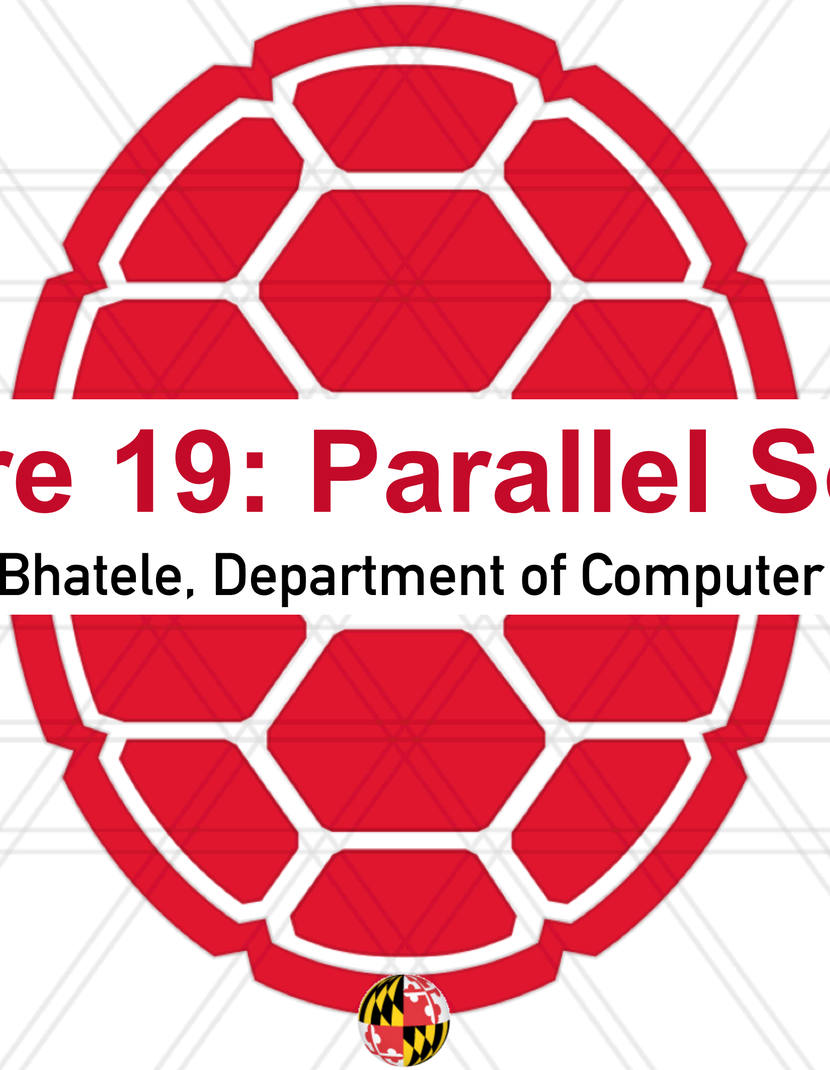
High Performance Computing Systems (CMSC714)



