2008 High End Computing File Systems and I/O Workshop

SAM^2 Toolkit: Scalable and Adaptive Metadata Management for High-End Computing

Hong Jiang, David Swanson
University of Nebraska

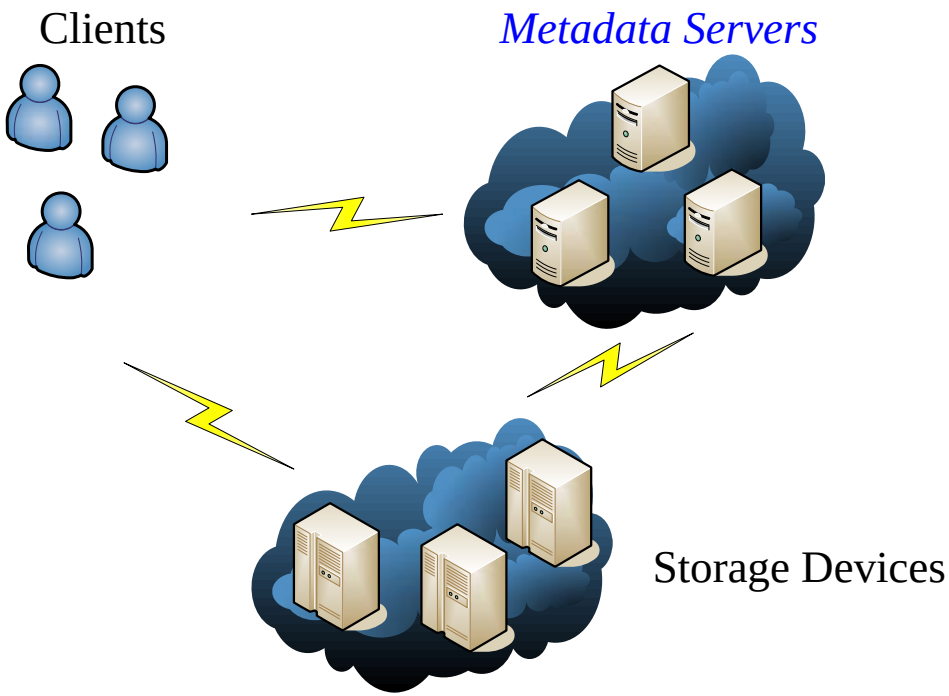
Yifeng Zhu
University of Maine

Jun Wang
University of Central Florida

Main Research Results

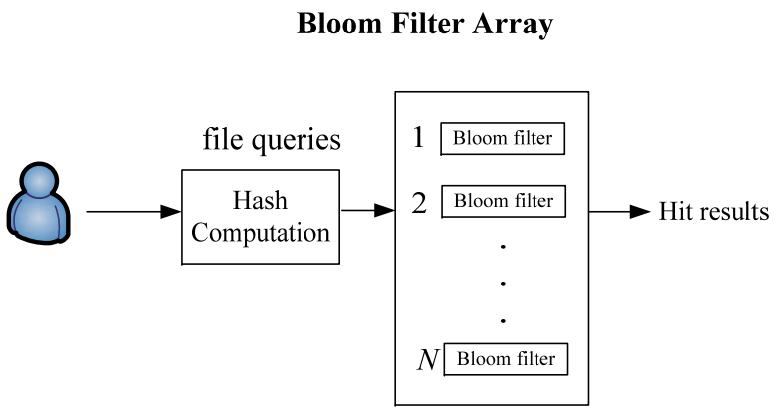
- **Performance Optimization Through Semantic Mining & Grouping**
  - **(ICDCS'08)** GHBA: Scalable and Adaptive Metadata Management in Ultra Large-scale File Systems
  - **(HPDC'08)** FARMER: File Access corRelation Mining and Evaluation Reference model for peta-scale file systems
  - **(CCGrid'08)** AMP: An Affinity-based Metadata Prefetching scheme for metadata servers
  - **(ICPP'08)** SOGP: Segment Structured On-disk Data Grouping and Prefetching
  - **(Ph.D. dissertation)** Metadata and Data Management in High Performance File and Storage Systems
- **Workload Modeling**
  - **(MASCOTS'08)** A Novel and Generic Model for Synthesizing Disk I/O Traffic Based on The Alpha-stable Process
  - **(Cluster'08)** A Novel Model for Synthesizing Parallel I/O Workloads in Scientific Applications
- **Energy Efficiency Optimization**
  - **(MASCOTS'08)** GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability
  - **(MASCOTS'08)** Energy Efficient Buffer Cache Replacement
  - **(ICPP'08)** Impacts of Indirect Blocks on Buffer Cache Energy Efficiency
  - **(Cluster'07, Best Paper Award)** Evaluating Memory Energy Efficiency in Parallel I/O Workloads

