

# Performance of the JMA NWP models on the PC cluster TSUBAME.

K.Takenouchi<sup>1)</sup>, S.Yokoi<sup>1)</sup>, T.Hara<sup>1)</sup>\*, T.Aoki<sup>2)</sup>,  
C.Muroi<sup>1)</sup>, K.Aranami<sup>1)</sup>, K.Iwamura<sup>1)</sup>, Y.Aikawa<sup>1)</sup>

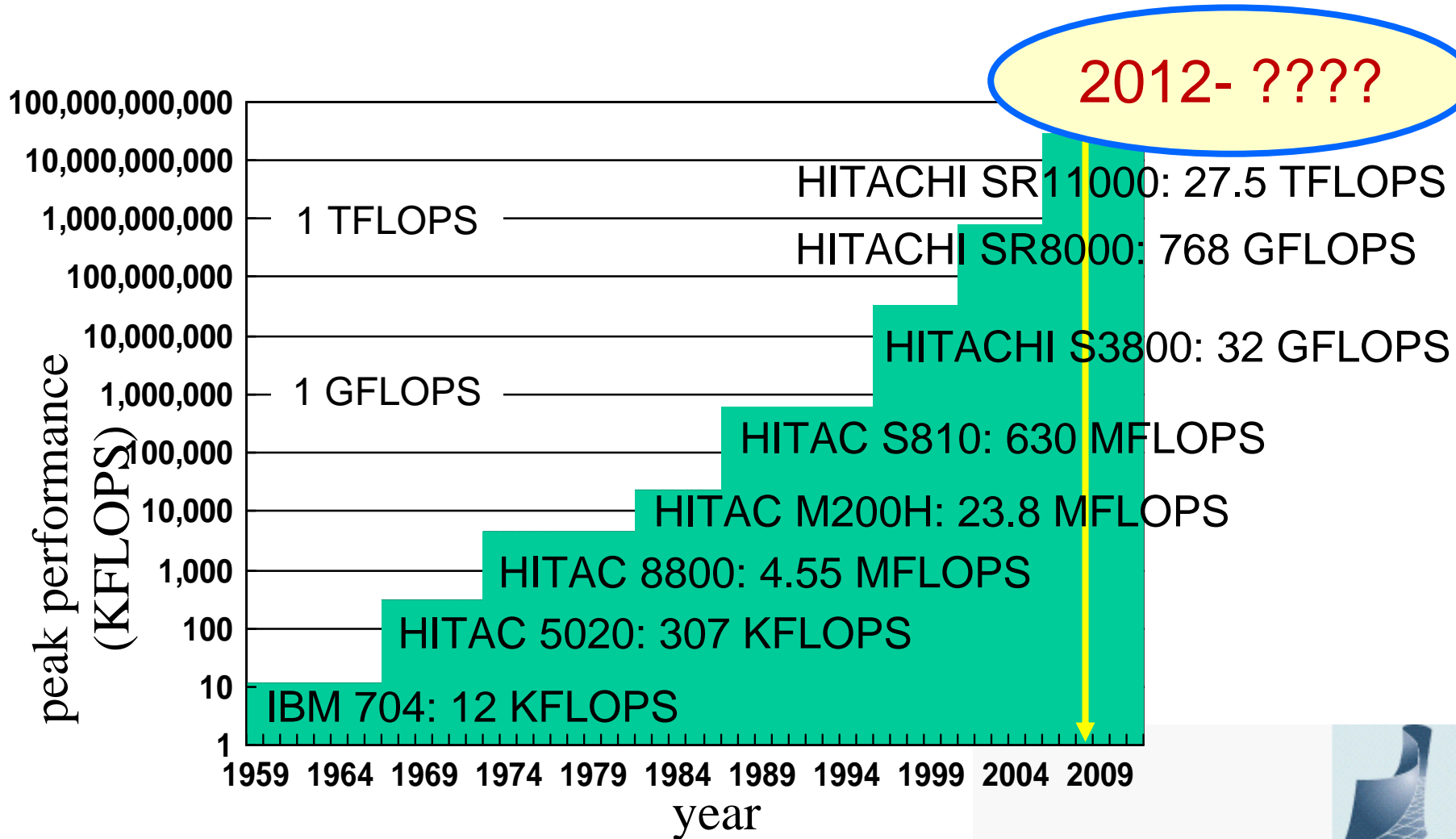
1) Japan Meteorological Agency (JMA)

