Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
H. Ghodselahi, Y. Maus
November 26, 2014

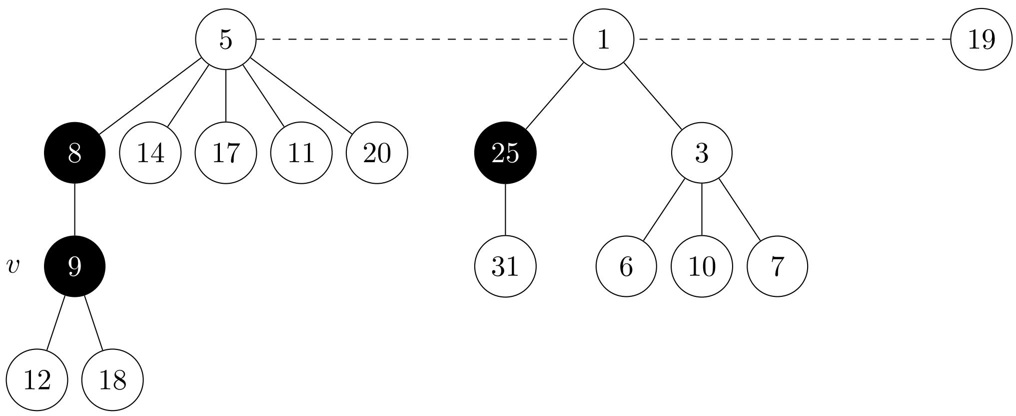
Algorithm Theory, Winter Term 2014/15
Problem Set 6

hand in (hard copied) by Thursday, 10:00, December 04, 2014, either before the lecture or in the box corresponding to your group in building no. 51.

Exercise 1: Fibonacci Heaps (0.5+2.5+3 points)

(a) Explain why the operations *Insert* and *Decrease-Key* on a Fibonacci Heap are so called *lazy* operations.

(b) Consider the following Fibonacci heap (black nodes are marked, white nodes are unmarked). How does the given Fibonacci heap look after a *decrease-key*$(v, 2)$ operation and how does it look after a subsequent *delete-min* operation?



(c) Fibonacci heaps are only efficient in an amortized sense. The time to execute a single, individual operation can be large. Show that in the worst case, both the *delete-min* and the *decrease-key* operations can require time $\Omega(n)$ (for any heap size n).

Hint: *Describe an execution in which there is a delete-min operation that requires linear time and describe an execution in which there is a decrease-key operation that requires linear time.*

Exercise 2: Amortized Analysis (4 points)

We are given a data structure $\mathcal{D}$, which supports the operations `put` and `flush`. The operation `put` stores a data item in $\mathcal{D}$ and has a running time of 1. Further, if $\mathcal{D}$ contains $k \geq 0$ items, the operation `flush` deletes $\lceil k/2 \rceil$ of the k data items stored in $\mathcal{D}$ and its running time is equal to k.
Prove that both operations have constant amortized running time by using the potential function method.