Implementation of Parallel Radix Sort using MPI

CSE 633: Parallel Algorithms
Dr. Russ Miller
What is Radix Sort?

It is a non-comparison based sort, best suited for sorting Integers

Comes under stable sorting algorithm

Two types of radix sort, LSD and MSD

Uses counting sort
An Example

Let $n$ be the number of integers
If $i$ is the largest integer, let $k$ be the number of digits in $i$.

For integers from 1 to 9999, $i = 9999$ and $k = 4$

Example set of integers:

10, 5, 6, 24, 14, 3

$n = 6$, $i = 24$, $k = 2$
Example (Cont.)

Input: 10, 05, 06, 24, 14, 03

1. Sort by units place

10, 03, 24, 14, 05, 06

Note: If two numbers are same, preserve the initial order. (Stable sort)

2. Sort by tens place

03, 05, 06, 10, 14, 24
Analysis

It takes $O(n)$ time to sort by units place
It takes another $O(n)$ to sort by tens place

Total Sorting time: $O(kn)$

In the example $k = 2$, therefore running time is $O(kn)$

What if $i$ (Largest integer) is unknown?
Do an iteration over the data to find the largest Integer
Parallel Implementation

**Step 1:** If the data is initially present in a single processor, distribute it to all other processors

**Step 2:** Convert the numbers to base 2 (Binary)

In base 10, we proceeded from Least Significant Digit to Most Significant Digit

For parallel implementation we choose a group of g bits
Parallel Implementation (Step 2)

If $p = \text{Number of processors}$

Then we choose $g$ such that,

$$2^g = p$$

$$g = \log_2 p$$

For example,

if $p = 4$, then $g = 2$. We take 2 bits at a time

00, 01, 10, 11
Parallel Implementation

**Step 3:** Do an interprocess communication such that

All numbers ending in bits 00 are sent to Processor $P0$
All numbers ending in bits 01 are sent to Processor $P1$
and so on…

**Step 4:** Perform counting sort locally on these processors

**Step 5:** Calculate the global prefix-sum of the number of integers in each processor

**Step 6:** Using the index calculated in previous step put back the integers in a temporary array and this serves as input to next iteration.
Charts
Serial: Speed of Processing

Number of Integers vs Run Time

- 100
- 1000
- 10,000
- 100,000
- 1,000,000
- 10,000,000
Parallel: Speed of Processing

Number of Integers / Run Time
Thank You!