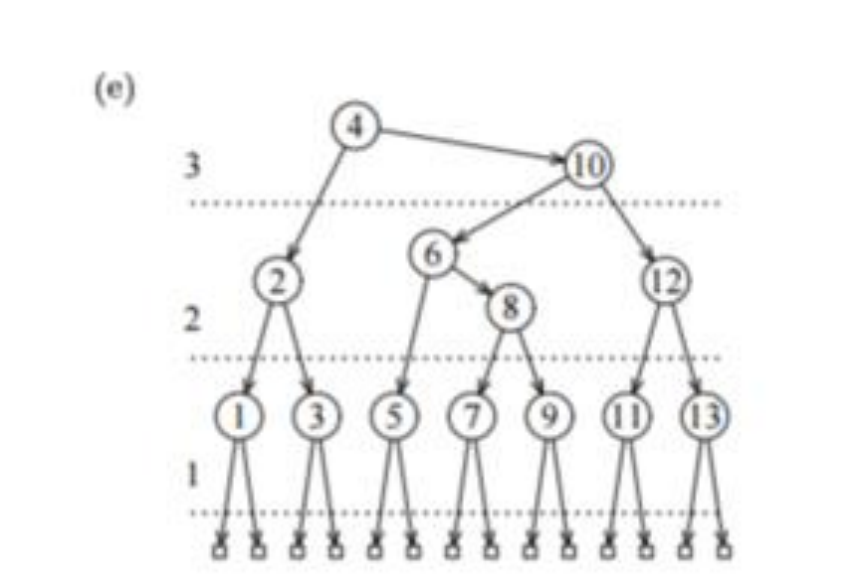
Балансировка

```
template <typename T>
Node * split(Node * node)
{
  if (node != nullptr && node->right != nullptr && node->right->right != nullptr)
    //Проверяем, одного ли уровня текущий узел и его правый внук
    if (node->level == node->right->right->level)
    {
      Node * rightChild = node->right;
      node->right = rightChild->left;
      rightChild->left = node;
      rightChild->level += 1;
      node = rightChild;
    }

  return node;
}
```

