Triply-Logarithmic Upper and Lower Bounds for Minimum, Range Minima, and Related Problems with Integer Inputs

O. Berkman\textsuperscript{1} Y. Matias\textsuperscript{2} P. Ragde\textsuperscript{3}

\textsuperscript{1} King's College London\textsuperscript{\dagger}
\textsuperscript{2} AT&T Bell Laboratories\textsuperscript{\ddagger}
\textsuperscript{3} University of Waterloo\textsuperscript{\S}

Abstract. We consider the problem of computing the minimum of $n$ values, and several well-known generalizations (prefix minima, range minima, and all-nearest-smaller-values (ANSV) problems) for input elements drawn from the integer domain $[1..s]$ where $s \geq n$. Recent work \cite{4} has shown that parallel algorithms that are sensitive to the size of the input domain can improve on more general parallel algorithms. The cited paper demonstrates an $O(\log \log \log s)$-step algorithm on an $n$-processor PRIORITY CRCW PRAM for finding the prefix-minima of $n$ numbers in the range $[1..s]$. The best known upper bounds for the range minima and ANSV problems were previously $O(\log \log n)$ (using algorithms for general input). This was also the best known upper bound for computing prefix minima or even just the minimum on the COMMON CRCW PRAM; this model has a $\Theta(\log n/\log \log n)$ time separation from the stronger PRIORITY model when using the same number of processors. In this paper we give simple and efficient algorithms for all of the above problems. These algorithms all take $O(\log \log \log s)$ time using an optimal number of processors and $O(ns^{2})$ space on the COMMON CRCW PRAM. We also prove a lower bound demonstrating that no algorithm is asymptotically faster as a function of $s$, by showing that for $s = 2^{2^{2^{\Theta(\log n \log \log n)}}}$ the upper bounds are tight.

1 Introduction

Let $A = (a_1, \ldots, a_n)$ be an array of input elements. Denote by $MIN(i, j)$ the minimum over $a_i, \ldots, a_j$. We consider the following problems:

- The minimum problem: find $MIN(1, n)$.
- The prefix minima problem: find $MIN(1, i)$ for all $i, 1 \leq i \leq n$.

\textsuperscript{\dagger} Dept. of Computer Science, King's College London, The Strand, London WC2R 2LS, England.
\textsuperscript{\ddagger} AT&T Bell Laboratories 600 Mountain Ave., Murray Hill, NJ 07974-0636, USA.
\textsuperscript{\S} Dept. of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1.
- The **range minima** problem: build a data structure that will permit a constant-time answer to any query \( \text{MIN}(i, j) \) for any \( 1 \leq i < j \leq n \).
- The **all nearest smaller values (ANSV)** problem: find for all \( i, 1 < i < n \), the maximum \( j, j < i \), such that \( a_j < a_i \) (the "left match" of \( a_i \)) and the minimum \( k, k > i \), such that \( a_k < a_i \) (the "right match" of \( a_i \)).

In this paper we consider the above problems when the elements of \( A \) are drawn from the integer domain \([1..s]\) where \( s \geq n \). We show:

**Theorem 1.** Each of the above problems can be solved on the **COMMON** CRCW PRAM in \( O(\log \log \log s) \) time using \( n/\log \log \log s \) processors and \( O(ns^\epsilon) \) space.

**Theorem 2.** Any \( n \)-processor **PRIORITY** CRCW PRAM algorithm for computing the minimum (and thus any algorithm for the other three problems) takes \( \Omega(\log \log \log s) \) time for any \( s, s > 2^{2\sqrt{\log \log n}} \).

### 1.1 The model of computation

The model of parallel computation used in this paper is the concurrent-read concurrent-write (CRCW) parallel random access machine (PRAM). The CRCW PRAM model employs synchronous processors, all having access to a shared memory with allowed concurrent access. There are several sub-models of CRCW PRAM regarding the conflict resolution rule in case of a concurrent writing. In the **COMMON** model, several processors may attempt to write simultaneously at the same location only if they write the same value; **COMMON** thus forbids write conflicts. Following [13, 20], Boppana [7] gave a lower bound of \( \Omega(n/\log \log n) \) for computing the **Element Distinctness** problem on an \( n \)-processor **COMMON**. This problem can be solved in constant time on models that allow write conflicts. Such models include: (i) **TOLERANT**, where if two or more processors attempt to write to the same cell in a given step then the contents of that cell does not change; (ii) the stronger **ARBITRARY**, in which a concurrent writing results in an arbitrary winner among the writing processors; and (iii) the yet stronger **PRIORITY** in which a write conflict is resolved by having the processor with highest priority succeed. Algorithms running on **PRIORITY** or **ARBITRARY** might not be transferable to **COMMON** without a significant slowdown or loss of efficiency.

A parallel algorithm is said to be **optimal** if its time-processor product is (asymptotically) equal to the lower bound on the time complexity of any sequential algorithm for the problem. A primary goal in parallel computation is to design optimal algorithms that also run as fast as possible.

### 1.2 Related Work

We review below previous and related results for the four problems considered in this paper.

**Sequential algorithms** Gabow, Bentley, and Tarjan [14] gave a linear-time preprocessing algorithm for range minima that results in constant-time query retrieval. The ANSV problem has a simple linear-time algorithm using a stack.