Data and Metadata Decoupled Structure



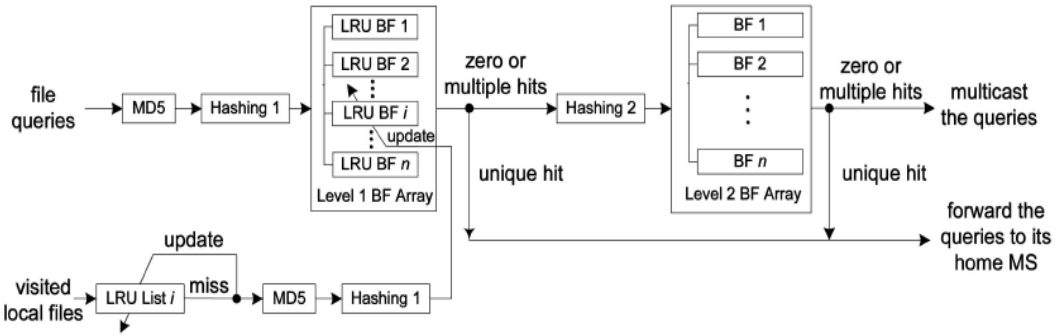
- A single *metadata server* (*MDS*) can still become a severe performance bottleneck.
- In large-scale storage systems, *multiple metadata servers* are desirable for improving scalability.

Fast Metadata Lookups



A simple and naïve solution, but introducing problems:

