

# CSE 361 Complexity of Sequential and Parallel Algorithms

## Exam III; Help Sheet

1. **ALGEBRAIC PROBLEMS.** A degree- $n$  polynomial can be evaluated at a given point in  $O(n)$  time. Lagrangian interpolation algorithm runs in  $O(n^3)$  time whereas Newton's interpolation algorithm takes  $O(n^2)$  time.

Two degree- $n$  polynomials can be multiplied in  $O(n \log n)$  time. A degree- $n$  polynomial can be evaluated at  $n$  given arbitrary points in  $O(n \log^2 n)$  time. Also, interpolation of a polynomial presented in value form at  $n$  arbitrary points can be done in  $O(n \log^3 n)$  time.

2. **INTRACTABLE PROBLEMS.** A problem  $\pi$  is said to be  $\mathcal{NP}$ -hard if  $\pi' \leq \pi$  for every  $\pi' \in \mathcal{NP}$ . Equivalently, a problem  $\pi$  is  $\mathcal{NP}$ -hard if  $\pi' \leq \pi$  where  $\pi'$  is known to be  $\mathcal{NP}$ -hard. A problem  $\pi$  is  $\mathcal{NP}$ -complete if  $\pi$  is in  $\mathcal{NP}$  and  $\pi$  is  $\mathcal{NP}$ -hard.

We showed that the following problems are  $\mathcal{NP}$ -complete: SAT, Clique, NodeCover, 3SAT, Chromatic Number Decision Problem, Subset Sum, and Partition.

3. **PARALLEL ALGORITHMS.** The model we used was the PRAM (Parallel Random Access Machine). Processors communicate by writing into and reading from memory cells that are accessible to all. Depending on how read and write conflicts are resolved, there are variants of the PRAM. In an Exclusive Read Exclusive Write (EREW) PRAM, no concurrent reads or concurrent writes are permitted. In a Concurrent Read Exclusive Write (CREW) PRAM, concurrent reads are permitted but concurrent writes are prohibited. In a Concurrent Read Concurrent Write (CRCW) PRAM both concurrent reads and concurrent writes are allowed. Concurrent writes can be resolved in many ways. In a Common CRCW PRAM, concurrent writes are allowed only if the conflicting processors have the same message to write (into the same cell at the same time). In an Arbitrary CRCW PRAM, an arbitrary processor gets to write in cases of conflicts. In a Priority CRCW PRAM, write conflicts are resolved on the basis of priorities (assigned to the processors at the beginning).

We presented a Common CRCW PRAM algorithm for finding the Boolean AND of  $n$  given bits in  $O(1)$  time. We used  $n$  processors. As a corollary we gave an algorithm for finding the minimum (or maximum) of  $n$  given numbers in  $O(1)$  time using  $n^2$  Common CRCW PRAM processors. The following results were also proven: The maximum of  $n$  arbitrary numbers can be found in 1)  $\tilde{O}(1)$  time using  $n$  CRCW PRAM processors; 2)  $O(\log \log n)$  time using  $\frac{n}{\log \log n}$  CRCW PRAM processors. Also, the maximum of  $n$  integers in the range  $[1, n^{O(1)}]$  can be found in  $O(1)$  time using  $n$  CRCW PRAM processors.

We also discussed a CREW PRAM algorithm for the prefix computation problem. This algorithm uses  $n$  processors and runs in  $O(\log n)$  time on any input of  $n$  elements. (For the prefix computation problem the input is a sequence of elements from some domain  $\Sigma$ :  $k_1, k_2, \dots, k_n$  and the output is another sequence:  $k_1, k_1 \oplus k_2, \dots, k_1 \oplus k_2 \oplus k_3 \oplus \dots \oplus k_n$ , where  $\oplus$  is any binary associative and unit-time computable operation on  $\Sigma$ .)

We also proved the following theorems: 1) Prefix computation on  $n$  elements can be done using  $\frac{n}{\log n}$  EREW PRAM processors in  $O(\log n)$  time; 2) If a parallel algorithm runs in time  $T$  on a  $P$ -processor PRAM, it can be simulated on a  $P'$ -processor PRAM in time  $O(PT/P')$  as long as  $P' \leq P$ ; and 3) We can sort  $n$  given numbers in  $\tilde{O}(\log n)$  time given  $n$  CRCW PRAM processors.