

# Understanding and Improving Storage Accesses at the K computer

Yuichi Tsujita
RIKEN AICS

UIOP Workshop @DKRZ, Mar. 23, 2017

# Outline

- Overview of the K computer and its file systems
- Activities for high availability and performance
  - Alleviation of MDS load using loop-back file systems
  - Elimination of client evicts
  - Optimization for alleviating interference by huge data accesses
- Future plan (and hopes)

# Overview of the K computer and its file systems

# System configuration of the K computer

40 m x 40 m

**Full System**

**Compute Rack × 864**


**10.6(11.3)PFLOPS**
**1.27(1.34)PiB**

4000mm x 800mm

**2 Cabinets**

**Compute Rack × 4**

**Disk Racks × 1**

**49.2(52.4)TFLOPS**
**6.00(6.38)TiB**

800mm x 800mm

**Compute Rack**

**SB × 24**

**IOSB × 6**

**12.3(13.1)TFLOPS**
**1.50(1.59)TiB**

500mm x 500mm

**System Board(SB)**

**Node × 4**

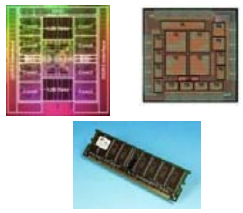
**512GFLOPS**
**64GiB**

**Node**

**CPU × 1**

**ICC × 1**

**memory**

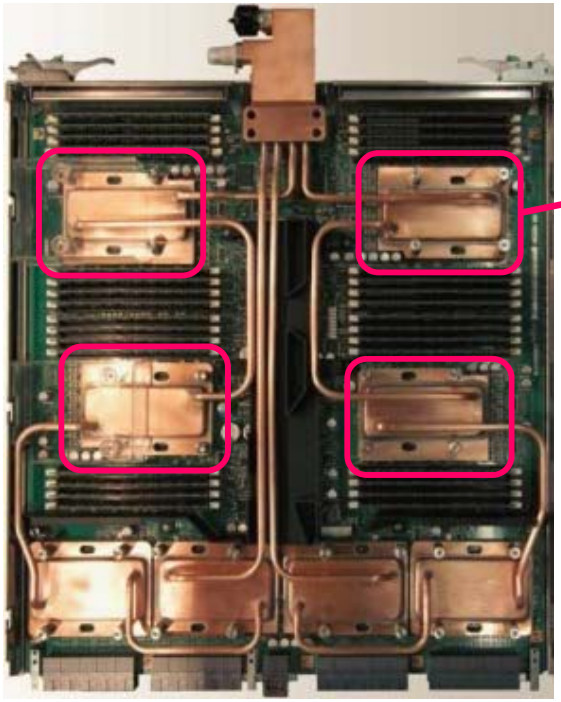


**128GFLOPS**
**16GiB**

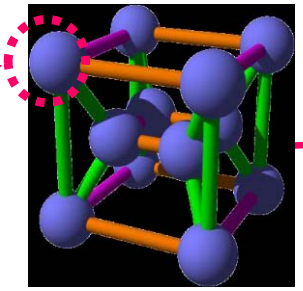
( )included IO node performance and memory capacity

# Tofu Interconnect

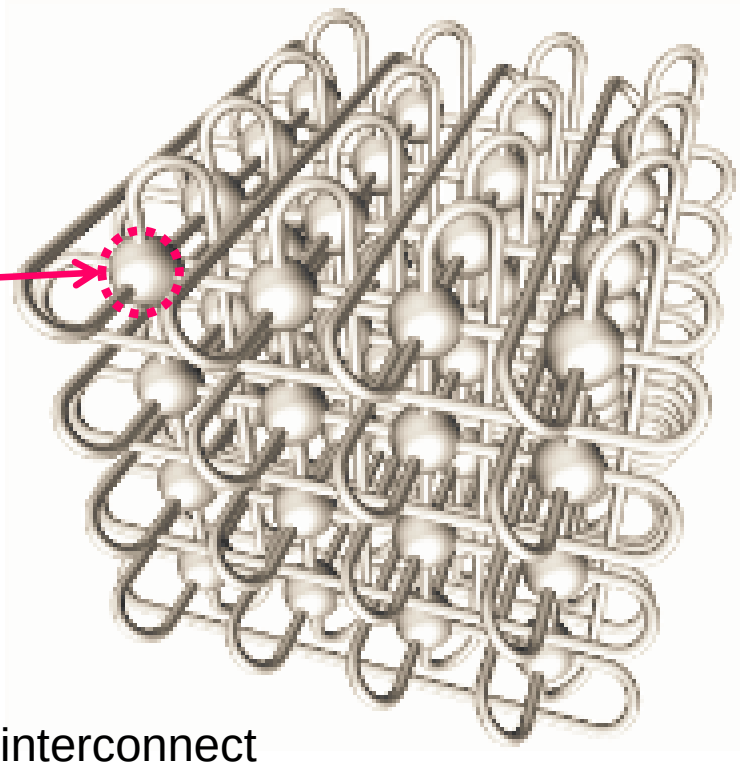
- Compute nodes with the Tofu interconnect
  - Tofu : Torus Fusion



1 system board
(4 compute nodes)



1 Tofu-unit
(2x3x2 nodes)

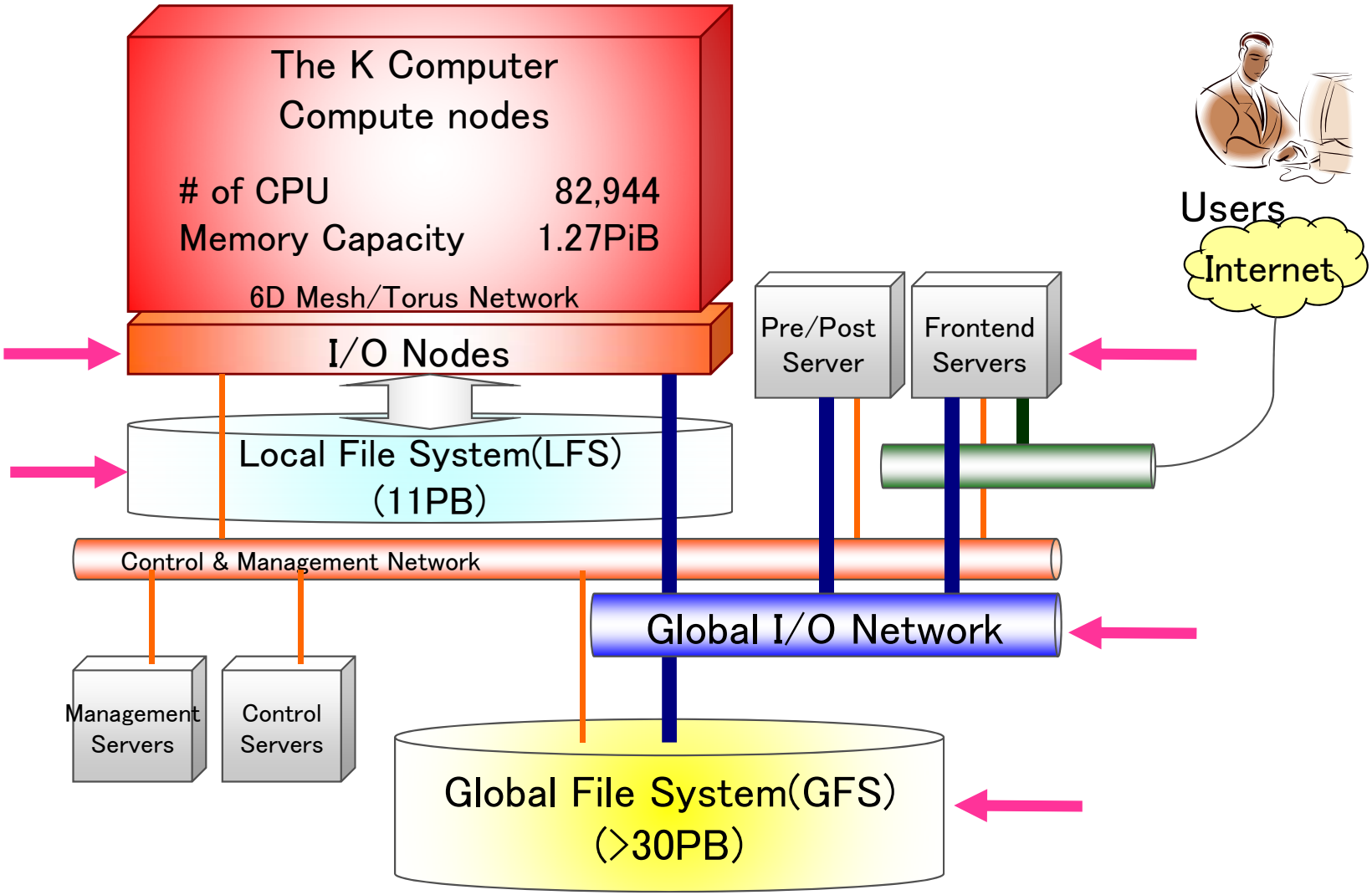


6D mesh/torus using the Tofu interconnect

・Axis : X, Y, Z, a, b, c
・X,Z,b : torus (Z=0: I/O node), Y, a, c : Mesh
・Network size : (X, Y, Z, a, b, c) = (24, 18, 17, 2, 3, 2)