- *Query accuracy;*
- *Updating costs.*

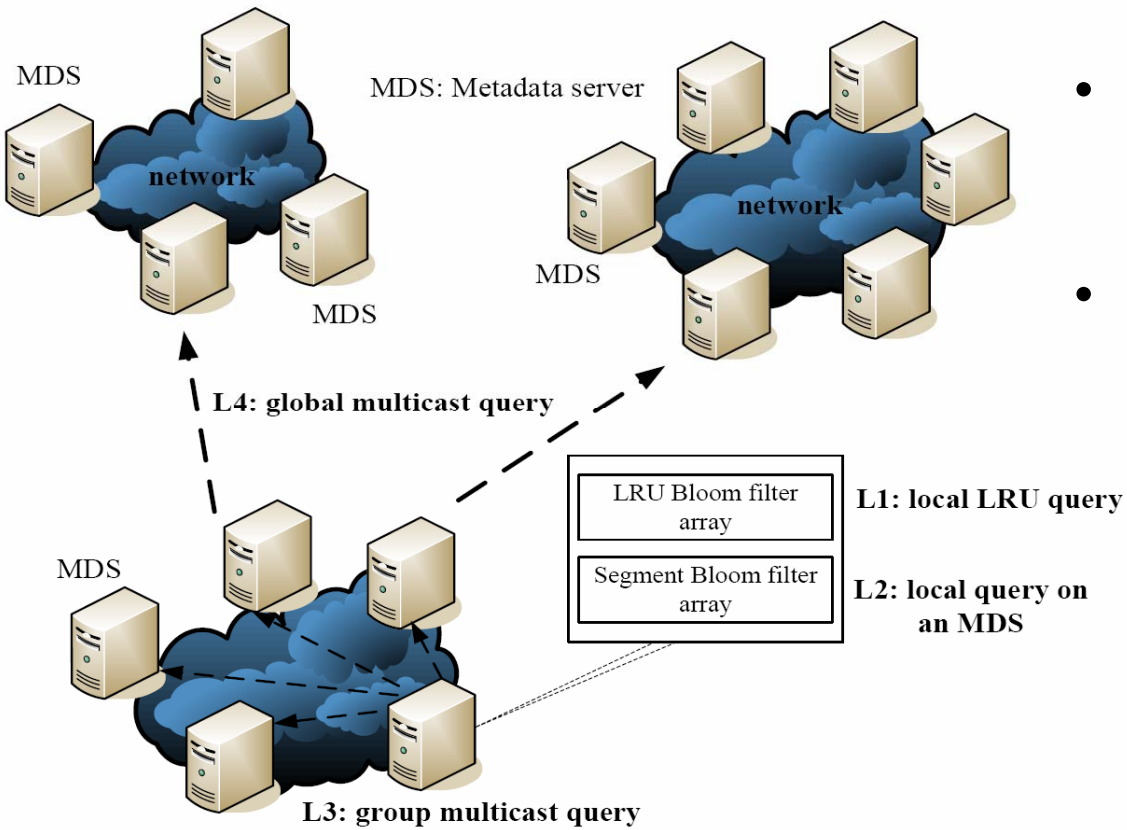


An improved solution, still a centralized :

- *Space overheads;*
- *Query latency;*
- *Dynamic operations..*

Y. Zhu, H. Jiang, J. Wang and F. Xian, "HBA: Distributed Metadata Management System for Large Cluster-based Storage", *IEEE Transaction on Distributed and Parallel Systems*, Vol. 19, No. 6, pp. 750-763, June 2008

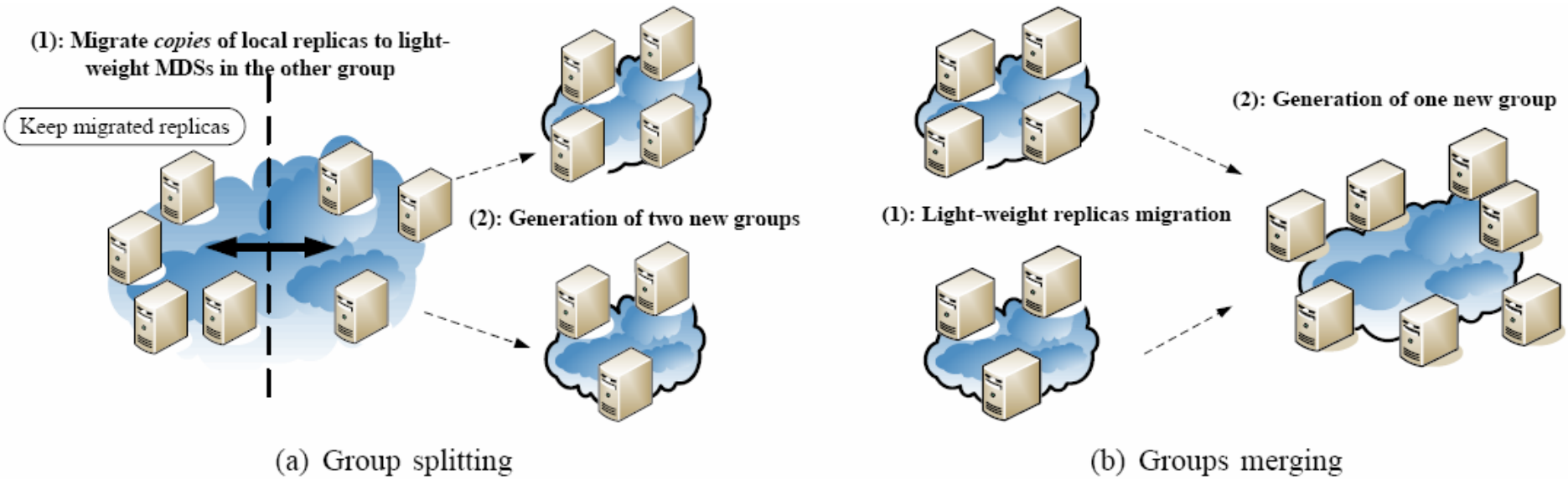
G-HBA: Group-based Hierarchical Bloom filter Array



- The basic idea is to decentralize metadata management among multiple groups of MDSs;
- ***Design objective***:
  - Improving scalability and query efficiency.
  - Supporting dynamic and self-adaptive operations in ultra large-scale file systems.

