

# Sample sort

9/28/07

## Bucket Sort

- In a bucket sort, we divide the range of the input numbers into equal sized intervals called buckets.
- If the numbers are uniformly distributed, each bucket can be expected to have roughly identical number of elements placed in them.
- Elements in the buckets are locally sorted.

## Parallel Bucket Sort

- Parallelizing bucket sort is relatively simple. We can select  $p$  intervals.
- A given processor is responsible for a range of values and each individual processor runs through its local list to assign each of its elements to the corresponding processor.
- The elements are sent to the destination processors using a single all-to-all personalized communication.
- Each then processor sorts all of its elements.

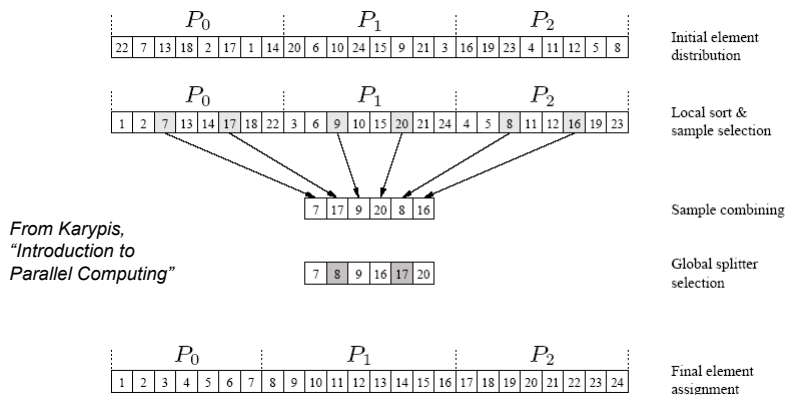
## Picking splitters

- The most important aspect of a bucket sort assigning appropriate numeric ranges to processors.
- Ideally, the  $n$  elements should be divided into  $p$  blocks of size  $O(n/p)$  and sorted using our favorite serial algorithm (e.g., quicksort).
- To do so, we choose  $p - 1$  evenly spaced elements per processor.

# Splitters

- The  $p(p - 1)$  elements selected from all the blocks represent the sample used to determine the buckets.
- This scheme guarantees that the number of elements ending up in each bucket is less than  $2n/m$ .

# Parallel Sample Sort



Sample sort performed on an array with 24 elements using three processors.

## Parallel Bucket and Sample Sort

- The splitter selection scheme can itself be parallelized.
- Each processor generates the  $p - 1$  local splitters in parallel.
- All processors share their splitters using a single all-to-all broadcast operation.
- Each processor sorts the  $p(p - 1)$  elements it receives and selects  $p - 1$  uniformly spaced splitters from them.

## Analysis

- Sorting  $n/p$  elements requires time  $\Theta((n/p)\log(n/p))$
- The selection of the  $p - 1$  sample requires time  $\Theta(p)$  and the time for an all-to-all broadcast is  $\Theta(p^2)$
- The time required to sort the  $p(p - 1)$  sample elements is  $\Theta(p^2 \log p)$
- Each process can *insert* these  $p - 1$  splitters in its local sorted block of size  $n/p$  by performing  $p - 1$  binary searches in time  $\Theta(p \log(n/p))$ .

## Parallel Sample Sort

- The total time is given by:

$$T_P = \overbrace{\Theta\left(\frac{n}{p} \log \frac{n}{p}\right)}^{\text{local sort}} + \overbrace{\Theta(p^2 \log p)}^{\text{sort sample}} + \overbrace{\Theta\left(p \log \frac{n}{p}\right)}^{\text{block partition}} + \overbrace{\Theta(n/p)}^{\text{communication}}. \quad (5)$$