QoE-Aware Video Storage Power Management Based on Hot and Cold Data Classification

Hwangje Han and Minseok Song
Inha University in South Korea

NOSSDAV 2018

Outline

- Introduction
- Basic idea
- Proposed schemes
  - Hot data determination algorithm
  - Hot disk allocation algorithm
  - Bandwidth allocation algorithm
- Experimental results
- On-going works
- Conclusions

Introduction – Background

- Dynamically adaptive streaming over HTTP (DASH) requires a lot of storage for store all the transcoded versions, thereby consuming a significant disk power.
- A major consequence of this high power consumption is heat, which may accelerate an increase in the failure rate for disk drives.
- What is worse, high energy consumption also impedes the expansion of servers, so it is essential to limit storage power consumption.

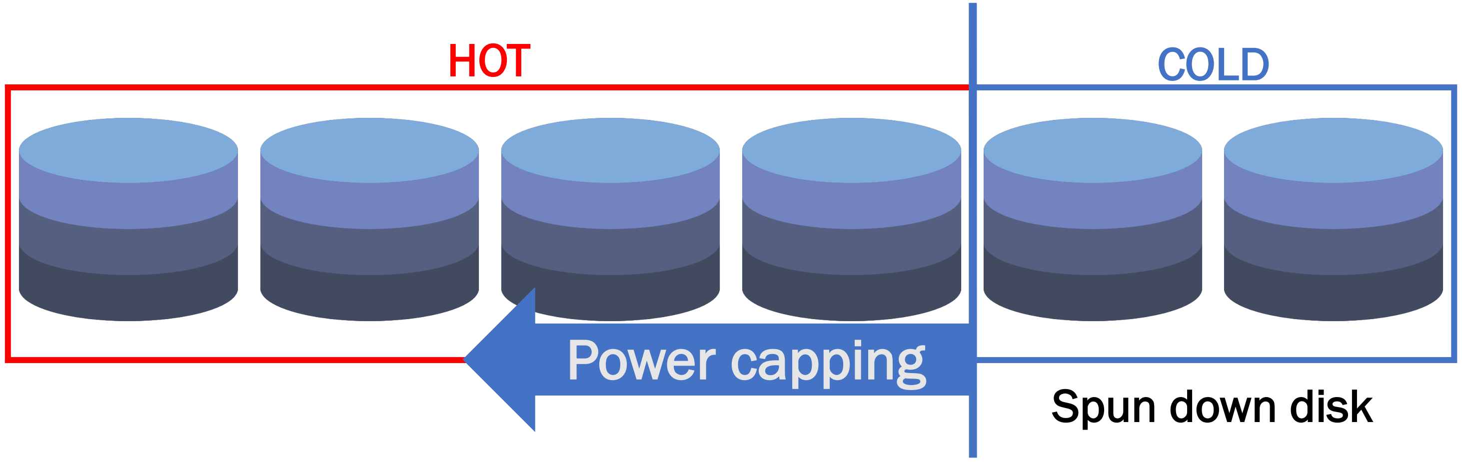
Introduction – Motivation

- DASH inherently requires a lot of transcoding operations.
  - This also requires a lot of storage space to store transcoded versions, resulting in high disk power consumption to maintain many storage devices.
  - How many disks are required ?
    - 2000 videos, 7 versions => 61 terabytes
    - 100,000 videos => about 1500 disks (2TB)

If the disk idle power is 4W, then 6,000W is required even though there is no request

Basic idea – Architecture

A disk array is divided into hot and cold zones, and disks in the cold zone are spun down to save energy consumption, allowing only the data on the hot disk to be read.

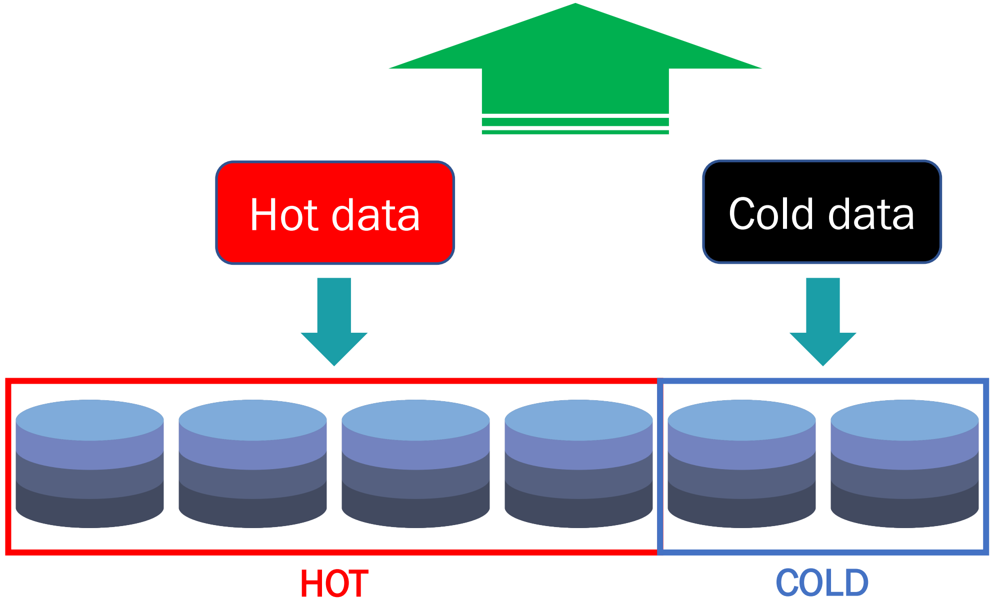


Number of hot disks determine the amount of power consumed

Basic idea – Hot data determination

- Hot data determination:
  - Hot data determination determines the data to be stored on the hot disk to maximize overall QoE gain.