(ICDCS'08) GHBA: Scalable and Adaptive Metadata Management in Ultra Large-scale File Systems

Dynamic and Adaptive Group Reconfigurations



(a) Group splitting (b) Groups merging

Identify optimal group size M through a simple benefit function:

$$\Gamma = \frac{U_{G\text{-}HBA}(throu.)}{U_{G\text{-}HBA}(space)} = \frac{1}{U_{G\text{-}HBA}(laten.) * U_{G\text{-}HBA}(space)}$$

where

$$U_{G\text{-}HBA}(space) = \frac{N-M}{M}$$

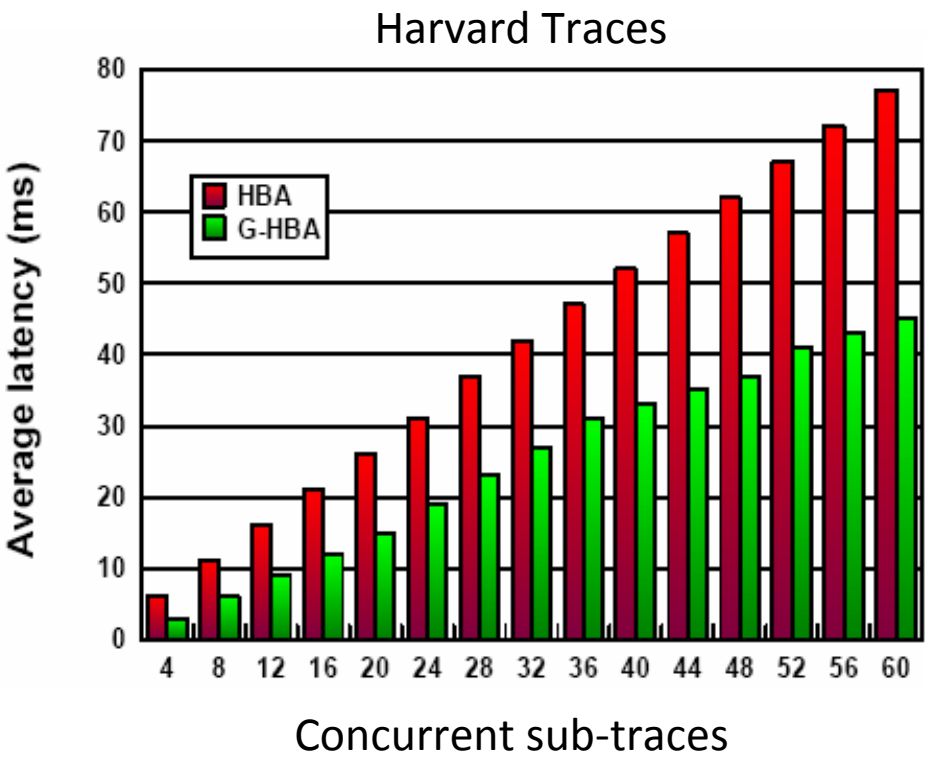
$$U_{G-HBA}(laten.) = D_{LRU} + (1-P_{LRU})D_{L2} + (1-P_{LRU})(1-\frac{P_{L2}}{M})D_{group} + (1-P_{LRU})(1-\frac{P_{L2}}{M})^{M}D_{net.}$$

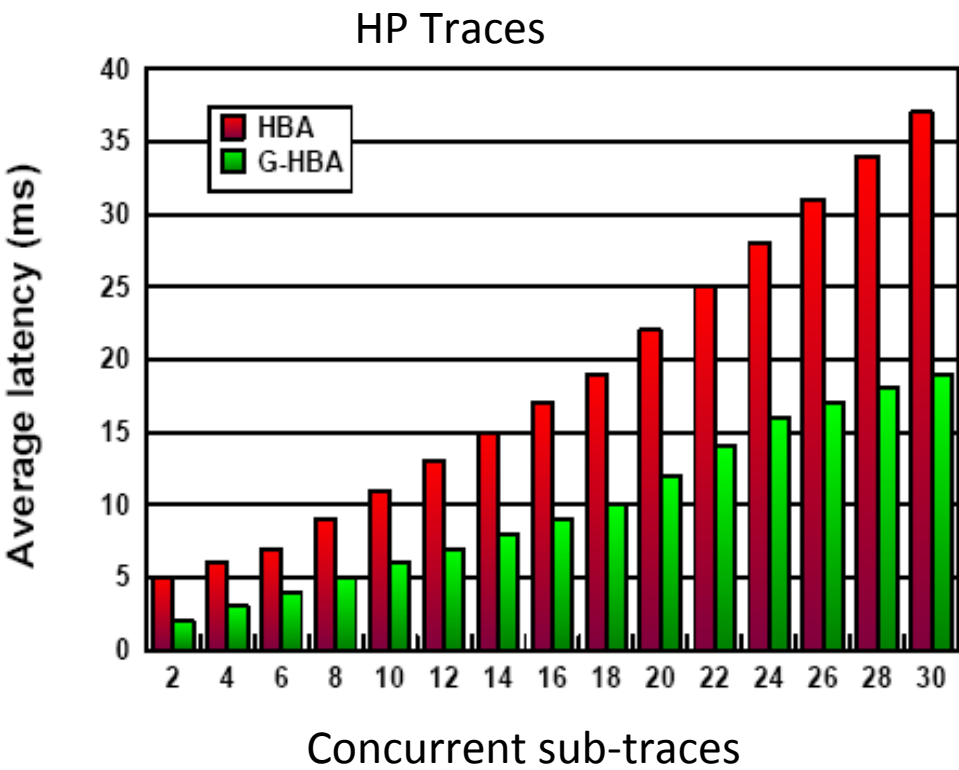
Table 1. Symbol representations.

