Collective communication costs in MPI

Suraj Kumar

Inria & ENS Lyon

Email:*suraj.kumar@inria.fr*

CR12: September 2023

https://surakuma.github.io/courses/daamtc.html

Cost model

- The links of the network are bidirectional
- Each process can send and receive at-most one message at the same time
- Time taken to send a message with n bytes between any two nodes – $T = \alpha + n\beta$
  - α: latency cost per message, β: transfer time per byte
- In case of reduction operation, γ: computation cost per byte
- P is the total number of processes

Allgather

p0	p1	p2	p3	p4	p5
a_0	a_1	a_2	a_3	a_4	a_5

Initial data

p0	p1	p2	p3	p4	p5
a_0	a_0	a_0	a_0	a_0	a_0
a_1	a_1	a_1	a_1	a_1	a_1
a_2	a_2	a_2	a_2	a_2	a_2
a_3	a_3	a_3	a_3	a_3	a_3
a_4	a_4	a_4	a_4	a_4	a_4
a_5	a_5	a_5	a_5	a_5	a_5

Data after operation

Ring algorithm

- Takes total $P - 1$ steps
- In each step, process i sends its contribution to process $i + 1$ and receives the contribution from process $i - 1$
- $T_{ring} = (P - 1)\alpha + \left(\frac{P-1}{P}\right) n\beta$
- n is the total amount of data gathered on each processor

Recursive doubling algorithm for Allgather

p0	p1	p2	p3	p4	p5	p6	p7
a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7

Initial data

a_0	a_0	a_2	a_2	a_4	a_4	a_6	a_6
a_1	a_1	a_3	a_3	a_5	a_5	a_7	a_7

a_0	a_0	a_0	a_0	a_4	a_4	a_4	a_4
a_1	a_1	a_1	a_1	a_5	a_5	a_5	a_5
a_2	a_2	a_2	a_2	a_6	a_6	a_6	a_6
a_3	a_3	a_3	a_3	a_7	a_7	a_7	a_7

a_0	a_0	a_0	a_0	a_0	a_0	a_0	a_0
a_1	a_1	a_1	a_1	a_1	a_1	a_1	a_1
a_2	a_2	a_2	a_2	a_2	a_2	a_2	a_2
a_3	a_3	a_3	a_3	a_3	a_3	a_3	a_3
a_4	a_4	a_4	a_4	a_4	a_4	a_4	a_4
a_5	a_5	a_5	a_5	a_5	a_5	a_5	a_5
a_6	a_6	a_6	a_6	a_6	a_6	a_6	a_6
a_7	a_7	a_7	a_7	a_7	a_7	a_7	a_7

- Assume P is a perfect power of 2
- In each step $k (0 \leq k < \lg P)$, processes that are at 2^k distance exchange their data
- $T_{rec\_dbl} = \lg P \alpha + \left(\frac{P-1}{P}\right) n\beta$
- Requires adaptation when P is not a-power-of-two

Bruck's algorithm for Allgather

p0	p1	p2	p3	p4	p5
a_0	a_1	a_2	a_3	a_4	a_5

Initial data

p0	p1	p2	p3	p4	p5
a_0	a_1	a_2	a_3	a_4	a_5
a_1	a_2	a_3	a_4	a_5	a_0

After step 0

p0	p1	p2	p3	p4	p5
a_0	a_1	a_2	a_3	a_4	a_5
a_1	a_2	a_3	a_4	a_5	a_0
a_2	a_3	a_4	a_5	a_0	a_1
a_3	a_4	a_5	a_0	a_1	a_2

After step 1

p0	p1	p2	p3	p4	p5
a_0	a_1	a_2	a_3	a_4	a_5
a_1	a_2	a_3	a_4	a_5	a_0
a_2	a_3	a_4	a_5	a_0	a_1
a_3	a_4	a_5	a_0	a_1	a_2
a_4	a_5	a_0	a_1	a_2	a_3
a_5	a_0	a_1	a_2	a_3	a_4

After step 2

p0	p1	p2	p3	p4	p5
a_0	a_0	a_0	a_0	a_0	a_0
a_1	a_1	a_1	a_1	a_1	a_1
a_2	a_2	a_2	a_2	a_2	a_2
a_3	a_3	a_3	a_3	a_3	a_3
a_4	a_4	a_4	a_4	a_4	a_4
a_5	a_5	a_5	a_5	a_5	a_5

After local shift

- $T_{bruck} = \lceil \lg P \rceil \alpha + \frac{P-1}{P} n\beta$
- In each step $k (0 \leq k < \lceil \lg P \rceil)$, process i sends data to process $(i - 2^k)\%P$ and receives data from process $(i + 2^k)\%P$