2) Tokyo Institute of Technology

\* Presenter Today

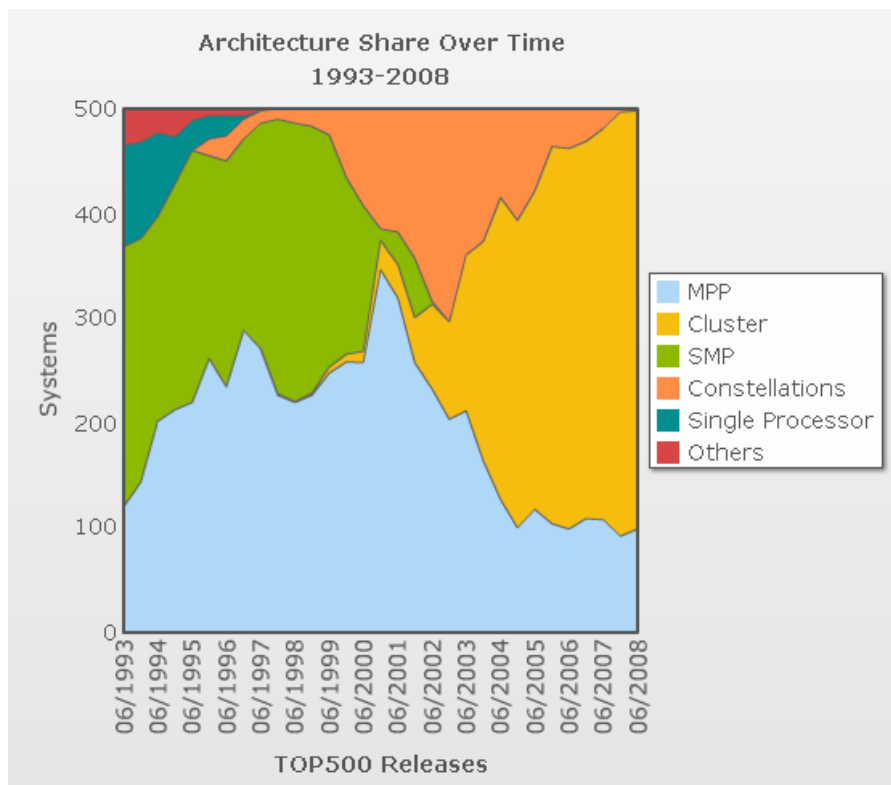


# History of Supercomputers at JMA





# Present and Future of Supercomputer



- In TOP500, since 2000 The proportion of Cluster has been rising remarkably.

In future, PC cluster or massively Parallel Scalar?

Reference:TOP500.Org

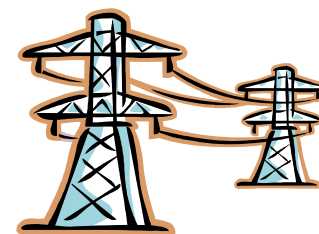
Can PC Cluster be a part of the next supercomputer system for NWP ?





# PC Cluster compared with current “Supercomputer”

- Advantages
  - Much less expensive!
  - Require much less electric power!



These are crucial for procurement as an operational computer.



# PC Cluster compared with current “Supercomputer”

- Disadvantages to be considered
  - Each processor is less powerful.
    - CPU processing performance (**Intel vs Power**)
    - Transfer speed from memory (**Memory Bandwidth**)
  - It should be compensated with more processors/nodes.
    - Excellent scalability is necessary.
      - It is meaningless if a huge number of processors are required to be comparable with “supercomputer” in regard to expense and electric power.
  - Poorness on
    - **Communication latency between nodes.**
    - Amount of memory



# TSUBAME

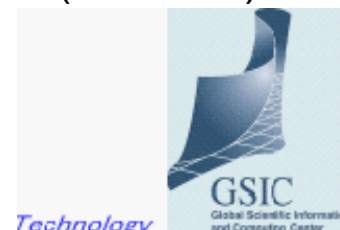
TSUBAME is the PC Cluster of Tokyo Institute of Technology.

