EE 382V Parallel Algorithms Vijay Garg Summer 2020

Assignment 1
Due: June 16
Instructor: Professor Vijay K. Garg (email: garg@ece.utexas.edu)

The goal of this assignment is to learn basic techniques in parallel algorithms. For this assignment, you can work in groups of two.

The assignment must be submitted on Canvas by as a zip file with the name **eid1_eid2.zip**, which must include a pdf with the answers to problems 1-4, as well as the code for problems 5,6.

Code instructions: Starter code and coding instructions can be found in the class github website: `https://github.com/vijaygarg1/sum20-Parallel-algs/assignment/hw1`

1. **(10 points)** Suppose that an array does not have all elements that are distinct. Show how you can use any algorithm that assumes distinct elements for computing the maximum to solve the problem when elements are not distinct.

2. **(10 points)** Give a parallel algorithm on a CREW PRAM to determine the largest odd number in a given array of positive integers. Assume that the array has size n and the number of processors available is also n. The size n may not be a power of 2. Your algorithm should not take more than $O(\log n)$ time.

3. **(15 points)** Give a parallel algorithm on a CREW PRAM with time complexity $O(\log n)$ and work complexity $O(n)$ to compute the inclusive parallel prefix sum of an array of size n by combining two algorithms: sequential prefix sum that takes $O(n)$ time and $O(n)$ work and a non-optimal parallel prefix algorithm that takes $O(\log n)$ time and $O(n \log n)$ work.

4. **(15 points)** Given an integer array A and two numbers x and y, give a parallel algorithm on a CREW PRAM to compute an array D such that D consists only of entries in A that are greater than or equal to x and less than or equal to y. The order of entries in D should be same as that in A.

5. **(20 points)** This is a programming assignment. Give a parallel program that takes as input a real vector x and returns its Euclidean length.

6. **(30 points)** This is a programming assignment. Give a parallel program that takes as input a sorted array A of integers of size n and returns another sorted array B such that B has no duplicates. You are allowed to use an additional $O(n)$ space in your algorithm.