AICS 京 K computer

# Overview of the K computer




FEFS is used for both LFS and GFS.
(FEFS: Fujitsu Exabyte File System based on Lustre technology)

# File system at the K computer

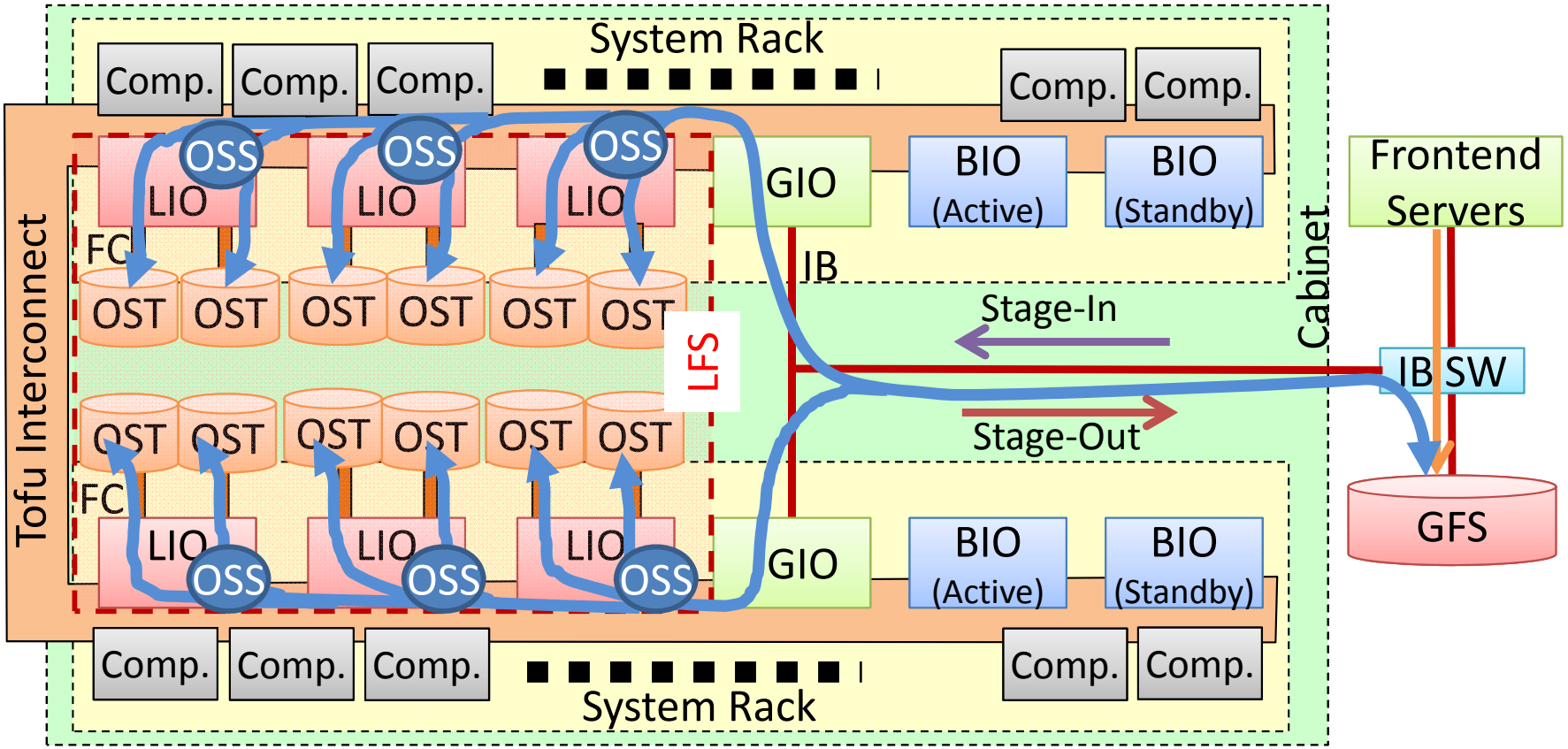
- Organization of file systems at the K computer
  - LFS : Performance oriented
    - for high performance I/O during computation
  - GFS : Capacity oriented
    - for huge data storing and high redundancy

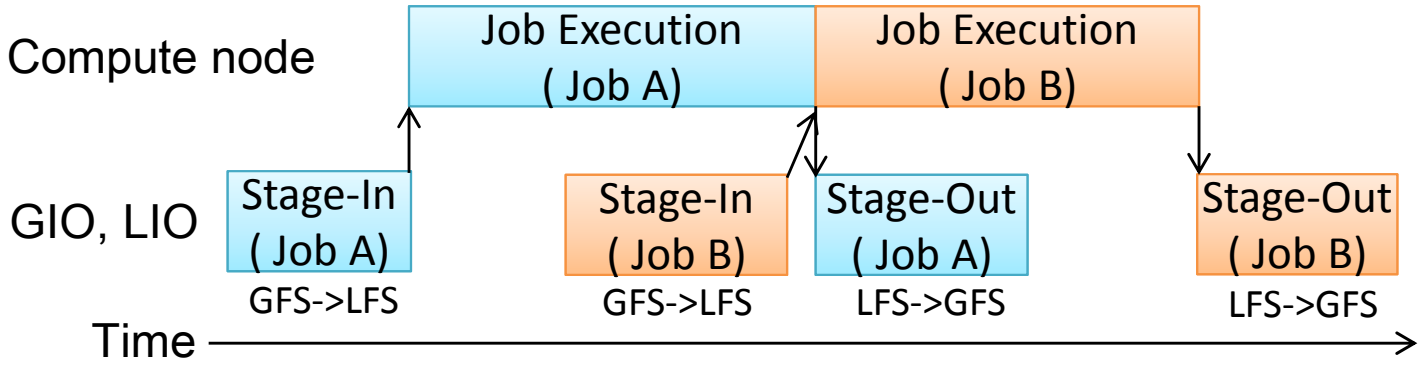
| File system | LFS | GFS * |
|---|---|---|
| Total volume size | ~ 11 PB | > 30 PB |
| # volumes | 1 | 8 |
| # OSSs | 2,592 | 90 |
| # OSTs | 5,184 | 2,880 |
| Disk system of OST | RAID5+0 | RAID6 |

* GFS configuration will be changed due to some upgrades later.

