Marburg University

Department of Computer Science
Prof. Dr. Sebastian Wild

Date: 2026-05-08
Version: 2026-05-08 22:47

# Exercise Sheet 3 for Advanced Data Structures (Summer 2026)

***Hand In:*** *Until 2026-05-17 18:00, on ILIAS.*

## Problem 1

30 points

Suppose we fix $n$ distinct keys and are given a sequence $m$ of accesses to these keys, namely $a_1, \ldots, a_m$. We want to find the optimal way of maintaining these keys as a binary search tree. More precisely, we represent the keys as a binary search tree together with a *cursor* which is originally at the root. The values are in some arbitrary binary search tree originally. At cost 1, we may move the cursor to a neighbour, rotate the tree along any edge adjacent to the cursor, or "find" the currently sought element, resetting the cursor to the root. The goal is to apply these operations such that the cursor finds, in order, the nodes for keys $a_1, \ldots, a_m$, while minimising the total cost of all the operations.

Give a *fixed-parameter tractable* solution for this problem. More precisely, give a solution that computes the optimal operation sequence (offline) in time $f(n) \times m^c$ for some fixed function $f$ and constant $c$, chosen by you.

## Problem 2

50 points

Consider the adaptive linked-list scenario (as for the move-to-front rule) for accessing keys in a linked list.

Assume that at cost $k$, we can inspect and *arbitrarily permute* the first $k$ items in the list. As before, each access to an item must find it, so if the current access $a_i$ is the $k$th element in the list, we must pay at least cost $k$ (but may opt to pay more, to read and permute a longer prefix of the list).

Show that in this model, *no* online algorithm can be $O(1)$ competitive.