Performance of the Supercomputer Fugaku for Breadth-First Search in Graph500 Benchmark

Masahiro Nakao (masahiro.nakao@riken.jp)†

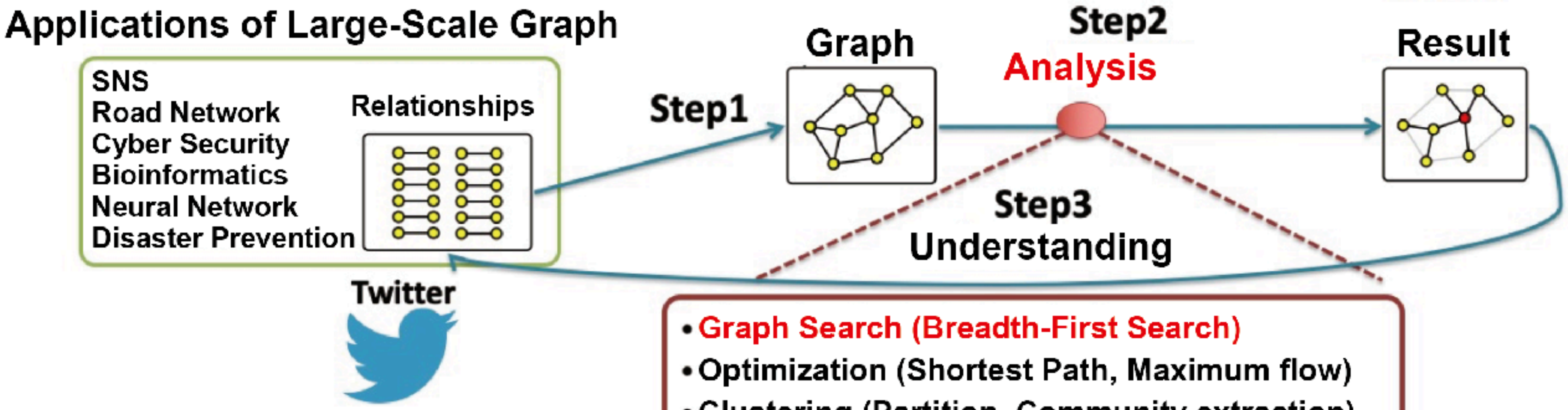
Koji Ueno‡, Katsuki Fujisawa*, Yuetsu Kodama†, Mitsuhisa Sato†

† RIKEN Center for Computational Science

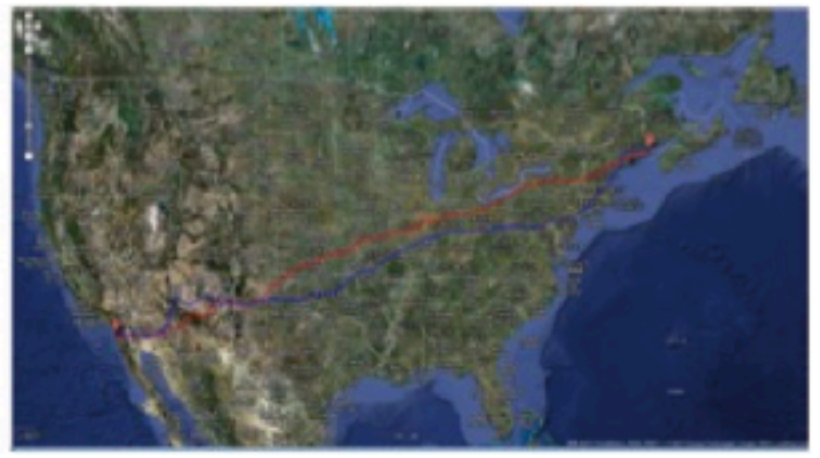
‡ Fixstars Corporation

* Institute of Mathematics for Industry, Kyushu University

Background

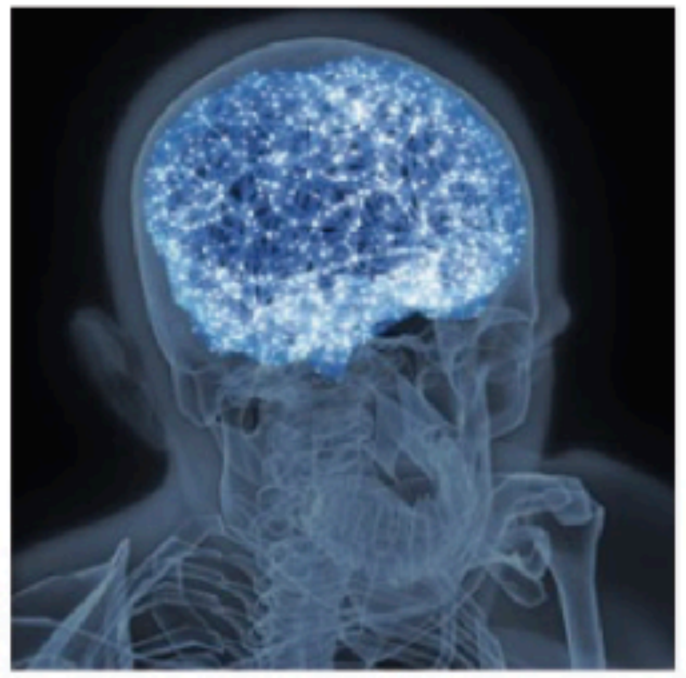


Social Networking Service
61.6M vertices & 1.4B edges



Road Network
24M vertices & 58M edges

Cyber Security
15B access/day



Neural Network
89B vertices & 100T edges

http://opt.imi.kyushu-u.ac.jp/lab/jp/activities.html

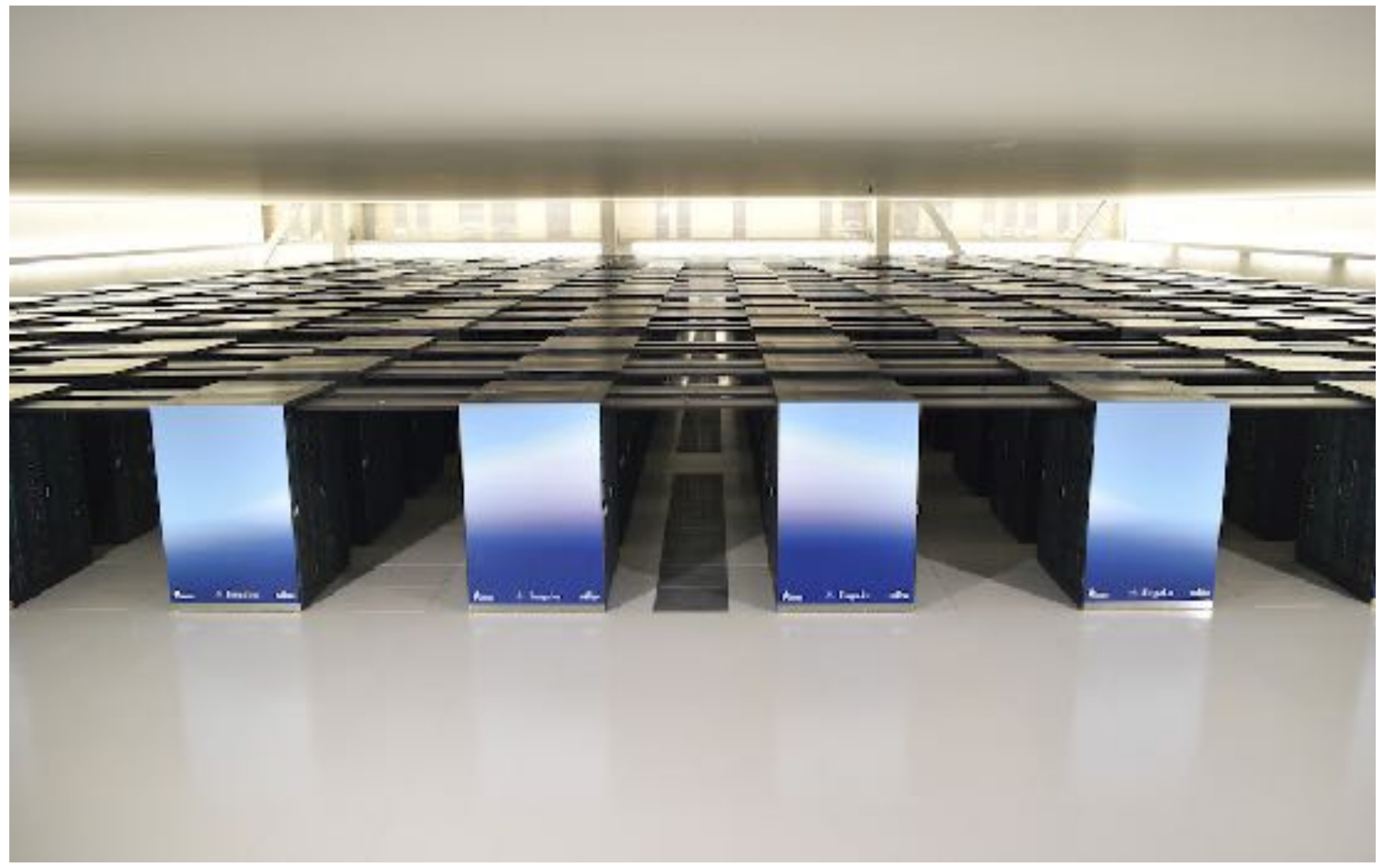
Breadth First Search on Distributed Memory System

- Breadth First Search (BFS)
  - The most fundamental graph algorithm
  - A kernel used in Graph500 list
- Large scale supercomputers
  - Consist of thousands of distributed memory nodes
  - Graph algorithm that runs efficiently on such systems is a challenging research

The K computer : 88,128 nodes[1]

[1] Koji Ueno et al: Efficient breadth-first search on massively parallel and distributed-memory machines.
Data Science and Engineering, (2016)

Supercomputer Fugaku : 152,352 nodes



Graph500

https://graph500.org

- Graph500 is a competition for evaluating performance of large-scale graph processing
- The performance unit is a traversed edges per second (TEPS)
  - 1GTEPS : Search 1 billion edges per second
- Graph500 list is updated twice a year (June and November in BoFs of ISC and SC)
  - The K computer ranked first 10 times from 2014 to 2019
  - Supercomputer Fugaku ranked first in June and November 2020
  - New Graph500 list will be announced at the ISC BoF on July 1
- In graph500, an artificial graph called the “Kronecker graph” is used
  - Some vertices are connected to many other vertices while numerous others are connected to only a few vertices
  - Social network is known to have a similar property

Objective

- This presentation describes the performance tuning of BFS for the Graph500 submission in 2020 and experimental evaluation results conducted on Fugaku