# Data-staging

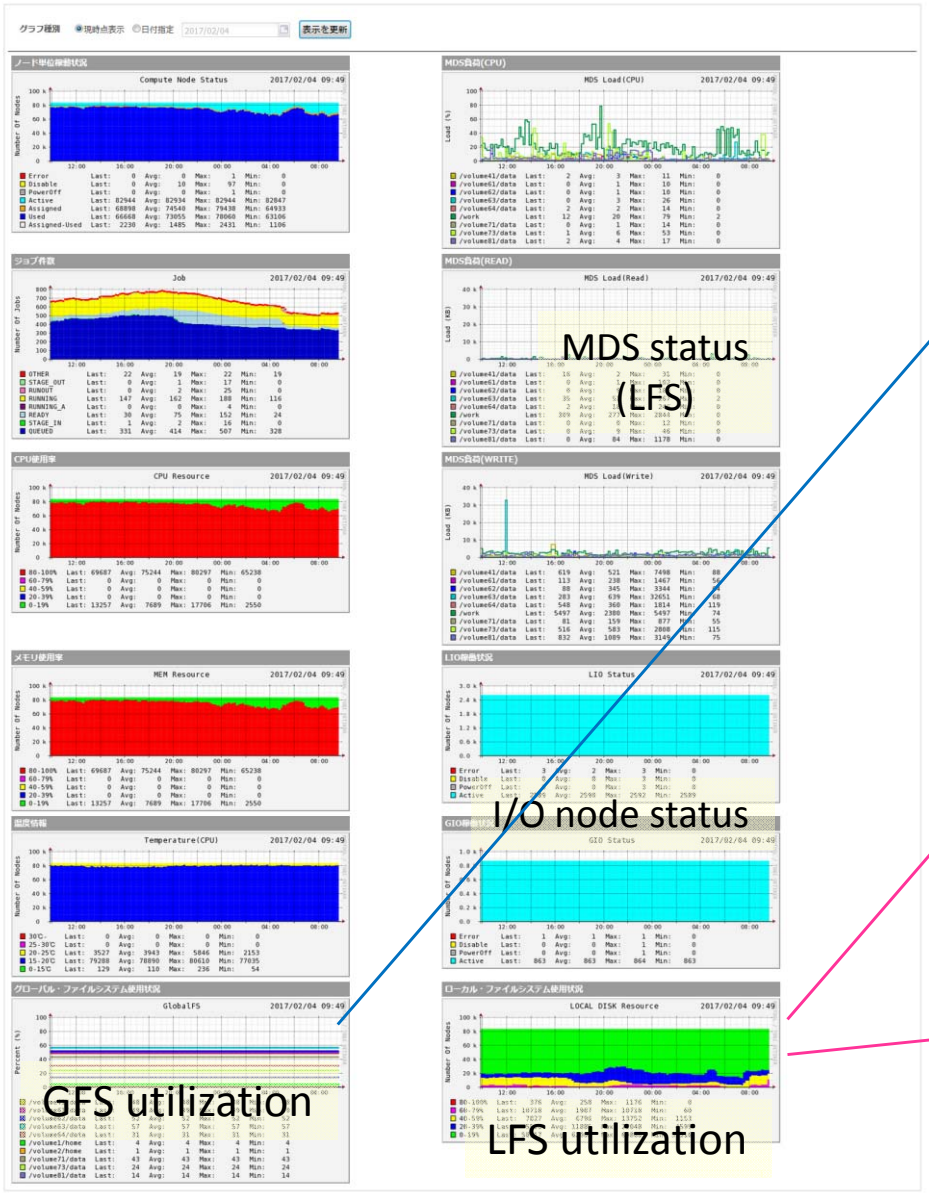
- Asynchronous data staging for high efficient job scheduling

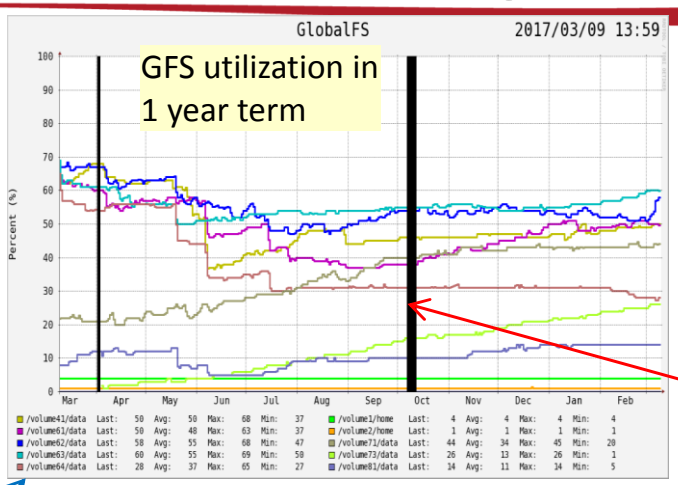


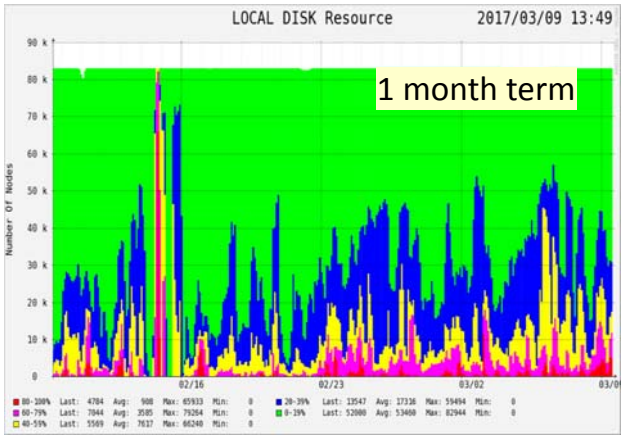
# K computer monitoring

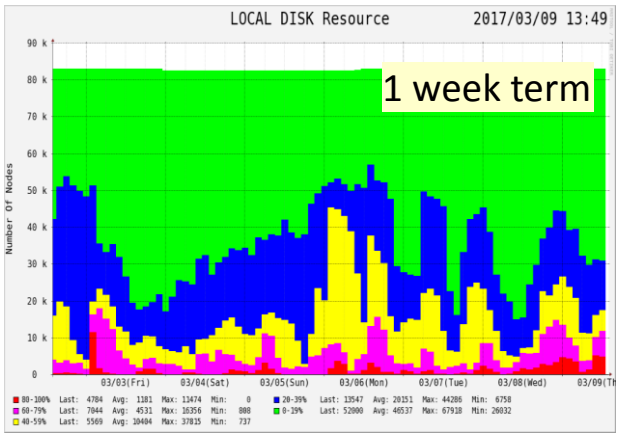
- Administrator's portal




system shutdown for maintenance

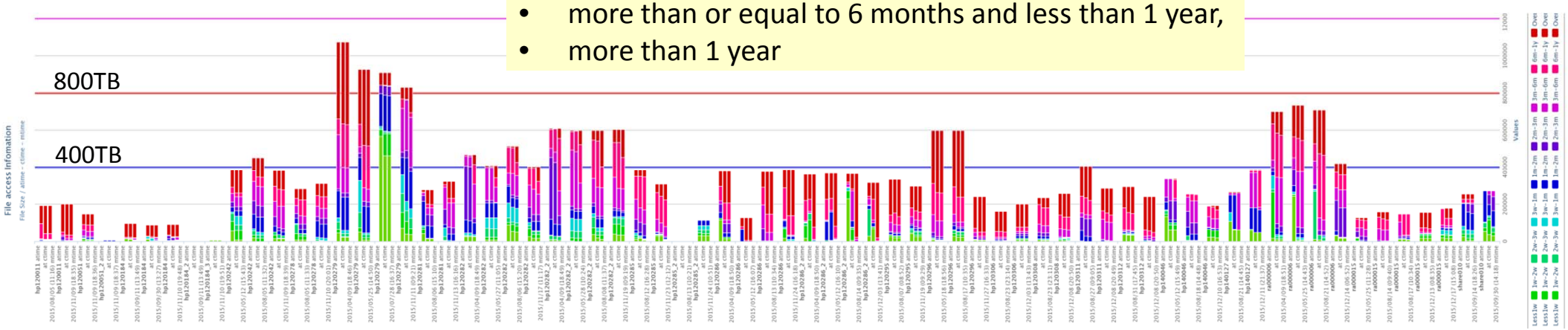


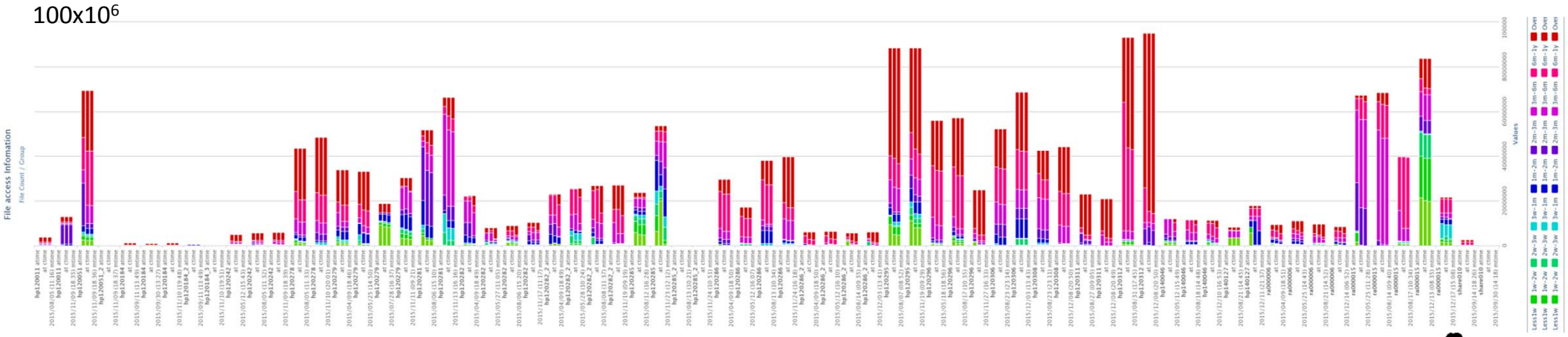
# Timestamp monitoring at GFS volumes

- Analysis of time spent after the final updated times (atime, ctime, mtime) among major user groups regarding
  1. file size, and
  2. the number of files

