Communication Benchmark Projection

Khalid

April 4, 2025

Abstract

We are making projection about the improvements in scale-up and scale-out

1 Introduction

Estimating the time of collectives considering dominant network parameters

2 Variables

Here we list important variables and their theoretical max values:

1. B_{MDFI}: MDFI Bandwidth (For tile-to-tile communication) = $\sim$ 220 GB/s (from internal communication, needs verification), 196 GB/s (ALCF acceptance data, internal)
2. B_{Xe}: Xe–link Bandwidth (For GPU-to-GPU) communication = $\sim$ 53 GB/s [Int23], 15 GB/s (ALCF acceptance data, internal)
3. n_M: Number of ranks participating through MDFI
4. n_X: Number of ranks participating through Xe–link
5. Z: Message size for a collective
6. B_{D2H}: Bandwidth for Device-to-Host: 55 GB/s (PCIe 5, ALCF internal Data)
7. B_{H2S}: Bandwidth for Host-to-Switch: 55 GB/s (PCIe 5)
8. B_{S2N}: Bandwidth for Switch-to-NIC: 55 GB/s (PCIe 5)
9. B_{NIC}: NIC Bandwidth: 25 GB/s (23 GB/s, ALCF internal testing)
10. $B_{comp.}$: GPU Memory Bandwidth: 1024 GB/s

3 Example: Inter-node Allreduce

In this example, we will attempt to reason about one of the most expensive operations in our benchmark – the inter-node allreduce with a 2GB data volume. To estimate the total time to solution, we trace out the data movement path:

$$\boxed{\begin{array}{c}\text{GPU (D2H, through PCIe 5)} \rightarrow \text{DDR (H2S, through PCIe 5)} \rightarrow \text{Fabric (through NIC)} \\ \rightarrow \text{DDR (H2D, through PCIe 5)} \rightarrow \text{GPU}\end{array}}$$

Each rank, except the first rank has to follow the data path to complete the allreduce. Therefore a total of (n-1) data transfers and (n-1) summations are done. The ranks have in total (n_l) links to complete the transfer.

Therefore, the time taken for this case (N2, R1) will be:

$$t_{dp}^{N2,R1} = \left(\frac{Z}{B_{D2H}} + \frac{Z}{B_{H2S}} + \frac{Z}{B_{NIC}} + \frac{Z}{B_{D2H}} + \frac{Z}{B_{comp.}}\right) \times \frac{2(n_t-1)}{n_l} \tag{1}$$

So, if we have 1 Ranks per node,

$$n_t = 2 \times n_{ranks\_per\_node} = 2 \times 1 = 2 \tag{2}$$
$$n_l = \#\ of\ links = 1 \tag{3}$$

3.1 Alternative Model (Scale-up + Scale-out)

This model is coming from a collaboration with Maria Garzaran from Intel. The fundamental idea is to perform the Allreduce in the reduce-scatter+allgather format. The data flow is such that, each tile divides the data in half (Z/2) and exchange it parallelly through the MDFI. Each tile then divide their portion of the data based on the number of Xe-links it is connected to. The bandwidth difference between the Xe-links and MDFI is such that, in the case of large message sizes, all of MDFI transfer can be hidden behind the Xe-link communication. Hence, the first order estimates for cases othe than tile-to-tile allreduces (N1,R2 cases with ZE_AFFINITY=0,1, for example), MDFI can be assumed to be negligible. For small message sizes, this can become important.

$$t_{dp}^{N2,R1} = \overbrace{\frac{Z/2}{n_{Xe} \times B_{Xe}}}^{Reduce-Scatter\ Scale-up} + \overbrace{\frac{2Z}{0.8 \times n_{NICs\_per\_node} \times B_{NIC}} \times \left(\frac{n_{nodes}-1}{n_{nodes}}\right)}^{Allreduce\ Scale-out} + \underbrace{\frac{Z/2}{n_{Xe} \times B_{Xe}}}_{Allgather\ Scale-up} \tag{4}$$

For this model, the assumptions are:

1. The maximum achievable total NIC bandwidth is 80% of the full capacity, hence the factor of 0.8.
2. The compute cost of the Allreduce summation is negligible.
3. We don't include the Host-device data copy cost, assuming the data is already in the device.