Objectives: Maximization of overall QoE

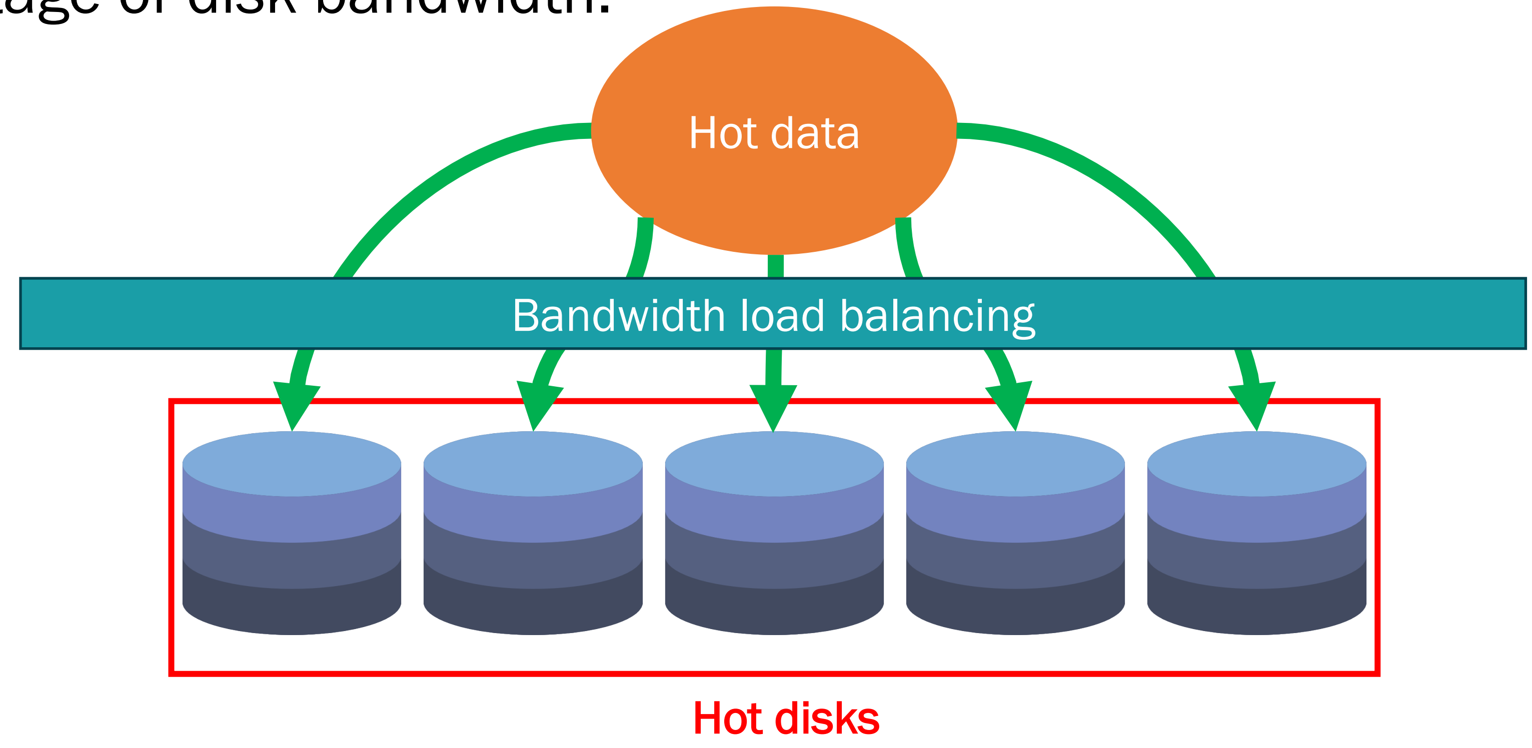


Segment 1	Segment 2	Segment 3	Segment 4
1080p	1080p	1080p	1080p
900p	900p	900p	900p
720p	720p	720p	720p
600p	600p	600p	600p
480p	480p	480p	480p
360p	360p	360p	360p
240p	240p	240p	240p

...

Basic idea – Hot disk allocation

- Hot disk allocation: A hot zone can be composed of multiple disks. It is therefore essential to distribute video segments across disks evenly; otherwise, although segments are stored on hot disks, they may not be accessed due to the shortage of disk bandwidth.