<timestamp grouping>
- within 1 week,
- more than or equal to 1 week and less than 2 weeks,
- ...
- more than or equal to 6 months and less than 1 year,
- more than 1 year




Amount of file sizes with timestamp grouping in each major user group



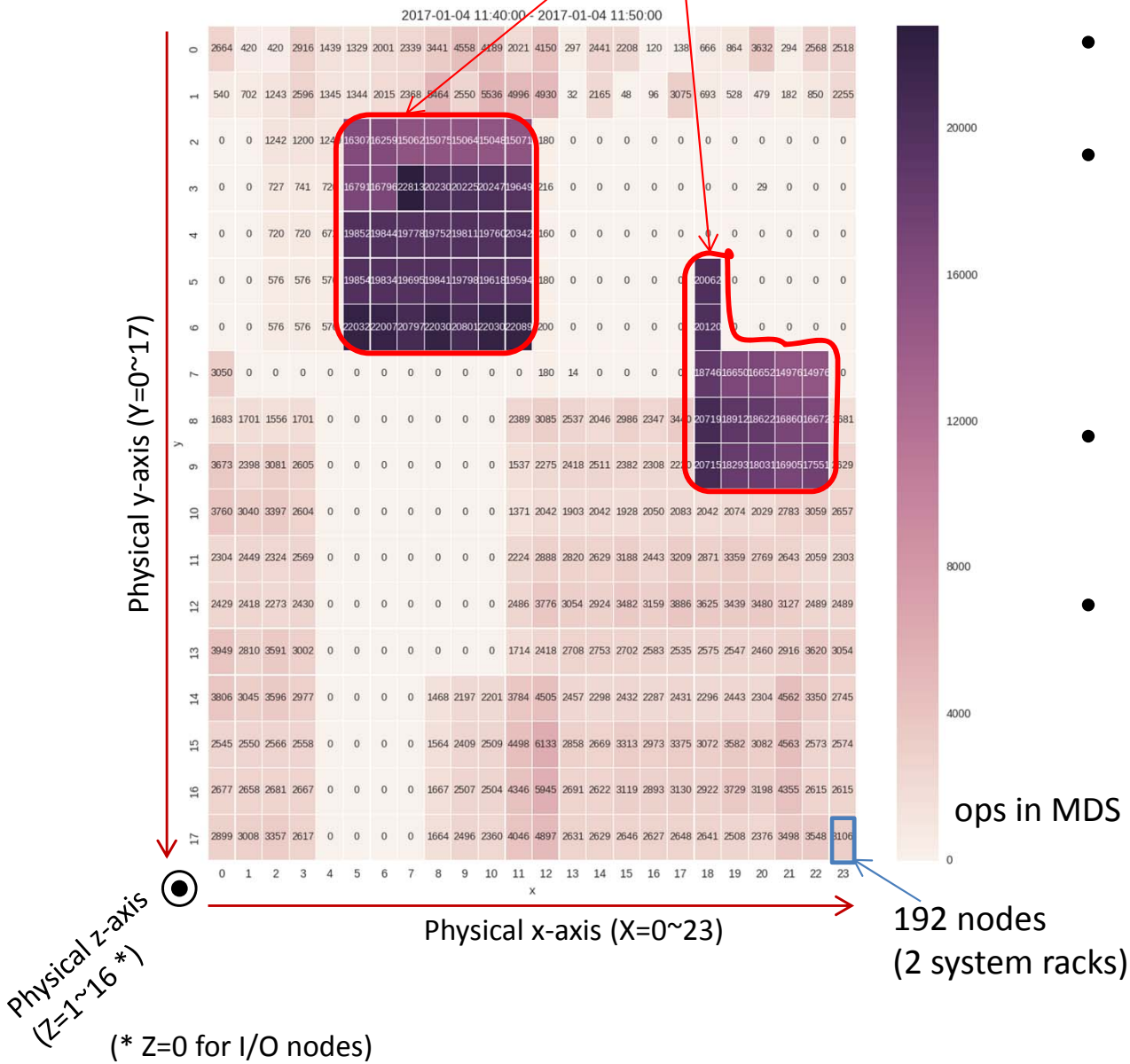

Amount of the number of files with timestamp grouping in each major user group

# Activities for high availability and performance

- Alleviation of MDS load using loop-back file systems
- Elimination of client evicts
- Optimization for alleviating interference by huge data accesses

# High load of MDS (LFS)

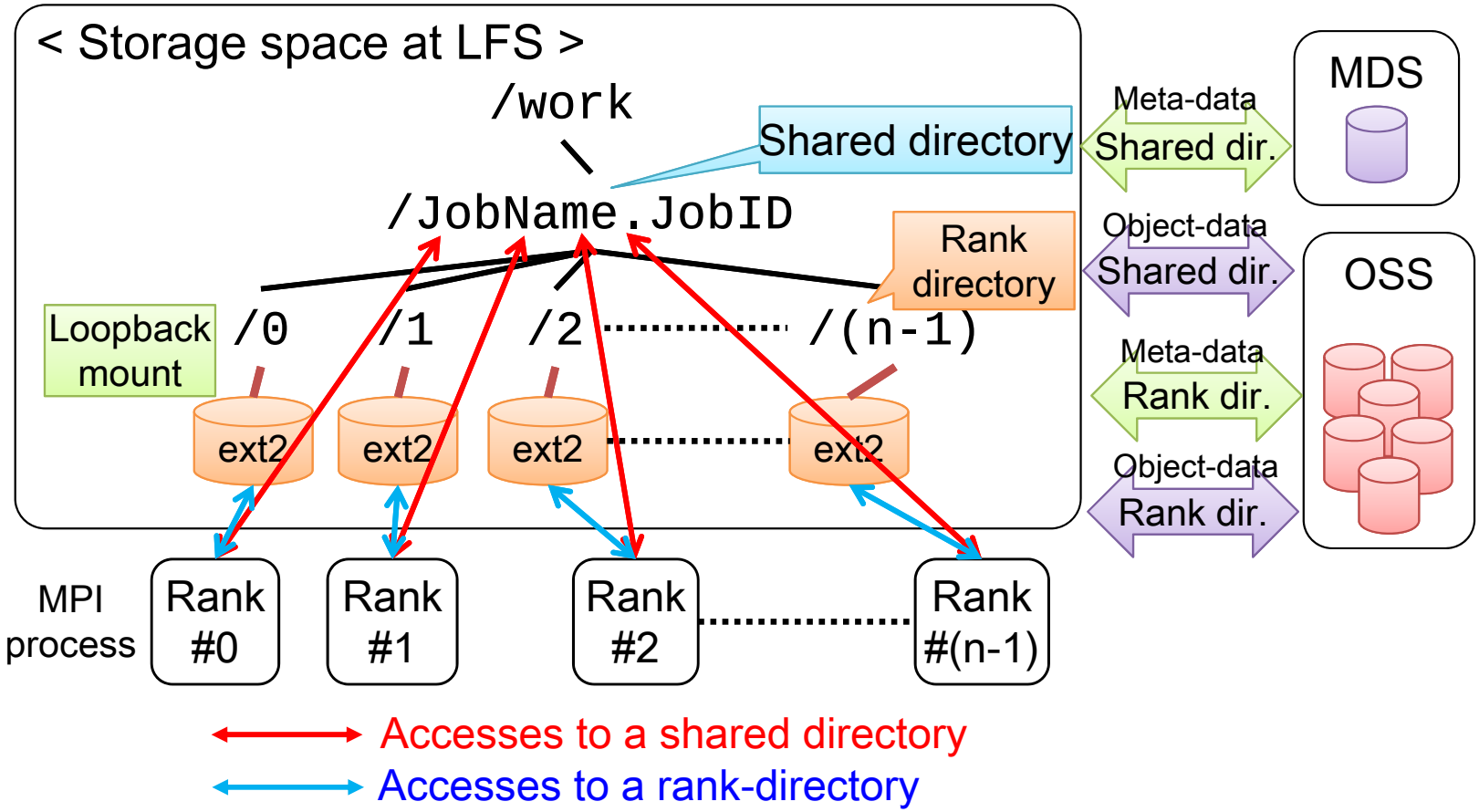
Compute nodes which generated huge number of requests to MDS of LFS




- Many file accesses(open, close, …) lead to high load of MDS.
- High load of MDS on LFS may affect many user applications accessing LFS.

- Providing loop-back file systems (rank-directory) to alleviate high MDS load
- Rank-directory is recommended for user applications which access many files.

# Rank-directory (loopback file system)

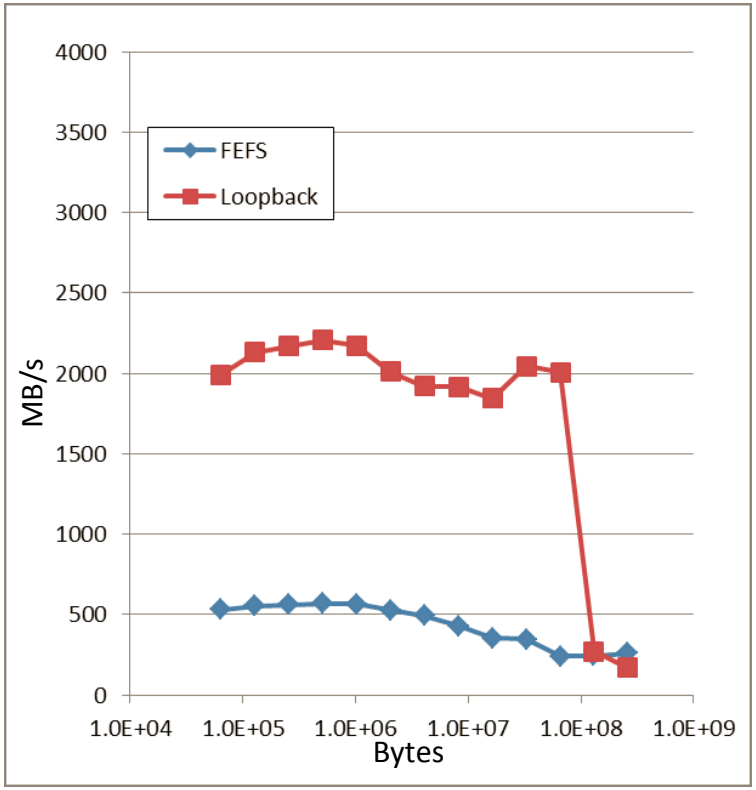



- Reducing MDS accesses leads to effective utilization of LFS.
- I/O accesses in rank-directories are free from slowdown of MDS performance.

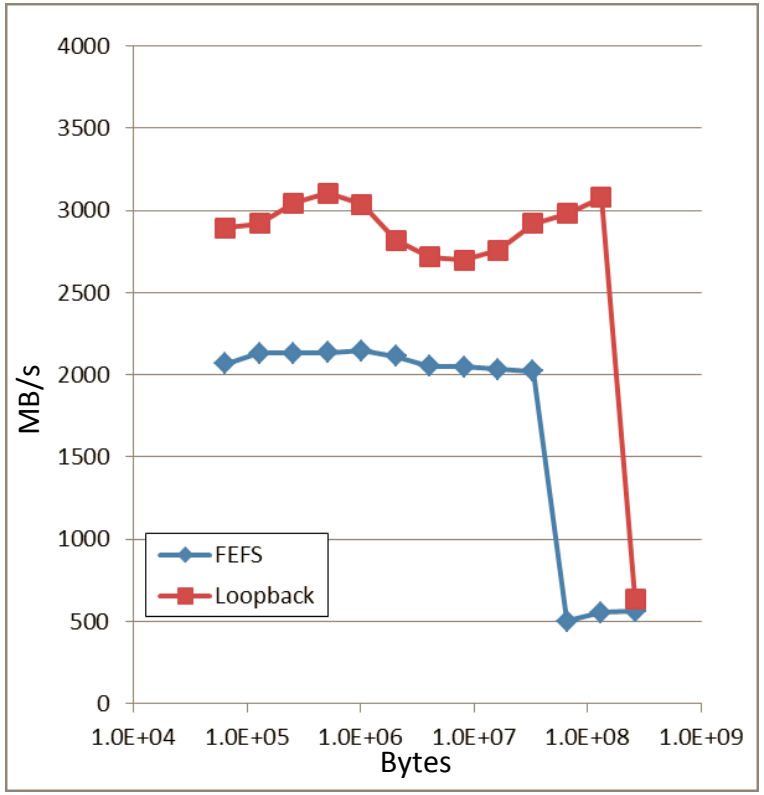
# Single node I/O performance evaluation by using IOzone

- FEFS (shared directory among nodes) vs. loopback
- Loopback outperformed FEFS for smaller data size with the help of file system cache.

write (64KB I/O block)



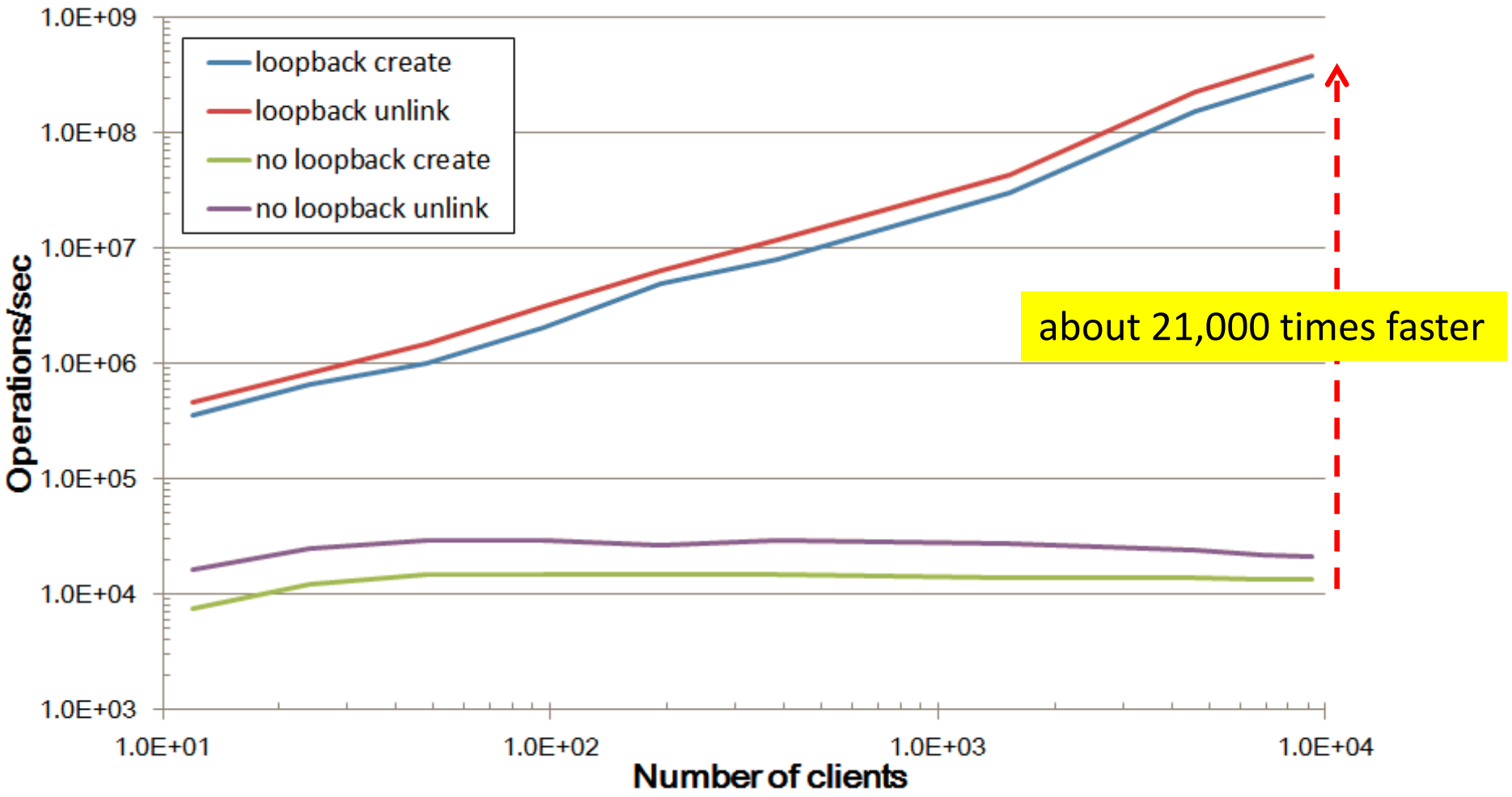

read (64KB I/O block)




K. Yamamoto F. Shoji, A. Uno, S. Matsui, K. Sakai, F. Sueyasu, and S. Sumimoto,
"Analysis and Elimination of Client Evictions on a Large Scale Lustre Based File System," LUG'15

# Total metadata access performance

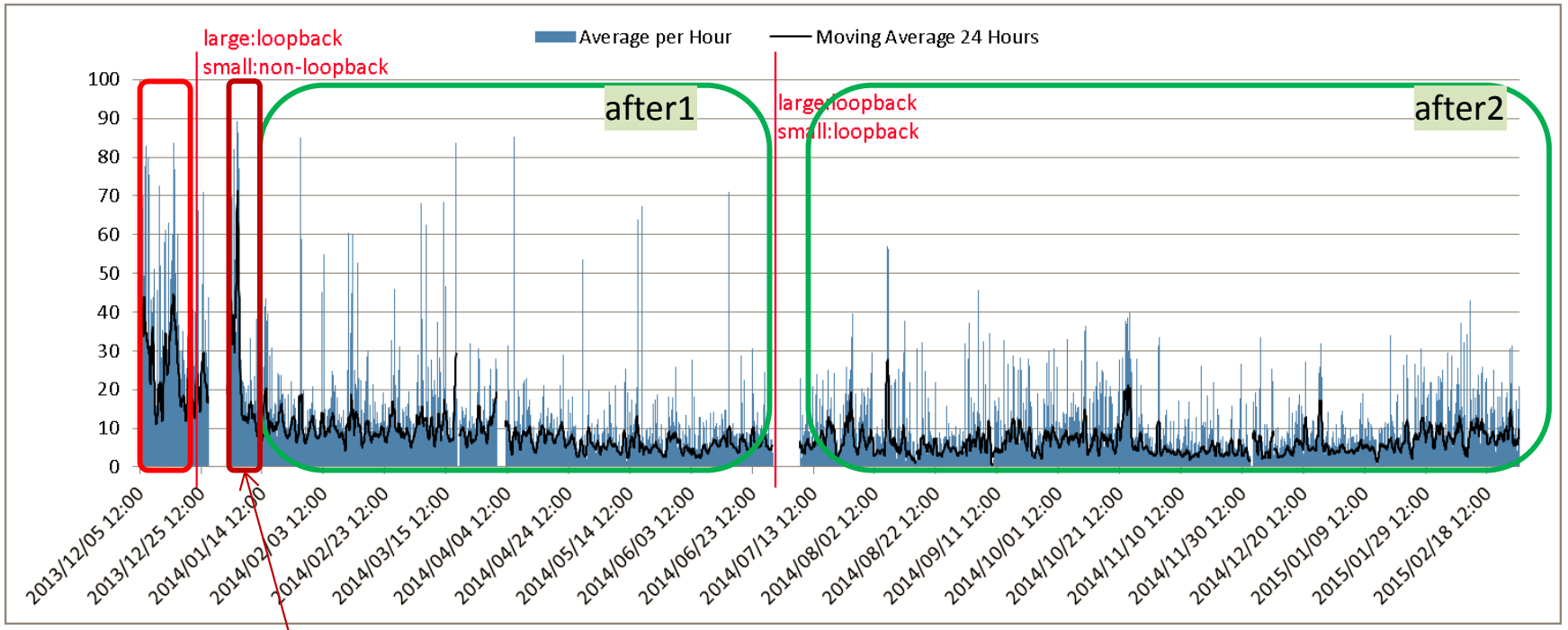
- Create 26K ops/node, unlink 37K ops/node by mdtest (100 files/node)
- Rank directory (loopback) scales with a large number of processes.




K. Yamamoto, F. Shoji, A. Uno, S. Matsui, K. Sakai, F. Sueyasu, and S. Sumimoto,
"Analysis and Elimination of Client Evictions on a Large Scale Lustre Based File System," LUG'15

# Impact for MDS load average

- MDS CPU load




< MDS CPU load over time before and after loopback introduction through two steps(after1 and after2) >

* Some large class job did not use loopback.

- MDS load average per hour: reduced to 1/3.5
- Peak occurrence times per day (over 50%, 70%): reduced to 1/30

K. Yamamoto, F. Shoji, A. Uno, S. Matsui, K. Sakai, F. Sueyasu, and S. Sumimoto,
"Analysis and Elimination of Client Evictions on a Large Scale Lustre Based File System," LUG'15

# Eviction problem

- Eviction
  - File server evicts a client when a client does not work properly, e.g. no response to requests from servers.
- Impact of eviction
  - I/O accesses of running jobs on the node will fail.
    - In many cases, jobs affected by evictions are aborted.

  - Frequent evictions led to a decrease in node utilization seriously.

# Triggers of evictions

- Many of evictions were related with
  - OSS failover
    - Hardware failure
    - Software failure
  - System board maintenance
    - CPU failure
    - Memory failure
    - Interconnect failure

# Mitigation of evictions

- Elimination of client evictions that we have done
  - Step 1: Eliminating evictions during system board maintenance by system operation level
  - Step 2: Eliminating evictions during system board maintenance by improvement of file system level

- The two fixes reduced eviction occurrence ratio by a 1/72.

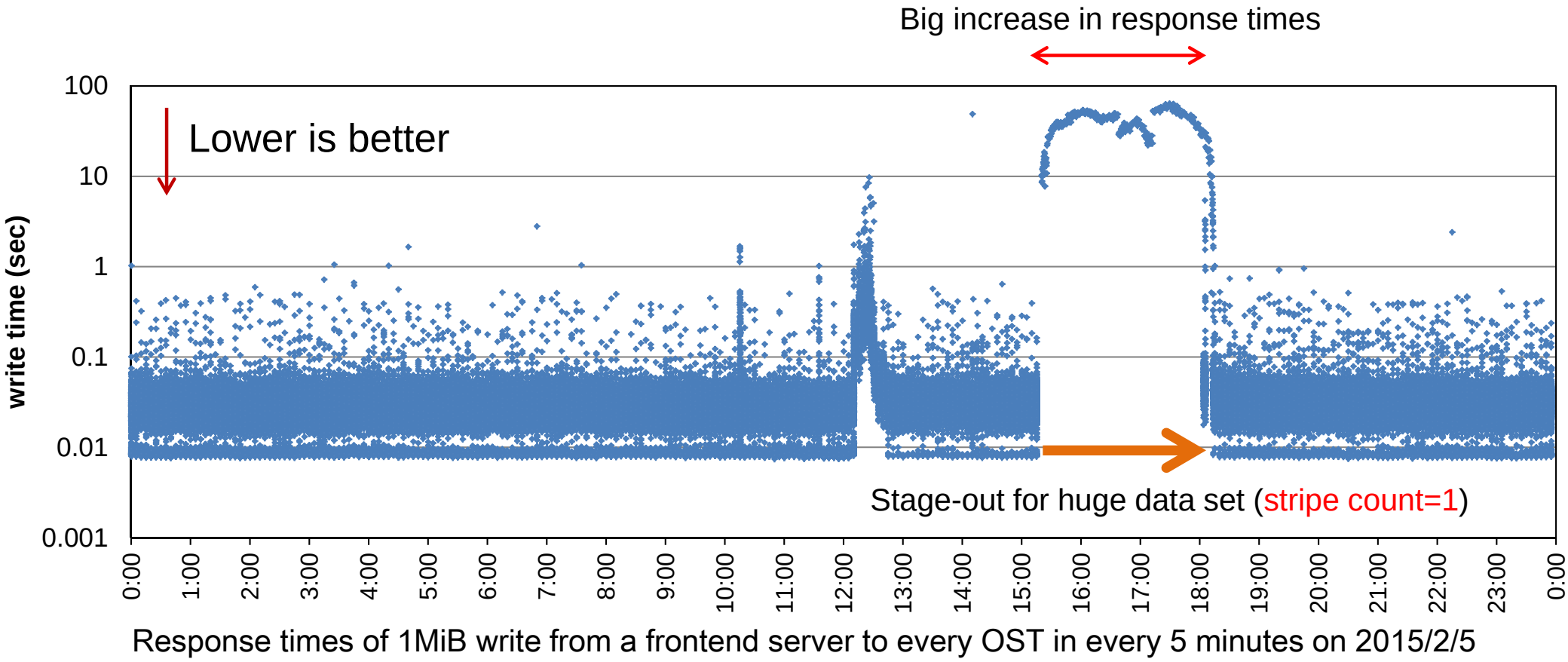
Eviction occurrence ratio/node

| Before | After | Improvements |
|---|---|---|
| 0.47 | 0.0065 | 1/72 |

K. Yamamoto, F. Shoji, A. Uno, S. Matsui, K. Sakai, F. Sueyasu, and S. Sumimoto,
"Analysis and Elimination of Client Evictions on a Large Scale Lustre Based File System," LUG'15

# Interference due to heavy data staging

- Increase in response time in GFS accesses due to heavy data staging




Response times of 1MiB write from a frontend server to every OST in every 5 minutes on 2015/2/5

- We have already adopted stripe count selection simply based on file size in stage-out phase. => Success in mitigation interference so far.
- For more optimization, we have examined impact of stripe count and QoS function of FEFS.

