

Zad 2.

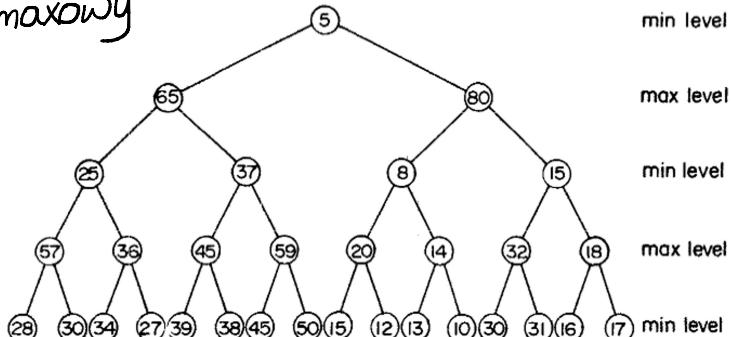
niedziela, 5 marca 2023 19:44

2. (1pkt) Napisz w pseudokodzie procedury:

- 1) • przywracania porządku
- 2) • usuwania minimum
- 3) • usuwania maksimum

z kopca minimaksowego. Przyjmij, że elementy tego kopca pamiętane są w jednej tablicy (określ w jakiej kolejności). Użyj pseudokodu na takim samym poziomie szczegółowości, na jakim zostały napisane w Notatce nr 2 odpowiednie procedury dla zwykłego kopca.

kopiec
min-maxowy



2. MIN-MAX HEAPS

Given a set S of values, a min-max heap on S is a binary tree T with the following properties:

- 1) T has the heap-shape
- 2) T is min-max ordered: values stored at nodes on even (odd) levels are smaller (greater) than or equal to the values stored at their descendants (if any) where the root is at level zero. Thus, the smallest value of S is stored at the root of T , whereas the largest value is stored at one of the root's children; an example of a min-max heap is shown in Figure 1

FIGURE 1. Sample of a Min-Max Heap. The heap condition alternates between minimum and maximum from level to level.

RESEARCH CONTRIBUTIONS

Programming
Techniques
and Data Structures

Ian Munro
Editor

Min-Max Heaps and Generalized Priority Queues

M. D. ATKINSON, J.-R. SACK, N. SANTORO, and T. STROTHOTTE

1) procedure repair($K[1..n]$) $O(n \log n)$
 i $\leftarrow \lfloor \frac{n}{2} \rfloor$ ↳ liscie sa juz dobrymi kopcami wiec zaczynamy w predostatniej warstwie
 dopoki $i \geq 0$
sprawdzamy na jakim poziomie jessemu level $\leftarrow \lfloor \log_2 i \rfloor$
 jeśli level parzysty to
 zepchnij-min($K[1..n]$, i)
 wpp
 zepchnij-max($K[1..n]$, i)
 i $\leftarrow i - 1$
 procedure zepchnij-min($K[1..n]$, i)
 jeśli wiezchotek i ma dzieci to
 m \leftarrow indeks najmniejszego spośród dzieci i wnukow (jeśli istnieją)
 wiezchotka i
 jeśli m to wnuk i to
 jeśli $K[m] < K[i]$ to
 zamień $K[m]$ z $K[i]$
 nie wiemy nic o ... m \leftarrow ojciec m
 jeśli $K[m] > K[\text{ojciec } m]$ to
 zamień $K[m]$ z $K[\text{ojciec } m]$
 jeśli
 na poziomie min wszyscy mają być wieksi, dlatego minimum

nie wiemy nic o
 relacji $K[i]$ i $K[\text{ojciec } m]$
 więc dopewniamy się
 że nic nie
 zepsuliśmy i
 ewentualnie
 naprawiamy

zamień $K[m]$ z nulu
 jeśli $K[m] > K[\text{ojciec } m]$ to
 zamień $K[m]$ z $K[\text{ojciec } m]$
 zepchnij - $\min(K[1..n], m)$
 w przeciwnym wypadku
 jeśli $K[m] < K[i]$ to
 zamień $K[m]$ z $K[i]$

jeśli
 $K[m] \geq K[i]$
 to struktura
 kopca
 zachowana

nie musimy
 robić nic więcej
 bo taka sytuacja
 występuje tylko jeśli
 nie ma wnuków

(spychamy do dobrego
 kopca minmaxowego,
 zatem jeśli minimum jest
 na poziomie max to znaczy
 że nie ma wnuków)

procedure zepchnij-max
 analogicznie tylko
 zmienione $<$ na $>$
 i m największy

2) procedure delmin($K[1..n]$)

$K[1] \leftarrow K[n]$
 zepchnij-min($K[1..n-1]$, 1)

3) procedure delmax($K[1..n]$)

$m \leftarrow$ indeks największego z elementów
 $K[2] \cup K[3]$

$K[m] \leftarrow K[n]$
 zepchnij-max($K[1..n-1]$, m)