

# Лекция 4 от 21.01.2016

## Двоичное дерево поиска

Так как курс всё же называется «Алгоритмы и *структуры данных*», рассмотрим такую структуру данных, как **двоичное дерево поиска**.

Во-первых, это двоичное дерево — есть узлы и связи между ними, при этом у каждого узла не более двух детей. В вершинах дерева — числа, при этом в левом поддереве узла все числа не больше, чем в самом узле; в правом же поддереве, наоборот, не меньше.

Введём обозначения: пусть  $x$  — некоторый узел. Тогда

- $x.key$  — число в узле;
- $x.left$  — левый потомок;
- $x.right$  — правый потомок;
- $x.p$  — родитель.

При этом считаем, что у каждого узла есть оба наследника, но некоторые могут узлами не являться и иметь значение NULL.

Указанные выше свойства двоичного дерева поиска можно записать так:

$$y \in \text{Tree}(x.left) \implies y.key \leq x.key$$

$$y \in \text{Tree}(x.right) \implies y.key \geq x.key$$

Что с этим деревом можно делать? Для начала, это дерево можно обойти так, чтобы перечислить элементы в порядке возрастания:

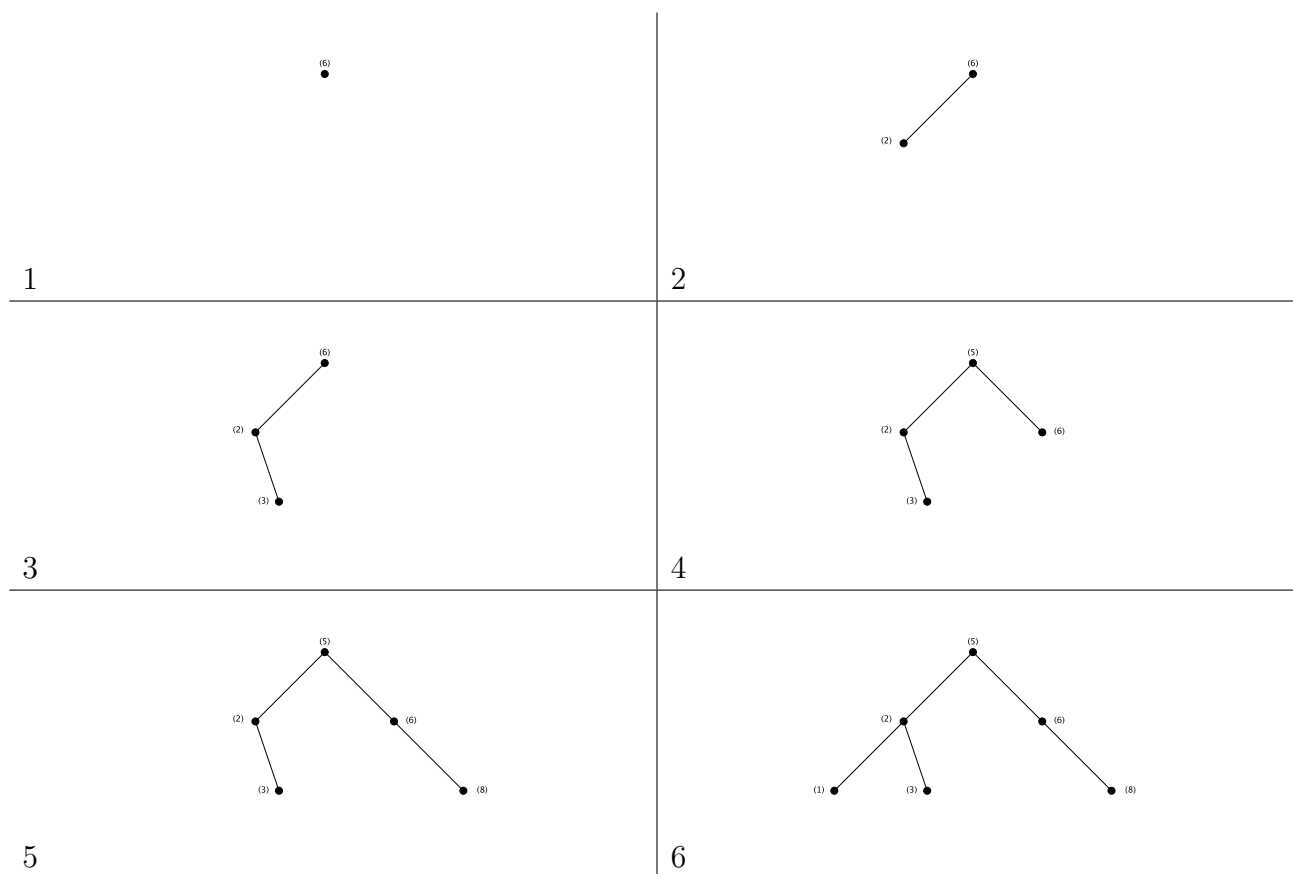
```
inorder_tree_walk(x)
  if x != NULL then
    inorder_tree_walk(x.left)
    output x.key
    inorder_tree_walk(x.right)
```

Сложность алгоритма —  $\Theta(n)$ ; каждый узел мы посещаем не более одного раза, при этом операции внутри узла занимают константное время.

