FUJITSU

shaping tomorrow with you

Erasure Code with Shingled Local Parity Groups for Efficient Recovery from Multiple Disk Failures

Takeshi Miyamae, Takanori Nakao, Kensuke Shiozawa
Fujitsu Laboratories Ltd.

October 5th, 2014 (HotDep'14)

FUJITSU

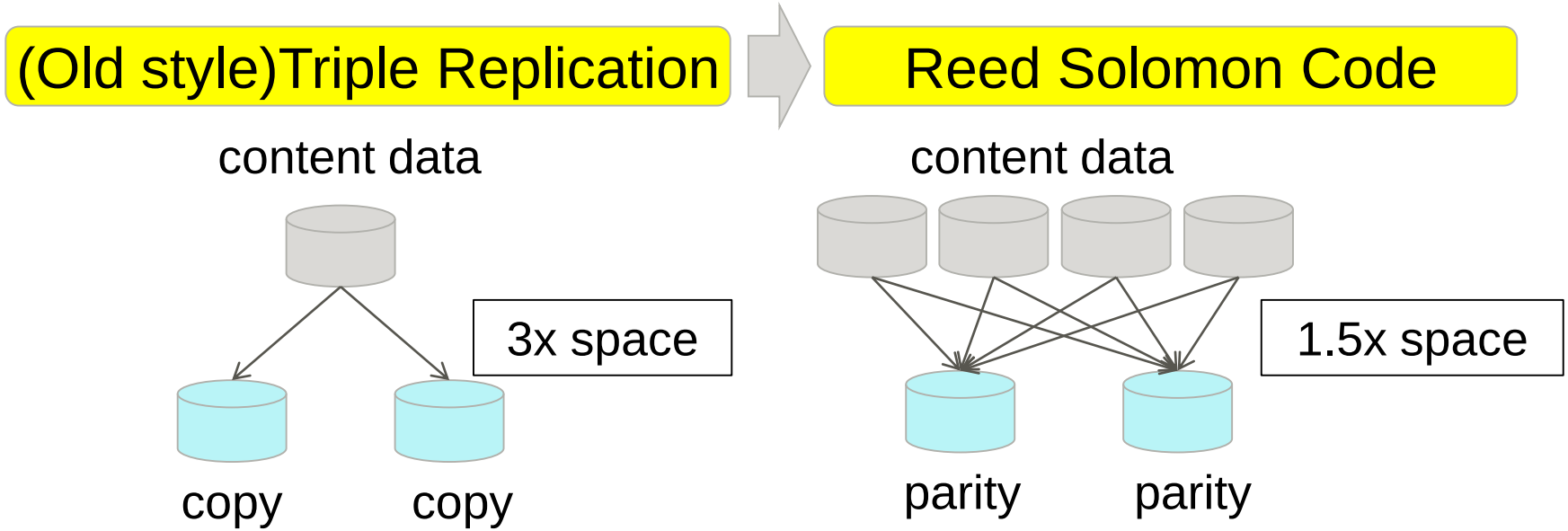
Contents

FUJITSU

1. Backgrounds and Our Proposal

Backgrounds (1)

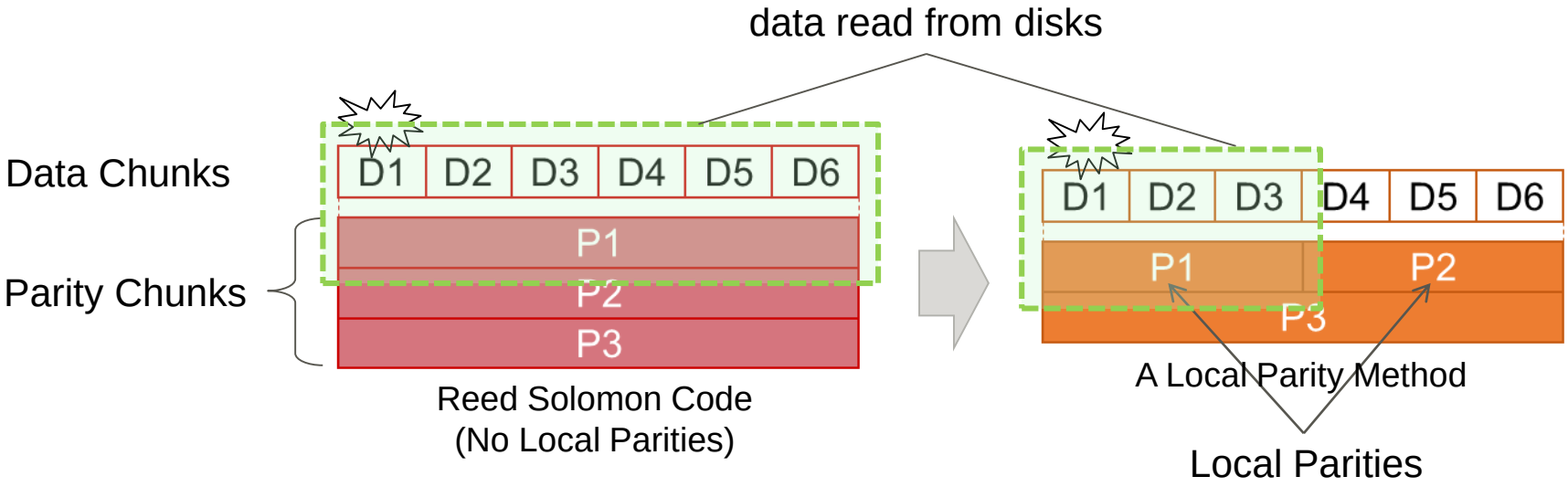
- Erasure codes for content data
  - Content data for ICT services is ever-growing
  - Demand for higher space efficiency and durability
  - Reed Solomon code (de facto erasure code) improves both



However, Reed Solomon code is not so recovery-efficient

Backgrounds (2)

- Local parity improves recovery efficiency
  - Data recovery should be as efficient as possible
    - in order to avoid multiple disk failures and data loss
  - Reed Solomon code is improved by local parity methods
    - data read from disks is reduced during recovery

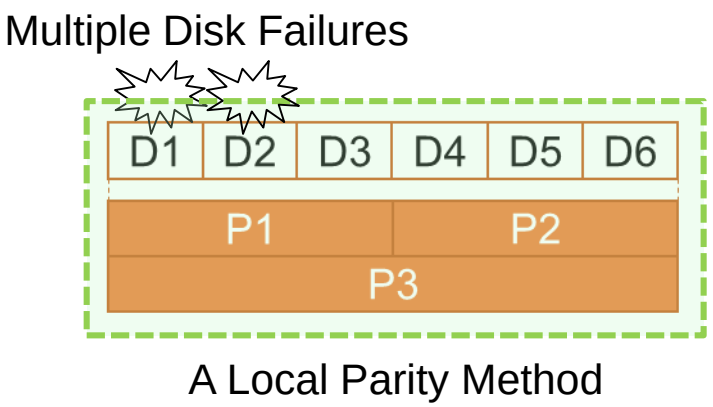


However, multiple disk failures is out of consideration

Our Goal

FUJITSU

- Local parity method for multiple disk failures
  - Existing methods are optimized for single disk failure
    - e.g. Microsoft MS-LRC, Facebook Xorbas
  - However, Its recovery overhead is large in case of multiple disk failures
    - because they have a chance to use global parities for recovery



