Sample Sort

1 Quicksort

1.1 Serial Quicksort

Quicksort operates on the divide-and-conquer principle. It involves three main steps:

1. **Pivot Selection:** Choose an element from the array as the pivot. The choice of pivot affects the algorithm's efficiency but is often the first element, the last element, a random element, or the median of a set of elements from the array.

2. **Partitioning:** Rearrange the array so that all elements less than the pivot come before it, and all elements greater than the pivot come after it. After this step, the pivot is in its final position.

3. **Recursion:** Recursively apply the above steps to the sub-arrays of elements before and after the pivot, until the entire array is sorted.

1.2 Parallel Quicksort

Assume we have n values and p processors, at first the values are evenly distributed into p. Then:

1. Pick a global pivot value and then broadcast to all the processors. Recall that the broadcast runtime expression is: $O(\tau \log p + \mu m \log p)$. Now the data size is 1, so the runtime is $O(\log p(\tau + \mu))$.

2. Partition locally. This will take $O(\frac{n}{p})$, this is the computation time.

3. Re-arrange partitions. In this step, each processor is communicating to two known processors using one-to-one communication (lower values to left, higher values to right), with data size as $O(\frac{n}{p})$. Therefore, the runtime is $O(\tau + \mu\frac{n}{p})$. This is the communication time at each level.

4. Split processors and recurse.

5. When each side only has only one processor, then we do the locally sort, the runtime is $O(\frac{n}{p} \log \frac{n}{p})$ (recall that sorting n values will take $O(n \log n)$).

Therefore, at the expected case (assuming no imbalance), we will can have $O(\log p)$ iterations before local sort. Therefore:

$$T_{comp} = O(\frac{n}{p}\log p + \frac{n}{p}\log\frac{n}{p}) = O(\frac{n}{p}\log\frac{n}{p})) \tag{1}$$

And the communication time is:

$$T_{comm} = O((\tau + \mu\frac{n}{p})\log p) \tag{2}$$

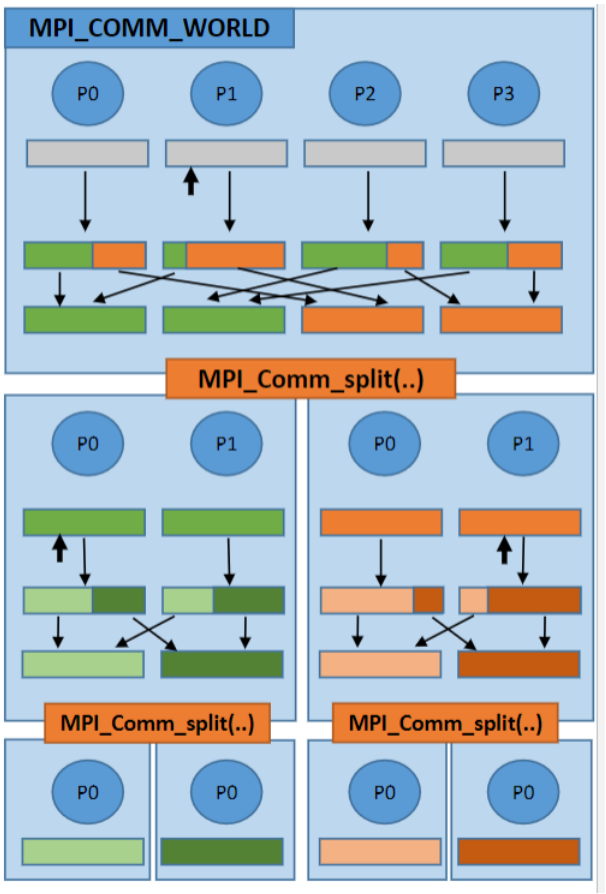


Figure 1: Parallel Quicksort

2 Sample Sort

2.1 Algorithm

The sample sort algorithm is similar with the parallel quicksort algorithm:

1. Sort locally, this will take runtime $O(\frac{n}{p}\log\frac{n}{p})$
2. Then find $p-1$ good splitters globally, the runtime will be discussed latter.

3. Use many-to-many communication to transfer values. Recall the expression of many-to-many runtime is $O(\tau p + \mu(R + S + p^2))$, here both R and S are $O(\frac{n}{p})$ and assume p^2 is relatively small, then the runtime will be $O(\tau p + \mu\frac{n}{p})$

4. Then locally merge p sequences with sorting, each sequence has $\frac{n}{p^2}$ elements. This will take runtime $O(\frac{n}{p^2} \cdot p \cdot \log p) = O(\frac{n}{p} \log p)$

2.2 Finding Good Splitters

The algorithm to find good splitters include:

1. Pick $p - 1$ equally spaced elements (local splitters) on each processor. Because the elements in each processor is sorted, so the runtime will be $O(1)$

2. Sort all $p(p - 1)$ lcoal splitters using bitonic sort. Recall that when $n > p$, the runtime could be expressed as:

$$T_{comp,BS} = O(\frac{n}{p} \log \frac{n}{p} + \frac{n}{p} \log^2 p) \tag{3}$$

$$T_{comm,BS} = O((\tau + \mu\frac{n}{p}) \log^2 p) \tag{4}$$

   In this case, we assume $n = O(p^2)$, therefore the runtime could be expressed as:

$$T_{comp,split} = O(p \log^2 p) \tag{5}$$

$$T_{comm,split} = O((\tau + \mu p) \log^2 p) \tag{6}$$

3. Now we pick $p - 1$ global splitters, using last local splitter on each processor

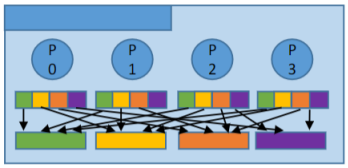
4. Finally we Allgather global splitters. Recall the expression of Allgather runtime is $O(\tau \log p + \mu mp)$. Now the data size is $O(1)$, therefore the runtime will be $O(\tau \log p + \mu p)$.

2.3 Final Runtime Analysis

Combine all of these, we have the runtime analysis shown below:

Sample sort

1. Sort locally — $O\left(\frac{n}{p}\log\frac{n}{p}\right)$
2. **Find $p-1$ "*good*" splitters**
3. Split local sequence into p segments according to splitters
4. Many-to-many communication — $O\left(\tau p+\mu\frac{n}{p}\right)$
5. Locally merge p sequences — $O\left(\frac{n}{p}\log p\right)$



Computation: $O\left(\frac{n}{p}\log\frac{n}{p}+\frac{n}{p}\log p+p\log^2 p\right)=O\left(\frac{n\log n}{p}+p\log^2 p\right)$

Communication: $O\left(\tau p+\mu\frac{n}{p}+(\tau+\mu p)\log^2 p\right)=O\left(\tau p+\mu\left(\frac{n}{p}+p\log^2 p\right)\right)$

Optimal for $n>p^2\log^2 p$.

Figure 2: Sample Sort