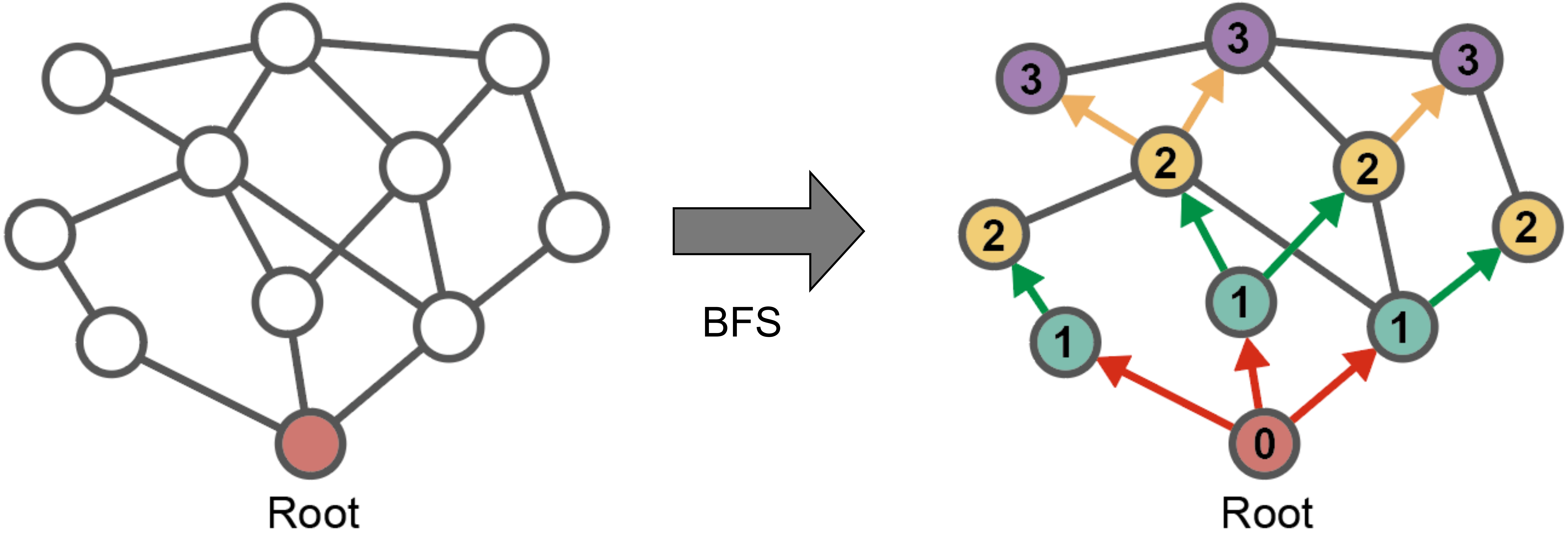
- Summary
  - Use a large-graph with 2.2 trillion vertices and 35.2 trillion edges (SCALE=41)
  - Archive 102,955 GTEPS
  - The result of Fugaku is 3.3 times better than that of the K computer

	June 2019			November 2020		
	NAME	SCALE	GTEPS	NAME	SCALE	GTEPS
1st	**K computer**	**40**	**31,302**	**Supercomputer Fugaku**	**41**	**102,955**
2nd	Sunway TaihuLight	40	23,756	Sunway TaihuLight	40	23,756
3rd	Sequoia	41	23,751	TOKI-SORA	36	10,813
4th	Mira	40	14,982	Summit	40	7,666
5th	SuperMUC-NG	39	6,279	SuperMUC-NG	39	6,279

Outline

- BFS in Graph500 Benchmark
- Supercomputer Fugaku
- Tuning BFS on Supercomputer Fugaku
- Full node evaluation

Overview of BFS in Graph500

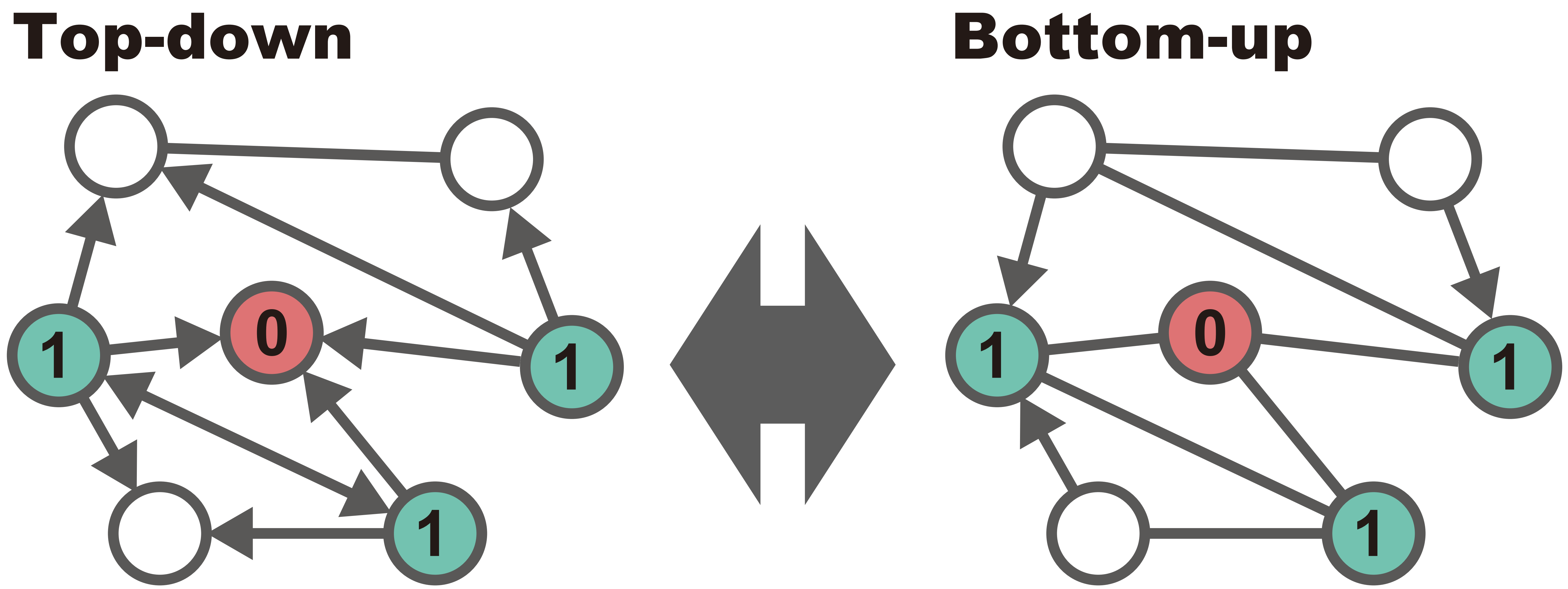


Input：Kronecker graph and root vertex **Output：BFS tree**

- Repeat BFS 64 times from different root vertex
- The harmonic mean of the 64 results is used as the final performance

Hybrid-BFS [Beamer, 2012] Scott Beamer et al. Direction-optimizing breadth-first search

- Hybrid-BFS runs while switching between **Top-down** and **Bottom-up**



Search for unvisited vertices from visited vertices

Search for visited vertices from unvisited vertices

- In the middle of BFS, the number of vertices being visited increases explosively, so it is inefficient in only **Top-down**
- Hybrid-BFS switches between **Top-down** and **Bottom-up** on the situation

2D Hybrid-BFS

[Beamer, 2013] Scott Beamer, et. al. Distributed Memory Breadth-First Search Revisited: Enabling Bottom-Up Search. IPDPSW '13.

- Adjacency matrix is distributed to a 2D process grid (R x C)

$$A = \left(\begin{array}{c|c|c} A_{1,1} & \cdots & A_{1,C} \\ \hline \vdots & \ddots & \vdots \\ \hline A_{R,1} & \cdots & A_{R,C} \end{array}\right)$$

- Communication only within the column processes and row processes
  - Allgatherv, Alltoallv, isend/irecv/wait
- The closer the R and C values are, the smaller the total communication size
- **Based on this 2D Hybrid-BFS, we implemented BFS with various ideas to improve performance[1]**

[1] Koji Ueno et al: Efficient breadth-first search on massively parallel and distributed-memory machines. Data Science and Engineering, (2016)

