Parallel Computing
COMP 633  Fall 2019

Written Assignment #1

Assigned:  Tue Aug 27
Due:  Tue Sep 10

Instructions:  work together with one other student in the class and turn in a single paper with both names.  If you prefer, you may work on your own.

I.  [6] The Work-Time (W-T) presentation of EREW sequence reduction (Algorithm 2 in PRAM handout) has work complexity $W(n) = O(n)$ and step complexity $S(n) = O(lg n)$.  Following the strategy of Brent’s theorem, the translation of this algorithm will yield a $p$ processor EREW PRAM program with running time

$$T(n, p) = O(n/p + lg n)$$

(a) Construct an alternate sequence reduction algorithm directly for the bare bones EREW PRAM with running time $T(n, p) = O(n/p + lg p)$.

(b) Explain why your solution to (a) cannot be expressed in the W-T model.

II.  [10] Let $A$ and $B$ be sets of integers with $|A| = m \leq n = |B|$. The elements of the sets are stored in increasing order in arrays $A[1..m]$ and $B[1..n]$, respectively (since $A$ and $B$ are sets, there are no duplicate elements in either of these arrays). Using this representation, construct a CREW W-T algorithm that determines whether $A \subseteq B$ in $O(lg n)$ steps and $O(n)$ work.

III.  [10] Choose one of the following two problems:

(a) Suppose we have a sequence $X[1..n]$ with values drawn from some large set $S$ so that $n = o(|S|)$, i.e. $S$ has asymptotically more than $n$ values. We want to determine whether any value occurs in $X$ more than once. Using hashing we can construct a sequential algorithm for this problem with expected sequential time $O(n)$ using $O(n)$ space. Construct a CRCW W-T algorithm using the arbitrary write-collision model for this problem with expected work complexity $O(n)$ and expected step complexity as small as you can get it using $O(n)$ space. Your complexity argument can be informal.
(b) Given a sequence $s[1..n]$, the maximum contiguous subsequence sum (mcss) of $s$ is the largest sum that can be formed from any contiguous subsequence of $s$ (including the empty subsequence, with sum zero), i.e.

$$mcss = \max_{1 \leq i \leq j \leq n} \left( \sum_{k=i}^{j} s_k \right)$$

When all elements of $s$ are positive the mcss is the sum of all elements in $s$. When all elements are negative the mcss is zero, corresponding to the sum of an empty subsequence. Here is an optimal sequential algorithm for this problem:

```plaintext
integer MCSS(sequence<integer> s)
    MaxSoFar, MaxEndingHere ← 0, 0
    for i = 1 to n do
        MaxEndingHere ← max(MaxEndingHere + s[i], 0)
        MaxSoFar ← max(MaxSoFar, MaxEndingHere)
    enddo
    return MaxSoFar
```

Design a work-efficient EREW algorithm in the Work-Time framework with step complexity $\Theta(\lg n)$ for this problem.