Categories of Parallel Sorting Algorithms

Comparison-Based Algorithms:

- Operate similarly to traditional sort algorithms.
- Examples:
  - Parallel Merge Sort
  - Parallel QuickSort
  - Bitonic Sort
  - Sample Sort

Non-Comparison-Based Algorithms:

- Use data characteristics (e.g., value ranges, bit representations).
- Examples:
  - Parallel Radix Sort
  - Counting Sort (for integer data)
  - Bucket Sort

When to Use Which Algorithm?

Small Datasets / Few Processors: Parallel Merge Sort or QuickSort

Sorted Input / Fixed Patterns: Bitonic Sort

Numeric Input / GPU Systems: Radix Sort

Massive Datasets in Clusters: Sample Sort with MPI

Parallel Merge Sort

Divide array into equal partitions for each processor.

Each processor sorts its subarray independently (in parallel).

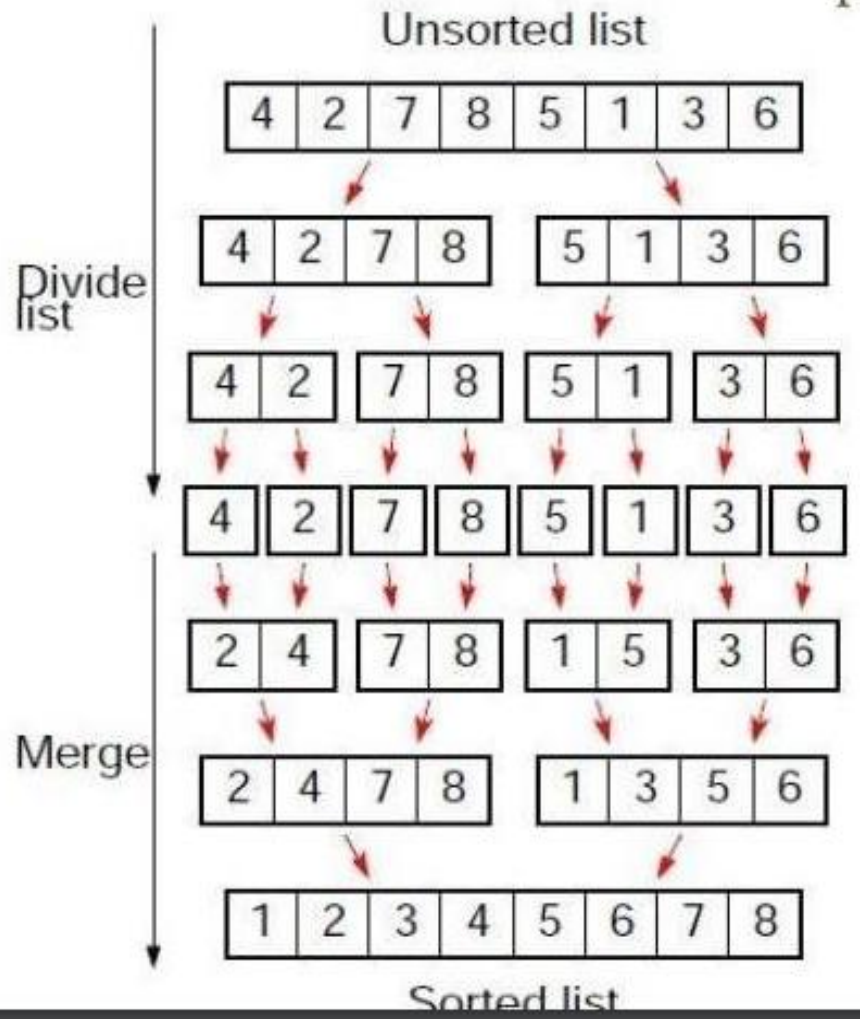
Parallel merging occurs in $\log_2(P)$ stages.

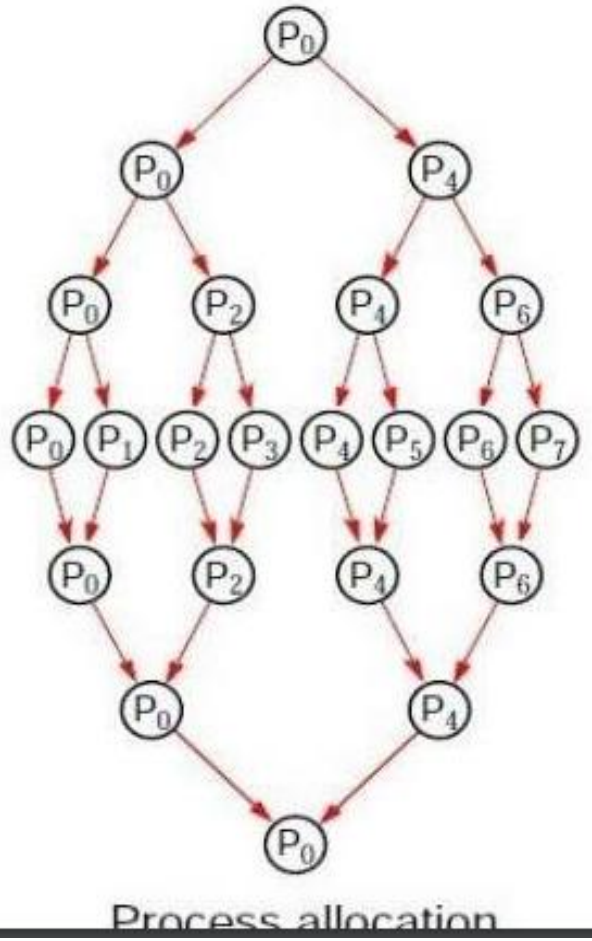
Time Complexity: O(n/P * log(n/P) + log P)