- Absolutely different PC cluster from what used to be called “PC Cluster”.
  - It can be **comparable with “supercomputer”** on number of CPUs, peak performance, amount of memory per node.
  - “Infiniband” for transferring between nodes (**2GB/s**)

	CPUs/Node	Peak/Node	Memory/Node
IBM eServer (SDSC DataStar)	8, 32	48GF~217.6GF	16~128GB
Hitachi SR11000 (U-Tokyo, Hokkaido-U)	8, 16	60.8GF~135GF	32~64GB
Fujitsu PrimePower (Kyoto-U, Nagoya-U)	64~128	532.48GF~799GF	512GB
The Earth Simulator	16	128GF	16GB
<b>TSUBAME</b> (Tokyo Tech)	<b>16</b>	<b>76.8GF+ 96GF</b>	<b>32~128(new)GB</b>
IBM BG/L	2	5.6 GF	0.5~1GB
Typical PC Cluster	2~4	10~40GF	1~8GB



By courtesy of  
Prof.Matsuoka  
(TITECH)









# Our Questions

- Can we use the latest scalar or PC Cluster like TSUBAME as an operational NWP system (at least as a part of the system)?
  - If we can, it brings possibility of much cost reduction, and gives us much more freedom of choice to procure the system.
- What should we users do to use them effectively?
- What is necessary for PC clusters so that NWP models can be operated on them, especially in regard with hardware and software by venders?

To try to answer them, we ran JMA NWP models on TSUBAME, and investigate its performance referring to the current operational supercomputer SR11000.

# Profile of TSUBAME & SR11000

Institute and machine name	GSIC TSUBAME 	JMA SR11000 
		
Total Performance	85TFlops	27.5TFlops
Total nodes	655 nodes ( 16CPU/1node )	210 nodes ( 16CPU/1node )
Peak/Node	<b>96GFlops</b>	<b>135GFlops</b>
Memory	32GB/node	64GB/node
Memory Bandwidth	<b>6.4GB/s</b>	<b>15GB/s</b>
Speed between nodes	Twoway <b>2GB/s</b>	Twoway <b>16GB/s</b>
CPU	AMD Opteron 880(2.4GHz) 885(2.6GHz)	POWER5+ ( 2.1GHz )





# Single node performances

- JMA-NHM with 1 node, parallelized with MPI

Running Time: SR11000:TSUBAME = 1:4.6  
(normalized peak performance)

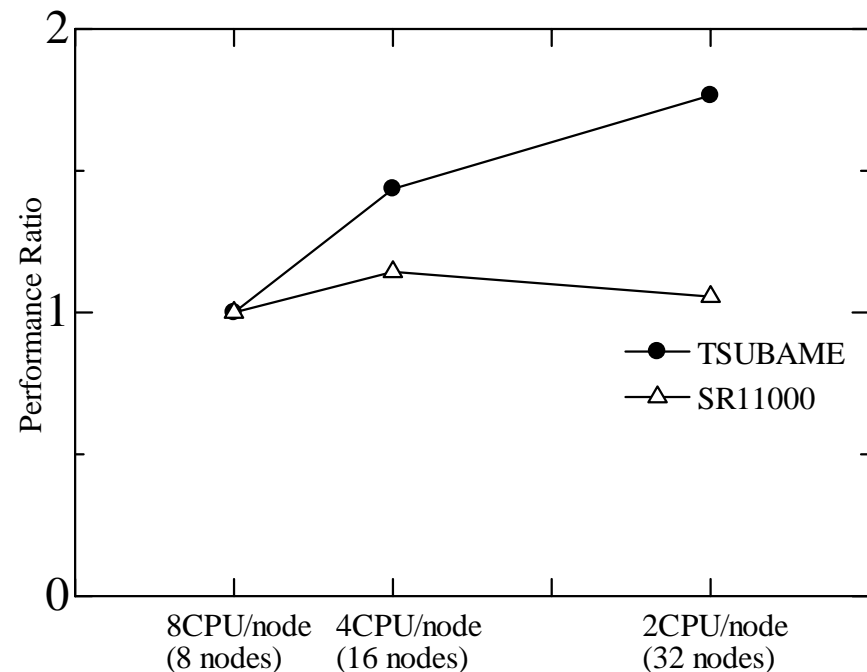
- The model is not tuned at all to be suitable for TSUBAME.
  - If tuned, the performance might get better.
- Major differences between SR11000 and TSUBAME

	TSUBAME	SR11K
Memory Bandwidth	6.4GB/s	15GB/s
Speed between nodes	2GB/s	16GB/s





# Influence of Memory Throughput



When the number of CPU per node is changed keeping total CPU number constant (64 CPUs), the smaller number of CPU per node, the better performance TSUBAME indicates.