Basic idea – Bandwidth allocation

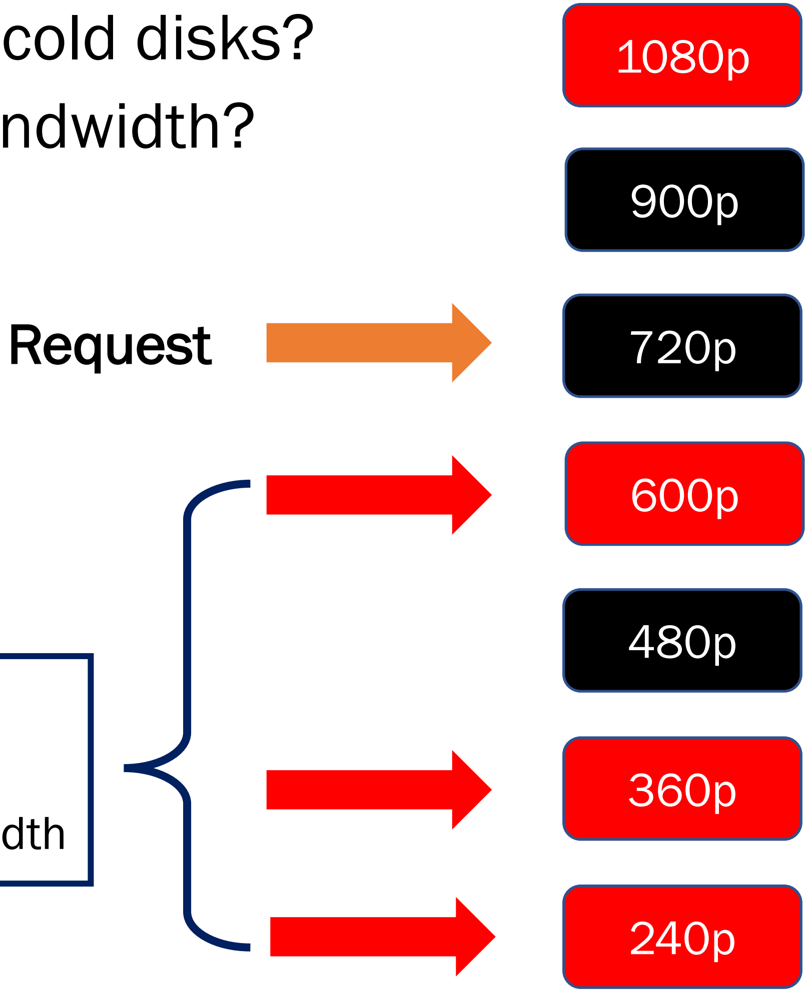
1. What if requested versions are stored on the cold disks?
2. What if hot disks do not have enough disk bandwidth?

Hot disk bandwidth allocation:

1. One of the lower version than requested on the hot disks should be streamed.
2. This inevitably results in QoE degradation, so bandwidth allocation should minimize such overall QoE degradation.

Optimization

QoE maximization subject to disk bandwidth



Basic idea – Summary

- We propose a new disk power management scheme for DASH-based storage video servers.

  - Three algorithms

    1) a hot data determination algorithm to determine versions to be stored on the hot disks with the aim of maximizing QoE gain
    2) a hot disk allocation algorithm to balance workloads among the hot disks
    3) a disk bandwidth allocation algorithm that allocates disk bandwidth to each request to limit power consumption consumed by a disk array with the aim of maximizing overall QoE

Hot data determination algorithm

Hot data determination algorithm

- QoE gain
  - Overall QoE gain, $G_{i,j}$ can be then defined to express the overall QoE achieved for the request to version $V_{i,j}$ as follows:
  - $G_{i,j} = p_{i,j} Q_{i,j}$

Video quality index for version $V_{i,j}$

Access probability of video version j of segment i, $V_{i,j}$, $(i = 1, \dots, N^{seg}\ and\ j = 1, \dots, N^{ver})$, where $\sum_{i=1}^{N^{seg}} \sum_{j=1}^{N^{ver}} p_{i,j}$.

Hot data determination algorithm

- We define the storage selection problem(SSP) as follows:
  - Find $X_{i,j}$ to maximize overall QoE gain subject to storage space

- Maximize $\sum_{i=1}^{N^{seg}} \sum_{j=2}^{N^{ver}-1} X_{i,j} G_{i,j}$
- Subject to $\sum_{i=1}^{N^{seg}} \sum_{j=1}^{N^{ver}} X_{i,j} S_{i,j} \leq S^{total};$ Total size of hot disks($N^{hot} S^{disk}$)