Now, explicitly writing the case of $t_{tp}^{N2,R4}$:

$$t_{dp}^{N2,R4} = \overbrace{\frac{Z/2}{n_{Xe} \times B_{Xe}}}^{Reduce-Scatter\ Scale-up} + \overbrace{\frac{2Z}{0.8 \times n_{NICs\_per\_node} \times B_{NIC}} \times \left(\frac{n_{nodes}-1}{n_{nodes}}\right)}^{Allreduce\ Scale-out} + \underbrace{\frac{Z/2}{n_{Xe} \times B_{Xe}}}_{Allgather\ Scale-up} \tag{5}$$
$$= \frac{Z/2}{2 \times B_{Xe}} + \frac{2Z}{0.8 \times 4 \times B_{NIC}} \times \left(\frac{2-1}{2}\right) + \frac{Z/2}{2 \times B_{Xe}}$$

Scale-out data, 1 rank per Node

	Efficiency	Model (ms)	Experiment (ms)
$t_{dp}^{2,1}$ (AR, 2 GB)	27%	108.7	402.7
$t_{dp}^{4,1}$ (AR, 2 GB)	29.3%	163.0	555.4
$t_{dp}^{8,1}$ (AR, 2 GB)	30%	190.2	636.6
$t_{tp}^{16,1}$ (AR, 2 GB)	30%	203.8	680.6

Scale-up and Scale-out data for up to 16 Nodes

	Efficiency	Model (ms)	Experiment (ms)
$t_{tp}^{2,1}$ (AR, 2 GB)	26.9%	108.7	402.7
$t_{tp}^{2,2}$ (AR, 2 GB)	14.8%	65.5	442.01
$t_{tp}^{2,4}$ (AR, 2 GB)	19%	93.8	494.25
$t_{tp}^{2,6}$ (AR, 2 GB)	17.7%	62.6	353.21
$t_{tp}^{2,8}$ (AR, 2 GB)	23%	46.9	203.2
$t_{tp}^{2,12}$ (AR, 2 GB)	16%	35.8	221.8
$t_{tp}^{4,12}$ (AR, 2 GB)	18.2%	42.6	233.0
$t_{tp}^{8,12}$ (AR, 2 GB)	18.5%	46.0	248.5
$t_{tp}^{16,12}$ (AR, 2 GB)	19.2%	47.7	247.9

Depending on the deployment of the benchmark, for example, in the case where the workload is running on a number of nodes, that is **not** a power of 2, the communication protocol may involve explicit copying of a data chunk to host and back. This involves using the PCIe bus, and each rank may do that in a parallel fashion. After including the overhead from the PCIe the timing can be estimated as:

$$\begin{aligned} t_{dp}^{N2,R1} = & \overbrace{\frac{Z/2}{n_{Xe} \times B_{Xe}}}^{Reduce-Scatter\ Scale-up} + \\ & \overbrace{\frac{2Z}{0.8 \times n_{NICs\_per\_node} \times B_{NIC}} \times \left(\frac{n_{nodes} - 1}{n_{nodes}}\right)}^{Allreduce\ Scale-out} + \\ & \underbrace{\frac{Z/2}{n_{Xe} \times B_{Xe}}}^{Allgather\ Scale-up} + \frac{2Z \times 2}{n_{ranks\_per\_node} \times B_{PCIe}} \end{aligned} \tag{6}$$

The first factor of 2 in the PCIe-term above represents the possible usage of PCIe in both reduce-scatter and allgather phase, and the second factor of 2 is to account for copying the data back and forth.

Scale-out data, 1 rank per Node, with PCIe

	Efficiency	Model (ms)	Experiment (ms)
$t_{dp}^{2,1}$ (AR, 2 GB)	74.4%	299.2	402.7
$t_{dp}^{4,1}$ (AR, 2 GB)	63.6%	353.5	555.4
$t_{dp}^{8,1}$ (AR, 2 GB)	59.8%	380.7	636.6
$t_{tp}^{16,1}$ (AR, 2 GB)	57.9%	394.2	680.6

