Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Harald Räcke
Chris Pinkau

Winter Semester 2013/14
Problem Set 3
November 5th, 2013

Parallel Algorithms

Due date: November 12th, 2013 **before** *class!*

Let $A = (a_1, \ldots, a_n)$ be an array whose elements are drawn from a linearly ordered set.

Problem 1 (10 Points)

The *left match* of $a_i, i \in \{1, \ldots, n\}$, is the element a_k (if it exists) such that k is the maximum index satisfying $k \in \{1, \ldots, i-1\}$ and $a_k < a_i$. Similarly, we can define the right match of a_i. The problem of finding the left and right matches of all the elements in A is called the problem of *all nearest smaller values* (ANSV).
Show how to solve the ANSV problem in $\mathcal{O}(1)$ time using $\mathcal{O}(n^2)$ operations.
Hint: Use Problem 4 from Problem Set 1.

Problem 2 (20 Points)

The *suffix-minima problem* is to compute for each $i \in \{1, \ldots, n\}$, the minimum element among $\{a_i, a_{i+1}, \ldots, a_n\}$. We can define the *prefix minima* in a similar way.

1. Design an $\mathcal{O}(1)$ time algorithm for computing the prefix and suffix minima of A, using a total of $\mathcal{O}(n^2)$ operations.
2. Use a $\sqrt{n}$ divide-and-conquer strategy to obtain an $\mathcal{O}(\log \log n)$ time algorithm. The total number of operations used must be $\mathcal{O}(n)$. Specify the PRAM model needed.

Problem 3 (10 Points)

1. Using an $\mathcal{O}(\log \log n)$ algorithm to compute the prefix (or suffix) minima of A, design an $\mathcal{O}(\log \log n)$ time algorithm for the range-minima problem using $\mathcal{O}(n \log n)$ operations.
2. Divide the array into subarrays to make this algorithm optimal, i.e. only $\mathcal{O}(n)$ operations must be used.