AICS K computer

# I/O workload aware stripe count

- Tuning scheme of stripe count (Cs) in stage-out

$$C_S = \left\lceil \frac{\alpha}{\beta} \times \frac{N_{OST}}{N_{IO} \times k_{stg}} \right\rceil \quad \text{, where} \quad \alpha = \left\lceil \frac{n_{stg}}{N_{OSS} \times l_{thr}} \right\rceil \quad \text{and} \quad k_{stg} = \min\left(\frac{n_{stg}}{N_{IO}}, k_{stg}^{\max}\right)$$

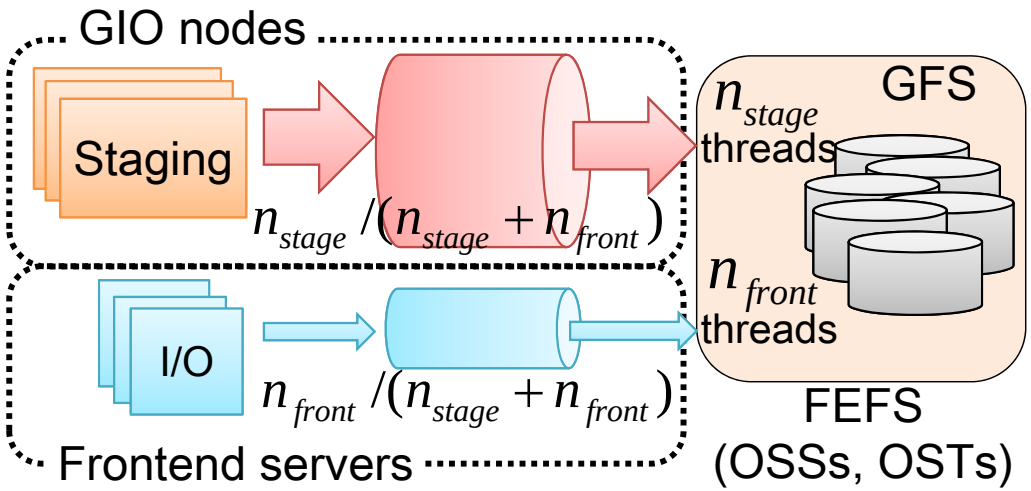
| | |
|---|---|
| $\alpha$ | The number of files that each OSS service thread manages |
| $\beta$ | Maximum acceptable variance in I/O workload among OSTs |
| $N_{OSS}$ | The number of OSSs |
| $N_{OST}$ | The number of OSTs |
| $N_{IO}$ | The number of I/O (GIO) nodes |
| $l_{thr}$ | Maximum number of service threads on each OSS |
| $k_{stg}$ | The number of files in staging at each GIO |
| $k_{stg}^{\max}$ | Maximum number of files that one GIO can manage |

Y. Tsujita, T. Yoshizaki, K. Yamamoto, F. Sueyasu, R. Miyazaki, and A. Uno, “Alleviating I/O Interference through Workload-Aware Striping and Load-Balancing on Parallel File Systems,” accepted in ISC’17