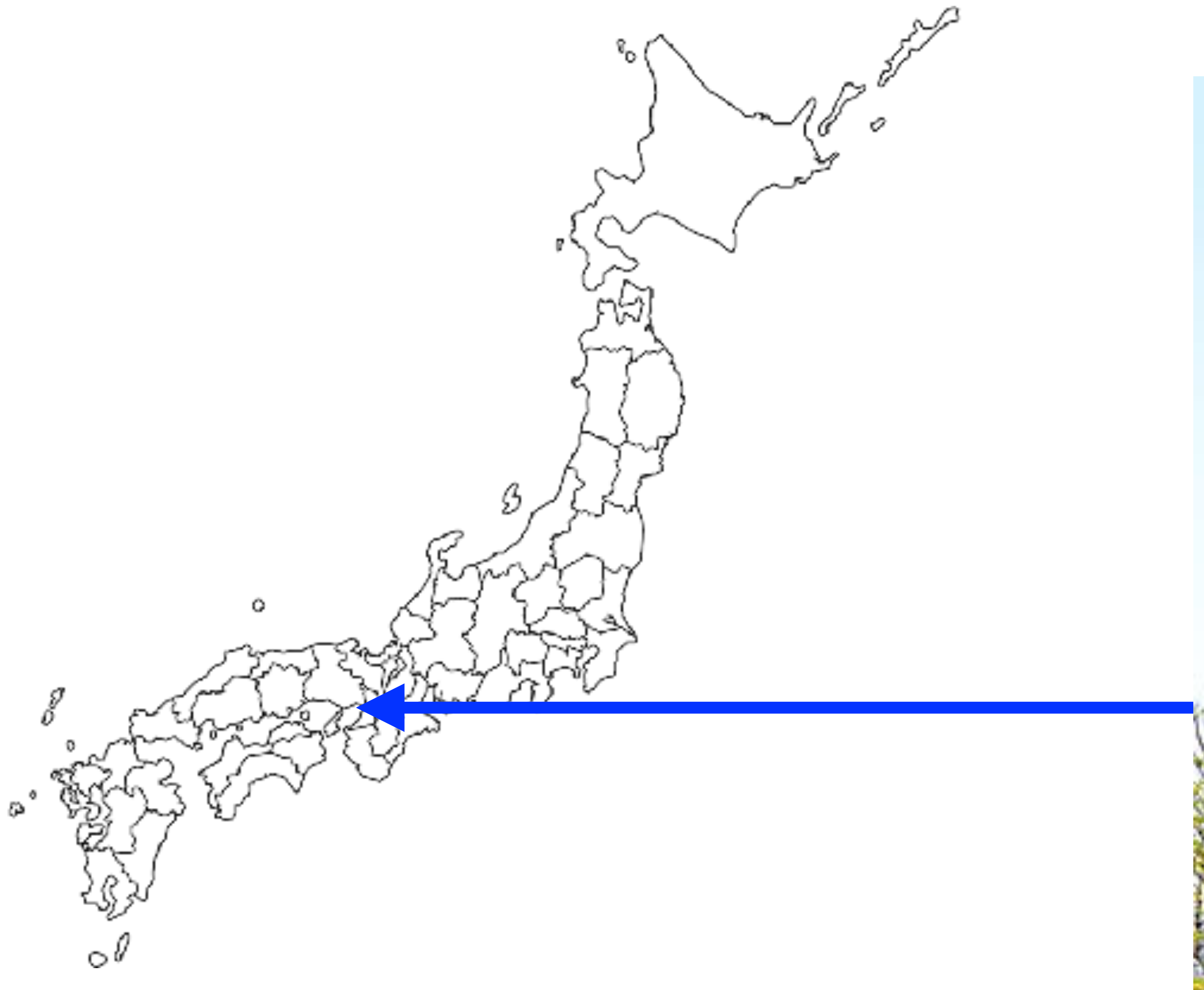
Outline

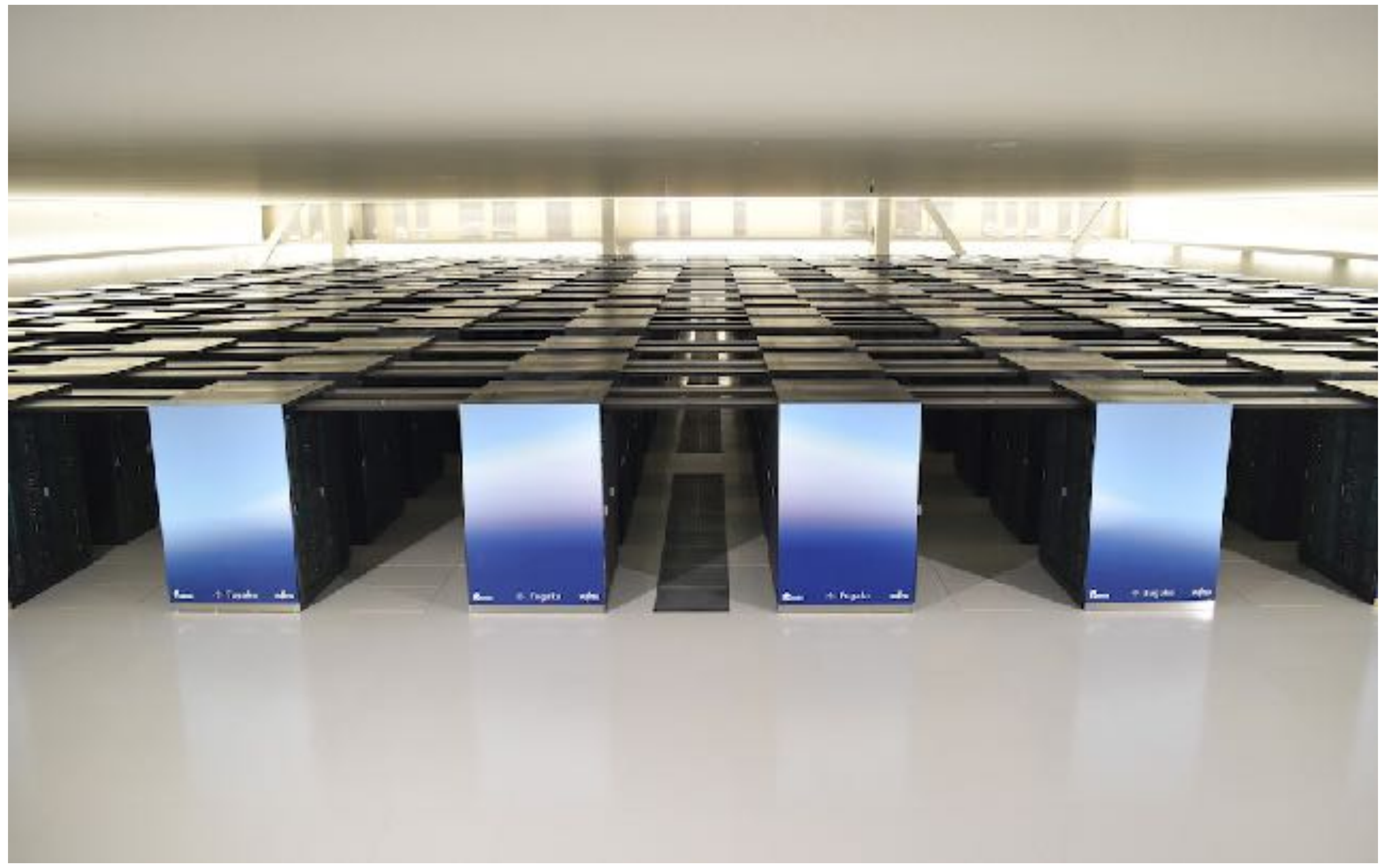
- BFS in Graph500 Benchmark
- Supercomputer Fugaku
- Tuning BFS on Supercomputer Fugaku
- Full node evaluation

Supercomputer Fugaku

- Supercomputer Fugaku, which is developed jointly by RIKEN and Fujitsu Limited based on Arm technology
- Located in RIKEN Center for Computational Science in Kobe, Hyogo, Japan
- 158,976 compute nodes
- Start sharing in March 2021

Note that the results in this presentation do not guarantee performance at the start of sharing because these are obtained before sharing.

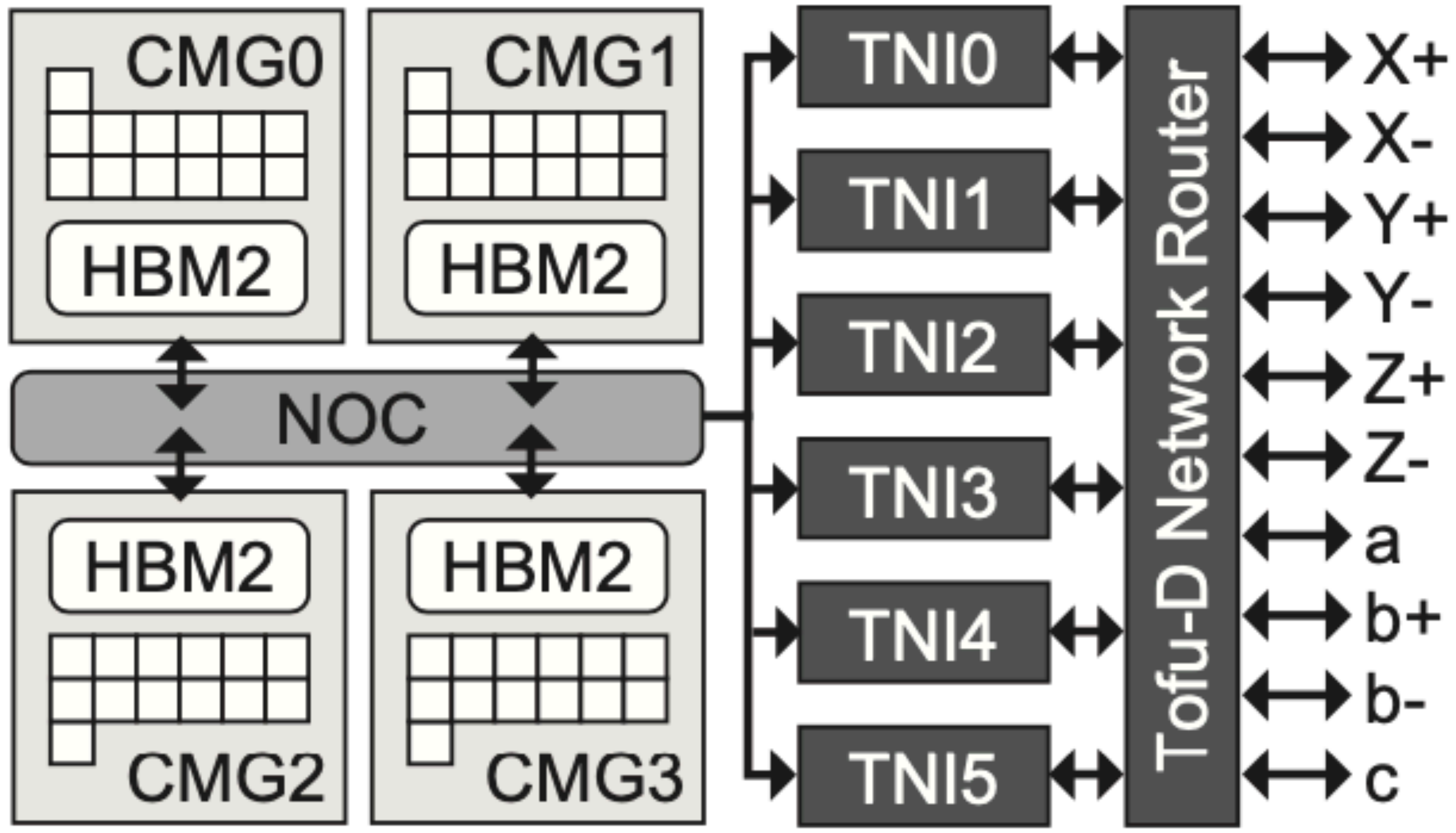




Specification of Computer Node

CPU	A64FX, 48+2/4cores, 2.0/2.2GHz, L2 8MB 3,072/3,379GFlops (double precision)
Memory	HBM2, 32GB, 1,024GB/s
Network	TofuD, 0.49 to 0.54μs (Latency) 6.8GB/s (Bandwidth)

CPU (A64FX)



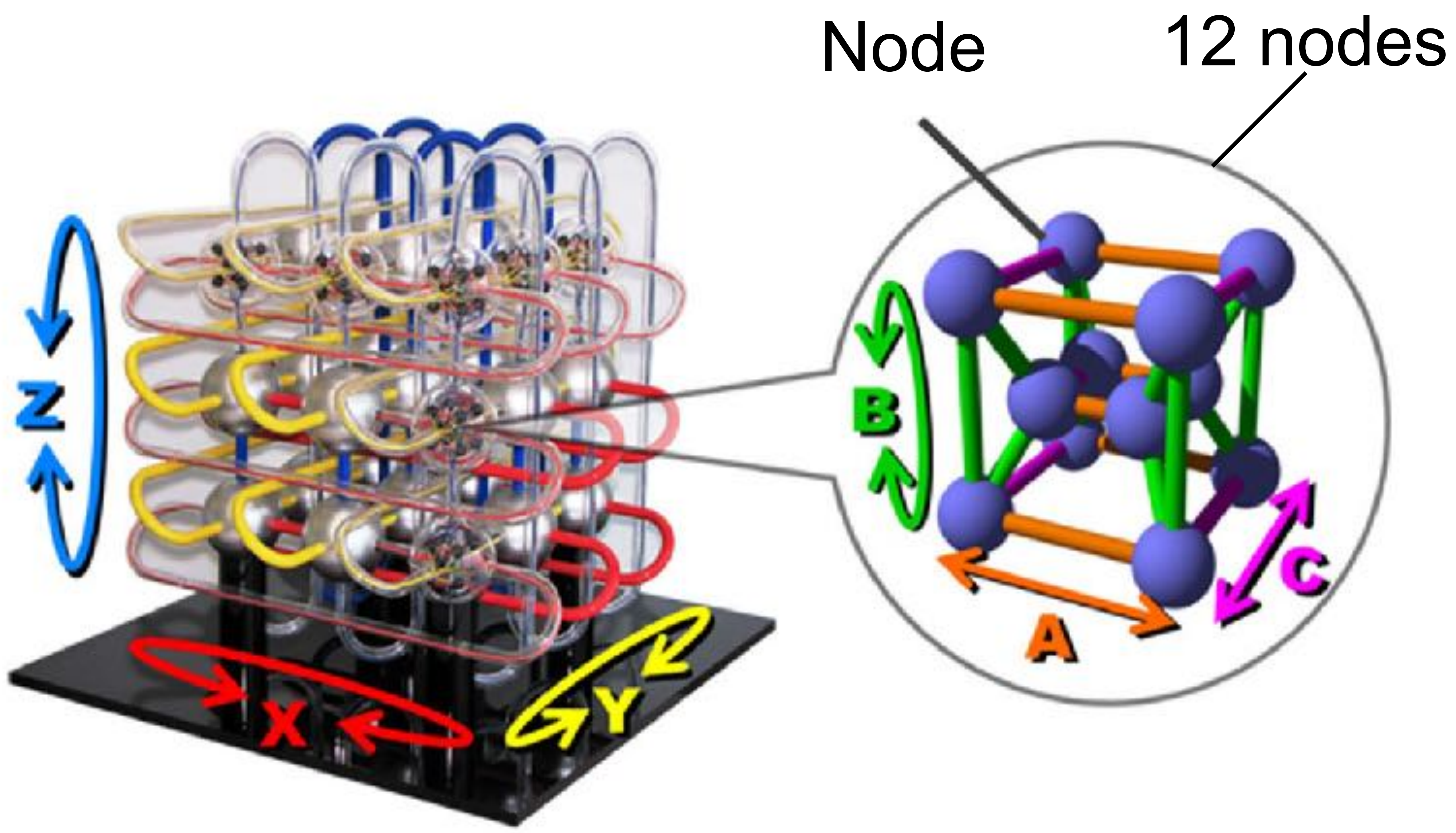
L2 Cache Coherent control between CMGs

- Each node has a single CPU
- Each CPU has 48 compute cores and 2/4 assistant cores. The assistant cores handle the interrupts OS and communication
- **2.0 GHz or 2.2 GHz for each job**
- Each CPU consists of 4 CMGs
  - Each CMG consists of 12 + 1 cores and 8GiB HBM2
  - **It is recommended that the number of processes per CPU is a divisor of 4**
- Each CPU has 10 network cables