  $X_{i,j} \in 0, 1, (i = 1, \ldots, N^{seg}, j = 2, \ldots, N^{ver} - 1)$

Storage in MB for version $V_{i,j}$

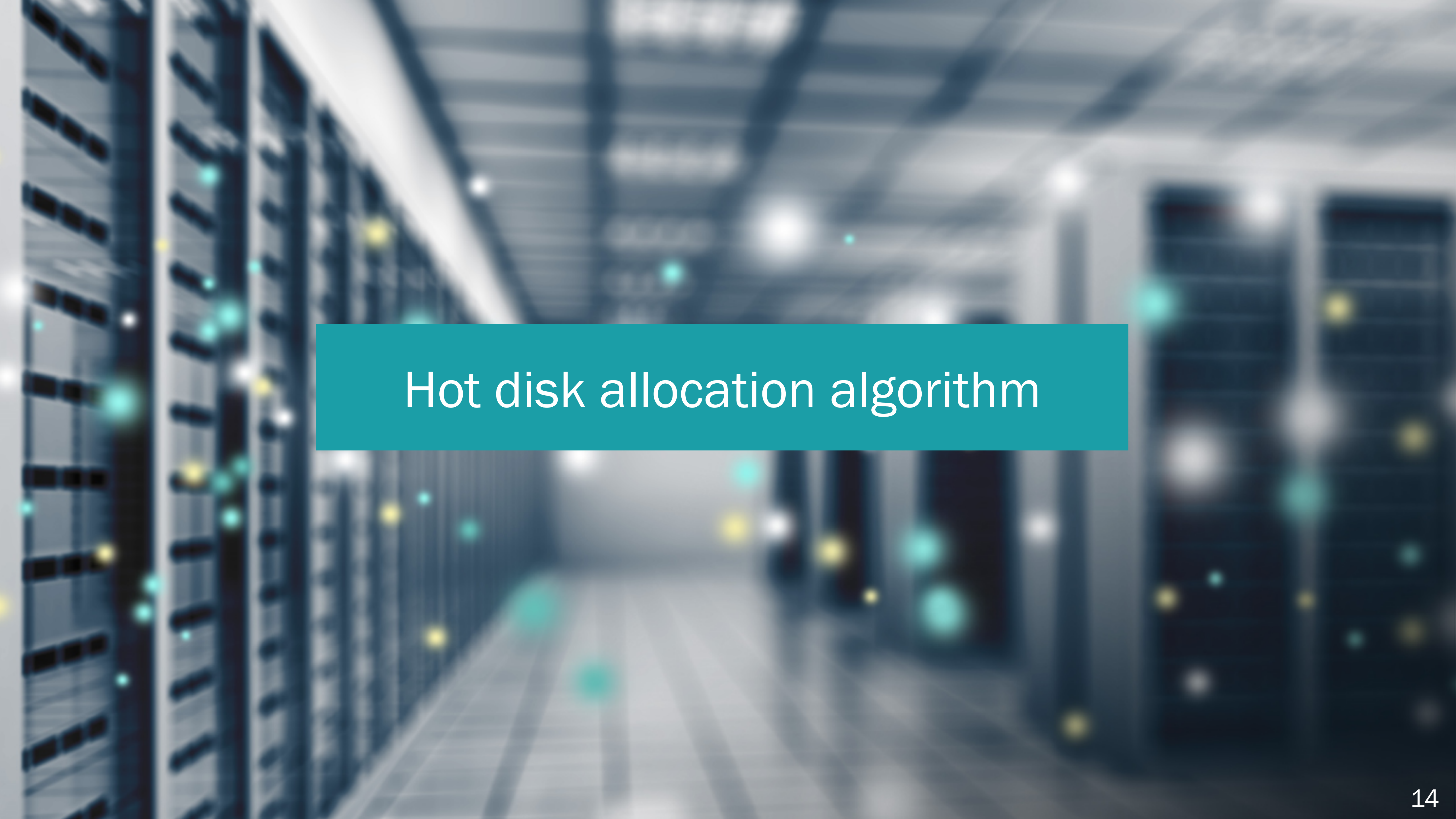
Hot data determination algorithm

- $I_{i,j}$ is defined for each version, $V_{i,j}$ as follows:

$$I_{i,j} = \frac{G_{i,j}}{S_{i,j}}$$

- Select $V_{i,j}$ in greedy technique by considering S^{total} in the order of larger $I_{i,j}$ value.

An array of all the parameters $I_{i,j}$: A;
Temporary variable: $t \leftarrow 0$;
All the values of $X_{i,j}$ are initialized to 0;
while $A \neq \phi$ **do**
    Choose a version $V_{i,j}$ with the highest value of $I_{i,j}$;
    if $t + S_{i,j} \leq S^{\mathrm{total}}$ **then**
        $t \leftarrow t + S_{i,j}$;
        $X_{i,j} \leftarrow 1$;
    end if
    $A \leftarrow A - \{I_{i,j}\}$;
end while



Hot disk allocation algorithm

Hot disk allocation algorithm

- S_k^{prob}
  - Sum of the access probabilities to disk k, $S_k^{prob}, (k = 1, \dots, N^{hot})$ can be expressed as follows:
  - $S_k^{prob} = \sum_{i=1}^{N^{seg}} Y_{i,k} p_i^{hot}$

Express the sum of overall probability for video segment i stored on the hot disks

Binary variable indicating whether video segment i is stored on disk k, $(Y_i = 1, \dots, N^{hot})$

Hot disk allocation algorithm

- We define the hot data allocation problem(HAP) as follows:
  - Find $Y_{i,k}$ that can balance the segment request probability for each disk k.

- Minimize $|max_k S_k^{prob} - min_k S_k^{prob}|$
- Subject to $\forall k, \left(k = 1, \ldots, N^{hot}\right) \sum_{i=1}^{N^{seg}} Y_{i,k} S_i^{hot} \le S^{disk}$,

$Y_{i,k} \in 0, 1, (i = 1, \ldots, N^{seg}, k = 1, \ldots, N^{hot})$

Hot disk allocation algorithm

- The data of the segment i having the highest P_i^{Hot} value is allocated to the disk k having the smallest storage capacity.
- Worst-case Allocation

Temporary variable: $t_k \leftarrow 0$, $(k = 1, \ldots, N^{\text{hot}})$;
Array of segments in non-increasing order of p_i^{hot} values: A;
All the values of $Y_{i,k}$ are initialized to 0;
while $A \neq \phi$ **do**
    Choose a segment i with the highest value of p_i^{hot};
    Assign segment i to disk k with the smallest value of t_k;
    $t_k \leftarrow t_k + S_i^{\text{hot}}$;
    $Y_{i,k} \leftarrow 1$;
    Remove segment i from A;
end while

Bandwidth allocation algorithm

Bandwidth allocation algorithm

- Service time
  - Service time to serve jth version of request m, and can be calculated as follows :
  - $T_{m,j} = \boxed{T_s} + \frac{\boxed{b_{V(m),j}}L}{tr}$

Bit-rate for the jth version of the requested video

Seek time

- Ratio parameter
  - Ratio parameters for the jth version of request m, $R_{m,j}$, where $X_{V(m),j} = 1$, are defined as follows:
  - $R_{m,j} = \frac{Q_{m,V_m^{req}} - Q_{i,j}}{T_{m,V_m^{req}} - T_{m,i}}$

- V(m) : video number for request m
- D(m) : disk number for request m
- Z_m : version number to be serviced to the client for request m

Bandwidth allocation algorithm

- We define the bandwidth allocation problem(BAP) as follows:
  - Maximize overall QoE subject to disk bandwidth constraint for each disk.
  - The total service time taken to read all the segments must not exceed the segment length, L to guarantee continuous delivery of video streams.

- Maximize $\sum_{m=1}^{N^{req}} Q_{V(m),Z_m}$
- Subject to $\forall\ disk\ k,\ \sum_{\forall m\ where\ D(m)=k} T_{m,Z_m} \leq L$

Bandwidth allocation algorithm – Heuristic

- The lowest value of $R_{m,low}$ in the set of $R_{m,j}$'s needs to be selected for the candidate version index, low, to minimize QoE degradation.

Array of all the values of $R_{m,j}$'s: A;
Temporary variables: t_k, $(k = 1, ..., N^{\text{hot}})$;
$t_k \leftarrow \sum_{\forall m \text{ where } D(m)=k} T_{m, V_m^{\text{high}}}$, $(k = 1, ..., N^{\text{hot}})$;
$\forall\ m, Z_m \leftarrow V_m^{\text{req}}$;
while $TRUE$ **do**
    if $\forall\ k, t_k \leq L$ **then**
        break;
    end if
    Choose the lowest value of $R_{m,low}$ in the array A and removes $R_{m,low}$ from A;
    if $low < Z_m$ and $t_{D(m)} + T_{m,low} - T_{m,Z_m} > L$ **then**
        $t_{D(m)} \leftarrow t_{D(m)} + T_{m,low} - T_{m,Z_m}$;
        $Z_m \leftarrow low$;
    end if
end while

Bandwidth allocation algorithm – Dynamic programming

- We developed a dynamic programming (DP) algorithm that runs for each disk
  - Let $V_{m,n}^{qoe}$ be the maximum QoE when n is the time length.
  - $V_{m,n}^{qoe}$ are all initialized to zero.
  - $V_{1,n}^{qoe}$ is calculated as follows:

$$V_{1,n}^{qoe} = \max_{j \in \{j | T_{m,j} \leq n\}} Q_{V(m),j} .$$

  - If $V_{m-1,n-T_{m,j}}^{qoe} > 0$, then $V_{m,n}^{qoe}$ can be updated using the following recurrence:

$$V_{m,n}^{qoe} = \max_{j=1,\ldots,V_m^{req}} \left\{ \max \left(V_{m,n-1}^{qoe}, V_{m-1,n-T_{m,j}}^{qoe} + Q_{V(m),j} \right) \right\}.$$

However, since the bandwidth allocation must be done in real time for each disk, we must use the heuristic algorithm BAA in consideration of time complexity.

Experimental results

- Simulation environments

- Monitoring period
  - 24 hours
- The access probability follows a Zipf distribution
  - Segment popularity: $\theta = 0.271$
- Requests arrival rate
  - Poisson process
  - Inter-arrival time : 1/sec
- Video characteristics (2000 videos)
  - Randomly selected between 2 and 3 hours

Experimental results

- Algorithm efficiency

- Hot data determination algorithm(HDA)
  - The maximum difference between HDA and upper bound was found to be almost zero, indicating that HDA yields a near-optimal solution to SSP.
- Hot disk balancing algorithm(HBA)
  - Table tabulated the percentage difference for $max_k S_k^{prob}$, $min_k S_k^{prob}$, which clearly shows that HBA can balance overall popularity over each disk

(HVP, 2TB)	(LVP, 2TB)	(HVP, 3TB)	(LVP, 3TB)
0.01%	0.04%	0.01%	0.02%

Experimental results

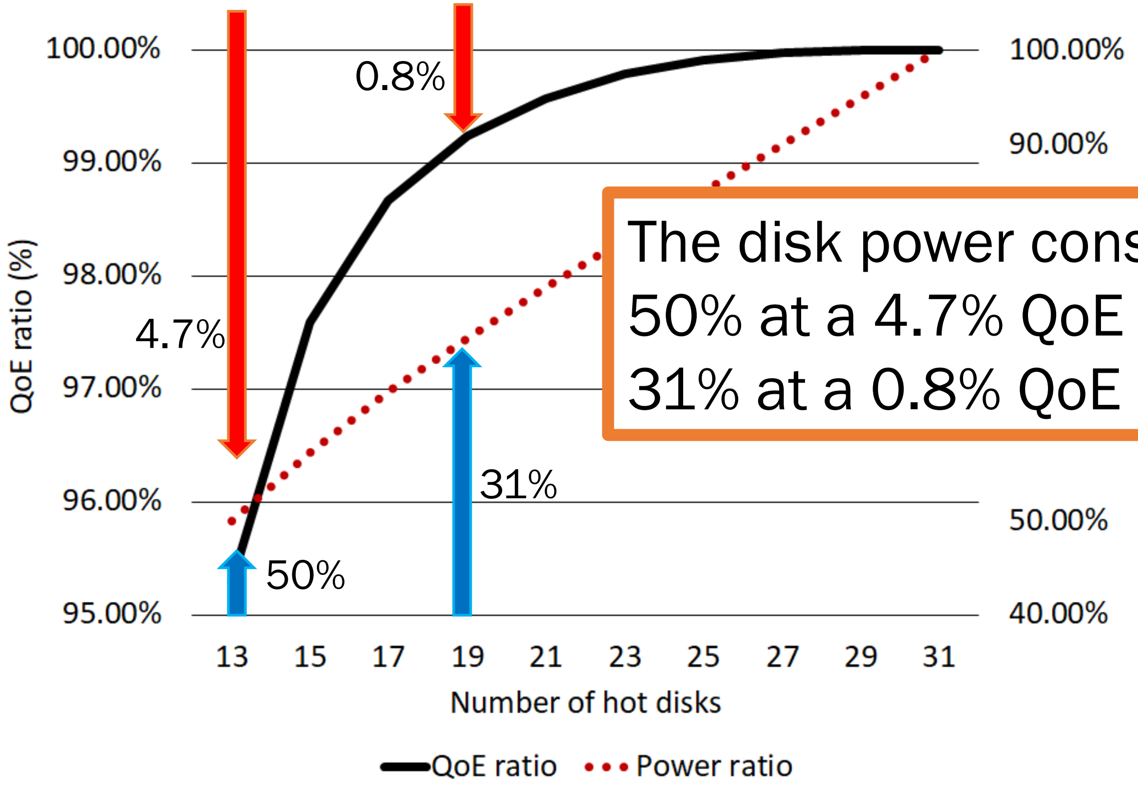
- Algorithm efficiency

- Bandwidth allocation algorithm(BAA)
  - Table shows that the average percentage difference between BAA and DP.
  - The difference is almost negligible, indicating that BAA produces a near-optimal solution within a reasonable time.
    - DP : 36.5s
    - BAA : 4.5ms

