AULA 8

Heapsort

CLRS 6
(veja tb CLRS B.5.3)

Heap sort

Rearranja $A[1\,..\,n]$ em ordem crescente.

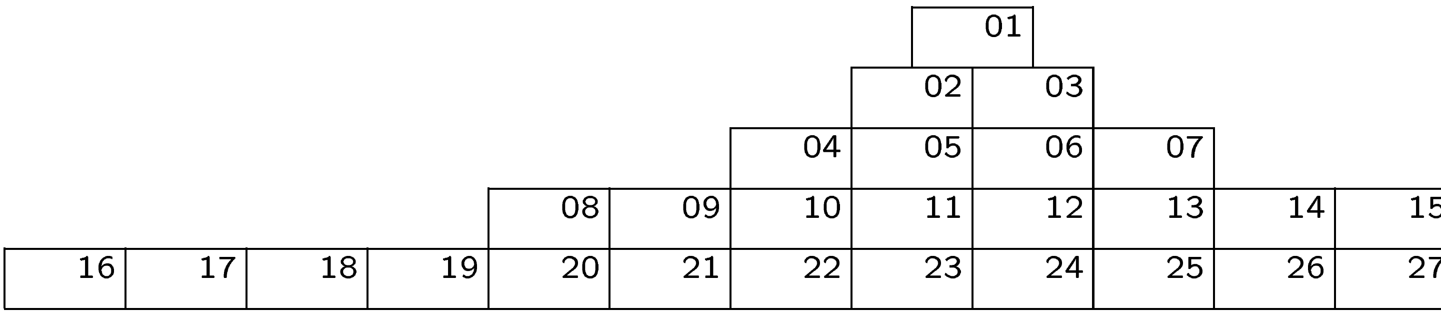
HEAPSORT (A, n)
0 BUILD-MAX-HEAP (A, n)
1 $m \leftarrow n$
2 para $i \leftarrow n$ descrescendo até 2
3   faça $A[1] \leftrightarrow A[i]$
4         $m \leftarrow m - 1$
5   MAX-HEAPIFY $(A, m, 1)$

1						m				n
66	44	55	44	22	33	11	77	88	99	99

Invariante: no início de cada execução de 2–5,

$A[m{+}1\,..\,n]$ grandes, ordem crescente
$A[1\,..\,m]$ pequenos, max-heap

Estrutura de heap



$A[1\,..\,29]$

filho esquerdo de i	$2i$
filho direito de i	$2i+1$
pai de i	$\lfloor i/2 \rfloor$
nível de i	$\lfloor \lg i \rfloor$ (nível p tem 2^p nós)
altura de i	$\lfloor \lg \frac{m}{i} \rfloor$
raiz ($i = 1$)	nível $= 0$
folha	altura $= 0$

$A[1\,..\,m]$ é um **max-heap** se $A[\lfloor \frac{i}{2} \rfloor] \geq A[i]$ para cada nó i.

Exercício 8.A

A **altura** de i em $A[1 \mathinner{\ldotp\ldotp} m]$ é o comprimento da mais longa seqüência da forma

$$f(i), f(f(i)), f(f(f(i))), \ldots ,$$

onde $f(i)$ vale $2i$ ou $2i+1$. Mostre que a altura de i é

$$\lfloor \lg \tfrac{m}{i} \rfloor$$

Exercício 8.B

Mostre que um heap $A[1\,..\,m]$ tem no máximo $\lceil m/2^{h+1} \rceil$ nós com altura h.

EXEMPLO: N_h = número de nós à altura h

m	$\lfloor m/2 \rfloor$	N_0	$\lceil m/2 \rceil$	$\lfloor m/4 \rfloor$	N_1	$\lceil m/4 \rceil$
16	8	8	8	4	4	4
17	8	9	9	4	4	5
18	9	9	9	4	5	5
19	9	10	10	4	5	5
20	10	10	10	5	5	5
21	10	11	11	5	5	6
22	11	11	11	5	6	6
23	11	12	12	5	6	6
24	12	12	12	6	6	6

Exercício 8.C

Mostre que $\lceil m/2^{h+1} \rceil \leq m/2^h$ quando $h \leq \lfloor \lg m \rfloor$.