Запишем алгоритм сортировки с помощью дерева:

```
tree_sort(a)
  t := Tree()
  for x in a do
    tree_insert(x, t)
  inorder_tree_walk(t)
```

Пусть  $a = [6, 2, 3, 5, 8, 1]$ . Тогда построение дерева может выглядеть так (может и по-другому, дерево далеко не всегда строится однозначно):



Заметим, что в общем и целом, это довольно похоже на алгоритм быстрой сортировки (особенно, если не двигать корневой элемент при вставке новых), а корень (под-)дерева — опорный элемент.

Время работы алгоритма:  $T(n) = 2T(\frac{n}{2}) + cn \implies \Theta(n \log n)$

Предположим, что слияние дорогое:

$T(n) = 2T(\frac{n}{2}) + cn^2 \implies O(n^2 \log n)$

HERE BE KARTINKA

$i$ -ый уровень:

- Число подзадач =  $2^i$
- Размер подзадачи =  $\frac{n}{2^i}$
- Время на решение подзадачи =  $c \left(\frac{n}{2^i}\right)^2$
- Всего работы =  $\frac{cn^2}{2^{2i}} \cdot 2^i = \frac{cn^2}{2^i}$

$$T(n) \leq \sum_{i=0}^{\log_2 n - 1} \frac{cn^2}{2^i} = cn^2 \sum \frac{1}{2^i} \leq 2cn^2 = O(n^2)$$

$$T(n) = kn^d$$

Удобно начинать индукцию с шага:

Пусть  $T(m) \leq km^d$  для  $m < d$ .

$$T(n) \leq 2T\left(\frac{n}{2}\right) + cn^2 \leq 2k\left(\frac{n}{2}\right)^d + cn^2 = 2k\frac{n^d}{2^d} + cn^2 = \{d=2\} = 2k\frac{n^2}{4} + cn^2 = \frac{k}{2}n^2 + cn^2 = \{k=2c\} = kn^2$$

$$\text{База: } T(2) \leq c \leq 2c \cdot 2^2$$

\*я перестал понимать происходящее и переписываю формулы\*

$$T(n) = aT\left(\frac{n}{2}\right) + cn \implies O(n^{\log_2 a})$$

$$T(n) \leq kn^d$$

$$T(n) = aT\left(\frac{n}{2}\right) + cn \leq ak\frac{n^d}{2^d} + cn = \{d = \log_2 a\} = kn^d + cn$$

$$T(n) \leq kn^d - ln$$

$$T(n) = aT\left(\frac{n}{2}\right) + cn \leq a\left(k\frac{n^d}{2^d} - l\frac{n}{2}\right) + cn = kn^d - al\frac{n}{2} + cn = kn^d - \left(\frac{al}{2} - c\right)n = \{l = \frac{al}{2} - c\} = kn^d - ln$$

$$\text{База: } T(2) \leq c \leq k \cdot 2^d - 2l$$

$$T(n) = T\left(\frac{n}{2}\right) + cn$$

$$T(n) = \sum_{i=0}^{\log_2 n} c\frac{n}{2^i} = cn \sum \frac{1}{2^i} \leq 2cn$$

Или с помощью метода частичной подстановки:

$$T(n) \leq kn^d$$

$$T(n) = T\left(\frac{n}{2}\right) + cn \leq \frac{kn}{2} + cn = kn^d \text{ аааааааааааа}$$

$$T(n) = aT\left(\frac{n}{b}\right) + cn^d$$

HERE BE KARTINKA

$i$ -ый уровень:

- Число подзадач  $= a^i$
- Размер подзадачи  $= \frac{n}{b^i}$
- Время на решение подзадачи  $= c\left(\frac{n}{b^i}\right)^d$
- Всего работы  $= c\left(\frac{n}{b^i}\right)^d \cdot a^i = cn^d\left(\frac{a}{b^d}\right)^i$

$$\text{Всего} \leq cn^d \sum_{i=0}^{\log_b n} \left(\frac{a}{b^d}\right)^i$$

1.  $a = b^d$ :  $O(n^d \log n)$
2.  $a < b^d$ :  $O(n^d)$
3.  $a > b^d$ :  $\sum \left(\frac{a}{b^d}\right)^i = O\left(\left(\frac{a}{b^d}\right)^{\log_b n}\right) \dots \dots \dots O(n^{\log_b a})$