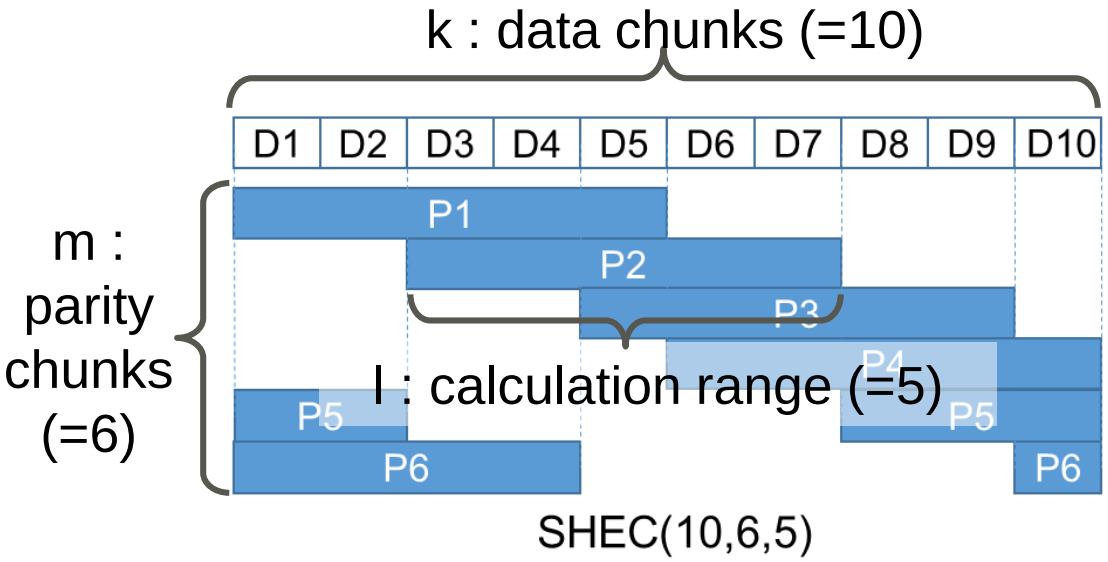
A Local Parity Method

Our goal is a method efficiently handling multiple disk failures

Our Proposal Method (SHEC)

FUJITSU

- SHEC (= Shingled Erasure Code)
  - An erasure code only with local parity groups
    - to improve recovery efficiency in case of multiple disk failures
  - The calculation ranges of local parities are shifted and partly overlap with each other (like the shingles on a roof)
    - to keep enough durability



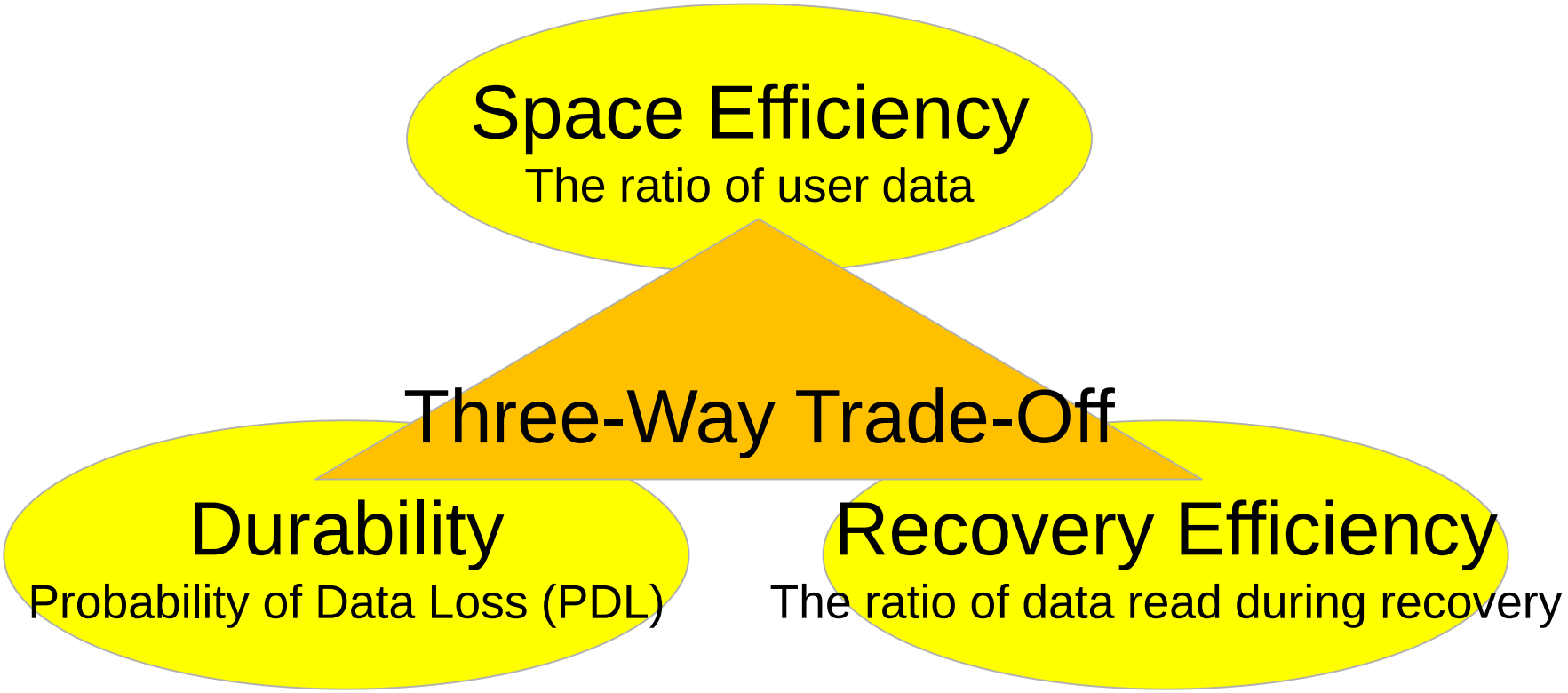
FUJITSU

2. SHEC's Theoretical Analysis

Erasure Code's Properties

FUJITSU

- We picked three erasure code's properties for SHEC's theoretical analysis

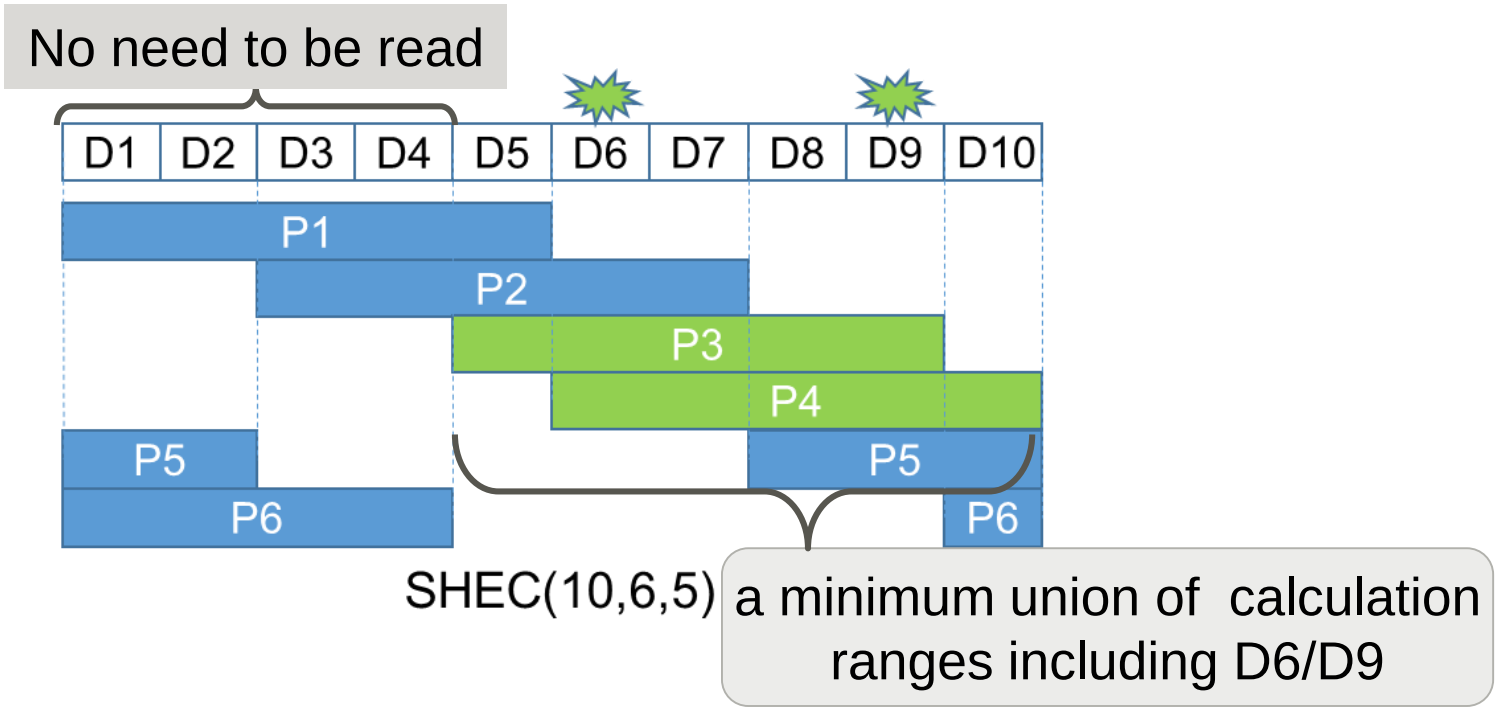


The properties satisfy a three-way trade-off relationship

SHEC's Recovery Efficiency

FUJITSU

- High Recovery Efficiency from Multiple Disk Failures
  - The amount of data read from disks is minimized
    - (e.g.) When D6/D9 break out, SHEC will select P3/P4 for recovery

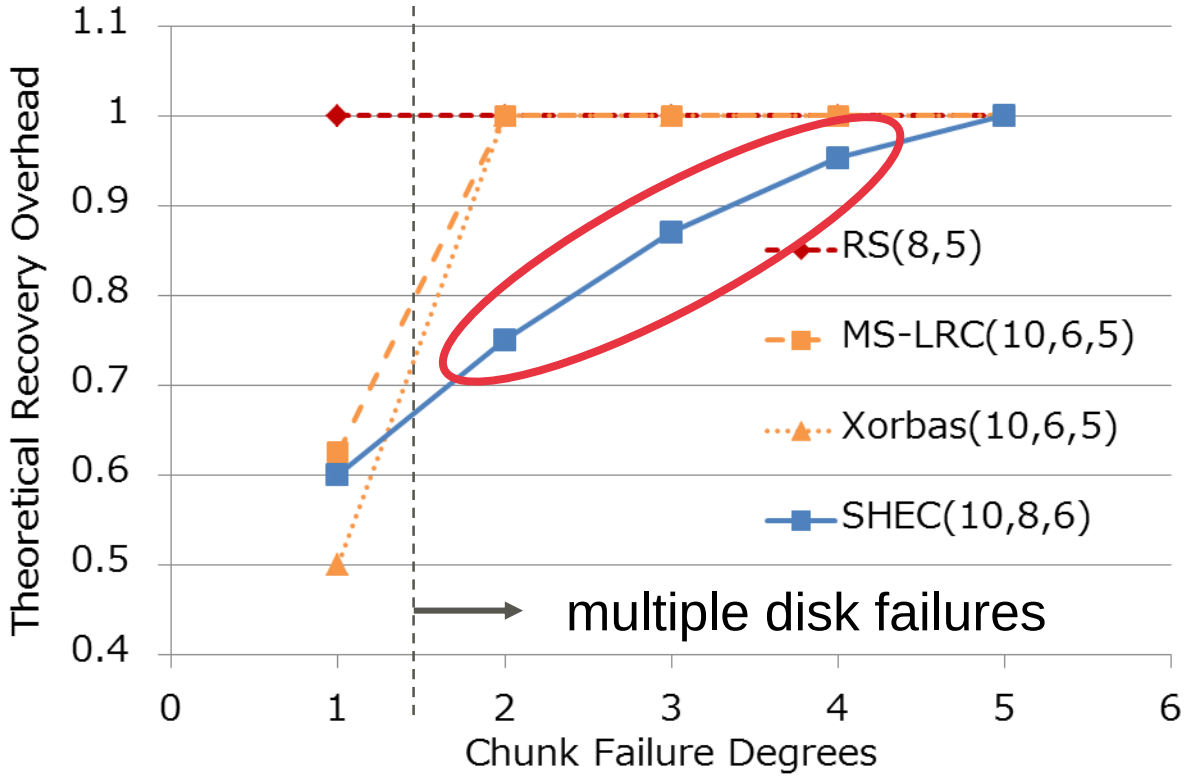


Recovery efficiency is one of the biggest features of SHEC

Comparison with Other Methods

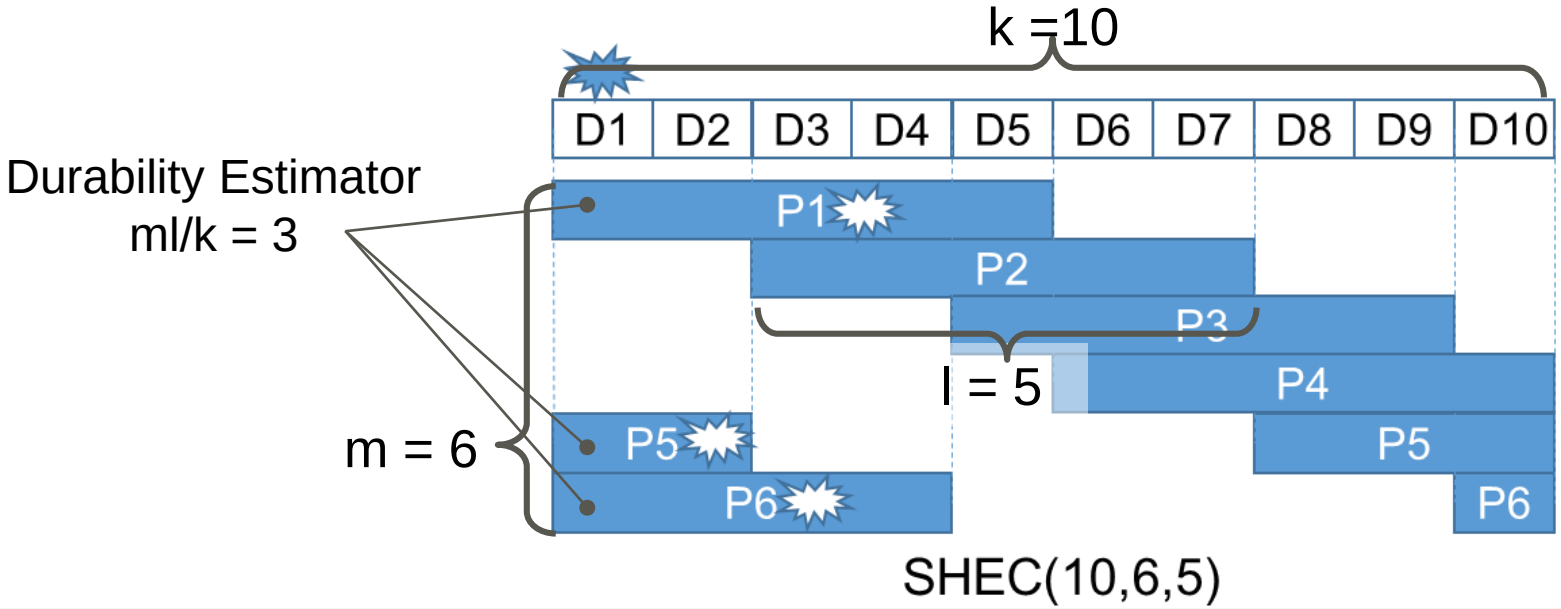
FUJITSU

- SHEC is expected to recover more efficiently than the other methods in case of multiple disk failures
  - Other methods : Reed Solomon, MS-LRC and Xorbas



SHEC's Durability

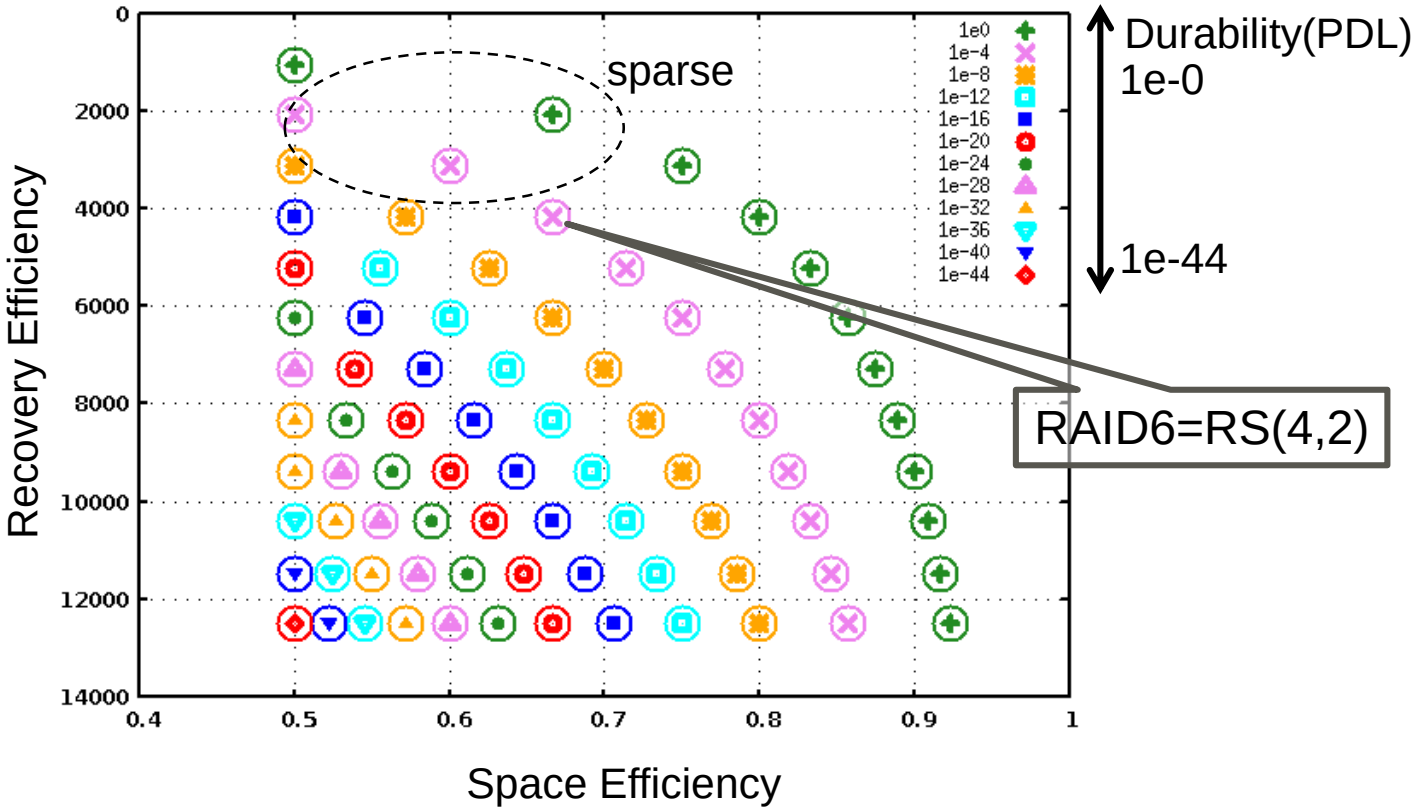
- Durability Estimator (=ml/k)
  - Indicates the number up to how many disks can be failed
  - Therefore, ml/k+1 disk failures can cause data loss
    - (e.g.) SHEC(10,6,5)'s durability estimator is three. Therefore, four failures of D1/P1/P5/P6 cause data loss because D1 cannot be recovered from the remaining chunks



Property Map of Reed Solomon code

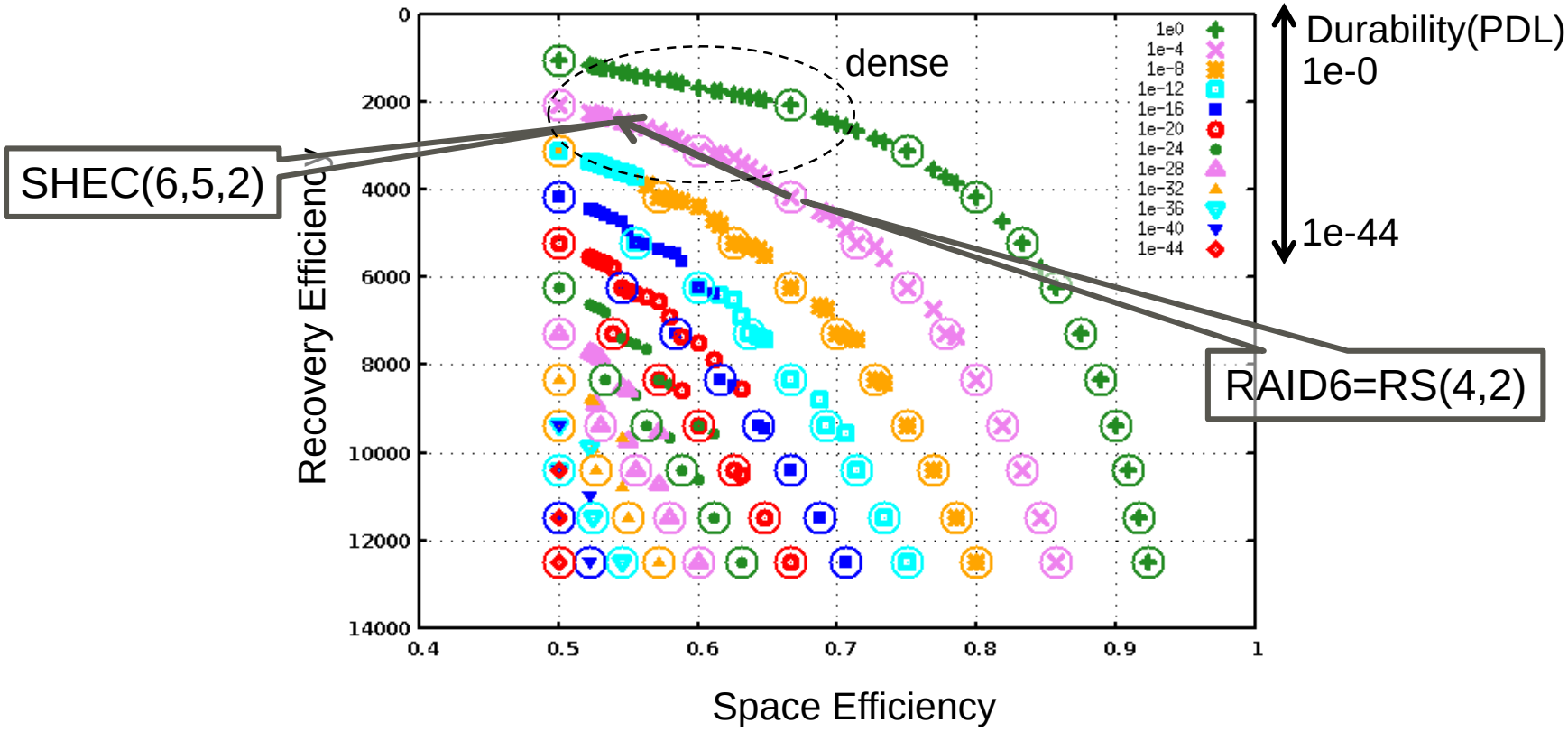
FUJITSU

- Upper area becomes sparse
  - Reed Solomon code has few recovery-efficient layouts



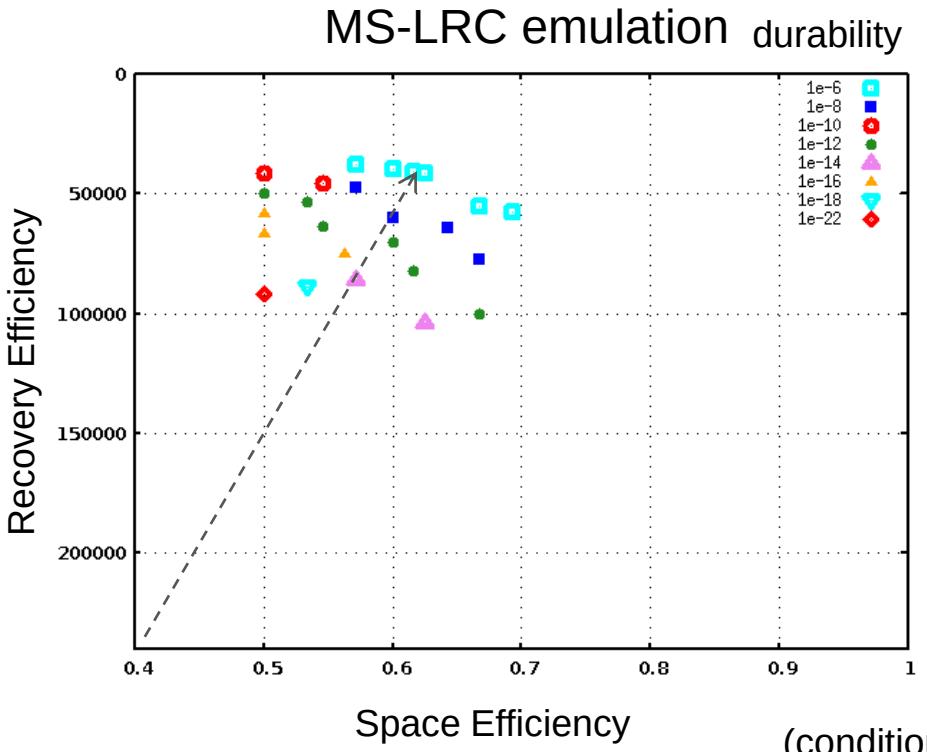
Property Map of SHEC

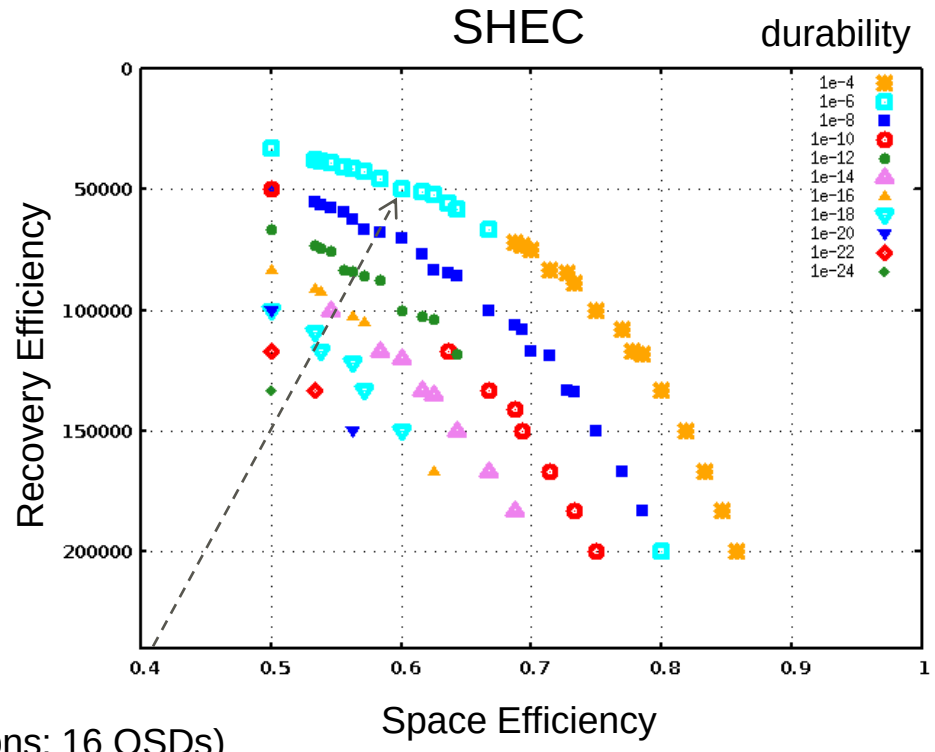
- Upper area is filled with SHEC-specific layouts
  - SHEC provides many recovery-efficient layouts
  - SHEC is more adjustable than Reed Solomon code



Comparison with MS-LRC (1)

- Single disk failure case
  - MS-LRC is plotted farther from the origin (= superior)
  - SHEC is plotted in a broader area (= more flexible)

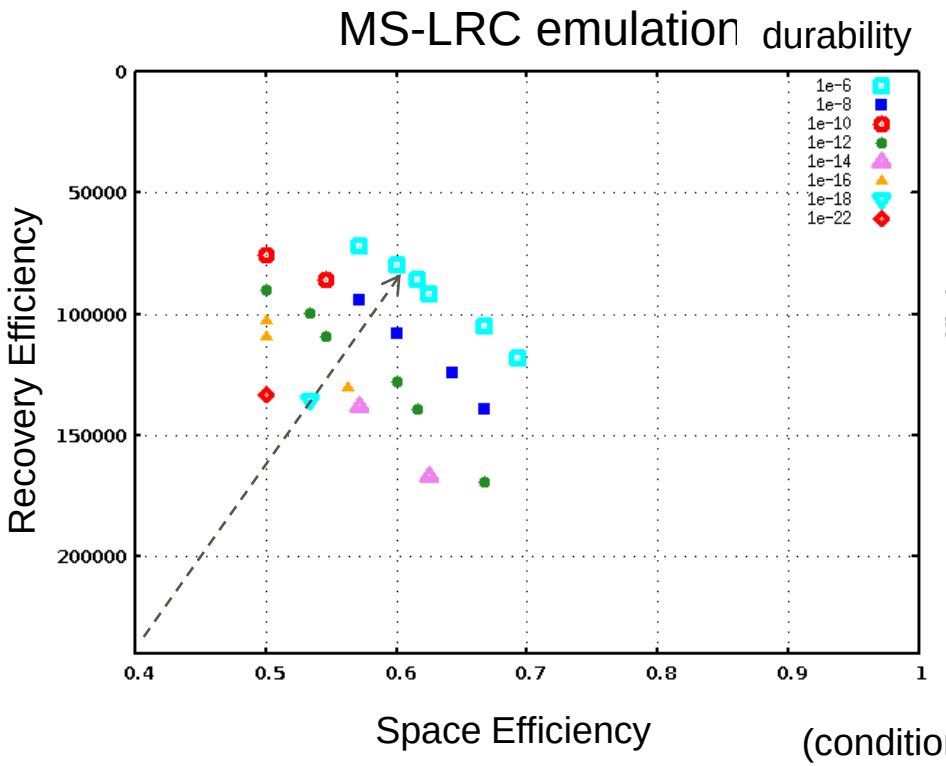


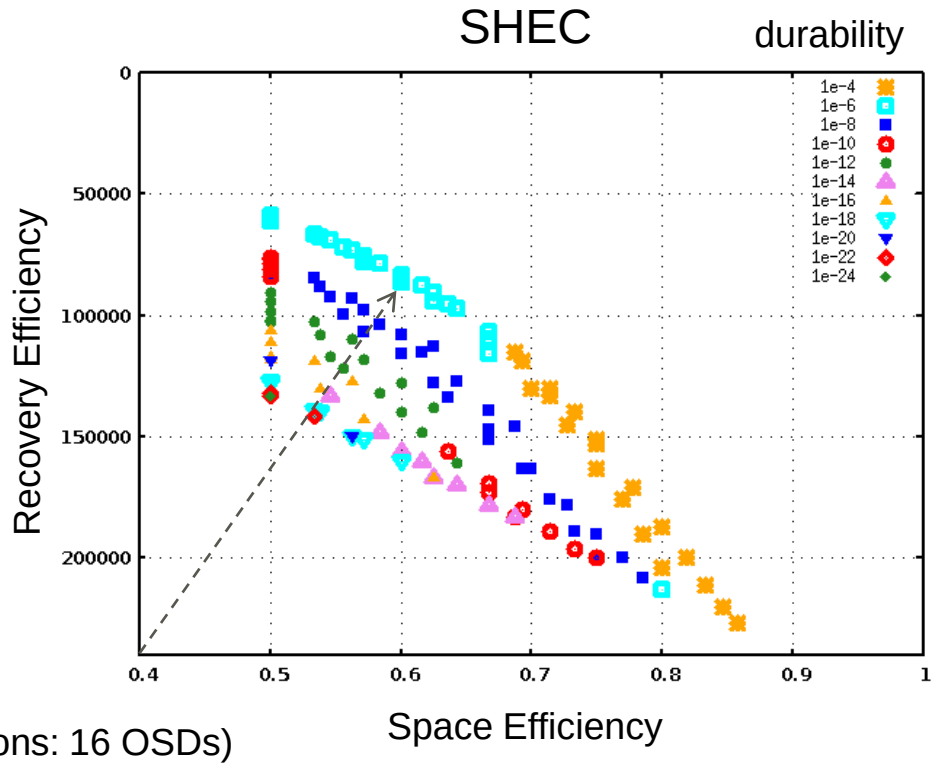


(conditions: 16 OSDs)

Comparison with MS-LRC (2)

- Double disk failures case
  - Both are plotted at the same distance from the origin
  - SHEC is plotted in a broader area (=more flexible)





(conditions: 16 OSDs)

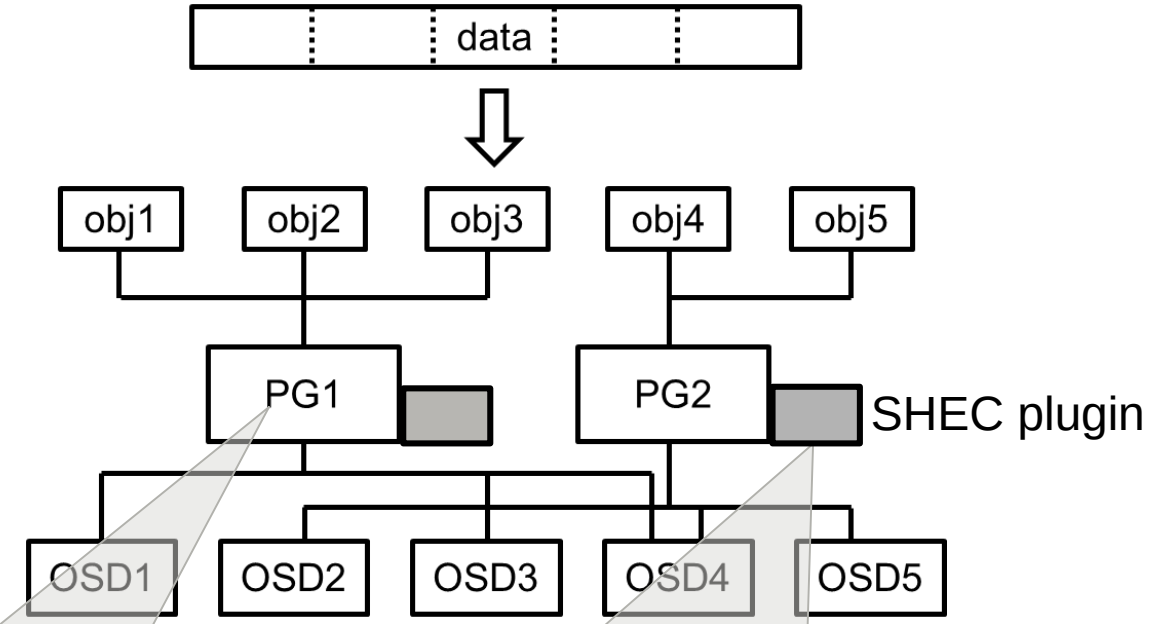
FUJITSU

3. SHEC's Experimental Evaluation

SHEC's Implementation on Ceph

FUJITSU

- SHEC is implemented as an erasure code plugin of Ceph, an open source scalable object storage



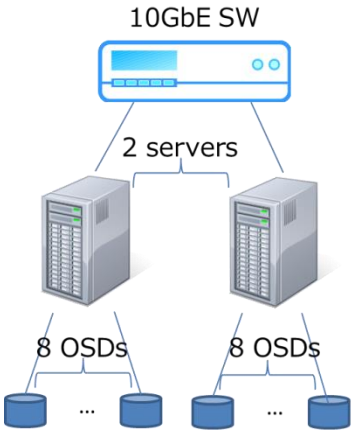
4MB objects are split into data/parity chunks, distributed over OSDs

encode/decode logic is separated from main part of Ceph Storage

Experiment of Recovery Efficiency

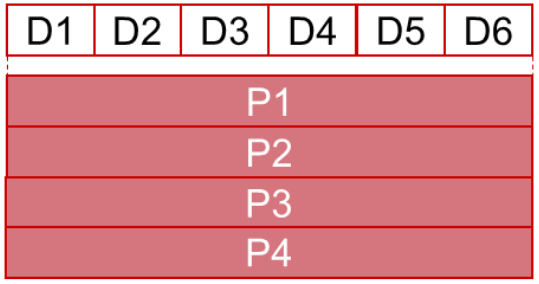
FUJITSU

- Experiment Abstract
  - Test items : Recovery completion time / Resource profiles
  - Failure degree : Double disk failures
  - Comparison : Reed Solomon RS(6,4) / SHEC(6,4,3)

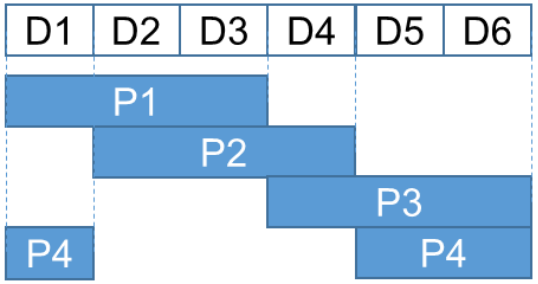


Machine	Fujitsu RX300 S7 (x2)
CPU	Xeon 300GHz (8 core)
Memory	32GB
HDD	300GB SAS (x8)
Network	10GbE NIC
OS	Ubuntu 14.04
Kernel	3.13.0-24
Ceph	v0.80.1 (Firefly)

Hardware and Software Setup



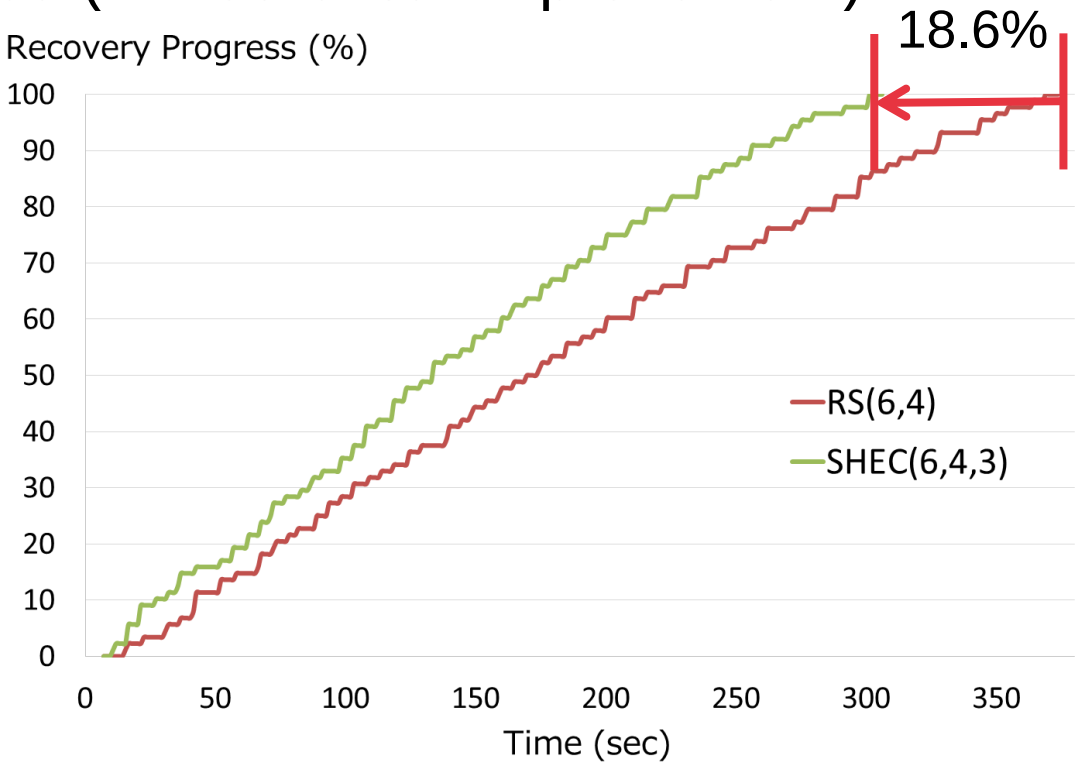
RS(6,4)