Efficient for shared-memory systems; merge phase is key bottleneck.

Parallel Merge sort:

Using a strategy to assign work to processors organized in a tree.

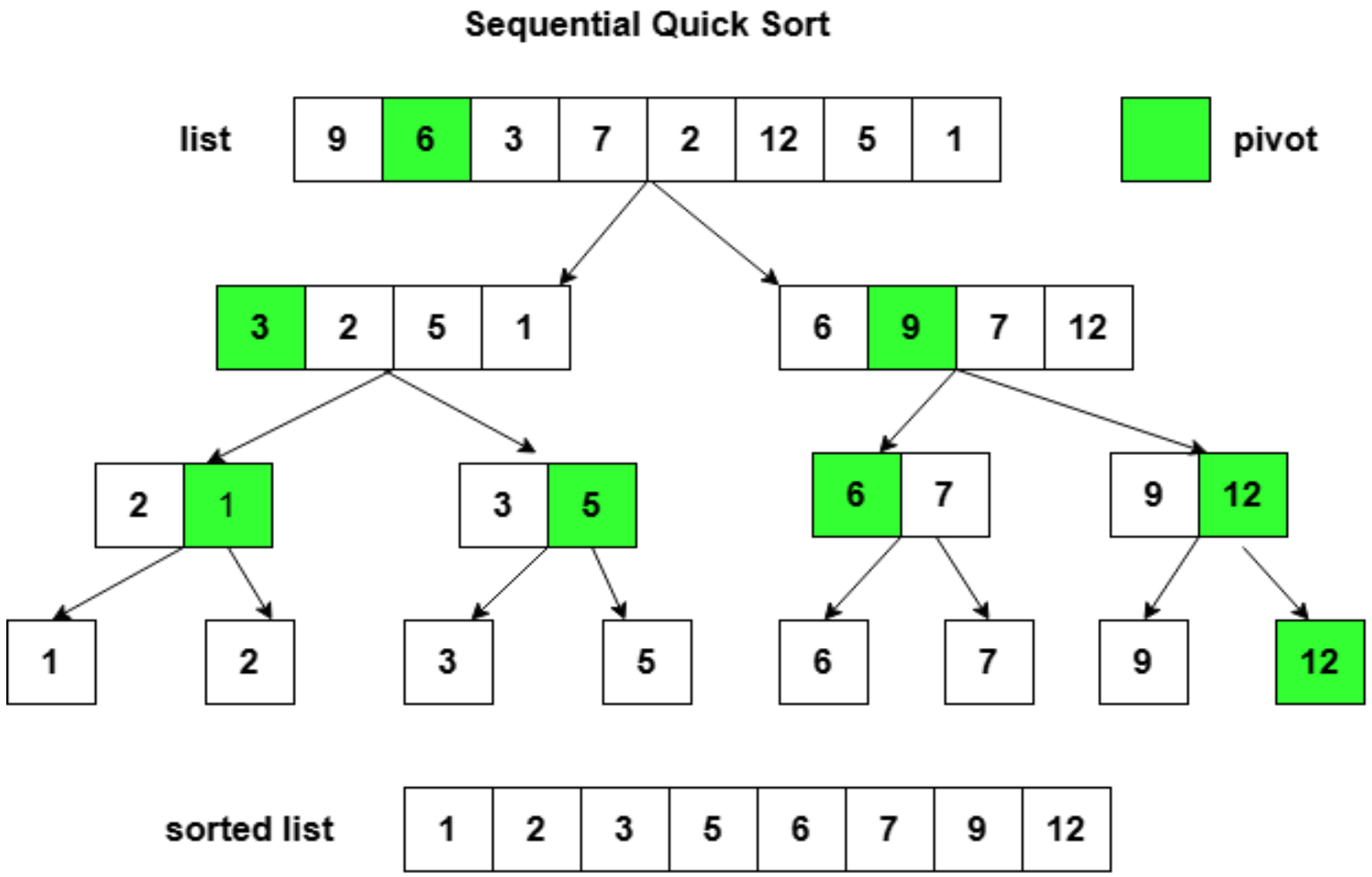




Parallel QuickSort

- Parallel pivot selection and partitioning.
- Each partition is recursively sorted in parallel.
- Highly efficient with balanced partitioning.
- Time Complexity: O(n log n / P)
- In-place sorting with minimal memory overhead.

Solved Example 1



Solved Example 2