Rotina básica de manipulação de max-heap

```
MAX-HEAPIFY (A, m, i)
l ← 2i
r ← 2i + 1
se l ≤ m e A[l] > A[i]
    então maior ← l
    senão maior ← i
se r ≤ m e A[r] > A[maior]
    então maior ← r
se maior ≠ i
    então A[i] ↔ A[maior]
          MAX-HEAPIFY (A, m, maior)
```

Recebe $A[1\,..\,m]$ e $i \geq 1$ tais que subárvores com raiz $2i$ e $2i+1$ são max-heaps. Rearranja de modo que subárvore com raiz i seja max-heap.

h := altura de i = $\lfloor \lg \frac{m}{i} \rfloor$

$T(h)$:= consumo de tempo no pior caso

Recorrência:

$$T(h) = T(h-1) + \mathrm{O}(1)$$

pois altura de $maior$ é $h-1$

Solução: $T(h) = \mathrm{O}(h)$

Prova?

Como $h \leq \lg m$, podemos dizer que consumo de tempo de MAX-HEAPIFY (A, m, i) é

$$\mathrm{O}(\lg m)$$

Construção de um max-heap

BUILD-MAX-HEAP (A, n)
para $i \leftarrow \lfloor n/2 \rfloor$ descrescendo até 1
    faça MAX-HEAPIFY (A, n, i)

$T(n) :=$ consumo de tempo no pior caso

$T(n) = \frac{n}{2}\,\mathrm{O}(\lg n) = \mathrm{O}(n \lg n)$.

Análise mais cuidadosa: $T(n) = \mathrm{O}(n)$.

PROVA:
Digamos que MAX-HEAPIFY (A, n, i) consome $h = \lfloor \lg \frac{n}{i} \rfloor$ unidades de tempo.
Então

$$
\begin{aligned}
T(n) &= \sum_{h=1}^{\lfloor \lg n \rfloor} \left\lceil \frac{n}{2^{h+1}} \right\rceil h \\
&\leq \sum_{h=1}^{\lfloor \lg n \rfloor} \frac{n}{2^h} h \\
&\leq n \Big(\frac{1}{2^1} + \frac{2}{2^2} + \frac{3}{2^3} + \cdots + \frac{\lfloor \lg n \rfloor}{2^{\lfloor \lg n \rfloor}} \Big) \\
&< n \frac{1/2}{(1-1/2)^2} \\
&= 2n
\end{aligned}
$$

Comentário: $x^0 + x^1 + x^2 + x^3 + \cdots = \dfrac{1}{1-x}$

$1x^0 + 2x^1 + 3x^2 + \cdots = \dfrac{1}{(1-x)^2}$

$1x^1 + 2x^2 + 3x^3 + \cdots = \dfrac{x}{(1-x)^2}$

Consumo de tempo do Heap sort

```
HEAPSORT (A, n)
0  BUILD-MAX-HEAP (A, n)
1  m ← n
2  para i ← n descrescendo até 2
3      faça A[1] ↔ A[i]
4             m ← m − 1
5             MAX-HEAPIFY (A, m, 1)
```

linha	consumo na linha
0	O(n)
1	O(1)
2	O(n)
3	O(n)
4	O(n)
6	nO(lg n)

Total O(n lg n)

TAREFA (AULA 8)

Exercício 8.B'
Mostre que um heap $A[1\,..\,m]$ tem no mínimo $\lfloor m/2^{h+1} \rfloor$ nós com altura h.

Exercício 8.D
Considere um heap $A[1\,..\,m]$; a raiz do heap é o elemento de índice 1. Seja m' o número de elementos do "sub-heap esquerdo", cuja raiz é o elemento de índice 2. Seja m'' o número de elementos do "sub-heap direito", cuja raiz é o elemento de índice 3. Mostre que

$$m'' \leq m' < 2m/3\,.$$

Exercício 8.E
Mostre que a solução da recorrência

$$\begin{array}{rcl} T(1) & = & 1 \\ T(k) & \leq & T(2k/3) + 5 \quad \text{para } k \geq 2 \end{array}$$

é $\mathrm{O}(\log k)$. Mais geral: mostre que se $T(k) = T(2k/3) + \mathrm{O}(1)$ então $\mathrm{O}(\log k)$. (Curiosidade: Essa é a recorrência do MAX-HEAPIFY (A, m, i) se interpretarmos k como sendo o número de nós na subárvore com raiz i).

Exercício 8.F
Escreva uma versão iterativa do algoritmo MAX-HEAPIFY. Faça uma análise do consumo de tempo do algoritmo.

Exercício 8.G
Discuta a seguinte variante do algoritmo MAX-HEAPIFY:

```
M-H (A, m, i)
  l ← 2i
  r ← 2i + 1
  se l ≤ m e A[l] > A[i]
      então A[i] ↔ A[l]
            M-H (A, m, l)
  se r ≤ m e A[r] > A[i]
      então A[i] ↔ A[r]
            M-H (A, m, r)
```