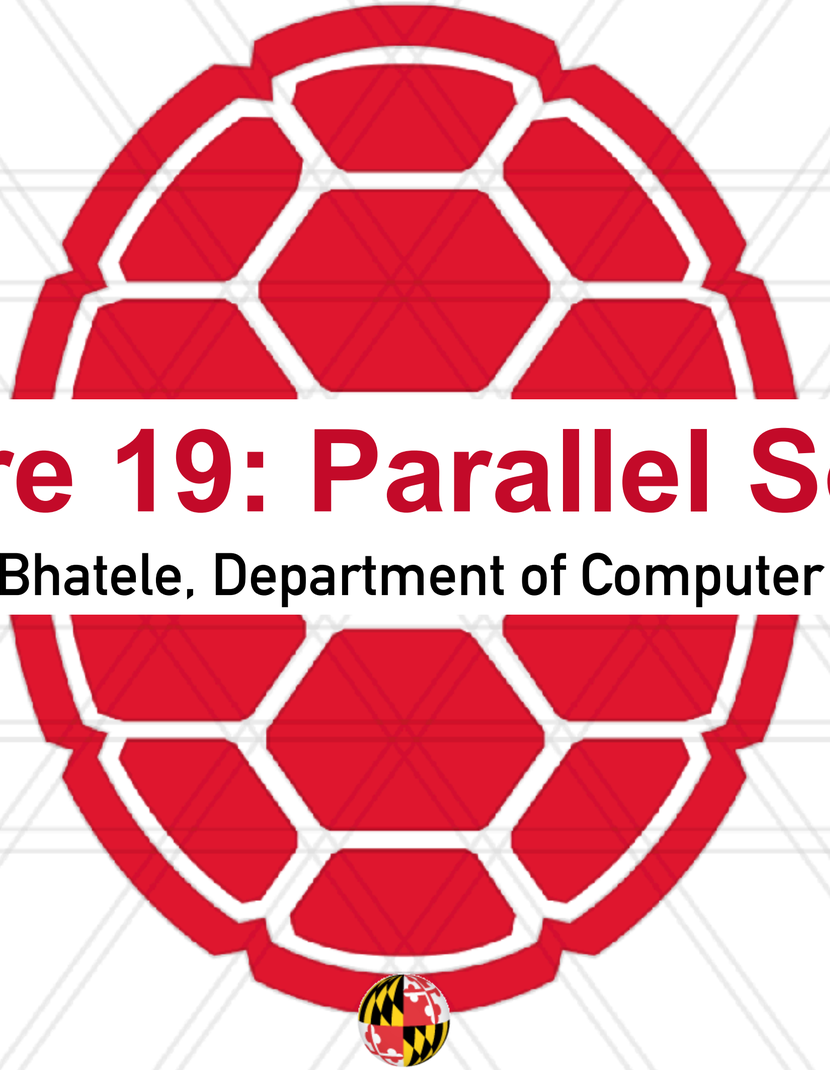
High Performance Computing Systems (CMSC714)



Lecture 19: Parallel Sorting

Abhinav Bhatele, Department of Computer Science

UNIVERSITY OF MARYLAND

Summary of last lecture

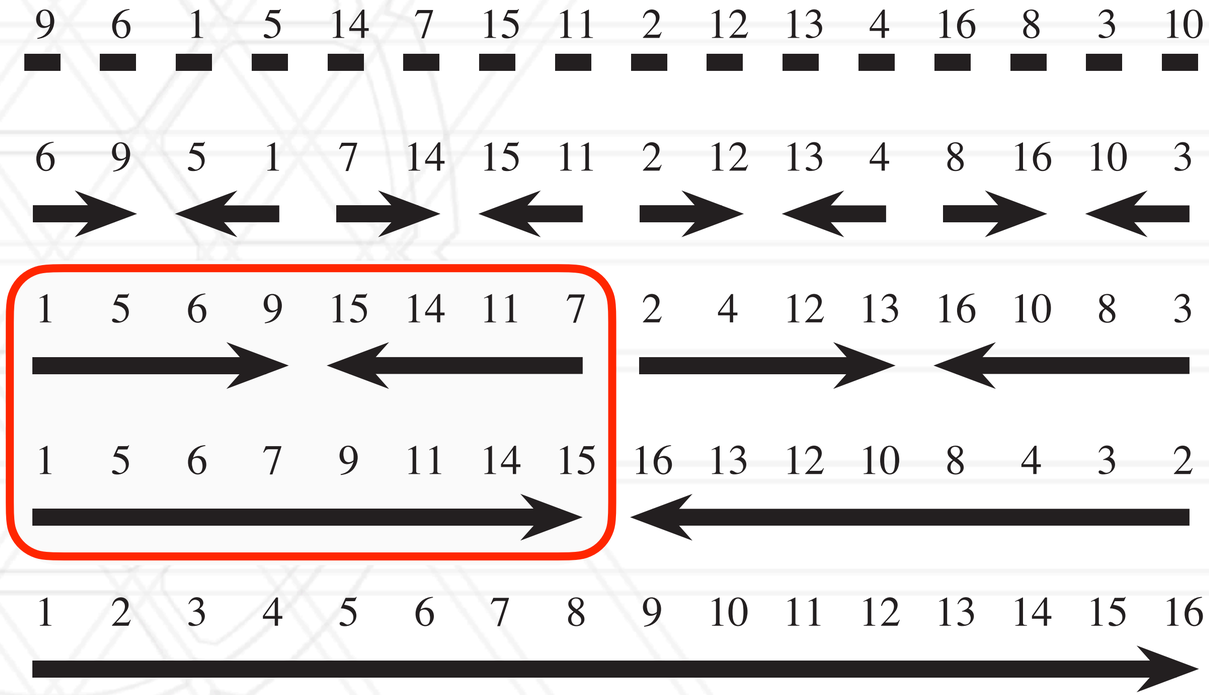
- Goal of auto-tuning: performance portability
- Selecting code variants, application/system parameters
- Model free vs. model-based
- Modeling: analytical, empirical, machine learning

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

Sample Sort

- Generalization of QuickSort
- Instead of selecting one pivot, we select s-1 samples randomly
  - This provides us with s-1 "splitters"
- Once sorted, these s-1 splitters create s buckets
- Keys are then placed in the appropriate bucket
- Call sample sort or quick sort recursively

Parallel Sample Sort

- Assumption: keys are distributed across all processors in the beginning
- Sample s keys randomly from each process
- Bring all keys s * p keys to one process
  - select p-1 splitters from this sorted sample
- Send all splitters to all processes
- Processes exchange data based on buckets
- Call some fast sorting algorithm locally

Parallel Radix Sort

- Instead of comparing keys in entirety, 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?

UNIVERSITY OF
MARYLAND

Abhinav Bhatele

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

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