Parallel Sorting Algorithms

KLA and Traditional
Parallel Sorting

• Shear Sort
• Radix Sort In Parallel
Shear Sort

• A very easy parallel algorithm for sorting two dimensional arrays. It sorts the rows and columns of the array in turn
• Input: Unsorted \( n \times m \) array
• Output: Sorted \( n \times m \) array where the data is sorted in a snake like order
KLA Shearsort

• $n \times n$ people to represent and hold the data
• $n$ people to be the processors
• Begin by sorting row-wise. Rows are alternatingly sorted in ascending and descending order
• Next step is to sort column-wise in descending order from the top of the mesh
• Repeat these two steps until no changes are made on a step
Initial state

After phase 1

After phase 2

After phase 3

After phase 4

After Phase 5: done
Shear sort complexity

• Complexity: $O(n \log n)$
• Work: $n^2 \log n$
• Nearly work optimal
Radix sort

- Radix sort is a non-comparative sorting algorithm that sorts integers by processing individual digits and grouping numbers by its digits. These groups are further sorted by the next significant digit.
- MSD Radix sort is parallelizable by handing each further group, known as buckets, to a free processor.
RADIX SORT WITH KLA

13 Processors
Each Processors has 3 Buckets

Each Level Places The Data into its Buckets According to the Following Underlined Digit

KLA Radix sort
362  291  207  207  |  237  237  216  211
436  362  436  253  |  318  216  211  216
291  253  253  291  |  216  211  237  237
487  436  362  362  |  462  268  268  268
207  487  487  397  |  211  318  318  318
253  207  291  436  |  268  462  462  460
397  397  397  487  |  460  460  460  462

**LSD Radix Sorting:**
Sort by the last digit, then by the middle and the first one

**MSD Radix Sorting:**
Sort by the first digit, then sort each of the groups by the next digit
### LSD Radix sort

<table>
<thead>
<tr>
<th>INPUT</th>
<th>1st pass</th>
<th>2nd pass</th>
<th>3rd pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>329</td>
<td>720</td>
<td>720</td>
<td>329</td>
</tr>
<tr>
<td>457</td>
<td>355</td>
<td>329</td>
<td>355</td>
</tr>
<tr>
<td>657</td>
<td>436</td>
<td>436</td>
<td>436</td>
</tr>
<tr>
<td>839</td>
<td>457</td>
<td>839</td>
<td>457</td>
</tr>
<tr>
<td>436</td>
<td>657</td>
<td>355</td>
<td>657</td>
</tr>
<tr>
<td>720</td>
<td>329</td>
<td>457</td>
<td>720</td>
</tr>
<tr>
<td>355</td>
<td>839</td>
<td>657</td>
<td>839</td>
</tr>
<tr>
<td>INPUT</td>
<td>1st pass</td>
<td>2nd pass</td>
<td>3rd pass</td>
</tr>
<tr>
<td>-------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>329</td>
<td>329</td>
<td>329</td>
<td>329</td>
</tr>
<tr>
<td>457</td>
<td>355</td>
<td>355</td>
<td>355</td>
</tr>
<tr>
<td>453</td>
<td>457</td>
<td>436</td>
<td>436</td>
</tr>
<tr>
<td>839</td>
<td>453</td>
<td>457</td>
<td>457</td>
</tr>
<tr>
<td>436</td>
<td>436</td>
<td>453</td>
<td>453</td>
</tr>
<tr>
<td>720</td>
<td>720</td>
<td>720</td>
<td>720</td>
</tr>
<tr>
<td>355</td>
<td>839</td>
<td>839</td>
<td>83</td>
</tr>
</tbody>
</table>

MDS Radix Sort
Radix sort in parallel complexity

• **Performance of Sequential Radix Sort is** $O(kn)$
  • $k$ is the number of digits in the number
  • $n$ is the number of elements

• **Performance of demonstrated Parallel Radix Sort is:**
  
  $O(n + \frac{n}{b} + \frac{n}{b^2} + \cdots + \frac{n}{b^k})$ where $b$ is the base of the numbers and $k$ is the maximum number of digits

  • $b_2 : O(2n)$
  • $b_4 : O(\frac{4}{3}n)$
  • $b_{10} : O(\frac{10}{9}n)$

  **Work:** $(n + \frac{n}{b} + \frac{n}{b^2} + \cdots + \frac{n}{b^k}) \times P$

  $P = b^k$