On the other hand, performance of SR11000 is almost independent.



The memory throughput of TSUBAME doesn't look sufficient.

To secure better performance, memory throughput can be an essential factor to be considered in designing.



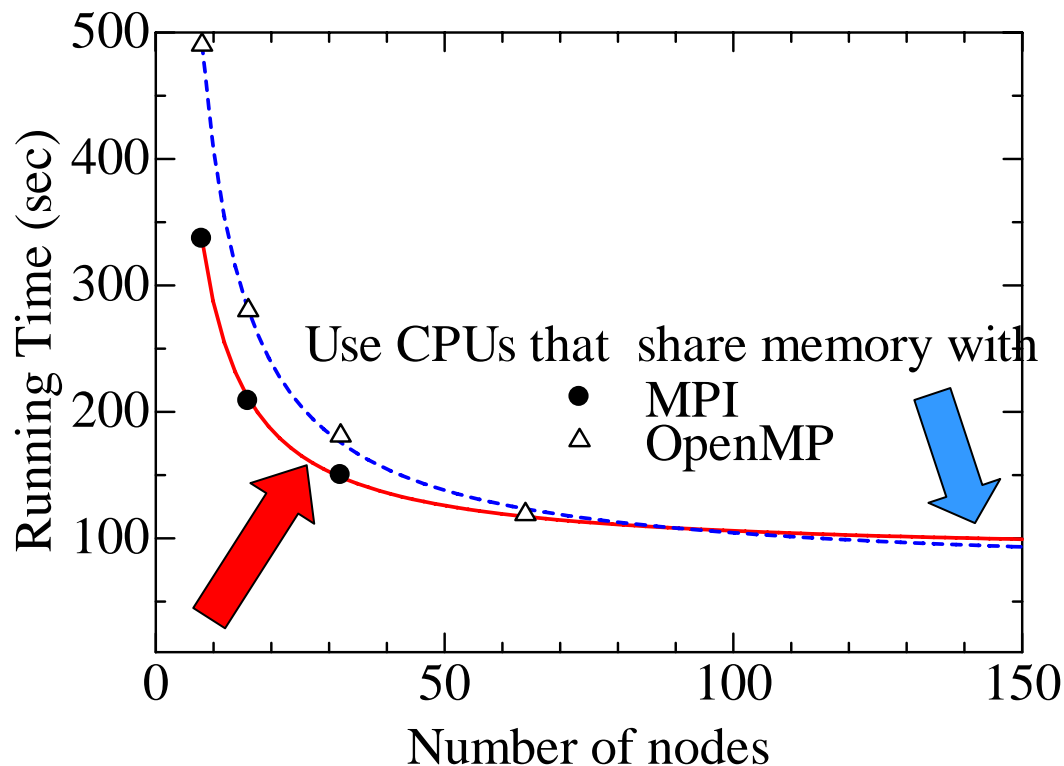
# To compensate the shortage of performance

- More nodes (CPUs) are required.
- Excellent scalability must be secured.
- Parallelization (for TSUBAME and SR11000)
  - Between nodes, MPI is regarded as de facto standard.
  - Between CPUs that share single memory
    - MPI as well
    - SMP (thread parallelization)
      - Automatic parallelization (only for SR11000)
      - OpenMP (with “Parallel code generator” by which MPI directives can be automatically inserted.)





# Which is better as parallelization of CPUs that share memory?



Fitted Amdahl's  $T = a + b/n$

( T: Running time,  
n: number of nodes )

- While “MPI” gives better performance with smaller number of nodes, increasing the number of MPI processes make it worse.
- “OpenMP” indicates so better scalability that it overturns “pure MPI” around 100 nodes on the running time.