Lecture 19: Parallel Sorting

Abhinav Bhatele, Department of Computer Science

UNIVERSITY OF MARYLAND

Summary of last lecture

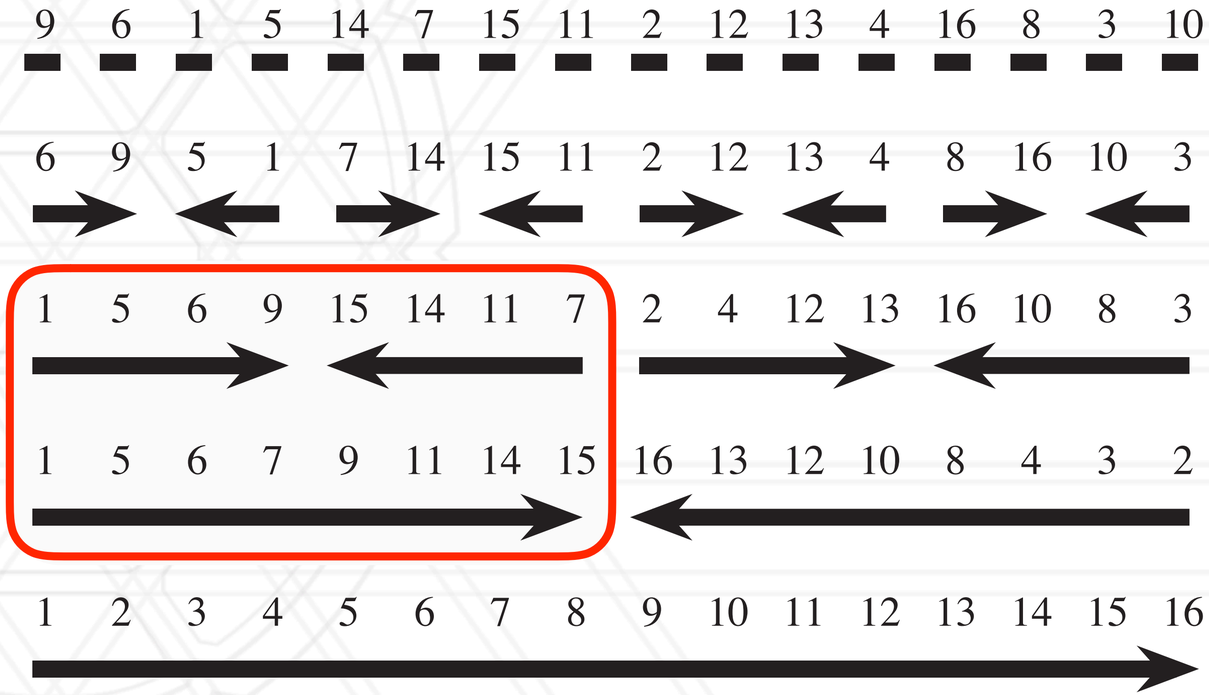
- I/O can become a bottleneck when other portions of the code scale well
  - Reading input datasets, writing numerical/scientific output, checkpointing
- Parallel file system required for high performance
- Different approaches
  - One process per file, shared file, shared files for subsets of processes
- Contention for metadata server and OSTs/disks

Parallel Sorting

- Sorting is used in many HPC codes
- For example, figuring out which particles/atoms are within a cutoff radius
- Two broad categories of parallel sorting algorithms:
  - Merge-based
  - Splitter-based

Review Bitonic Sort

- Merge-based algorithm: sort by merging bitonic sequences
- Bitonic sequence: increases monotonically then decreases monotonically
- At each step, merge a bitonic sequence



Review QuickSort

- Choose a pivot element from the unsorted list
- Move all elements < pivot before the pivot and all elements > pivot after the pivot
- Recursively apply this to the sublists before and after pivot

Parallel Sample Sort

- Instead of selecting one pivot, we select p-1 samples (if there are p processors)
  - This provides us with p-1 "splitters"
- These p-1 splitters create p buckets
- Keys are then sent to the appropriate bucket
- Why called sample sort? sample s keys randomly from each processor, sort sp keys and select p-1 splitters from this sorted sample

Parallel Radix Sort

- Instead of comparing keys, looks at k bits of each key in every step
  - k-bit radix sort looks at k bits in one step
- Move from least significant to most significant bits
- k bits leads to putting keys into 2^k buckets in a step
- Parallel version:
  - These buckets are assigned to p processes and key movement leads to all-to-all communication
  - To balance buckets across processes: use histograms to decide assignment of buckets to processes

Questions

An Improved Supercomputer Sorting Benchmark

- Can we talk about how the "plus scan" works as described in the "Scanning the Histogram" section? The paper essentially just states that it's something that happens.
- The sending of data clearly dominates the run time of the described radix sort. Is this always the case with parallel sorting?
- What does it mean to pipeline operations?
- How do GPUs fare in these sorting schemes?

Questions

A Comparison of Sorting Algorithms for the Connection Machine CM-2

- Are there commonly used libraries that implement these versions of parallel sorting algorithms? Are there other commonly used parallel sorting libraries?
- Can we go over the different sorting algorithms, the paper was confusing
- When is it important for sorting algorithms to be stable?

Questions?

UNIVERSITY OF
MARYLAND

Abhinav Bhatele

5218 Brendan Iribe Center (IRB) / College Park, MD 20742

phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu