Prof. Dr. A. Goerdt
Dr. L. Falke

TU Chemnitz
Sommersemester 2017
18.04.2017

Effiziente Algorithmen / Theoretische Informatik III

2. Übung

1. Aufgabe: Wir betrachten die *Binomialbäume* aus der Vorlesung.

(a) Welche Einschränkung hat ein Binomialbaum, der die *Heapeigenschaft* bezüglich der Schlüsselwerte erfüllt, gegenüber einem üblichen Heap?

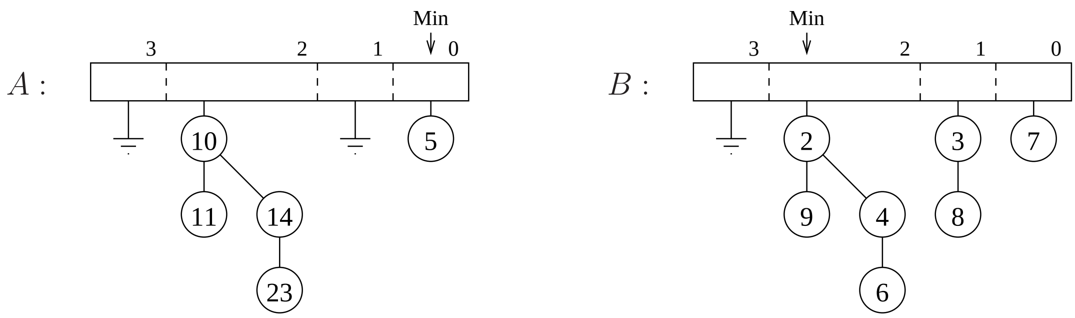
(b) Wie läßt sich auf Basis dieser Bäume eine Datenstruktur aufbauen, die *heapähnliche* Eigenschaften aufweist?

(c) Wie sind die *üblichen Heapoperationen* zu implementieren? Welche Laufzeiten ergeben sich?

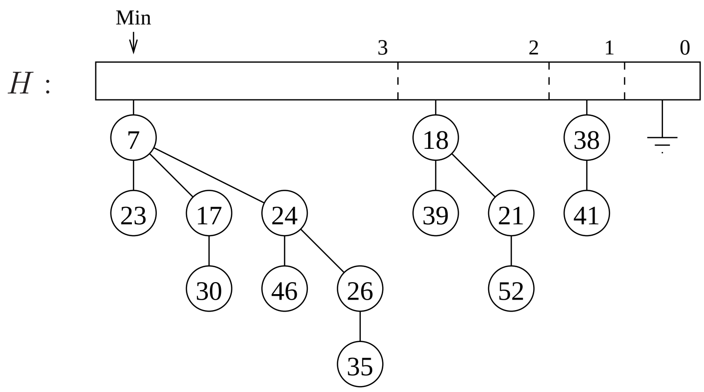
(d) Überlegen Sie, welche Operation sich im *binomialen Heap* zusätzlich mit einer *günstigen* Laufzeit realisieren läßt!

2. Aufgabe: Wir betrachten *binomiale Heaps*, wie in Aufgabe 1 definiert.

(a) Wir betrachten die folgenden zwei *binomialen Heaps* A und B. Führen Sie die Operation `meld(`A`,`B`)` aus.



(b) Führen Sie `DeleteMin(`H`)` auf folgendem binomialen Heap H aus:



3. Aufgabe: Wir betrachten eine Folge von Operationen auf Zahlen in *Binärdarstellung*. Die Laufzeit wird in der Anzahl der „verarbeiteten" Bits gezählt.

Beginnend bei der Zahl Null (0) wird n-mal hintereinander eine Eins (1) addiert.

(a) Was ist die *worst-case* Laufzeit einer einzelnen solchen Addition?

(b) Welche Laufzeit ergibt sich für die gesamte Folge von Operationen?

4. Aufgabe: Wir betrachten eine Folge $\sigma = \{\sigma_1, \sigma_2, \cdots, \sigma_n\}$ 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.