# SMP is more advantageous with plenty of nodes

- With a larger number of nodes, it is more advantageous to apply SMP than MPI.
  - In what follows in this presentation, results with SMP (OpenMP) are presented.
- However, SMP has something to be improved:
  - As the parallel code generator inserting OpenMP directives tends to divide codes into small blocks to be parallelized, the cost to synchronize all threads in the end of the blocks cannot be neglected.
  - It is a thinkable way to regard each of whole physical process as a block to be parallelized.
    - To do that, some modifications of algorithms within the models are needed.



# Scalability depending on models

- JMA NWP models of which scalability are measured.

## GSM: Global Spectral Model (TL319L60)

- Spatial discretization: **Spectral method**
- Each node is required to communicate with the other **all nodes** for grid  $\leftrightarrow$  wave transform.

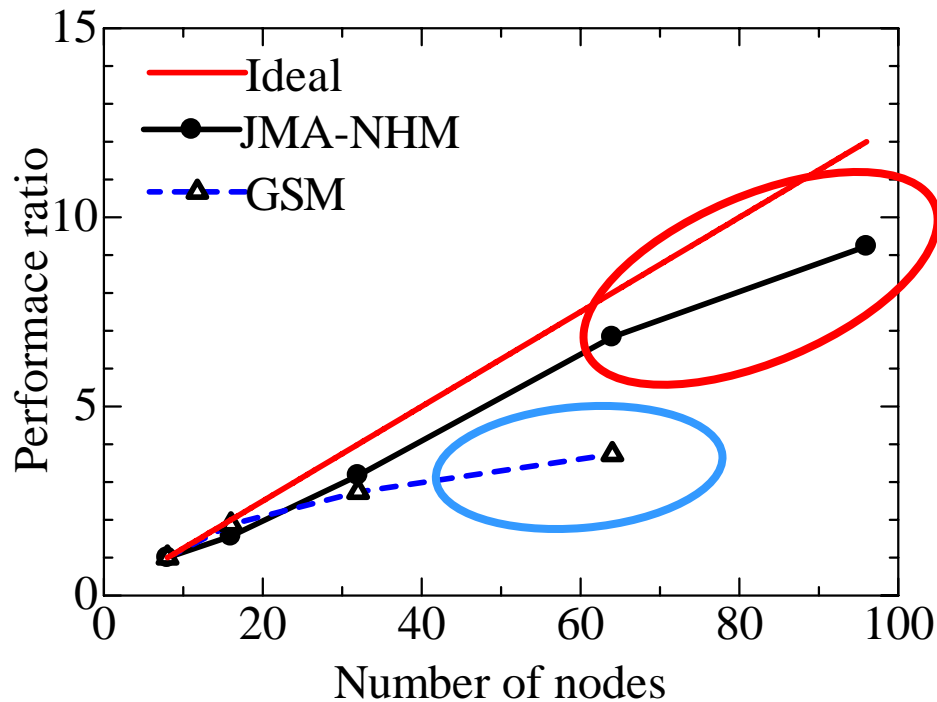
## JMA-NHM: Non-hydrostatic regional model (400x275L60)

- Spatial discretization: **Finite differential method**
- Each node is required to communicate with **only adjacent nodes**.

- This difference may reveal remarkable scalability contrasts between GSM and JMA-NHM.