CMG : Core Memory Group
NOC : Network on Chip
TNI: Tofu Network Interface

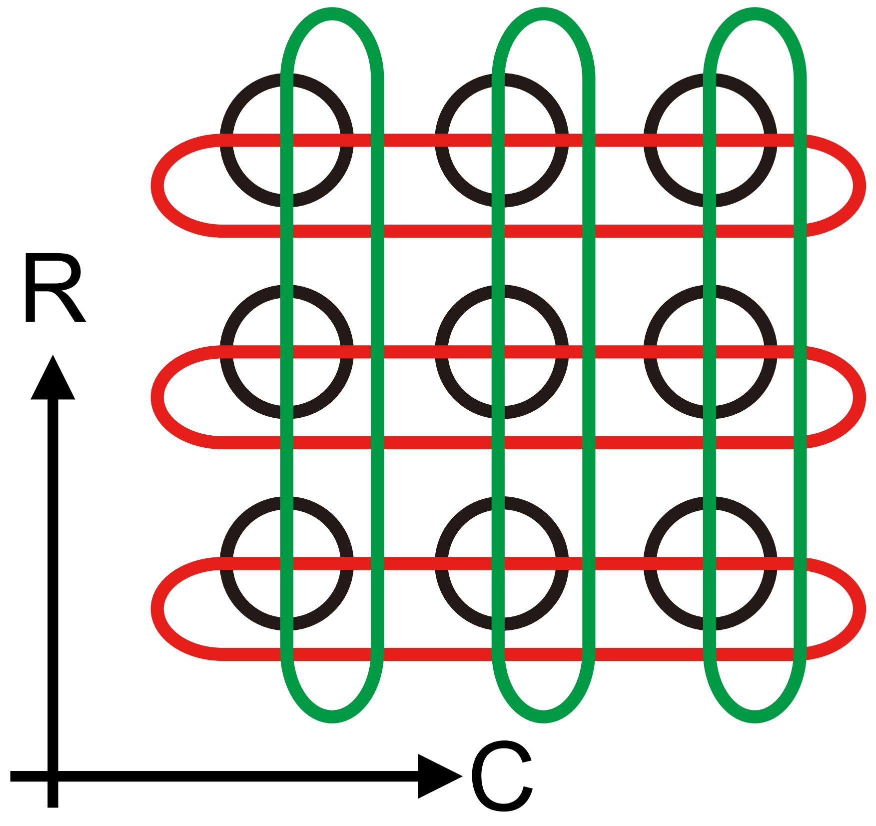
Network topology of Fugaku

- 6D mesh/torus : XYZabc-axis
  - The size of abc is fixed (a,b,c) = (2,3,2)
  - The size of XYZ depends on the system
  - The size of XYZ of Fugaku is (24,23,24)
    so it has 24*23*24*2*3*2 = 158,976 nodes



https://pr.fujitsu.com/jp/news/2020/04/28.html

- Process Mapping
  - Discrete assignment
  - 1D torus or mesh
  - **2D torus** or mesh
  - 3D torus or mesh



$$A = \left(\begin{array}{c|c|c} A_{1,1} & \cdots & A_{1,C} \\ \hline \vdots & \ddots & \vdots \\ \hline A_{R,1} & \cdots & A_{R,C} \end{array}\right)$$

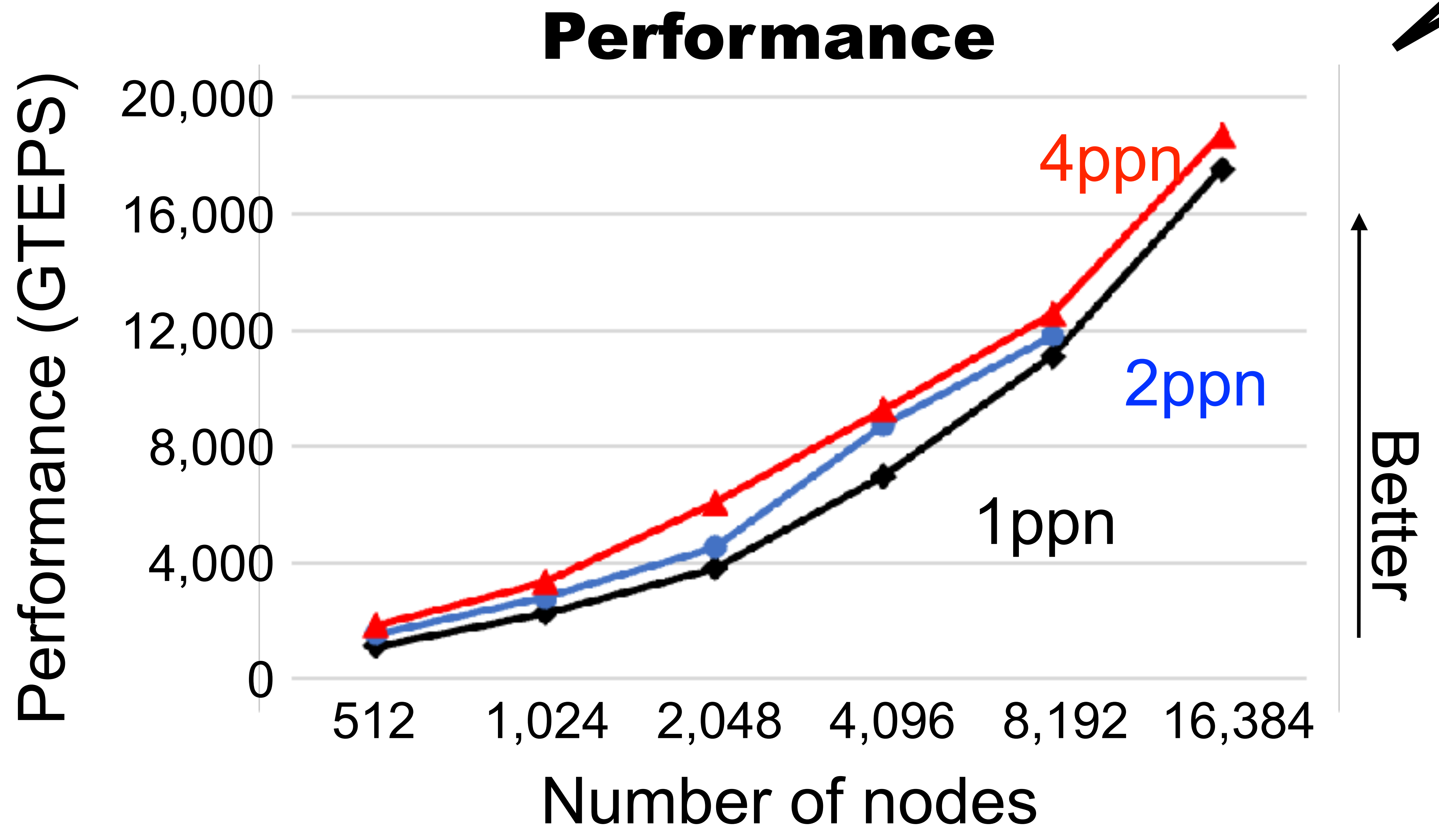
Outline

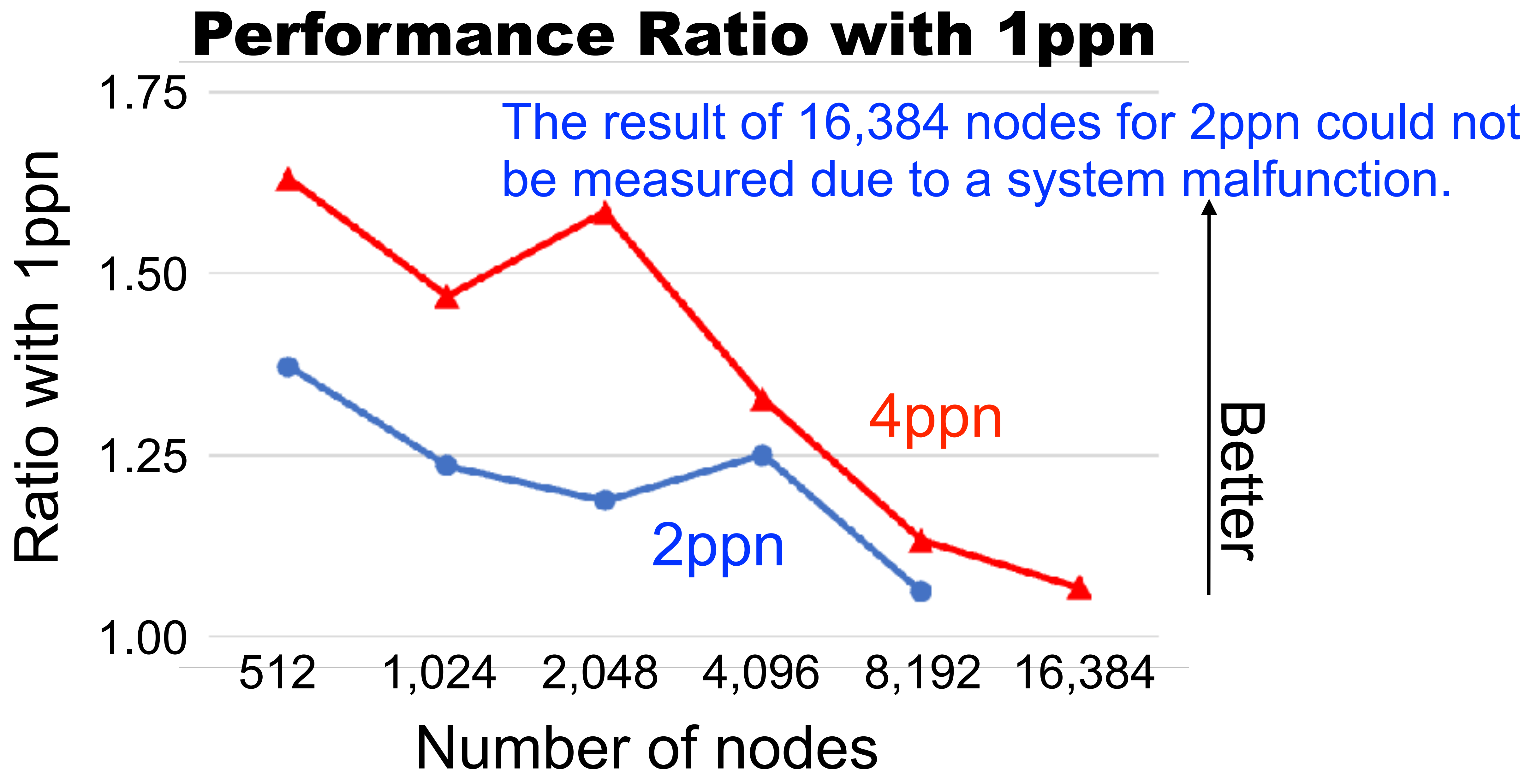
- BFS in Graph500 Benchmark
- Supercomputer Fugaku
- Tuning BFS on Supercomputer Fugaku
- Full node evaluation