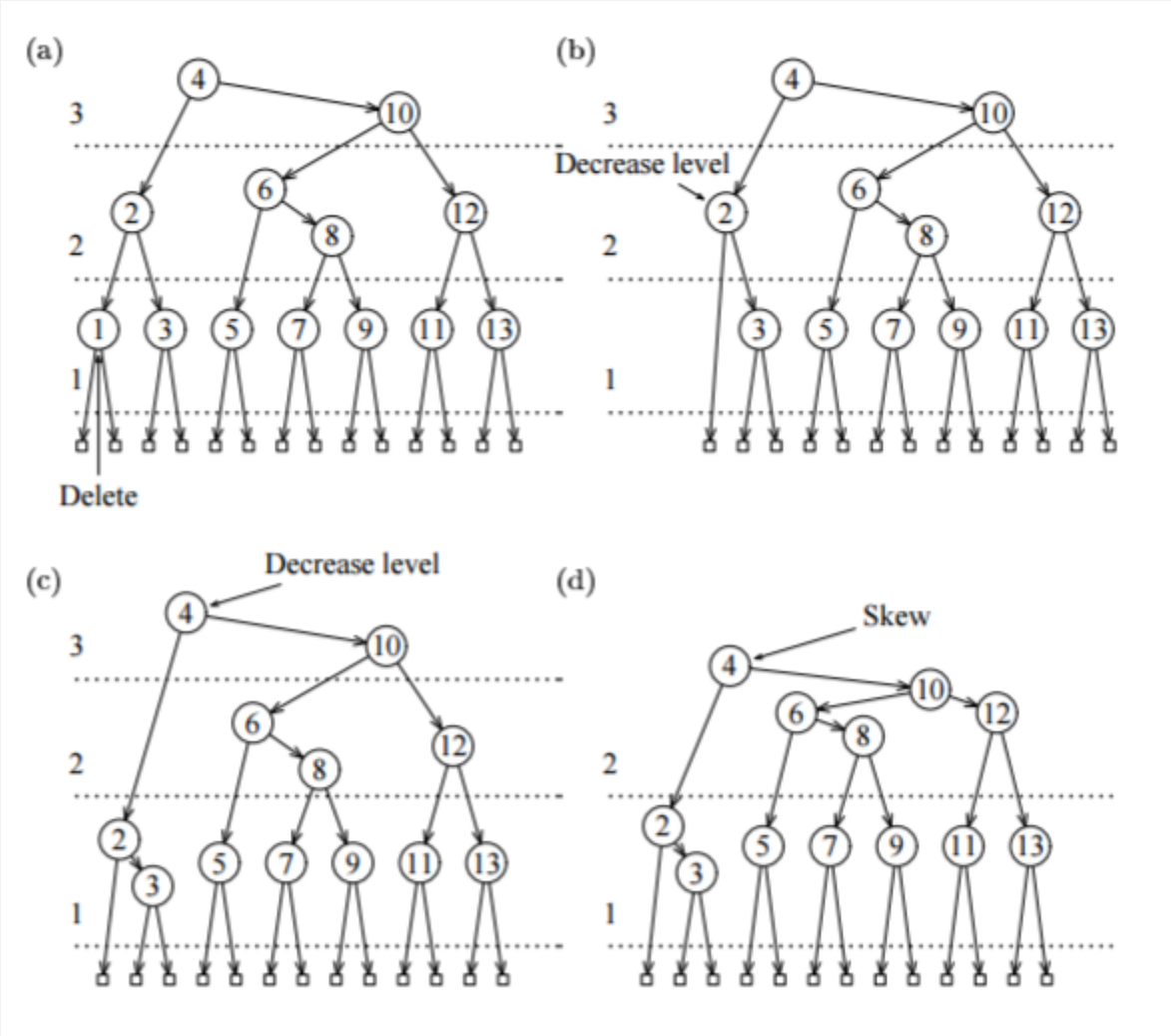


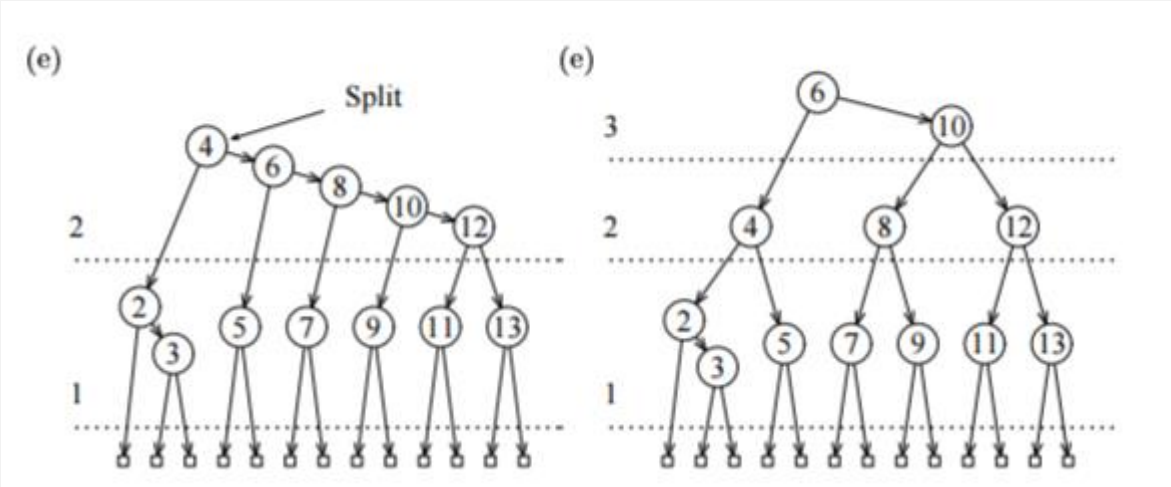
Вставка





Удаление





Использование

АА-деревья лучше использовать для создания и ведения упорядоченных списков, где нужно чаще искать значения, чем добавлять или удалять их

Например: базы данных, файловые системы, реализация стандартных библиотек, криптография, сетевые технологии