Scale-up and Scale-out data for up to 16 Nodes, with PCIe

	Efficiency	Model (ms)	Experiment (ms)
$t_{tp}^{2,1}$ (AR, 2 GB)	74.4%	299.2	402.7
$t_{tp}^{2,2}$ (AR, 2 GB)	36.3%	160.7	442.01
$t_{tp}^{2,4}$ (AR, 2 GB)	28.6%	141.5	494.25
$t_{tp}^{2,6}$ (AR, 2 GB)	26.6%	94.3	353.21
$t_{tp}^{2,8}$ (AR, 2 GB)	34.7%	70.7	203.2
$t_{tp}^{2,12}$ (AR, 2 GB)	23.3%	51.7	221.8
$t_{tp}^{4,12}$ (AR, 2 GB)	25.1%	58.5	233.0
$t_{tp}^{8,12}$ (AR, 2 GB)	24.9%	61.9	248.5
$t_{tp}^{16,12}$ (AR, 2 GB)	25.7%	63.7	247.9

To calculate the numbers in the table, I have used 42 GB/s for the PCIe 5 bandwidth, as it is more commonly reproduced in experiments at ALCF. 55 GB/s is the highest bandwidth that has been achieved, and 64 GB/s is the theoretical maximum. For the NIC bandwidth I have used 23 GB/s, which is more commonly reproduced.

4 Example: Intra-Node allreduce

$$t_{tp}^{N1,R2} = \left(\frac{Z}{B_{Xe}} + \frac{Z}{B_{comp.}}\right) \times \frac{2(n_t - 1)}{n_l} \tag{7}$$

So, if we have 2 Ranks per node,

$$n_t = 1 \times n_{ranks\_per\_node} = 1 \times 2 = 2 \tag{8}$$

$$n_l = \#\ of\ Xe - links = 1 \tag{9}$$

4.1 Alternative Model: Scale-up

This model is coming from a collaboration with Maria Garzaran from Intel.

$$t_{tp}^{N1,R2} = \overbrace{\frac{Z/n_{ranks\_per\_node}}{B_{Xe}}}^{Reduce-Scatter\ Scale-up} + \overbrace{\frac{Z/n_{ranks\_per\_node}}{B_{Xe}}}^{Allgather\ Scale-up} + \frac{Z/n_{ranks\_per\_node}}{B_{MDFI}} + \frac{Z/n_{ranks\_per\_node}}{B_{MDFI}} \tag{10}$$

	Efficiency	Model (ms)	Experiment (ms)
$t_{tp}^{1,2}$ (AR, 0.875 GB)	74%	58.3	74.2
$t_{tp}^{1,4}$ (AR, 0.875 GB)	66%	31.6	41.5
$t_{tp}^{1,12}$ (AR, 0.875 GB)	46%	10.5	19.7

5 Possible Bottlenecks and future improvements

For the scale-up case, the most likely bottleneck is coming from the efficiency of the Xe-links. According to product specification, Xe-links should have an unidirectional bandwidth of 53 GB/s, where as in the experiments we achieve 15 GB/s. The simple model presented here captures this effect.

For the scale-out case, the most likely bottleneck is coming from the NIC-bandwidth. Theoretically, each NIC has 25 GB/s bandwidth, whereas in experiments we can achieve 23 GB/s. The experiments performed using PyTorch distributed allreduce captures an effective NIC bandwidth of 8-11 GB/s, in some experiments even less. We may perform the same experiments using a lower-level language implementation (C, C++, Sycl) to identify whether the extra-overhead is coming from the Python frameworks.

Assuming that the NIC has to create temporary buffers in the DDR memory to inject data into the fabric, the available PCIe bandwidth can be another bottleneck. Future improvements may allow us to bypass this with NICs having the capability of directly injecting data (for example, larger registrars in the NIC).

In this analysis, we have assumed the compute cost to be negligible, and the GPUs are performing the allreduces without the CPU arbitration. If the compute has to be done involving the CPU registrars, then the core to NOC (network on chip) bandwidth can pose a significant bottleneck.

To test each of the above hypothesis, we need more careful experimentation and analysis.

References

[Int23] Intel. Intel Product Brief: 2023. https://www.intel.com/content/dam/www/central-libraries/us/en/documents/2023-01/data-center-gpu-max-series-product-brief.pdf, 2023. [Online; accessed 19-March-2025].