SHEC(6,4,3)

Recovery Completion Time

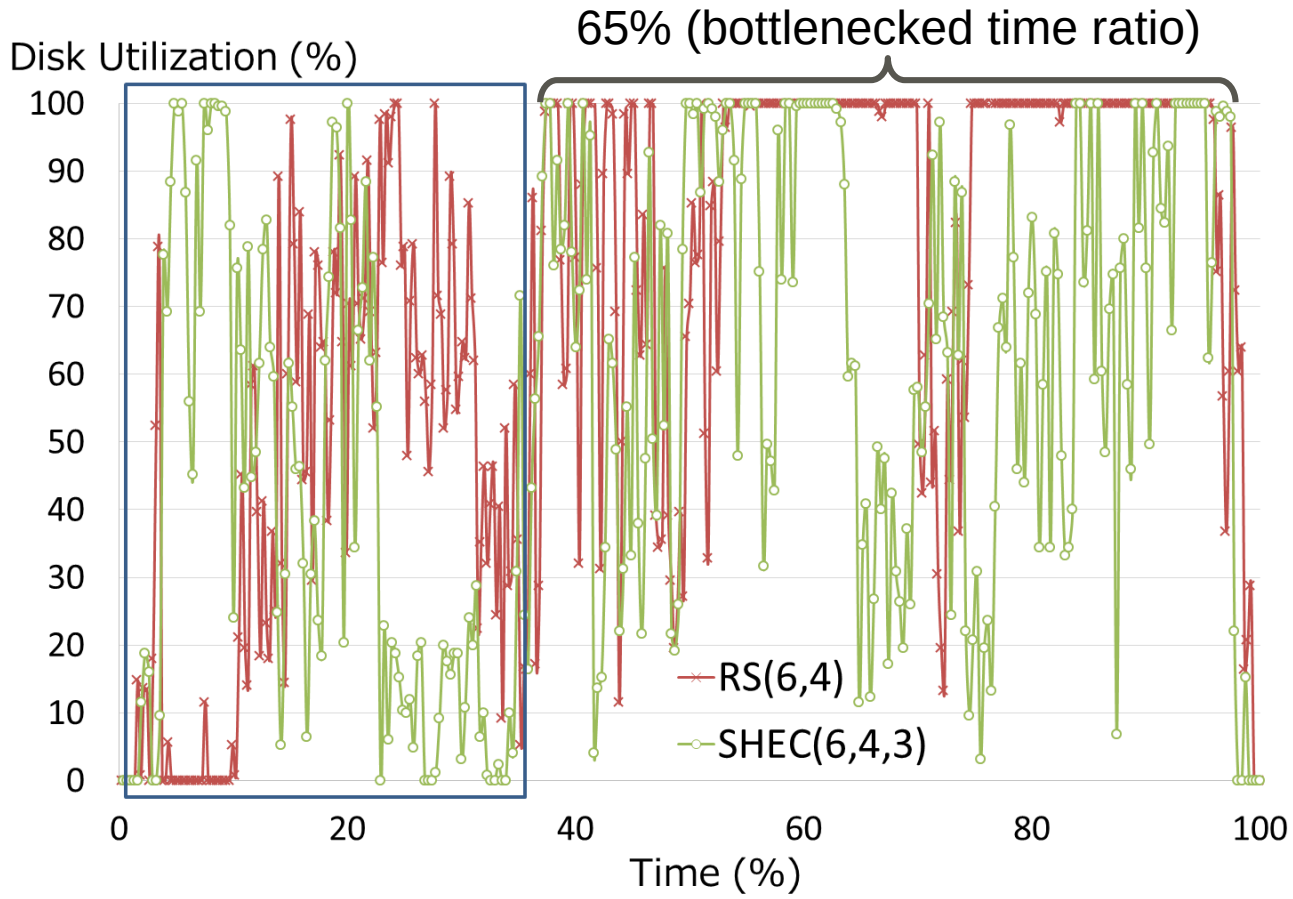
- SHEC's recovery completion time was 18.6% faster
  - OTOH, total amount of data read from disks was 26% decreased (= theoretical improvement)



Why were not these figures the same?

The Reason (= Disk utilization)

- Disks were only partly (65%) bottlenecked



There is 35% room for recovery time improvement

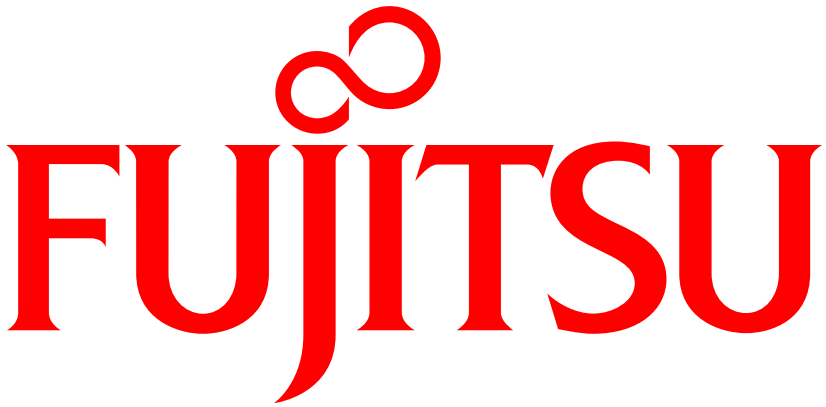
FUJITSU

4. Summary

Summary

FUJITSU

1. We proposed Shingled Erasure Code (SHEC)
   - SHEC is recovery-efficient especially in case of multiple disk failures
2. We found SHEC is more adjustable than Reed Solomon code
   - because SHEC provides many recovery-efficient layouts including Reed Solomon codes
3. We confirmed SHEC's recovery efficiency in an experiment
   - SHEC's recovery time was 18.6% faster than Reed Solomon code in case of double disk failures



shaping tomorrow with you