# Scalability on TSUBAME

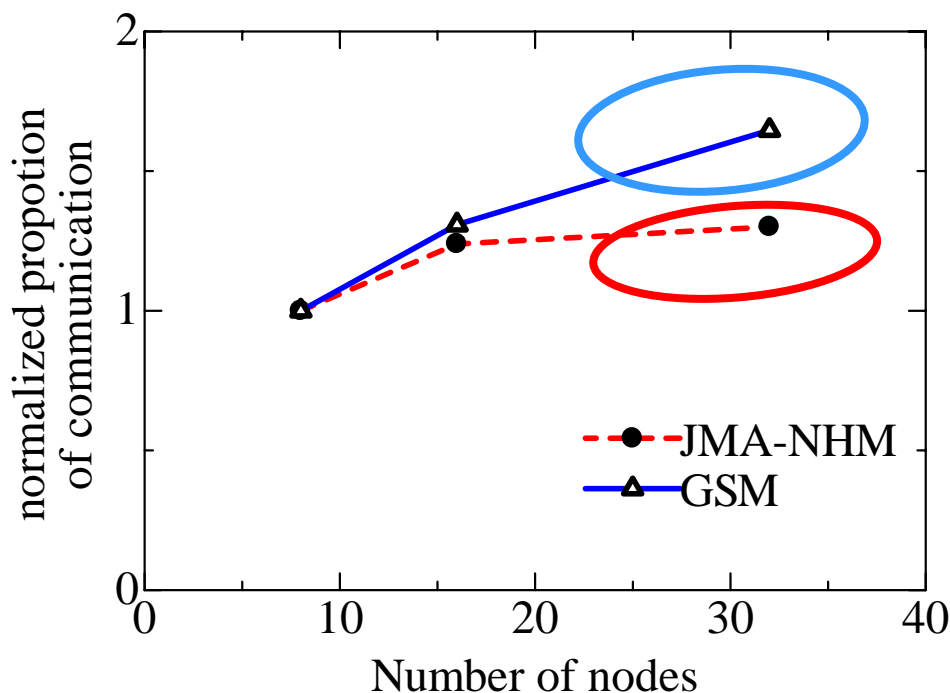


Normalized by the running time of 8 nodes.

The scalability of **JMA-NHM** is pretty good, while that of **GSM** gets almost saturated even with smaller nodes.



# Proportion of time of MPI comm.



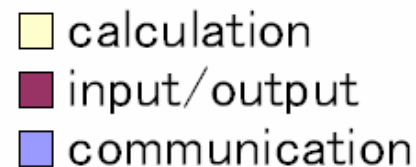
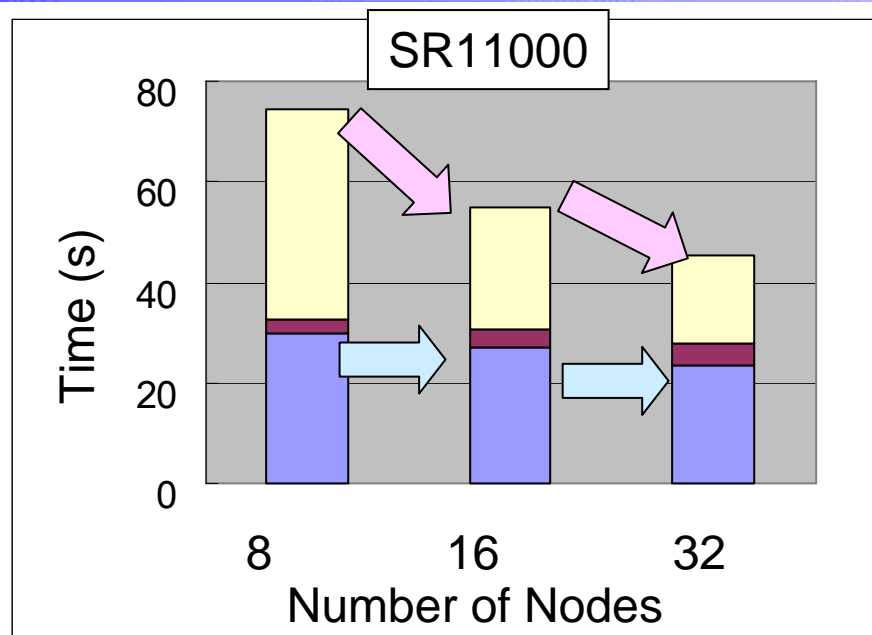
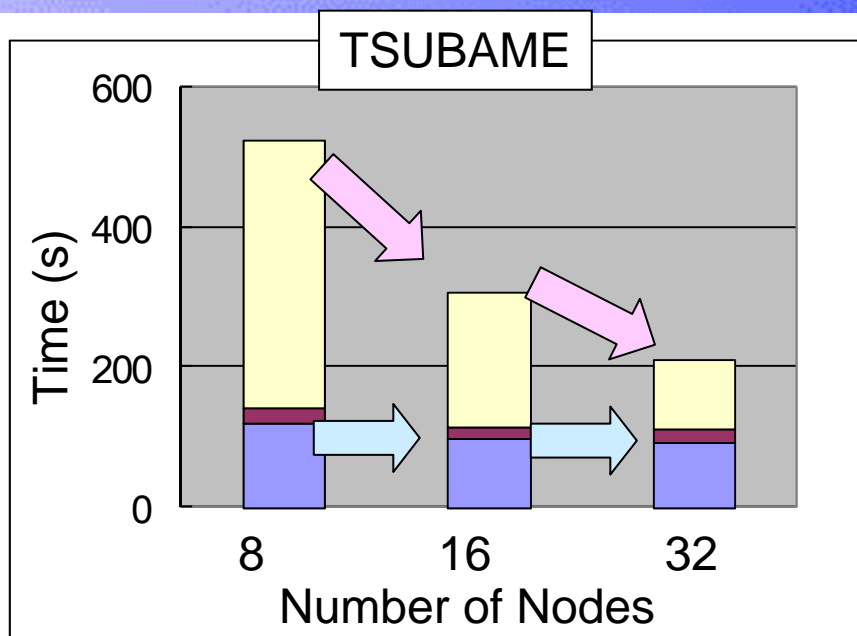
- Measured on SR11000
- Normalized by the proportion of MPI communication with 8 nodes.

