Fundamental Algorithms 9

Exercise 1 (Parallel sort)

BucketSortPRAM is a proposed parallel implementation of the BucketSort algorithm in a PRAM, $n$ processor model. copy is a sequential algorithm which takes an array $A$ and an array of arrays $B$ as arguments and copies the elements of $B$ into $A$ in order. The function $\text{index}$ distributes the array elements evenly into buckets, i.e. elements are assigned to buckets with (roughly) the same probability. Moreover, larger elements are assigned to larger buckets, i.e. $\text{index}(a) < \text{index}(b)$ implies $a < b$.

Algorithm 1: BucketSortPRAM

Input: $A$: Array[1..n]
Result: $A$ is sorted
$B \leftarrow $ Array[1..nb];
for $i = 1$ to $n$ in parallel do $\text{insert}(B[\text{index}(A[i])], A[i])$;
for $i = 1$ to $nb$ in parallel do $\text{BubbleSort}(B[i])$;
$\text{copy}(A, B)$;

1. For the two parallel loops in BucketSortPRAM, state for both arrays $A$ and $B$ whether there is concurrent or exclusive read / write access to their elements.

2. Implement CopyPRAM, i.e. a parallel version of copy You can use a constant-time $\text{len}$ function to determine the length of an array. What is the parallel complexity (depending on $nb$), and how many processors can your algorithm use?