# Performance improvements in GFS accesses

1. I/O workload-aware stripe count
   -> Balanced I/O workload among OSTs

2. Load-balancing among clients (QoS of FEFS)

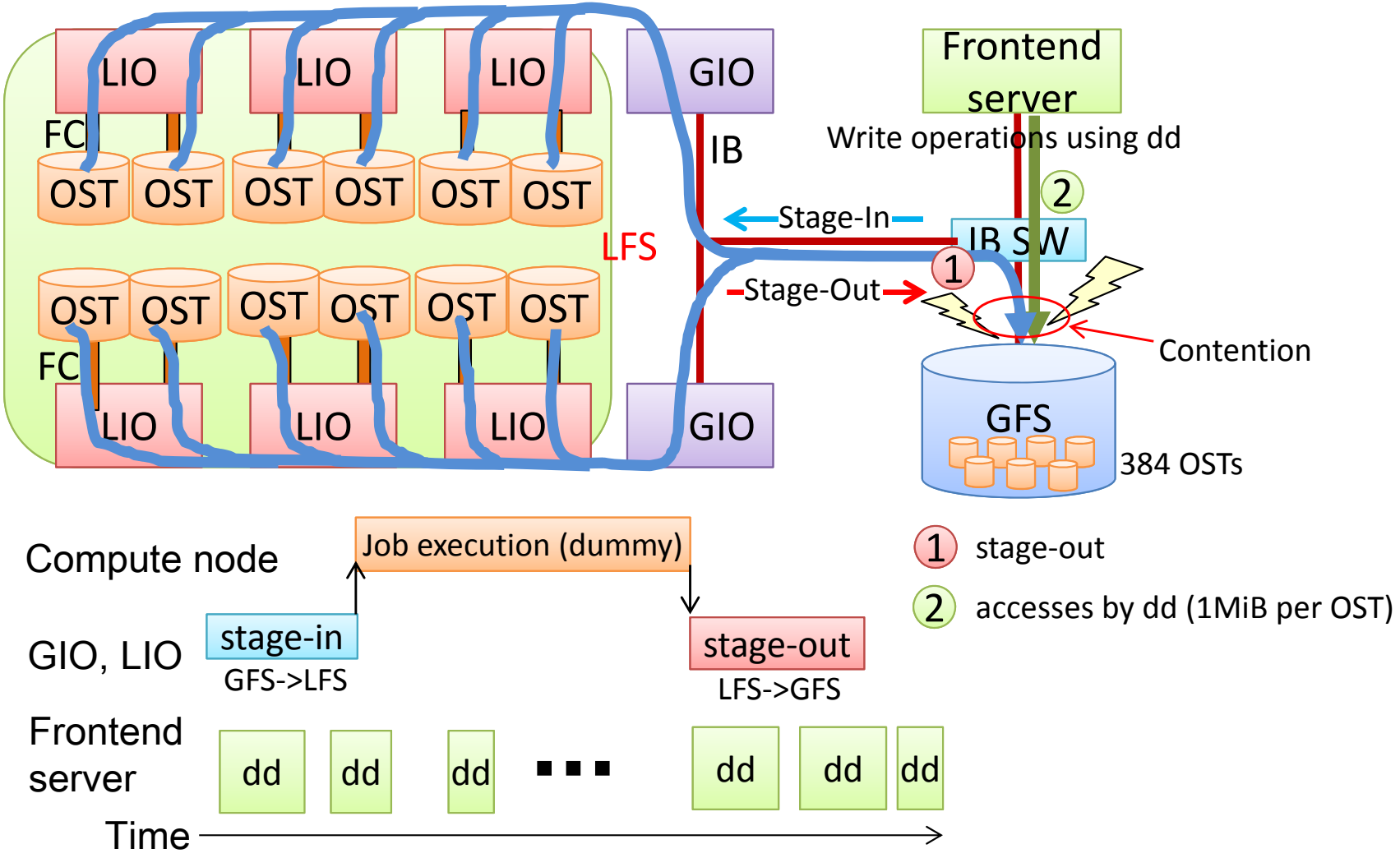



QoS function of FEFS utilized for data staging at the K computer

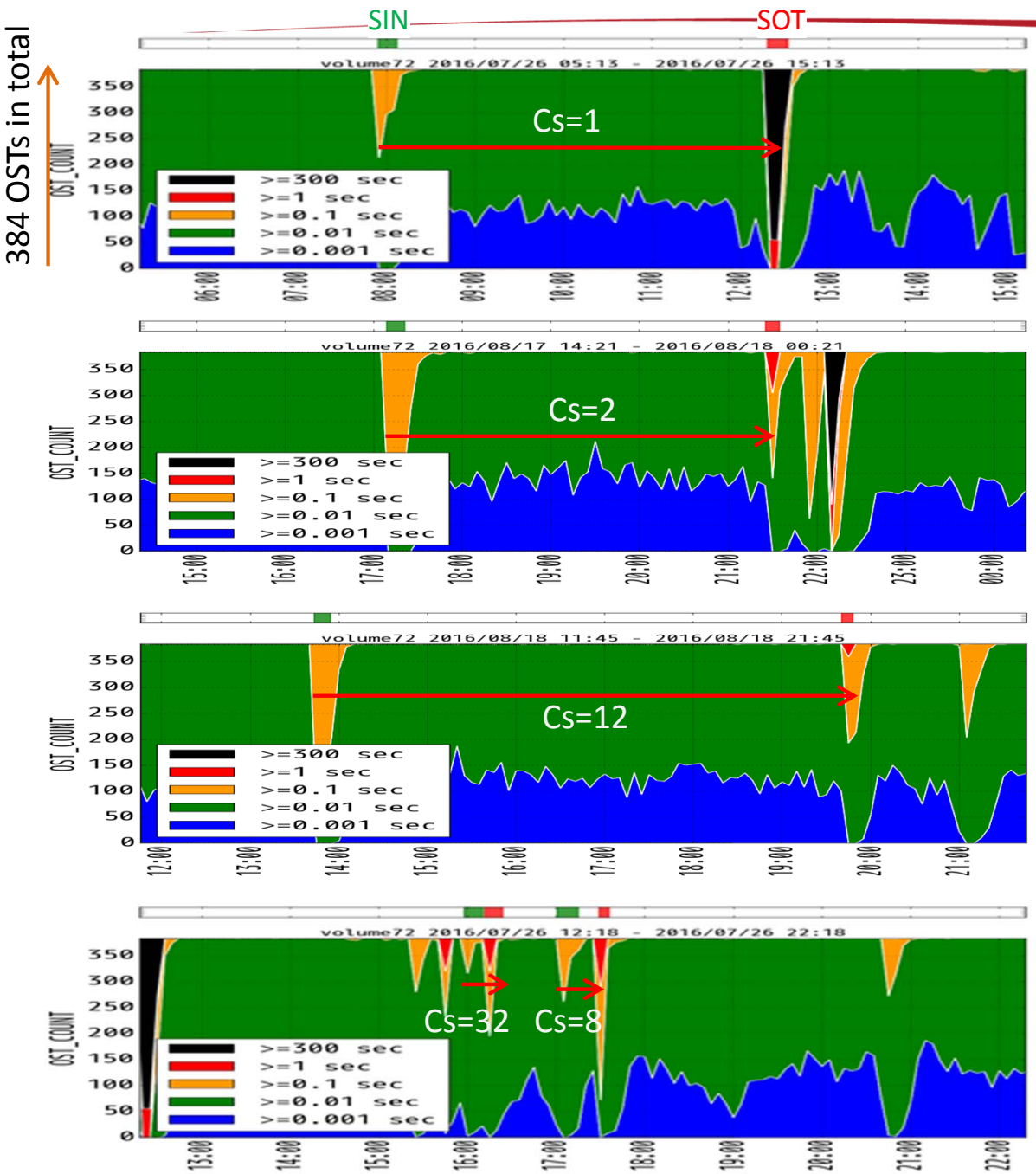
Limiting the maximum number of service threads for each client
-> Guaranteeing I/O bandwidth for each client

# I/O interference mitigation in data staging

- Evaluation setup (just doing stage-in and stage-out)



# Performance improvements in GFS accesses (stripe count tuning)




96 GIOs(12x24x2), 576 files (12GB/file)
One volume of GFS: 384 OSTs

Cs: Stripe count
SIN: Stage-in operation
SOT: Stage-out operation

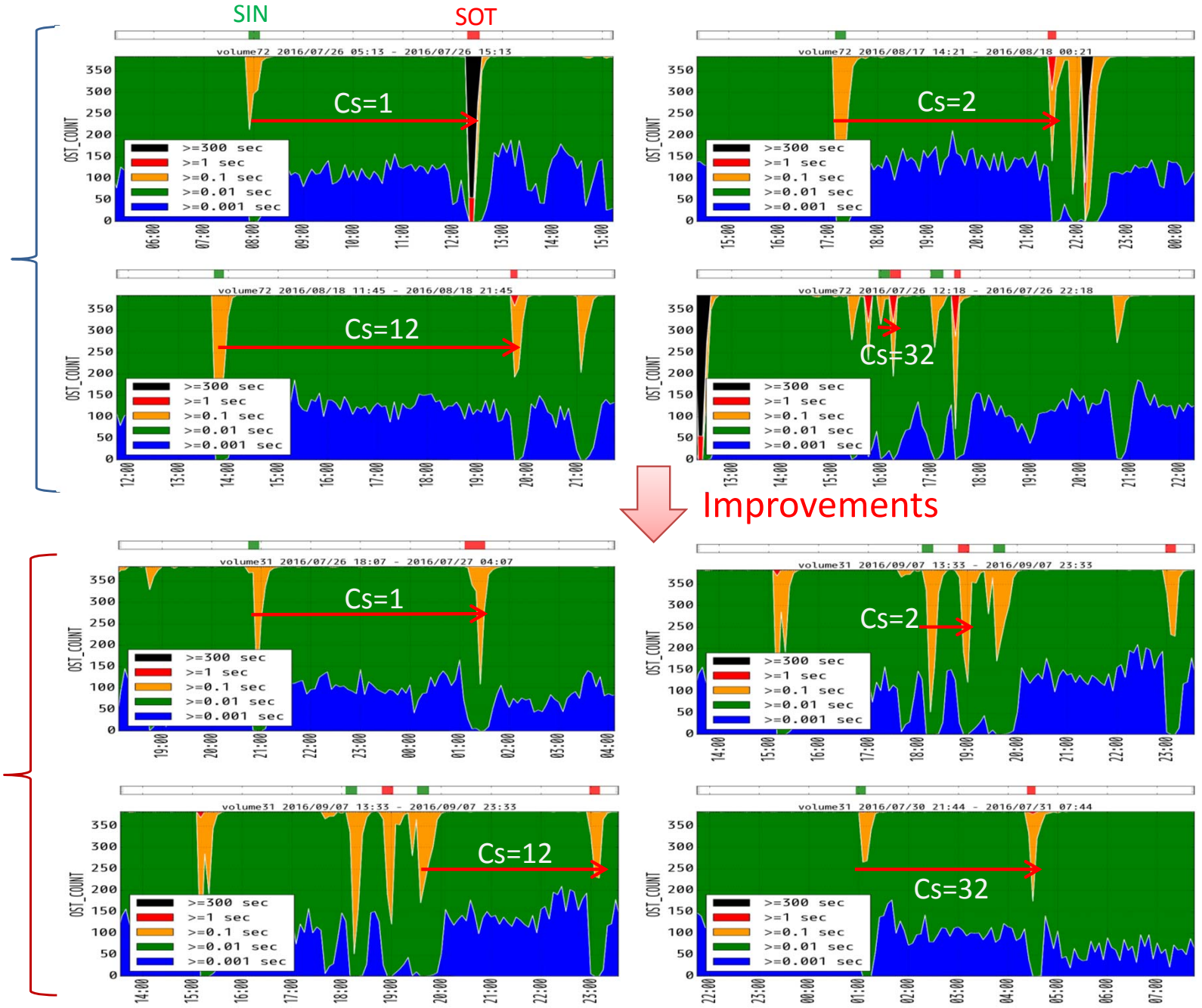
← Preferable stripe count

Preferable stripe count mitigated performance degradation due to huge data staging.

< Response time distribution in accessing a GFS from a frontend server over time >

# Performance improvements in GFS accesses with QoS function

96 GIOs(12x24x2), 576 files (12GB/file)



# Contribution to Lustre development by Fujitsu

- Incorporated function derived from R&D works for the K computer by Fujitsu

(* Until this time (Mar. 23, 2017))

| Jira [1] | Function | Landing |
|---|---|---|
| LU-2467 | Ability to disable pinging | Lustre 2.4 |
| LU-2466 | LNET networks hashing | Lustre 2.4 |
| LU-2934 | LNET router priorities | Lustre 2.5 |
| LU-2950 | LNET read routing list from file | Lustre 2.5 |
| LU-2924 | Reduce ldlm_poold execution time | Lustre 2.5 |
| LU-3221 | Endianness fixes (SPARC support) | Lustre 2.5 |
| LU-2743 | Errno translation tables (SPARC support) | Lustre 2.5 |
| LU-4665 | lfs setstripe to specify OSTs | Lustre 2.7 |

1. https://jira.hpdd.intel.com/projects/LU/issues/

# Future plan (and hopes)

- Improvements in analysis framework for log files from FEFS and associated components
  - Effective error detection to improve availability
  - Coupled analysis framework with user job information
- The way to guarantee heathy status of FEFS
  - Detection of malfunctioning OSTs
    - Considering the preferable and effective way for detection is in progress.

- Collaboration in file system management
  - High availability and performance
  - Advanced monitoring and log analysis scheme, etc.

# Acknowledgment

Special thanks to

- RIKEN AICS
  - F. Inoue, M. Iwamoto, F. Shoji, K. Sugeta, A. Uno, K. Yamamoto

- FUJITSU
  - H. Hida, N. Ikeda, S. Matsui, R. Miyazaki, M. Okamoto, R. Sekizawa, F. Sueyasu, S. Sumimoto, T. Yoshizaki

giving many information about their efforts described in this presentation.