Broadcast

It broadcasts n words from the root to all processes.

p0	p1	p2	p3	p4	p5
a					

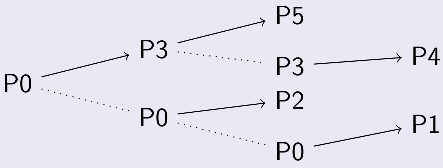
Initial setting

p0	p1	p2	p3	p4	p5
a	a	a	a	a	a

Data after operation

Bionomial tree algorithm

In the first step, the root sends data to process $(root + \frac{P}{2})$. This process and the root then act as new roots and recursively continue this algorithm.



- $T_{tree} = \lceil \lg P \rceil (\alpha + n\beta)$

Alternative approach: Scatter + Allgather

All-to-All

p0	p1	p2	p3	p4	p5
a_0	b_0	c_0	d_0	e_0	f_0
a_1	b_1	c_1	d_1	e_1	f_1
a_2	b_2	c_2	d_2	e_2	f_2
a_3	b_3	c_3	d_3	e_3	f_3
a_4	b_4	c_4	d_4	e_4	f_4
a_5	b_5	c_5	d_5	e_5	f_5

Initial settings

p0	p1	p2	p3	p4	p5
a_0	a_1	a_2	a_3	a_4	a_5
b_0	b_1	b_2	b_3	b_4	b_5
c_0	c_1	c_2	c_3	c_4	c_5
d_0	d_1	d_2	d_3	d_4	d_5
e_0	e_1	e_2	e_3	e_4	e_5
f_0	f_1	f_2	f_3	f_4	f_5

Data after operation

Algorithm by Thakur et al.

- In each step $k(1 \leq k < P)$, process i receives data from process $(i-k)\%P$ and send data to process $(i+k)\%P$
- $T = (P-1)\alpha + \frac{P-1}{P} n\beta$
- n is the total amount of data on any process in the beginning or end

Reduce-Scatter

p0	p1	p2	p3	p4	p5
a_0	b_0	c_0	d_0	e_0	f_0
a_1	b_1	c_1	d_1	e_1	f_1
a_2	b_2	c_2	d_2	e_2	f_2
a_3	b_3	c_3	d_3	e_3	f_3
a_4	b_4	c_4	d_4	e_4	f_4
a_5	b_5	c_5	d_5	e_5	f_5

Initial settings

p0	p1	p2	p3	p4	p5
x_0	x_1	x_2	x_3	x_4	x_5

Data after operation

$x_i = \text{Reduce}(a_i, b_i, c_i, d_i, e_i)$
Each process has n amount of data in the beginning.

Recursive halving algorithm (Assuming P is a perfect power of 2)

- Analogous to the recursive-doubling algorithm for Allgather
- In each step $k (1 \leq k \leq P)$, processes that are at $\frac{P}{2^k}$ distance exchange parts of their data
- Each process sends the data needed by all processes in the other half, receives the data needed by all processes in its own half
- $T_{rec\_half} = \lg P\alpha + \frac{P-1}{P} n\beta + \frac{P-1}{P} n\gamma$
- Requires adaptation when P is not a power-of-two

- With an adaptation of Bruck's algorithm:
  $T = \lceil \lg P \rceil \alpha + \frac{P-1}{P} n\beta + \frac{P-1}{P} n\gamma$

Reduce and Allreduce

Each process has n amount of data in the beginning.

Reduce

- With binomial tree algorithm, $T_{tree} = \lceil \lg P \rceil (\alpha + n\beta + n\gamma)$
- With Reduce-Scatter(Bruck's algorithm) + Gather(Binomial tree), $T = 2\lceil \lg P \rceil \alpha + 2\frac{P-1}{P} n\beta + \frac{P-1}{P} n\gamma$

Allreduce

- With Reduce-Scatter(Bruck's algorithm) + Allgather(Bruck's algorithm), $T = 2\lceil \lg P \rceil \alpha + 2\frac{P-1}{P} n\beta + \frac{P-1}{P} n\gamma$

R. Thakur, R. Rabenseifner, and W. Gropp (2005).
Optimization of Collective Communication Operations in MPICH
Int. J. High Perform. Comput. Appl. 19, 1 (February 2005), 49–66.

E. W. Chan, M. F. Heimlich, A. Purkayastha and R. A. van de Geijn
On Optimizing Collective Communication
IEEE International Conference on Cluster Computing, San Diego, CA, USA, 2004, pp. 145-155.