

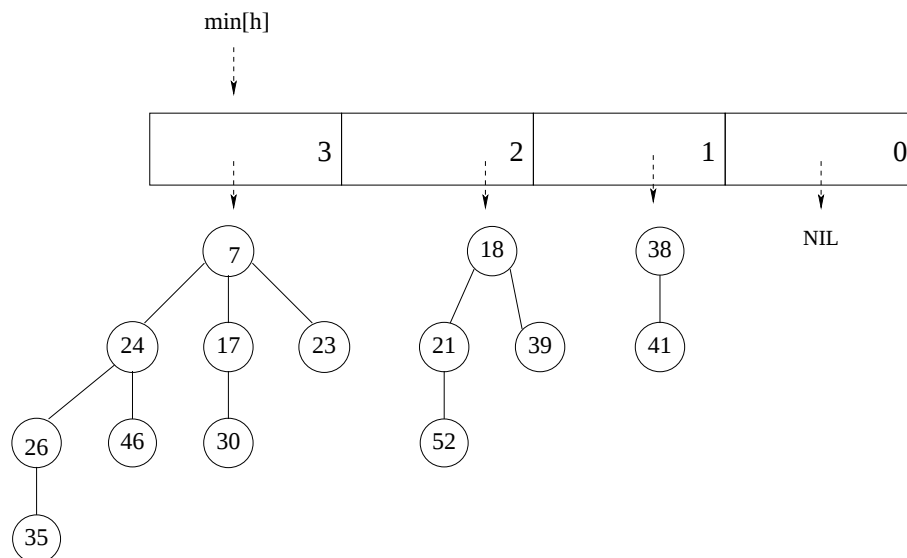
Effiziente Algorithmen / Theoretische Informatik III

2. Übung

1. Aufgabe: Im *Dijkstra-Algorithmus* führen unterschiedliche Datenstrukturen zu verschiedenen Laufzeiten. Überlegen Sie sich, warum der *Dijkstra-Algorithmus* nicht schneller als $O(|V| \cdot \log |V|)$ implementiert werden kann. Egal mit welcher Datenstruktur.

2. Aufgabe: Führen Sie eine amortisierte Analyse für die Addition von n Einsen (beginnend mit 0) in der Ternärdarstellung durch. Wie lange kann eine einzelne Addition dauern?

3. Aufgabe: Führen Sie $\text{DeleteMin}(h)$ auf folgendem binomialen Heap h aus:



4. Aufgabe: Wir betrachten eine Folge $\sigma = \{\sigma_1, \sigma_2, \dots, \sigma_n\}$ von Operationen auf einem anfangs leeren binomialen Heap mit „lazy“ meld.

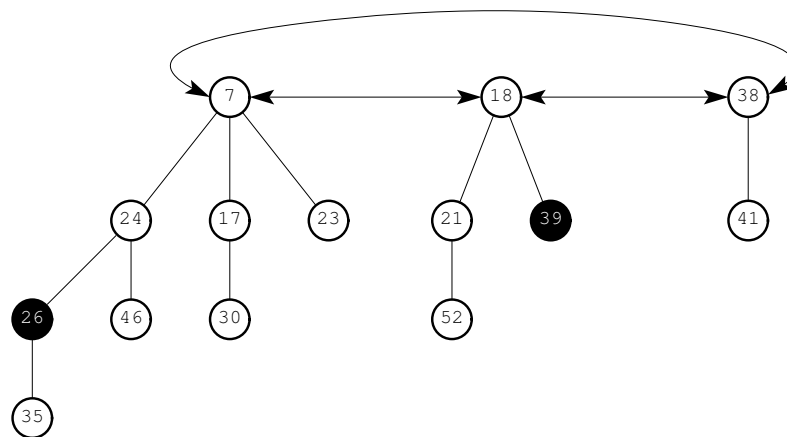
Dabei bestehe σ aus

- n_1 -mal *Insert*
- n_2 -mal *DeleteMin* und
- n_3 -mal *Minimum*

in einer beliebigen Reihenfolge. Beweisen Sie, daß σ einen Zeitaufwand von $O(n_1 + n_2 \cdot \log n_1 + n_3) = O(n \log n)$ mit $n = n_1 + n_2 + n_3$ besitzt.

5. Aufgabe: Sei F_i die Folge der Fibonaccizahlen. Also $F_i = F_{i-1} + F_{i-2}$ mit $F_0 = 0$ und $F_1 = 1$. Sei weiterhin $\varphi = \frac{1+\sqrt{5}}{2}$ der Wert des goldenen Schnittes. Zeigen Sie $F_i \geq \varphi^{i-2}$ für alle $i > 0$.

6. Aufgabe: Führen Sie $\text{DeleteMin}(h)$ auf folgendem Fibonacci-Heap h aus:



7. Aufgabe:

- (a) Geben Sie für jedes $n > 0$ eine Folge von Operationen für den Fibonacci-Heap an, so dass ein Baum der Art



mit n Knoten entsteht.

- (b) Zeigen Sie, dass die Operationen $\text{DeleteMin}(h)$, $\text{Cut}(i, h)$, $\text{DecreaseKey}(i, j, h)$ im Fibonacci-Heap h mit n Knoten eine im **worst-case**-Laufzeit von $\Theta(n)$ haben.