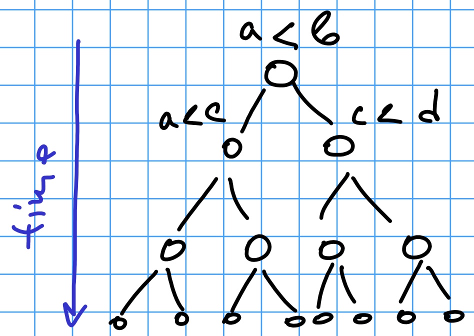
Сортировки

$|Output| = 1000 \cdot \max i$

$\log_2 |Output| = 14$ Код Шеннона

Нижняя оценка на сортировку сравнениями



Decision Trees

Вход: n чисел

$T(n) = height(T) + 1$

$T(n) \geq \Omega(\,?\,)$

$\forall$ эл-т. должен поучаствовать в сравнении $\Rightarrow \Omega(n/2)$

$\#$ листьев $\geq \#$ различных output-ов $= n!$

$h(Tree) \geq \lceil \log_2 \#$ листьев $\rceil$

$T(n) = h(Tree) \geq \log_2 n! = \log_2 \sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^n =$

$\geq \frac{n}{2} \cdot \log \frac{n}{2} = \Omega(n \log n)$

$= O(\log_2 n^n - \log e^n) =$
$= O(n \log n)$

Сортировки за $O(n^2)$

- Bubble Sort — n^2 сравн., n^2 перестановок
- Insertion Sort — n^2 $(n \log n)$ сравн., n^2 перест.
- Selection Sort — $\frac{n(n+1)}{2}$ сравн., n перест.

Сортировки за $O(n \log n)$

- Merge Sort $n \log n$, $O(n)$ памет
  стабилна

Stable: $(\underline{1},2)\ (3,4)\ (\underline{\underline{1}},5) \to (\underline{1},2)(\underline{\underline{1}},5)(3,4)$

- Heap Sort

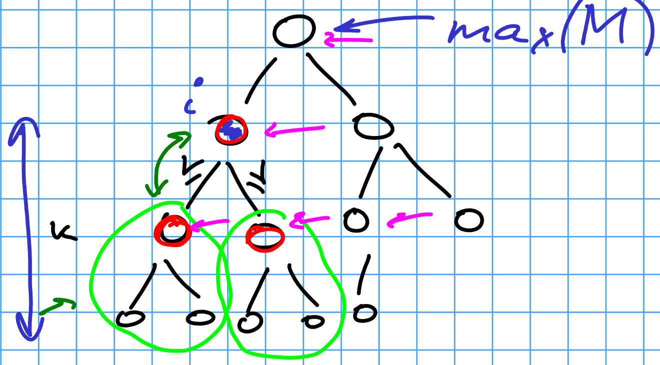
Куча (Heap) / Пирамида

M [| | | | | | | | | | | | | | | |] масив из n ел-ов.

$Left(i) = 2i$

$Right(i) = 2i+1$

$Parent(i) = \lfloor i/2 \rfloor$



Свойство Кучи:

$$M[Parent(i)] \geq M[i] \quad \forall i \leq Size(M)$$

```
Heapify(M, i):
// Left(i) и Right(i) - уже кучи
    largest = i
    if Left(i) ≤ Size(M) and M[Left(i)] > M[i]
        largest = Left(i)
    if Right(i) ≤ Size(M) and M[Right(i)] > M[largest]
        largest = Right(i)
    if i ≠ largest:
        M[i] <-> M[largest]
        Heapify(M, largest)
```

$T(Heapify(M, i)) = O(height(i))$

$O(\log n)$

BuildHeap(M):

for $i = \lceil Size(M)/2 \rceil$ to 1

Heapify(M, i)

$\frac{n}{2} \cdot O(\log n)$

$O(n \log n)$

$$\sum_{k=1}^{\log_2 n} \frac{n}{2^k} \cdot O(k) = n \cdot O\left(\sum_{k=1}^{\log_2 n} \frac{k}{2^k}\right) \leq n \cdot O\left(\sum_{k=1}^{\infty} \frac{k}{2^k}\right)$$

$$= O(n)$$

$$S = \sum_{k=1}^{\infty} \frac{k}{2^k} = 2$$

$$S = \sum_{k=1}^{\infty} \frac{1}{2^k} + \sum_{k=1}^{\infty} \frac{k-1}{2^k} = 1 + \frac{1}{2} S$$

$$S = 2$$

Extract-Max(M):

M[1] $\longleftrightarrow$ M[Size(M)]

Size(M) = Size(M) − 1

Heapify(M, 1)

$O(\log n)$

HeapSort(M[1..n])

BuildHeap(M) $\quad O(n)$

for i = 1 to n

Extract-Max(M) $\quad O(\log n)$

$O(n \log n)$

```
Insert (M, x)
    M[Size(M)+1] = x
    Size(M) = Size(M) + 1
    i = size(M)
    while i ≠ 1 and M[Parent(i)] < M[i]
        M[Parent(i)] <--> M[i]
        i = Parent(i)
```

$O(\log n)$

Сортировки за линейное время

- сортировка подсчётом

вход: массив целых чисел размера N
числа из $[0, K]$, где $K = O(N)$

```
Counting Sort (M[1, N]):
    C[1..K]
    for i = 1 to N
        C[M[i]] += 1
    S = 1
    for i = 1 to K
        S += C[i]
        C[i] = S - C[i]
    R[1..N]
    for i = 1 to N
        R[C[M[i]]] = M[i]
        C[M[i]] += 1
    return R
```

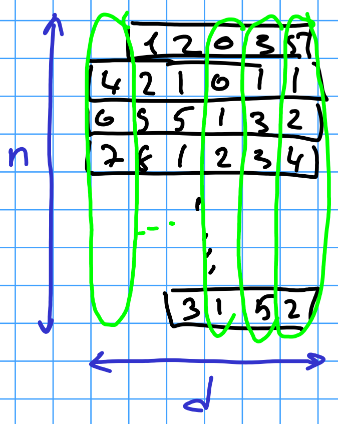
k | 0 1 0 3 0 5 |

| c 1 0 2 0 5 |

$O(n)$

- цифровая сортировка

Вход: n чисел длины d



$O(n \cdot d)$

← counting sort