- The proportion of running time occupied by MPI communication of **GSM** is more sharply increased than **JMA-NHM** with increasing number of nodes.
- That is one reason why the scalability of JMA-NHM is better.





# Dependency of running time of JMA-NHM on number of nodes



- The time for MPI communication is almost independent on number of nodes.
- It is because the cost of invoking MPI communication remains unchanged with number of nodes changed.
- To gain the performance, we should make an effort to reduce the time consumed by MPI communication.



# How to reduce MPI costs (1)

- Exclude global communications

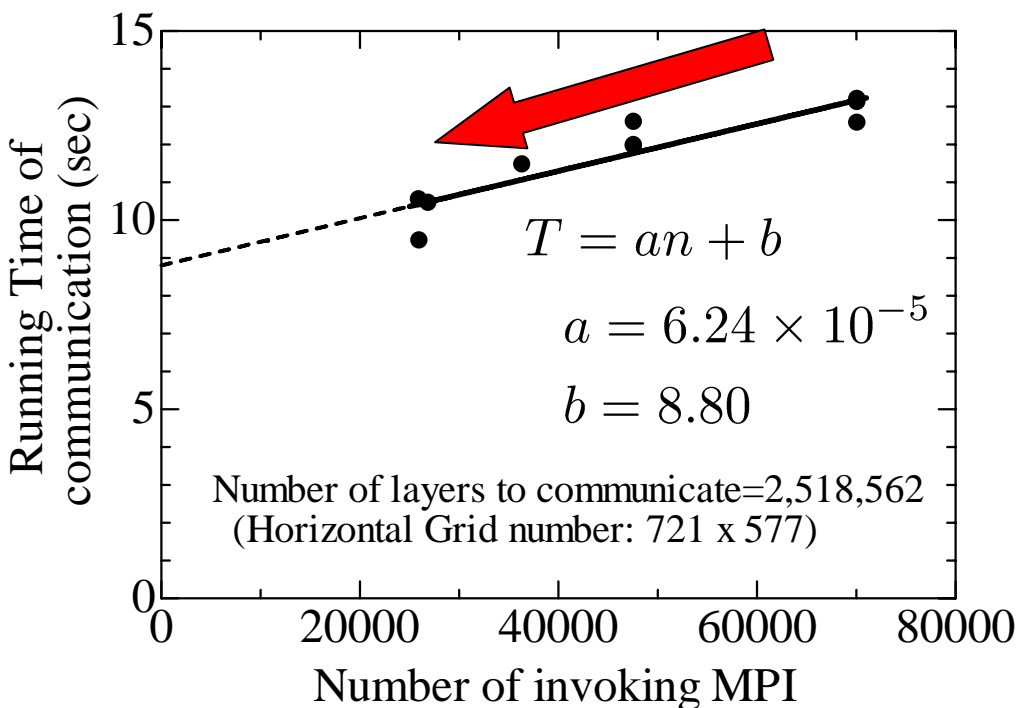
- Very expensive
- use only local exchanges (1:1, not 1: all, all:all)

- Reduce invoking MPI

- Cost of invoking MPI  $\sim O(1)$  (constant with # of nodes)
- Exchanges between MPI processes **should be gathered** as much as possible in one communication, which may require modifications of algorithms.