Number of processes per node (1/2)

- Process per node (ppn)
  - 1 process 48 threads per node (1ppn)
  - 2 processes 24threads per node (2ppn)
  - 4 processes 12threads per node (4ppn)

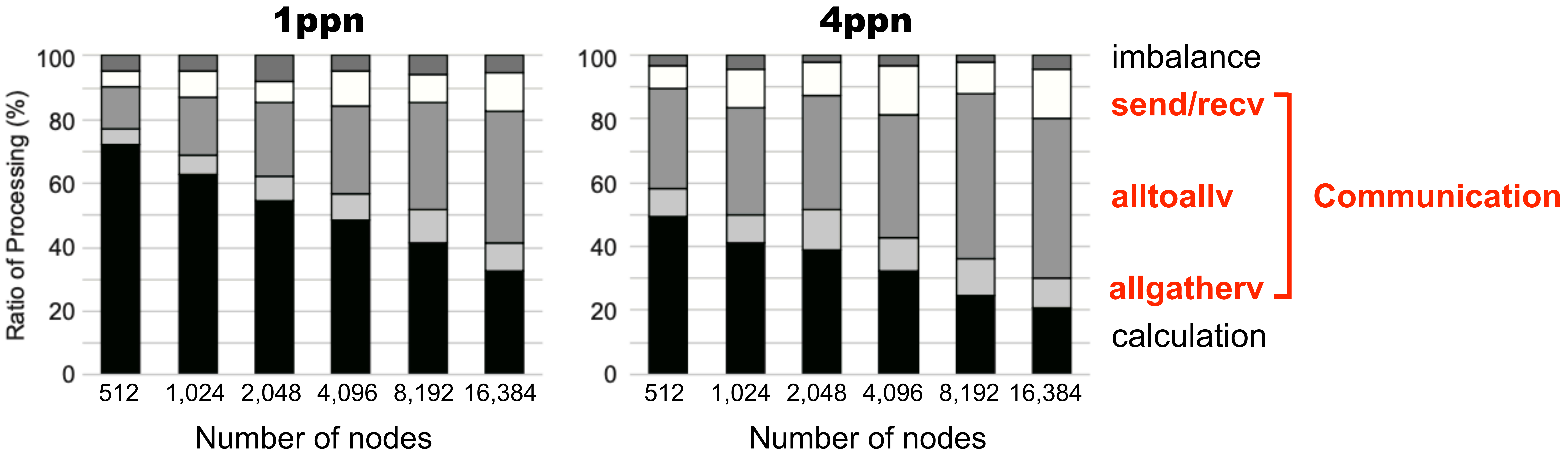
In the cases of 1ppn and 2ppn, the cache hit rate decreases because the memory accesses by threads cross the CMG.





- The larger the number of nodes, the smaller the performance difference

Number of processes per node (2/2)

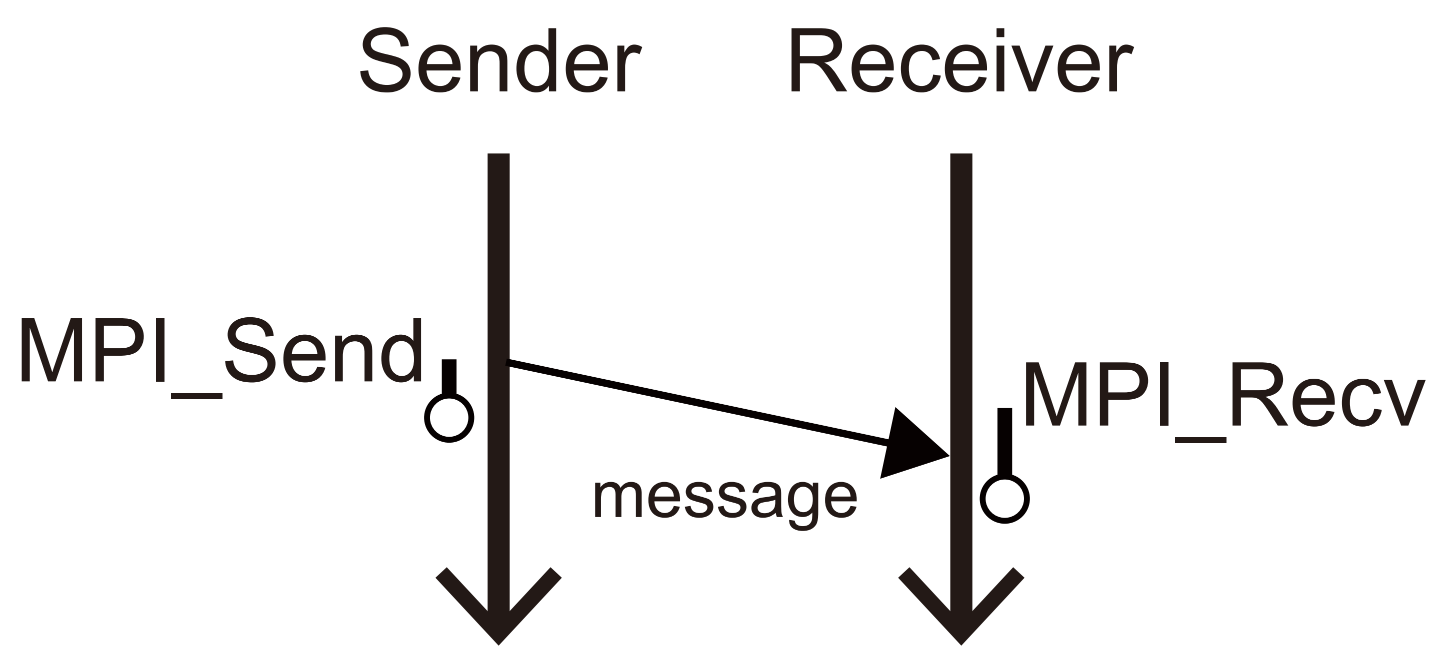


- As the number of nodes increases, the rate of communication increases
- 1ppn has a smaller rate of communication than 4ppn
- If the number of nodes is increased further, the communication ratio will increase.
- **Thus, we select 1ppn, which can bring out the full communication performance**

Use of Eager method (1/2)

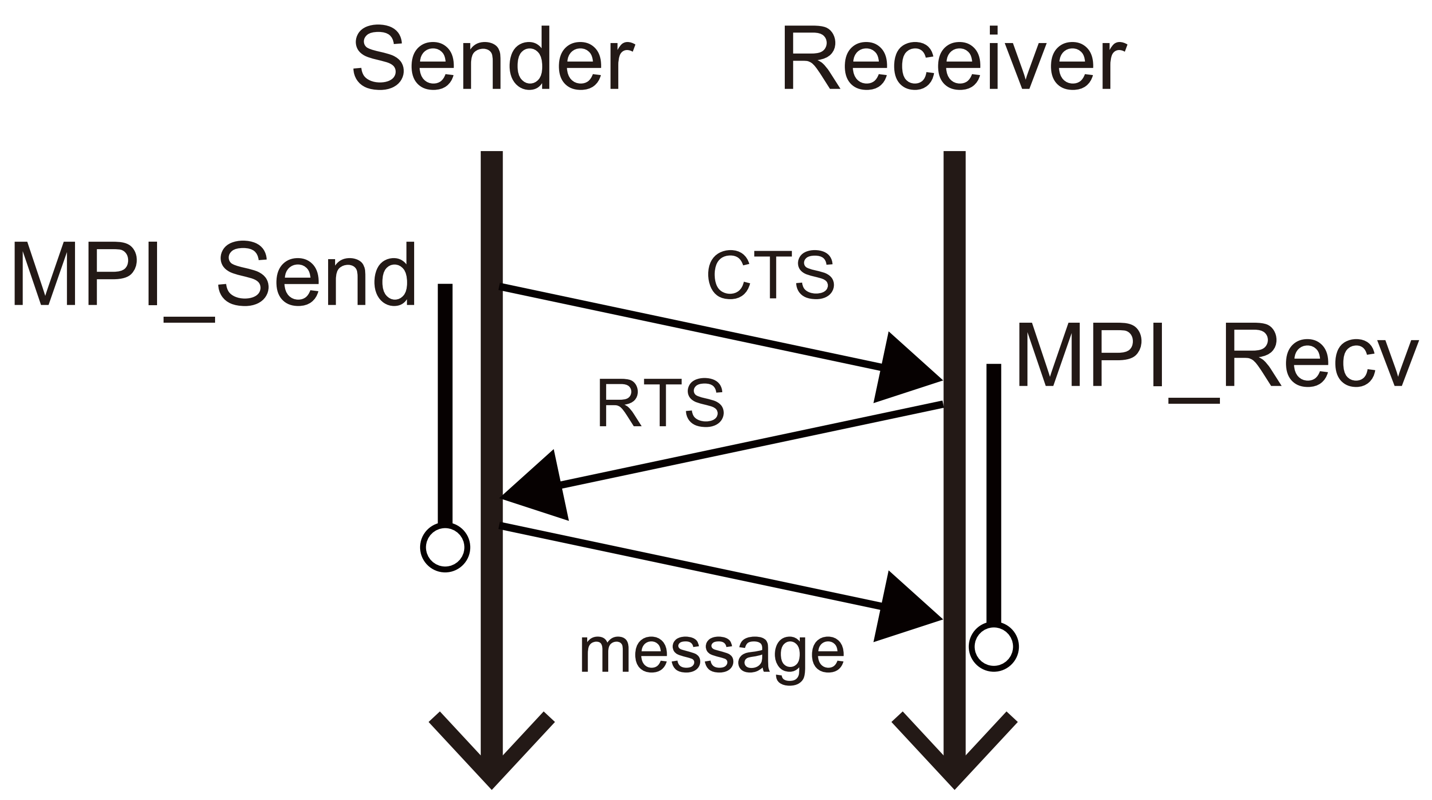
- In the point-to-point communication of most MPI implementations, the Eager and Rendezvous methods are implemented
- Although most MPI implementations switch the Eager and Rendezvous methods automatically depending on message size, optimal message size depends on application

Eager



Asynchronous communication that can start/end the message sending process regardless of the state of the sending/receiving processes

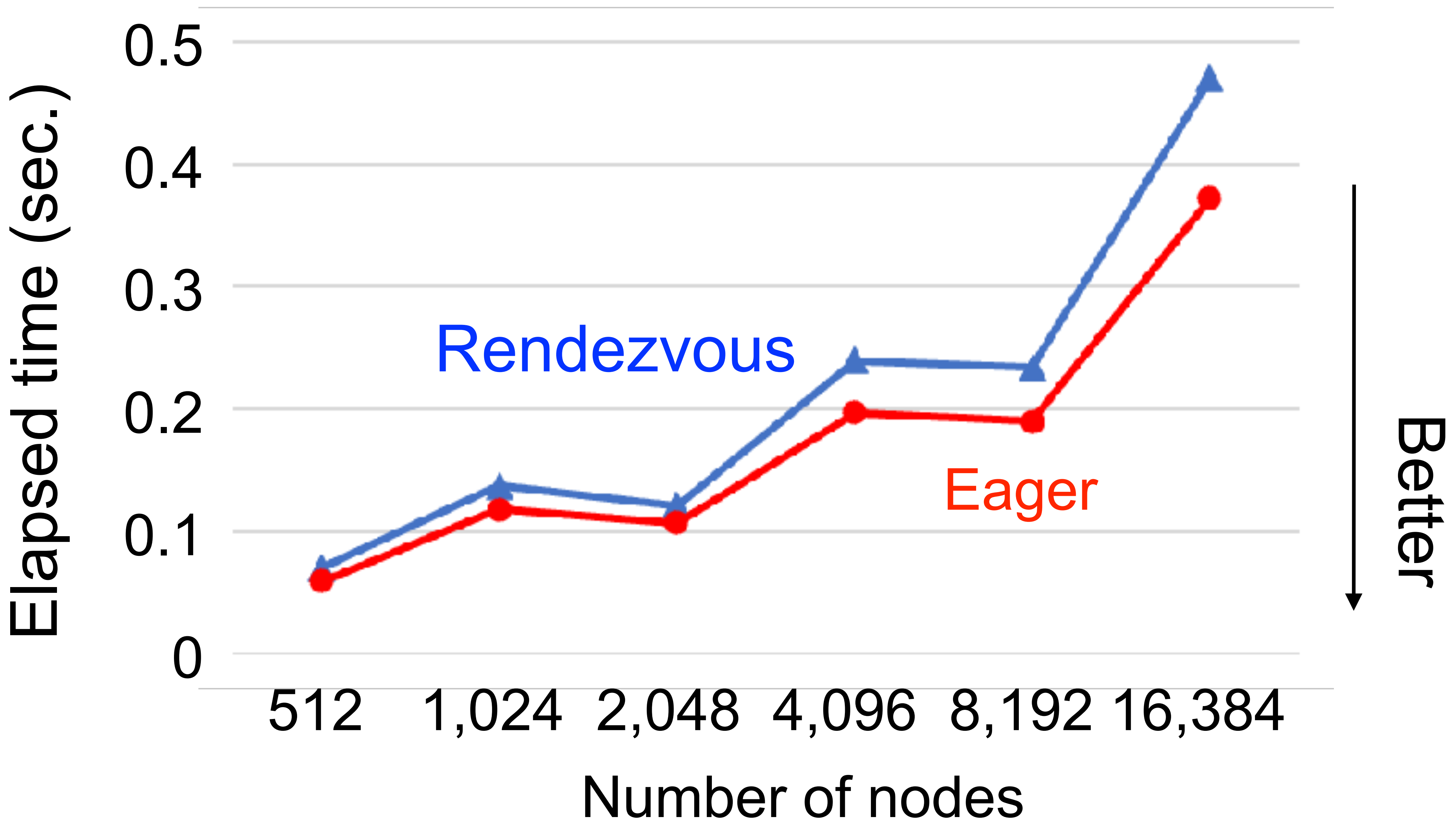
Rendezvous

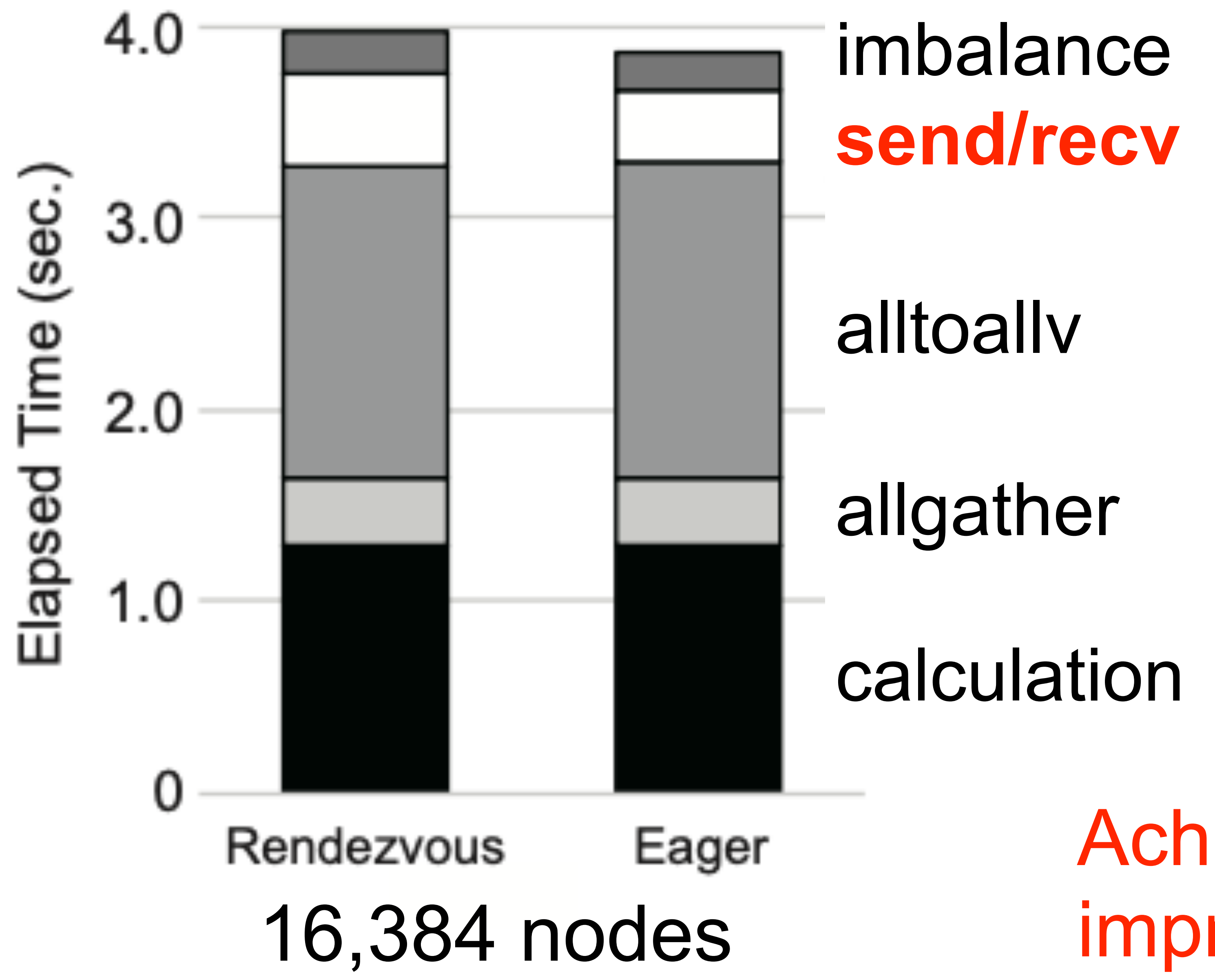


Synchronous communication to send/receive messages after both MPI processes are ready to communicate

Use of Eager method (2/2)

- In the default setting, Rendezvous was selected for all send/recv communication of BFS
- Fujitsu MPI library on Fugaku can set the threshold for switching between Eager and Rendezvous methods
  - We change the threshold to 512 Kbytes from default value to use Eager method
  - Since Fugaku's compute node has 32 Gbytes memory, the threshold is relatively small





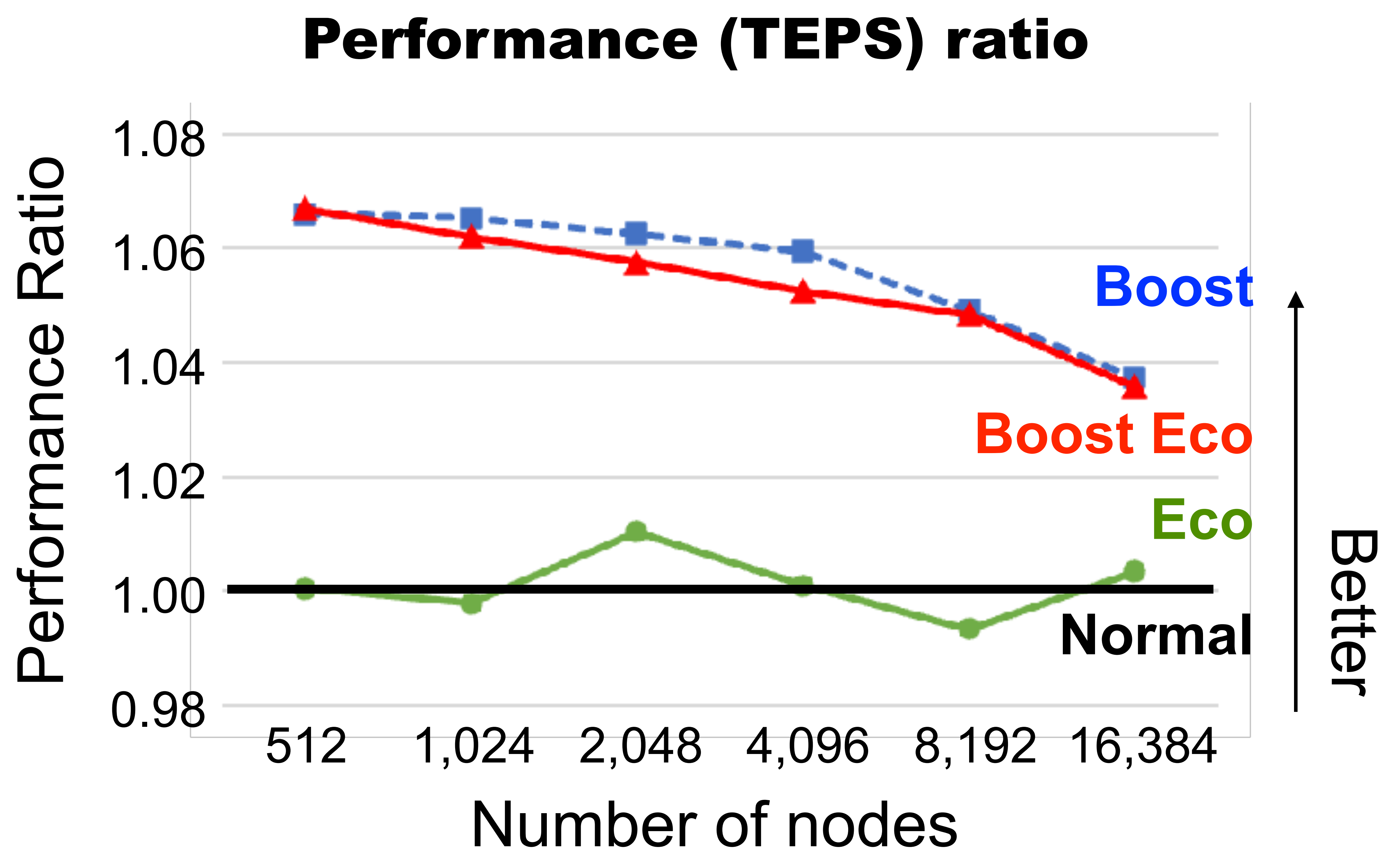
Achieved a performance improvement of 2.3%

