Problem 18

(4 points) Design a parallel algorithm that finds the maximum number in a sequence x1, . . . , xn of (not necessarily distinct) integers in the range 1 to n. Your algorithm should run in constant time on a CRCW Priority PRAM with n processors. Note that it is important here that the xis have restricted range. In a CRCW priority PRAM, each processor has a unique positive integer identifier, and in the case of write conflicts, the value written is the value that the processor with the lowest identifier is trying to write.

We will solve this problem using n processors in constant time by exploiting the fact that our input is bounded by n. Assign each processor to an item in the input, and have the processor read the value v=I[i] from this input. Now this processor will write its value v to a sorted memory location S[v] = v. Note that if conflicts occur, then a processor will be chosen according to the priority; however, this does not effect our result since any conflicts that occur on a given memory location will be attempting to write the same value.

Now we have a sorted list of the input, with gaps in it if a given value was not in the input. One could simply start at the last value in the list keep decrementing and checking last value until it is not null; however, in the worst case this would take linear time. Instead, we assign priorities to our processors in reverse order and parallelize the processes. "Reverse order" in this case means that P_{n-v} writes the 'v^{th}' item in the sorted list list. The result of this is that the n^{th} processor is written by P_0, and thus has the highest priority. All of these processors then write to a single memory location according to MAX = max(MAX, v), and if conflicts arise the higher value is taken according to priority.