# How to reduce MPI costs (2)



- Measured the running time of communication in JMA-NHM, varying the number of invoking MPI keeping amount of communication constant on SR11000 with 64MPI.
- The cost of communication decreases with the number of invoking MPI decreased.

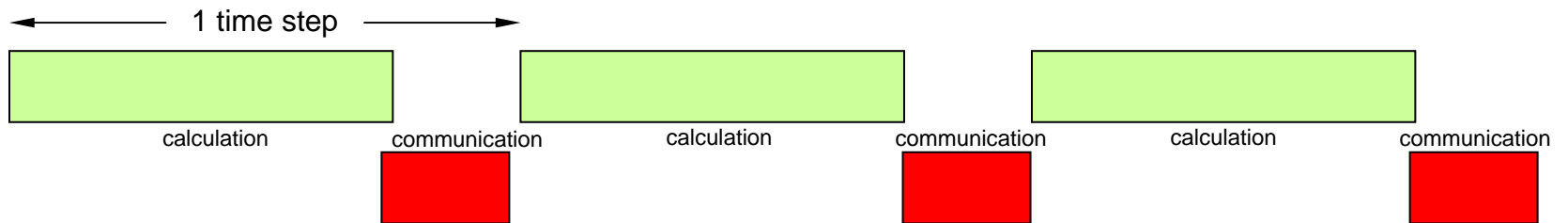
a: time to invoke one MPI process  
b: time to purely transfer the whole data (without invoking MPI)

To reduce time for communication, it is necessary for users

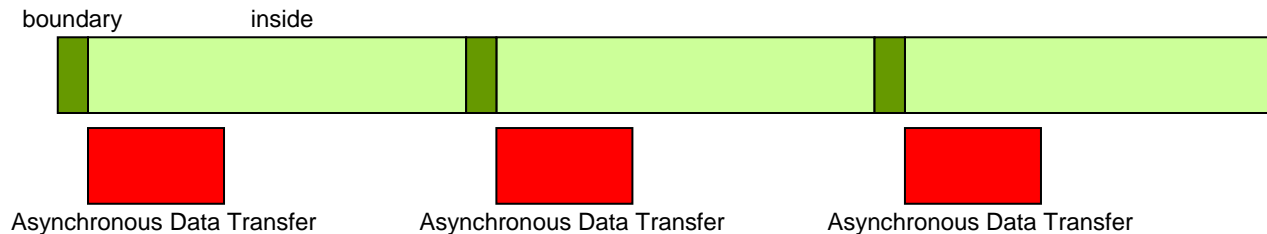
- to reduce the number of invoking MPI
- to reduce the amount of data to transfer between MPI processes (associated to b)



# How to reduce MPI costs (3)



## New Scheduling (Ideal)



- Only values on halos are exchanged with communication.
- Grid points near boundary associated to halos are calculated first, and calculation for inside region and asynchronous data transfer are executed simultaneously.



# Conclusions (1)

- TSUBAME has led us to the new generation of scalar computers and PC clusters.
  - **TSUBAME has comparable specifications** with the current supercomputers.
  - Its lessons affect the project T2K or Next-Generation Computer project in Japan, and it could be main streams of computers in the future.
  - NWP centers like JMA should be interested in them, and look for the ways to use them effectively.



## Conclusions (2)

- The latest PC clusters could be employed in NWP systems, but with some limitations.
  - Tunings to be suitable for scalars are required, of course.
  - The most problematic bottleneck is **communication between nodes**.
    - Less appropriate for models with **spectral scheme** or **fully implicit scheme**.
    - To reduce the cost, algorithm modifications are essential such as
      - **Reducing invoking MPI communications**
      - **Concealment of communications with asynchronous communication.**



# Conclusions (3)

- Not only us **users**, but also **venders** might have something to do for the operation of NWP models.
- We would like to eagerly expect further improvements, especially on
  - throughput of memory
  - latency of MPI communication**by venders.**