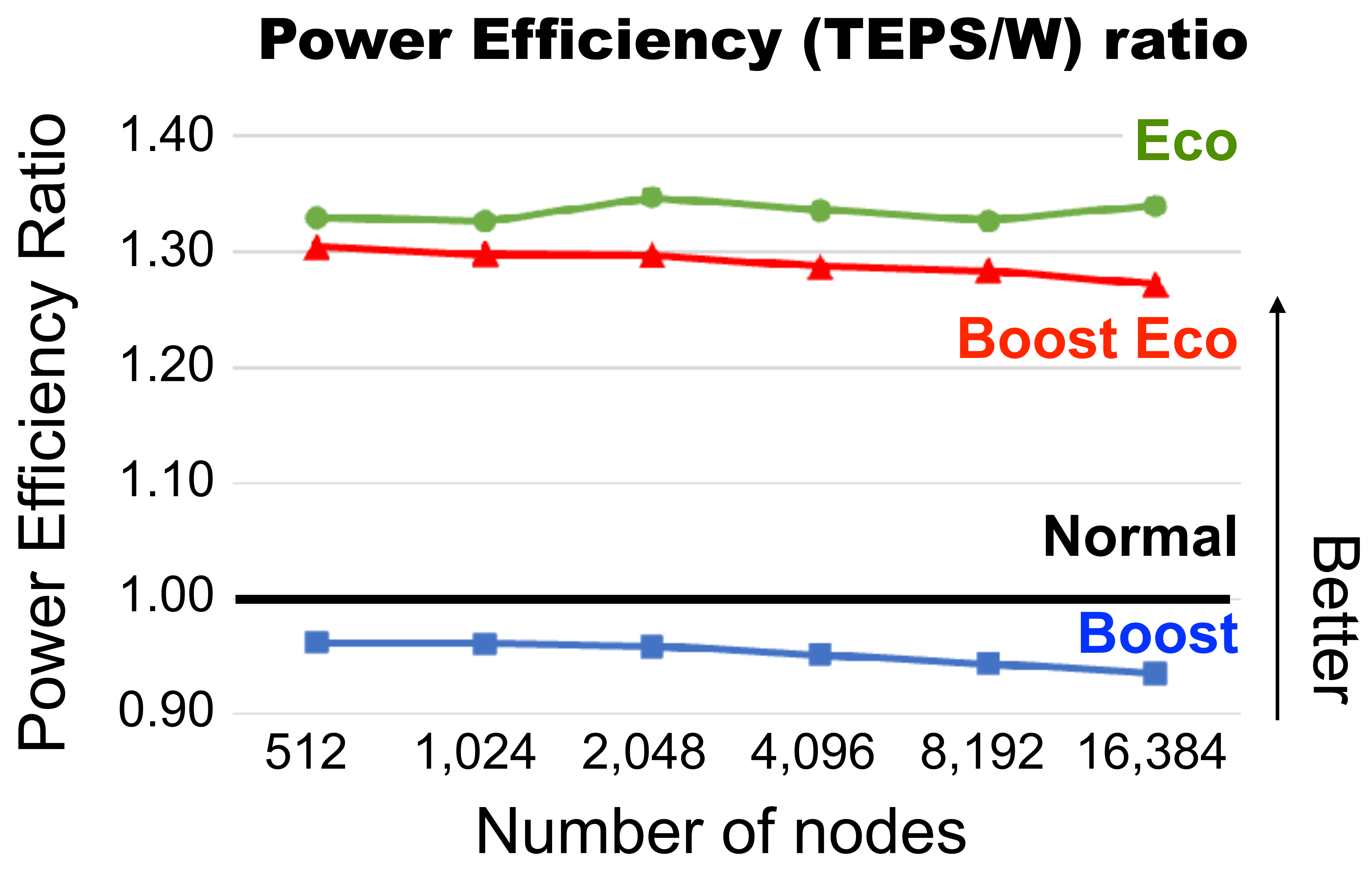
Power management (1/2)

- User can specify CPU frequency for each job
  - **Normal mode : 2.0 GHz**
  - **Boost mode : 2.2 GHz**
- **Eco mode :** Two floating-point arithmetic pipelines of A64FX are limited to **one**, and power control is performed according to the maximum power
  - Since BFS does not perform floating-point arithmetic, the use of Eco mode can be expected to reduce power consumption without affecting performance

- **Normal :** 2.0 GHz, **two** floating-point arithmetic pipelines (in previous evaluations)
- **Boost :** 2.2 GHz, **two** floating-point arithmetic pipelines
- **Normal Eco :** 2.0 GHz **one** floating-point arithmetic pipeline
- **Boost Eco :** 2.2 GHz **one** floating-point arithmetic pipeline

Power management (2/2)





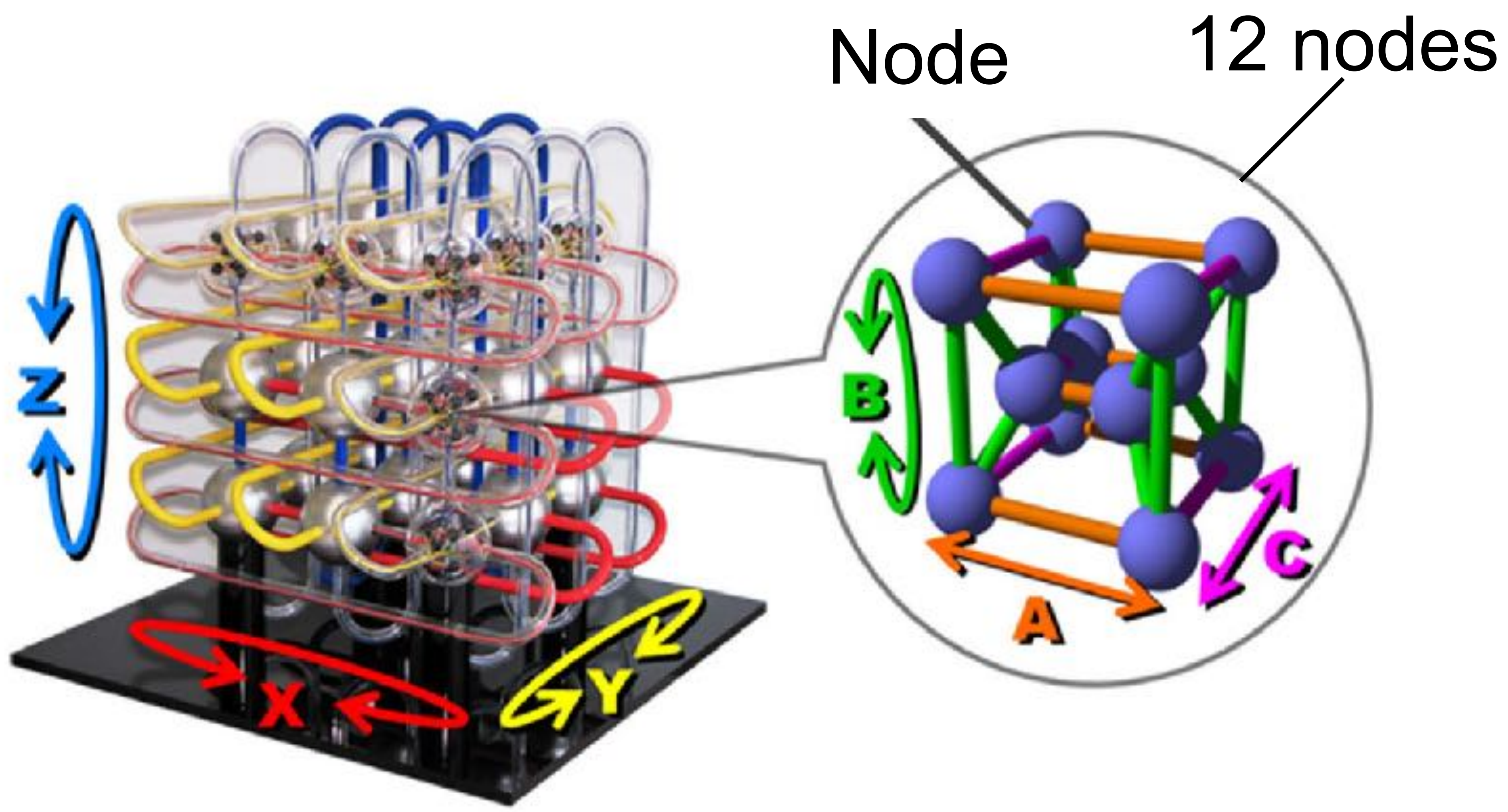
- **Boost Eco mode has a good balance between performance and power efficiency**
- The performance in Boost Eco mode is 3.6 % better than that in Normal mode
- The power efficiency in Boost Eco mode is 27.2 % better than that in Normal mode
- The results of Boost Eco mode for 16,384 nodes are 18,607 GTEPS and 1,408 kW