(HVP, 2TB)	(LVP, 2TB)	(HVP, 3TB)	(LVP, 3TB)
0.004%	0.005%	0.3%	0.02%

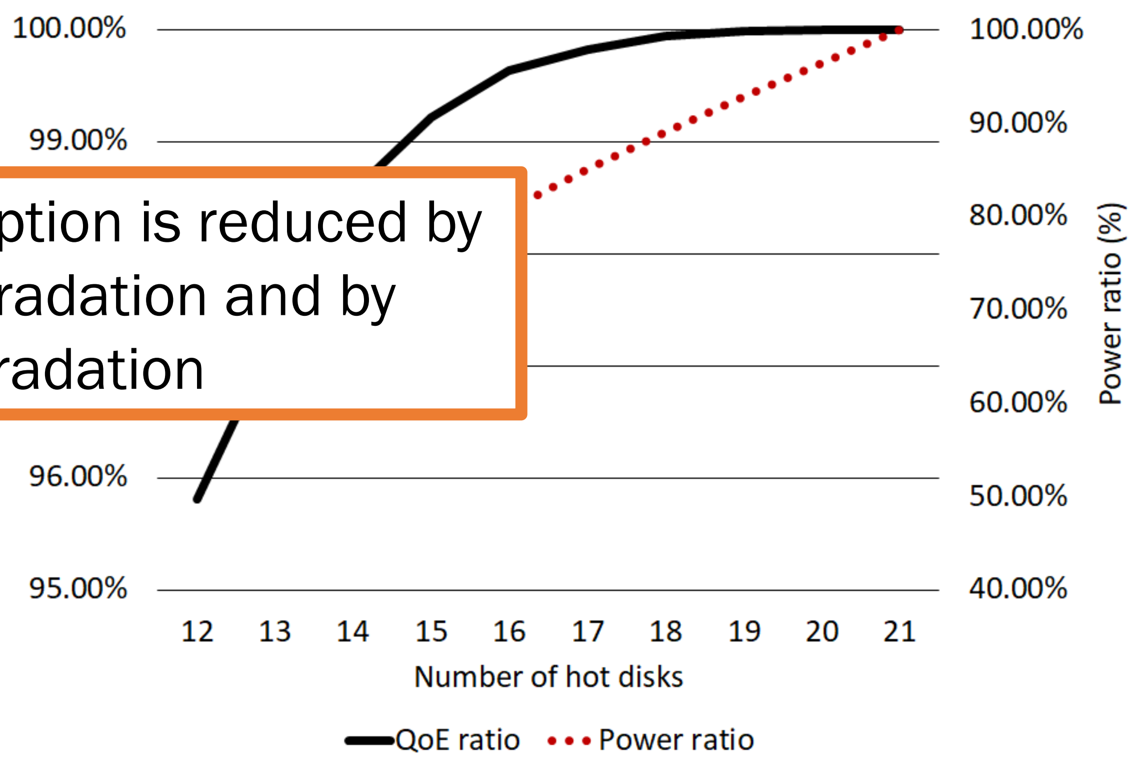
Experimental results

- Power and QoE results – HVP (High-version-popular) workloads



S^{disk} = 2TB

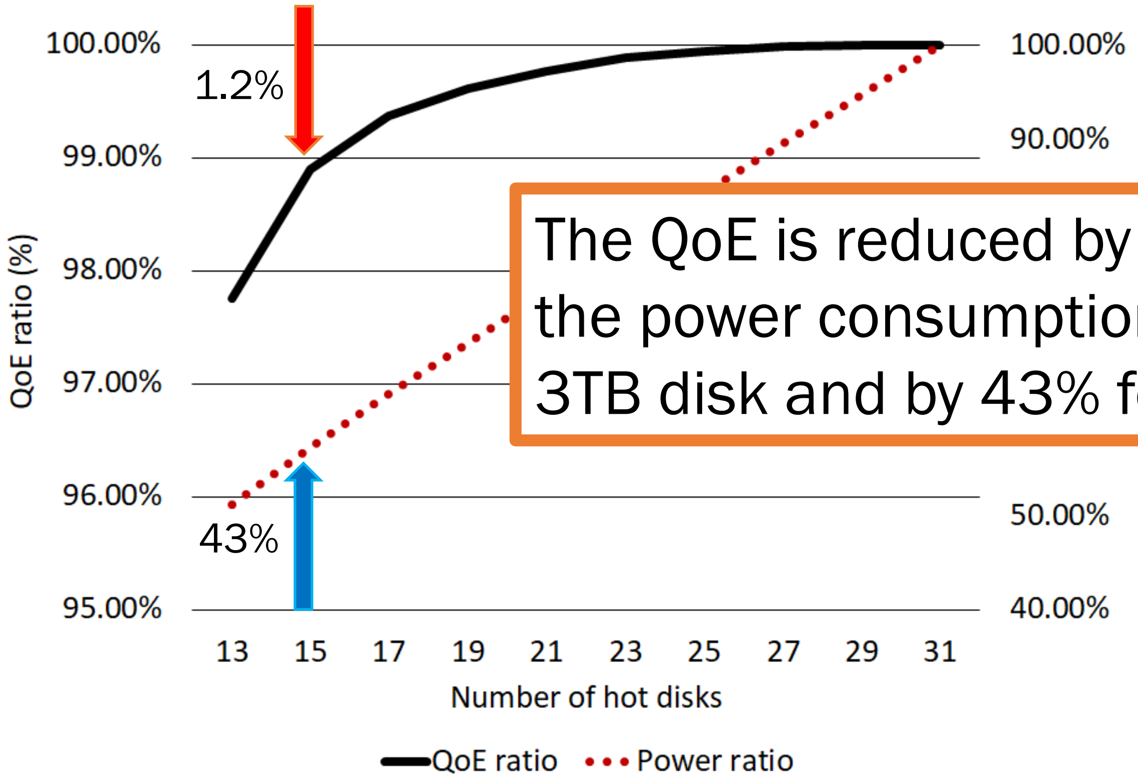
The disk power consumption is reduced by 50% at a 4.7% QoE degradation and by 31% at a 0.8% QoE degradation



