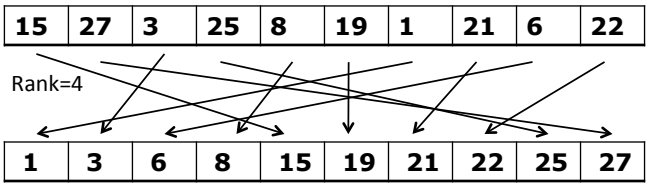
Enumeration Sort

Algorithm:

For each element compare how many elements are smaller => Rank or sorting order of this element. Put the elements in a new array according to the ranks.

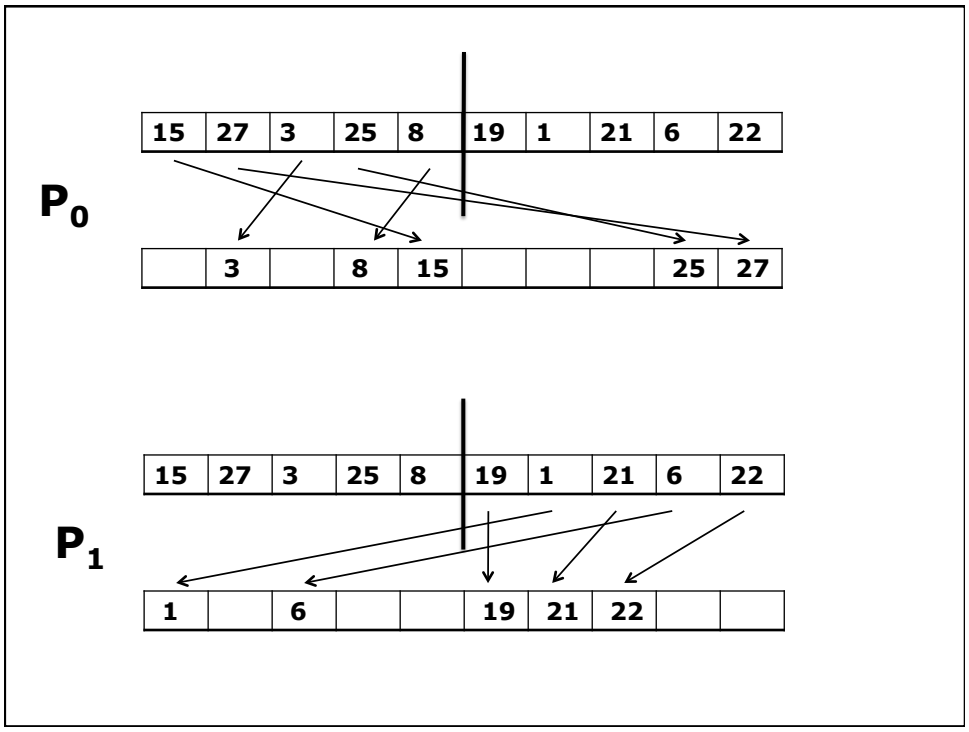


Parallelism:

Computing the rank for each element is a perfectly parallel operation but to do this we need to compare each element to all other elements.

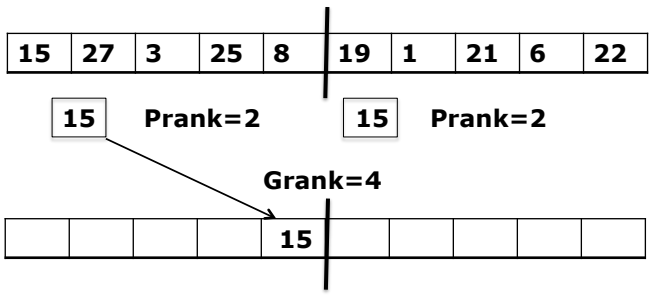
Alternative 1:

Copy all data to all processor but divide the work, i.e., let each processor compute the ranks for a subset of the elements. Merge (reduce) results then.



Alternative 2:

Split indata/outdata arrays and compute the rank for each element at a time, i.e., compute partial ranks on each processor in parallel and do a reduction of the rank.



Alternative 3:

Reformulate algorithm. Save ranks in a new array and compute outdata in the end using the rank-array, i.e., outdata[rank[i]]=indata[i];

In parallel, split indata and rank-arrays and compute partial ranks in parallel with *compute-and-shift*. Use an extra copy of the indata-array which is shifted p-times. In each step compute the partial ranks for each element and add to the rank-array.