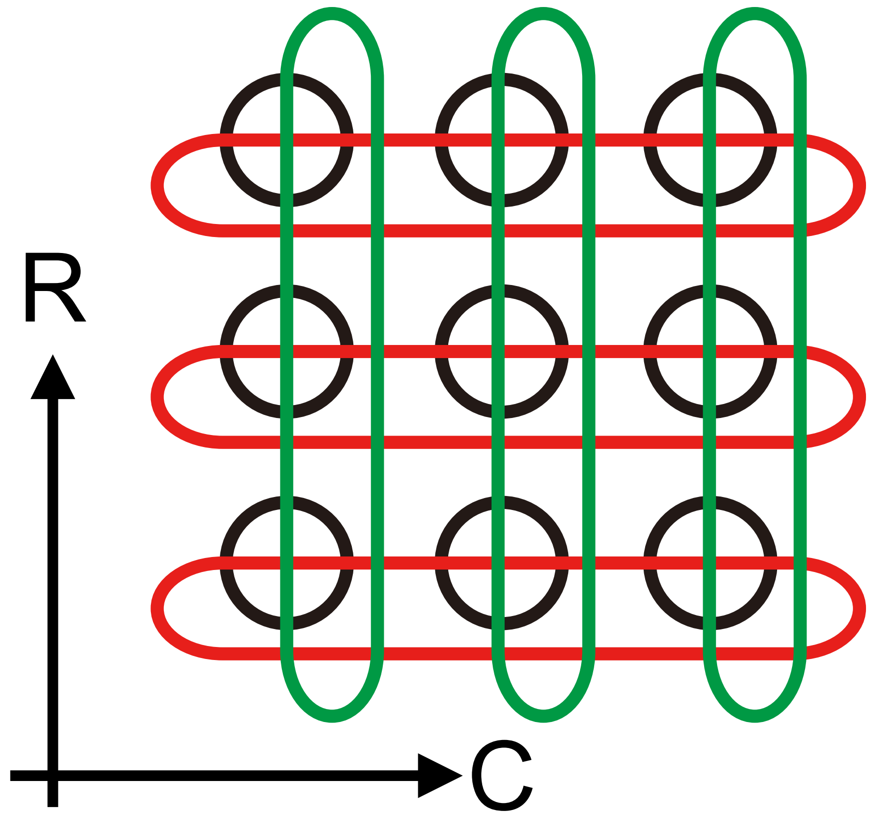
Outline

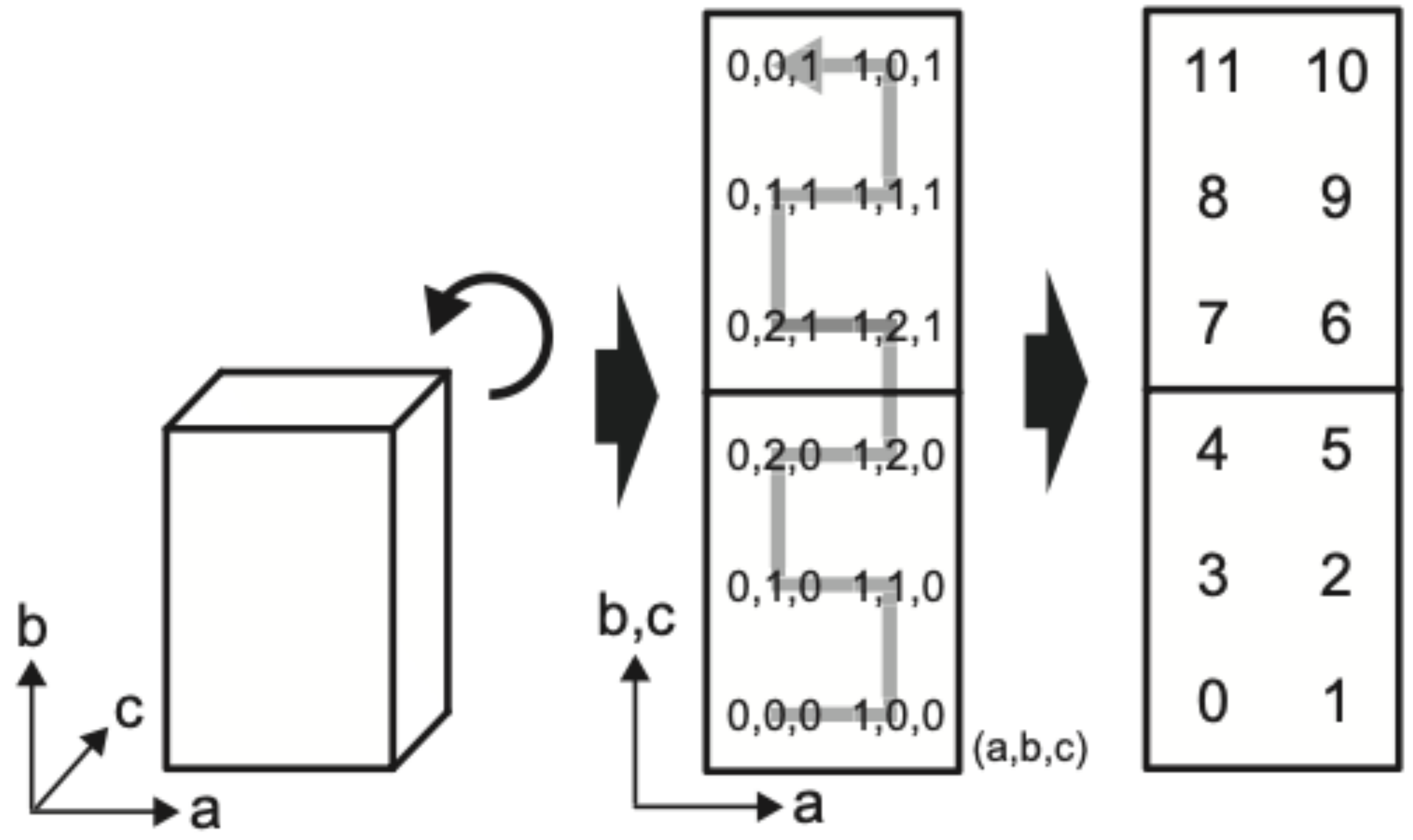
- BFS in Graph500 Benchmark
- Supercomputer Fugaku
- Tuning BFS on Supercomputer Fugaku
- Full node evaluation

Six-dimensional process mapping (1/2)

- The size of six axes in Fugaku network is (X, Y, Z, a, b, c) = (24, 23, 24, 2, 3, 2)
- It is desirable that the values of R and C process grid of BFS are close
- We assign the processes to (R, C) = (XY, Zabc) = (552, 288)
- Since neighborhood communication occurs in BFS, we assign the processes physically next to each other in row/column dimension



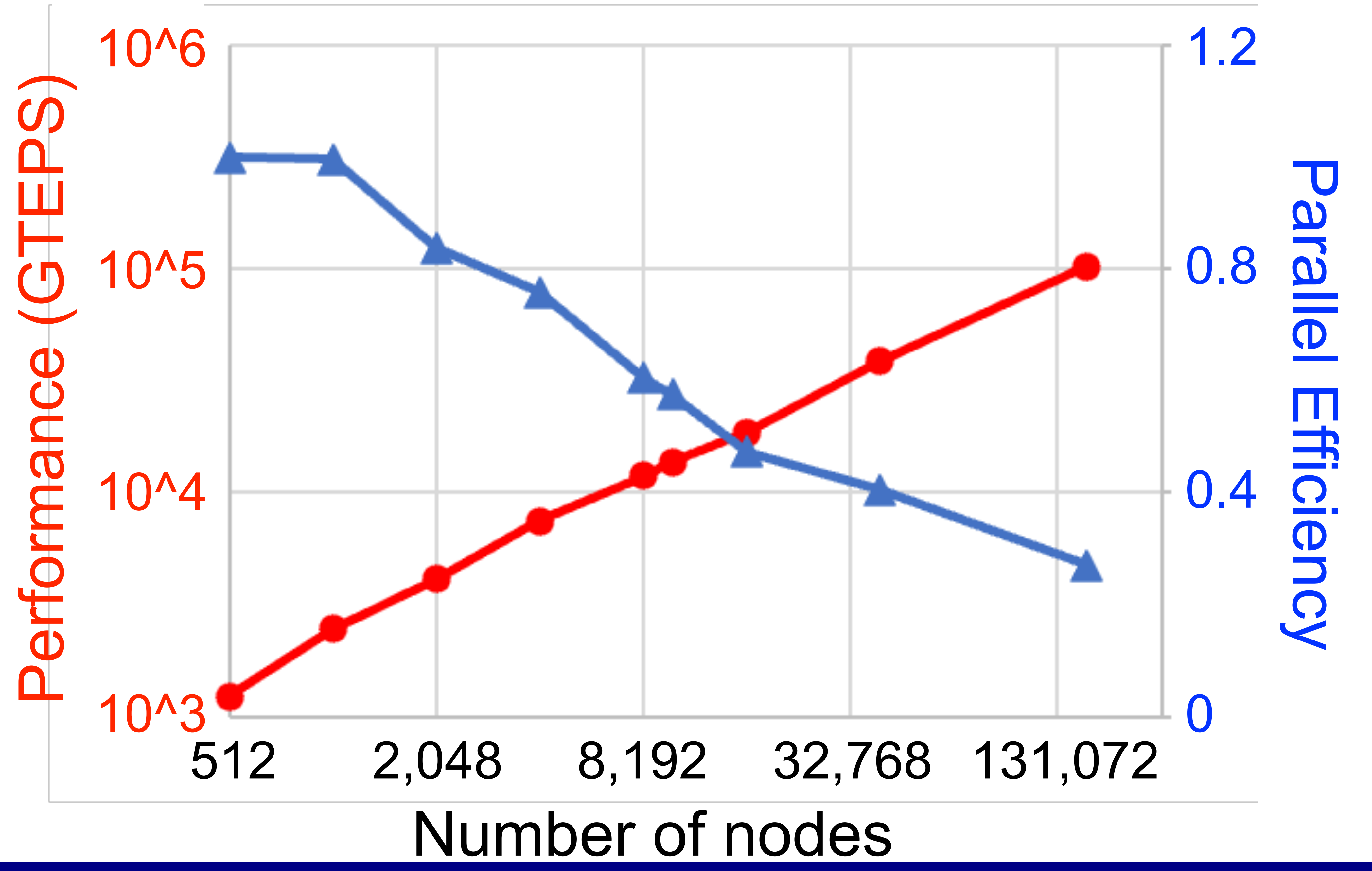


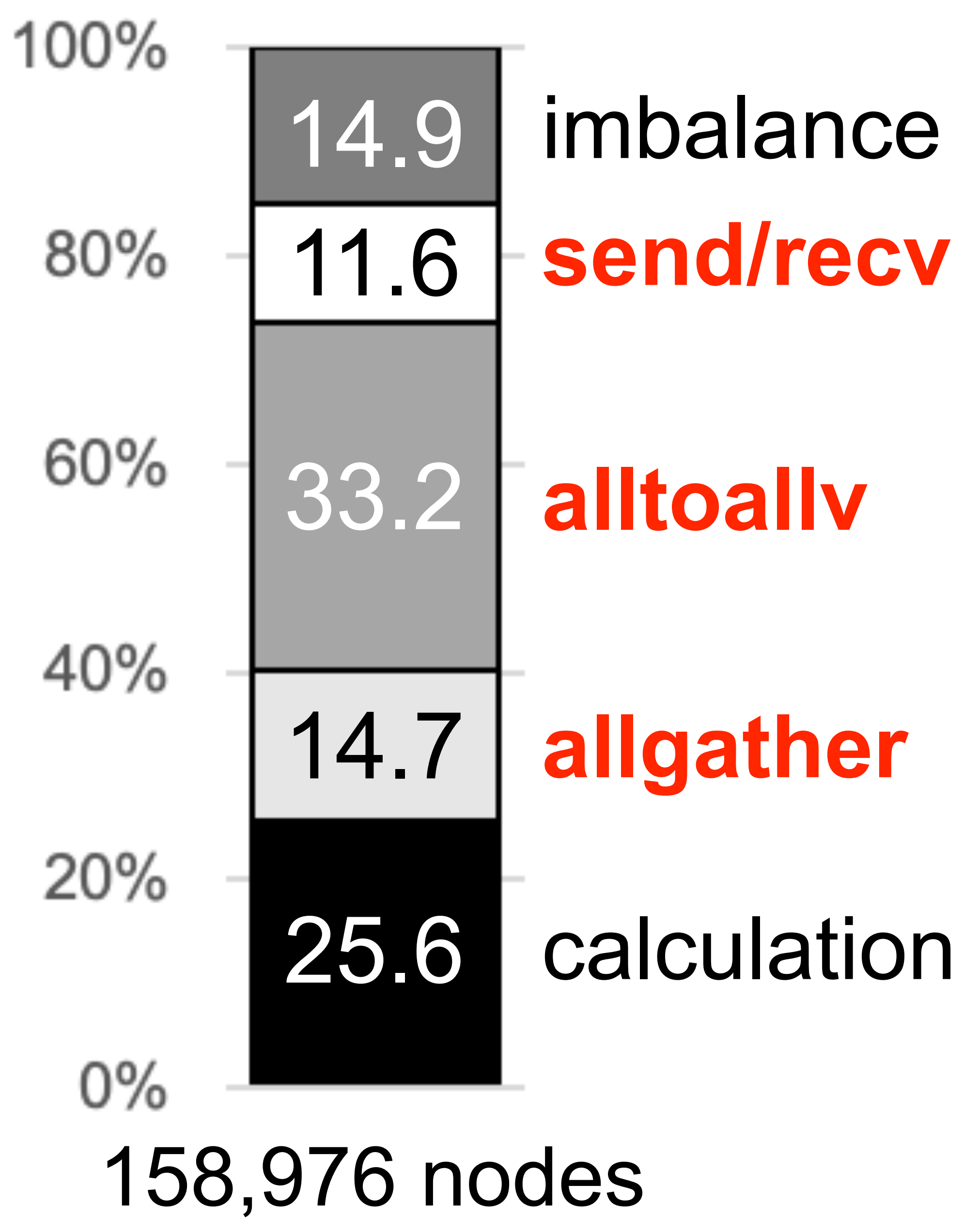


Six-dimensional process mapping (2/2)

- We evaluate the BFS performance for a large-scale graph consisting of about 2.2 trillion vertices and 35.2 trillion edges using the whole Fugaku system (158,976 nodes)
- Boost Eco mode
- **Performance: 102,955 GTEPS, Power: 14,961 kW, Efficiency: 6.9 MTEPS/W**
- Performance is 3.3 times that of the K computer (82,944 nodes), and power efficiency is 1.9 times that of IBM Sequoia (Blue Gene/Q)

The K computer did not measure power. At IBM Sequoia, the graph is the same size as Fugaku.





Summary

- Tune performance of BFS in Graph500 on Fugaku
- We evaluate the BFS performance for a large-scale graph consisting of about 2.2 trillion vertices and 35.2 trillion edges using the whole Fugaku system
- Achieve 102,955 GTEPS, resulting in the first position of Graph500 lists in 2020

- Future works
  - NUMA-aware optimization
  - Other graph algorithms (e.g. SSSP)