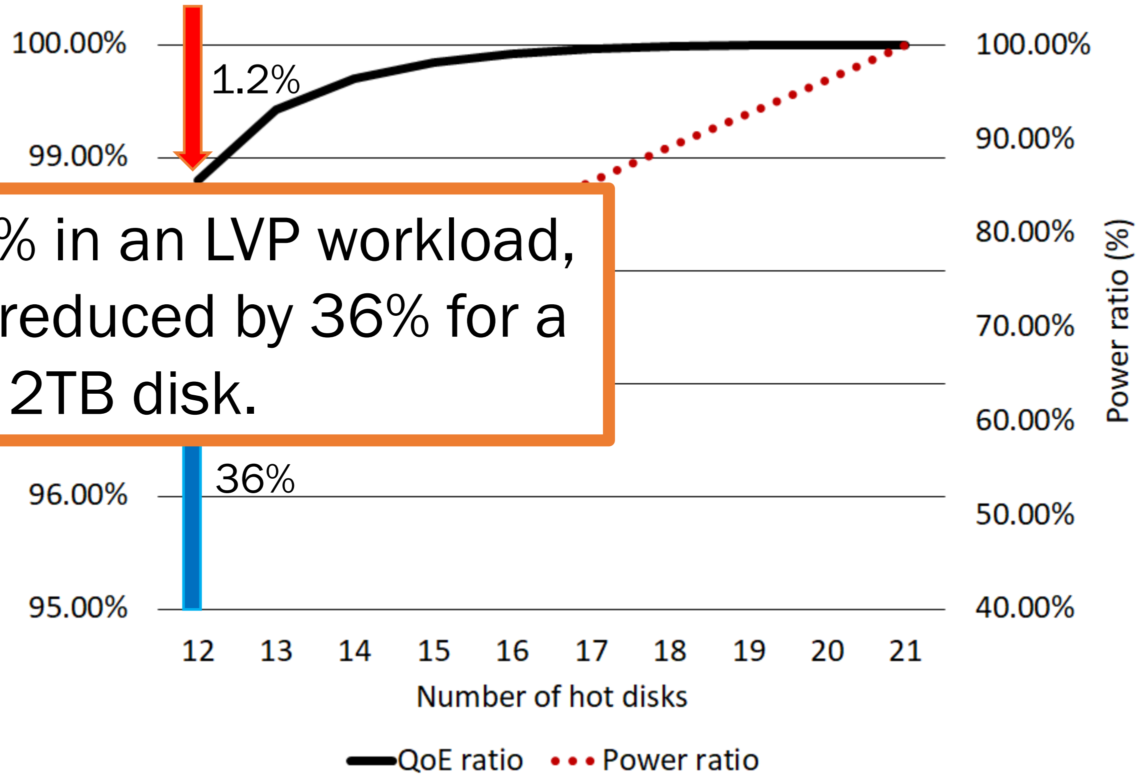
S^{disk} = 3TB

Experimental results

- Power and QoE results – LVP (Low-version-popular) workloads



S^{disk} = 2TB

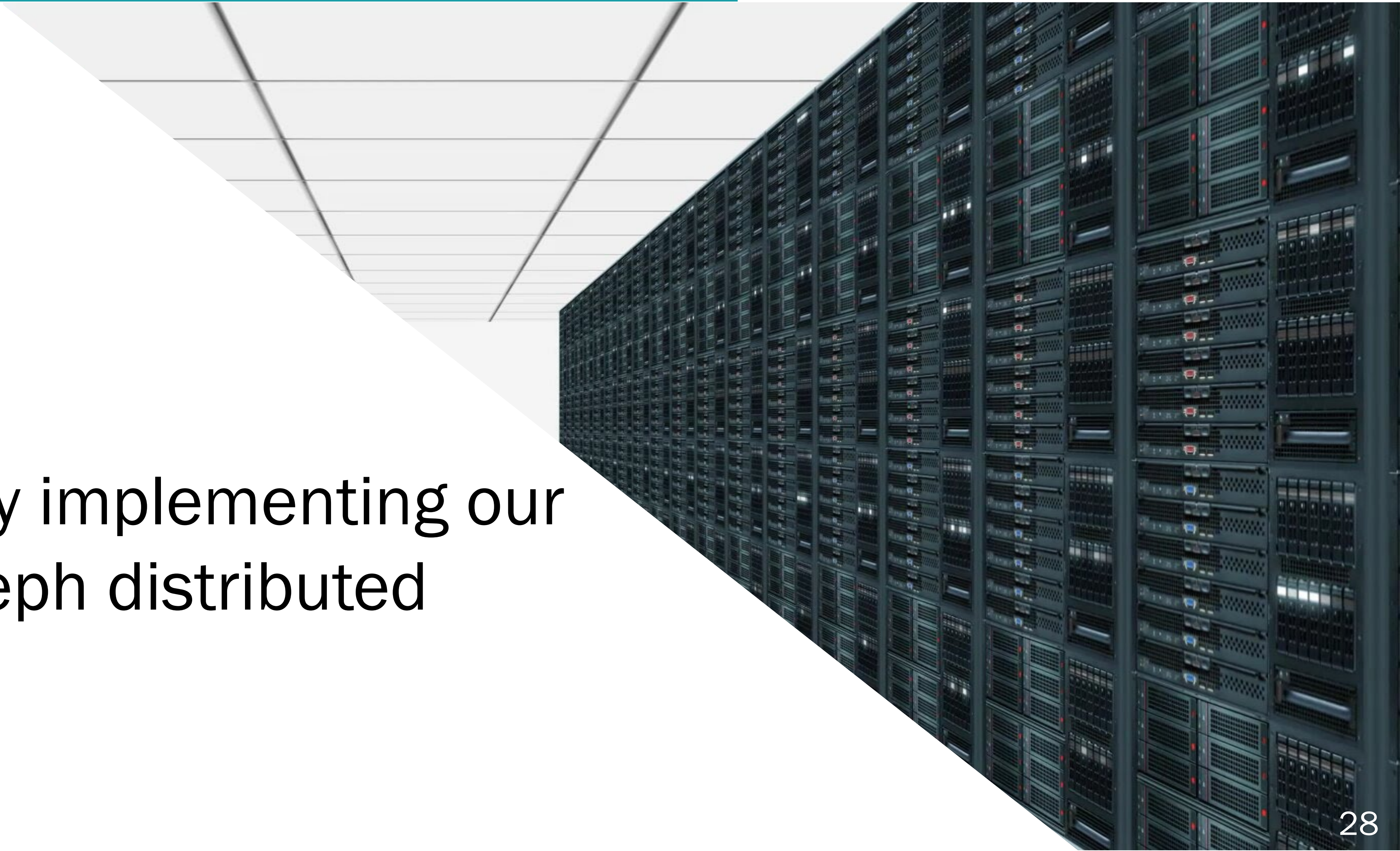


S^{disk} = 3TB

The QoE is reduced by 1.2% in an LVP workload, the power consumption is reduced by 36% for a 3TB disk and by 43% for a 2TB disk.

On-going work

ceph



We are currently implementing our scheme on a Ceph distributed storage system.

Conclusions

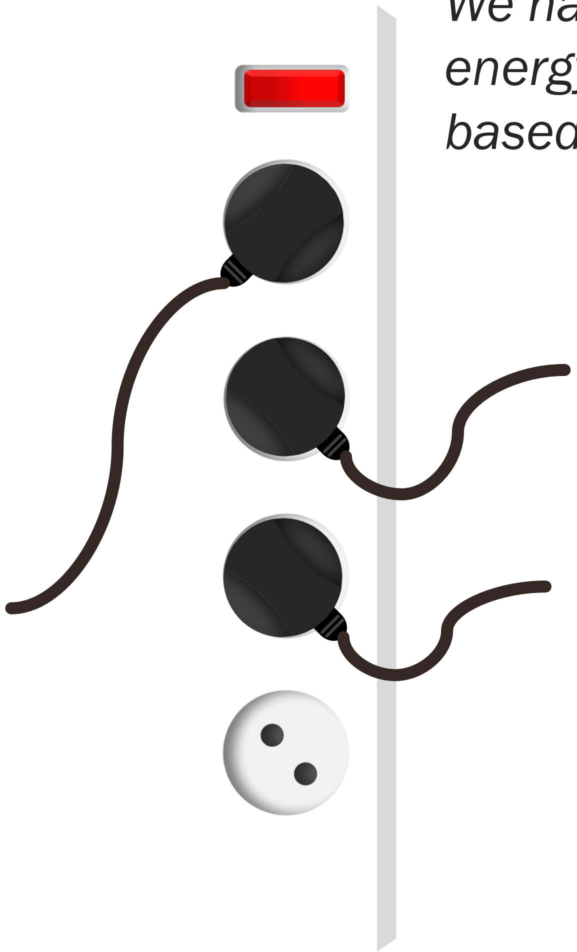
Power saving

29% ~ 46%

QoE degradation

1.5%

We have proposed a new scheme to minimize disk energy consumption in DASH-based video servers based on Hot and Cold storage concept



Hot Data Determination Algorithm

Determines the versions to be stored on hot disks

Hot Disk Balancing Algorithm

Balances the workloads among hot disks

Bandwidth Allocation Algorithm

Determines the bit-rate of segments with the aim of minimizing QoE degradation subject to disk bandwidth constraint

QnA

QoE ?

Table 1: Mapping between SSIM and QoE values [17].

$SSIM_{i,j}$	$Q_{i,j}$	Meaning
$SSIM_{i,j} \geq 0.99$	5	excellent
$0.95 \leq SSIM_{i,j} < 0.99$	$25SSIM_{i,j} - 19.75$	good
$0.88 \leq SSIM_{i,j} < 0.95$	$14.29SSIM_{i,j} - 9.57$	fair
$0.5 \leq SSIM_{i,j} < 0.88$	$3.03SSIM_{i,j} + 0.48$	poor
$SSIM_{i,j} < 0.5$	1	bad

We use mapping relationship between SSIM (compared with the highest quality) and MOS in the table. For this purpose, we measured SSIM values of several sample video clips.

For example, MOS of original workloads : 4.2

What does 1.5% degradation mean ?

MOS of our algorithm : 4.137