Performance Data and Interpretations

Q1: Ring communication

Blocking ring with deadlock

The original `ring.c` used blocking `MPI_Send`/`MPI_Recv` that deadlocks once messages exceed the MPI eager threshold (rendezvous mode requires the matching `Recv` to be posted). The largest message size that runs without deadlock is 4096 doubles (≈ 32 KB).

```
*** Assigned Granite Node:  grn059
Message Size = 1; 0.019 Gbytes/sec; Time = 0.066 sec
Message Size = 8; 0.094 Gbytes/sec; Time = 0.100 sec
Message Size = 64; 0.480 Gbytes/sec; Time = 0.104 sec
Message Size = 512; 1.993 Gbytes/sec; Time = 0.054 sec
Message Size = 4096; 3.000 Gbytes/sec; Time = 0.042 sec
```

Non-blocking fix and results

Fix: post `MPI_Irecv` before `MPI_Isend`, then `MPI_Waitall` for each send/recv pair. This ensures matching receives are posted before large messages enter rendezvous, eliminating deadlock.

```
*** Assigned Granite Node:  grn055
Message Size = 1; 0.015 Gbytes/sec; Time = 0.082 sec
Message Size = 8; 0.086 Gbytes/sec; Time = 0.111 sec
Message Size = 64; 0.460 Gbytes/sec; Time = 0.108 sec
Message Size = 512; 2.021 Gbytes/sec; Time = 0.053 sec
Message Size = 4096; 2.255 Gbytes/sec; Time = 0.056 sec
Message Size = 32768; 4.299 Gbytes/sec; Time = 0.030 sec
Message Size = 262144; 4.767 Gbytes/sec; Time = 0.028 sec
Message Size = 1048576; 4.176 Gbytes/sec; Time = 0.032 sec
```

Q2: Ping-Pong and α–β model

Single-node results

Msg (doubles)	Bytes	Time/message (µs)	Bandwidth (GB/s)
1	8	0.353	0.023
8	64	0.562	0.114
64	512	0.818	0.626

Msg (doubles)	Bytes	Time/message (µs)	Bandwidth (GB/s)
512	4,096	1.633	2.508
4,096	32,768	6.840	4.791
32,768	262,144	15.414	17.007
262,144	2,097,152	180.964	11.589
1,048,576	8,388,608	954.993	8.784

Two-node results

Msg (doubles)	Bytes	Time/message (µs)	Bandwidth (GB/s)
1	8	3.553	0.002
8	64	3.488	0.018
64	512	4.677	0.109
512	4,096	5.660	0.724
4,096	32,768	8.300	3.948
32,768	262,144	35.978	7.286
262,144	2,097,152	207.842	10.090
1,048,576	8,388,608	890.102	9.424

Alpha–beta model calculations

Model

$$T(m) = \alpha + \beta \cdot m$$

- α: latency (ns)
- β: time per byte (ns/byte)
- m: bytes

We estimate β using two large-message points (to minimize α's influence), then solve for α using a small-message point.

Single-node calculations

- Use $m_1 = 262,144$ bytes, $T_1 = 15,414$ ns and
- $m_2 = 8,388,608$ bytes, $T_2 = 954,993$ ns:

$$\beta_{\text{SN}} \approx \frac{T_2 - T_1}{m_2 - m_1} = \frac{954,993 - 15,414}{8,388,608 - 262,144} = \frac{939,579}{8,126,464} \approx 0.1156 \text{ ns/byte}$$

- Estimate α using $m = 512$ bytes, $T = 818$ ns:

$$\alpha_{\text{SN}} \approx T - \beta \cdot m = 818 - 0.1156 \cdot 512 \approx 818 - 59.2 \approx 758.8 \text{ ns}$$

- Implied peak bandwidth:

$$\text{BW}_{\text{SN}} \approx \frac{1}{\beta} \approx \frac{1}{0.1156} \text{ bytes/ns} \approx 8.65 \text{ GB/s}$$

Two-node calculations

- Use $m_1 = 2,097,152$ bytes, $T_1 = 207,842$ ns and $m_2 = 8,388,608$ bytes, $T_2 = 890,102$ ns:

$$\beta_{\text{2N}} \approx \frac{T_2 - T_1}{m_2 - m_1} = \frac{890,102 - 207,842}{8,388,608 - 2,097,152} = \frac{682,260}{6,291,456} \approx 0.1084 \text{ ns/byte}$$

- Estimate α using $m = 512$ bytes, $T = 4,677$ ns:

$$\alpha_{\text{2N}} \approx T - \beta \cdot m = 4,677 - 0.1084 \cdot 512 \approx 4,677 - 55.5 \approx 4,621.5 \text{ ns}$$

- Implied peak bandwidth:

$$\text{BW}_{\text{2N}} \approx \frac{1}{\beta} \approx \frac{1}{0.1084} \text{ bytes/ns} \approx 9.22 \text{ GB/s}$$

Interpretation

- **Bandwidth:** β (slope) is similar across single-node and two-node at large sizes, aligning with my measured GB/s near the largest messages. Mid-range deviations arise from eager/rendezvous protocol and cache effects.
-

  ### Consistency check: Using the largest points, BW derived from β aligns with reported GB/s (8.78 GB/s single-node, 9.42 GB/s two-node).

Q3: Distributed Conjugate Gradient (CG)

Results (N=1000)

Processes	Sequential time (s)	Parallel time (s)	Iterations	Speedup
2	0.0551	0.0287	13	1.92

Processes	Sequential time (s)	Parallel time (s)	Iterations	Speedup
4	0.0563	0.0148	13	3.81
8	0.0667	0.0087	13	7.63
16	0.0560	0.0149	13	3.76

Interpretation

- Near-linear speedups up to 8 processes.
-

  16-process speedup drops due to memory bandwidth contention and collective overhead when local work per rank shrinks.

All the log files and output files are included for review.