Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
M. Ahmadi, A. R. Molla, O. Saukh November 26, 2015

# Algorithm Theory, Winter Term 2015/16
# Problem Set 6

**hand in (hard copy or electronically) by 10:15, Thursday December 3, 2015, tutorial session will be on December 7, 2015**

## Exercise 1: 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.

## Exercise 2: Union-Find (4+4 points)

(a) In the lecture the union-by-size heuristic was introduced to guarantee shallow trees when implementing a Union-Find data structure. Another heuristic that can be used for `union(x,y)` is the union-by-rank heuristic. For the heuristic, the rank of a tree is defined as follows:

- The rank $r(T)$ of a tree $T$ consisting of only one node is 0.
- When joining trees $T_1$ and $T_2$ by attaching the root of tree $T_2$ as a new child of the root of tree $T_1$, the rank of the new combined tree $T$ is defined as $r(T) := \max\{r(T_1), r(T_2) + 1\}$.

When applying the union-by-rank heuristic, whenever combining two trees into one tree (as the result of a union operation), we attach the tree of smaller rank to the tree of larger rank (if both trees have the same rank, it does not matter which tree is attached to the other tree). Provide pseudo-code for the `union(x,y)` operation when using the union-by-rank heuristic.

Show that when implementing a Union-Find data structure by using disjoint-set forests with the union-by-rank heuristic, the height of each tree is at most $O(\log n)$.

(b) Demonstrate that the above analysis is tight by giving an example execution (of merging $n$ elements in that data structure) that creates a tree of height $\Theta(\log n)$. Can you even get a tree of height $\lfloor \log_2 n \rfloor$?