Symbol	Description
P_{LRU}	Unique hit rate in the LRU Bloom filters
P_{L2}	Unique hit rate in the 2nd level Bloom filters
D_{LRU}	Latency in the LRU Bloom filters
D_{L2}	Latency in the 2nd level Bloom filters
D_{group}	Latency in one group
$D_{net.}$	Latency in entire multicast network

Performance Evaluation

- Prototype Implemented in a Linux Cluster with 60 MDS
- Use trace playback tool FEUT to create reproducible benchmarks based on traces collected on actual systems
- Reduce latency by 48.% and 41.5% latency under HP & Harvard traces





(ICDCS'08) GHBA: Scalable and Adaptive Metadata Management in Ultra Large-scale File Systems

Metadata Prefetching Algorithm 1: Motivation

- File Correlation
  - Files accessed by the same user tend to have strong correlations
  - Individual program typically access the same files in the same order
  - Strong correlation among files stored in the same directory
- File correlation mining (existing approaches)
  - File access sequence mining (access frequency)
    - Last Successor, First Successor, Nexus
  - File semantic attribute mining (semantic similarity)
    - Semantically Smart Disk Systems (SDS), C-Miner
- Our contribution
  - Judiciously considers both file access sequences and semantic attributes simultaneously

(HPDC'08) FARMER: File Access corRelation Mining and Evaluation Reference model for peta-scale file systems

Metadata Prefetching Algorithm 1
FARMER: CoMiner

- Evaluating Semantic Distance
  - Use vector space model to measure semantic correlation between files

$$Sim(A,B) = \frac{|A \cap B|}{|max\ (A,B)|}$$

A, B: *semantic vectors* of file *A* and *B*

A∩B: equal items between A and B

max(A, B): maximum number of items

- Evaluating Access Frequency
  - Frequency of accesses in which file *B* is the successor of file *A* :

$$F(A,B) = N_{AB}/N$$

N_{AB}: the number of occurrences that file *B* is the successor of *A*

N: the total number of accesses of file *A*

FARMER: CoMiner(Cont')

- File Correlation Degree

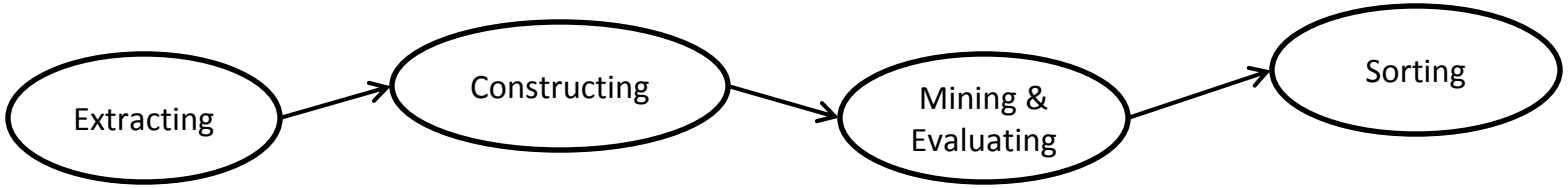
$$R(A,B) = Sim(A,B) \cdot p + F(A,B) \cdot (1-p)$$

p: a tunable weight

Sim(A,B): semantic distance,

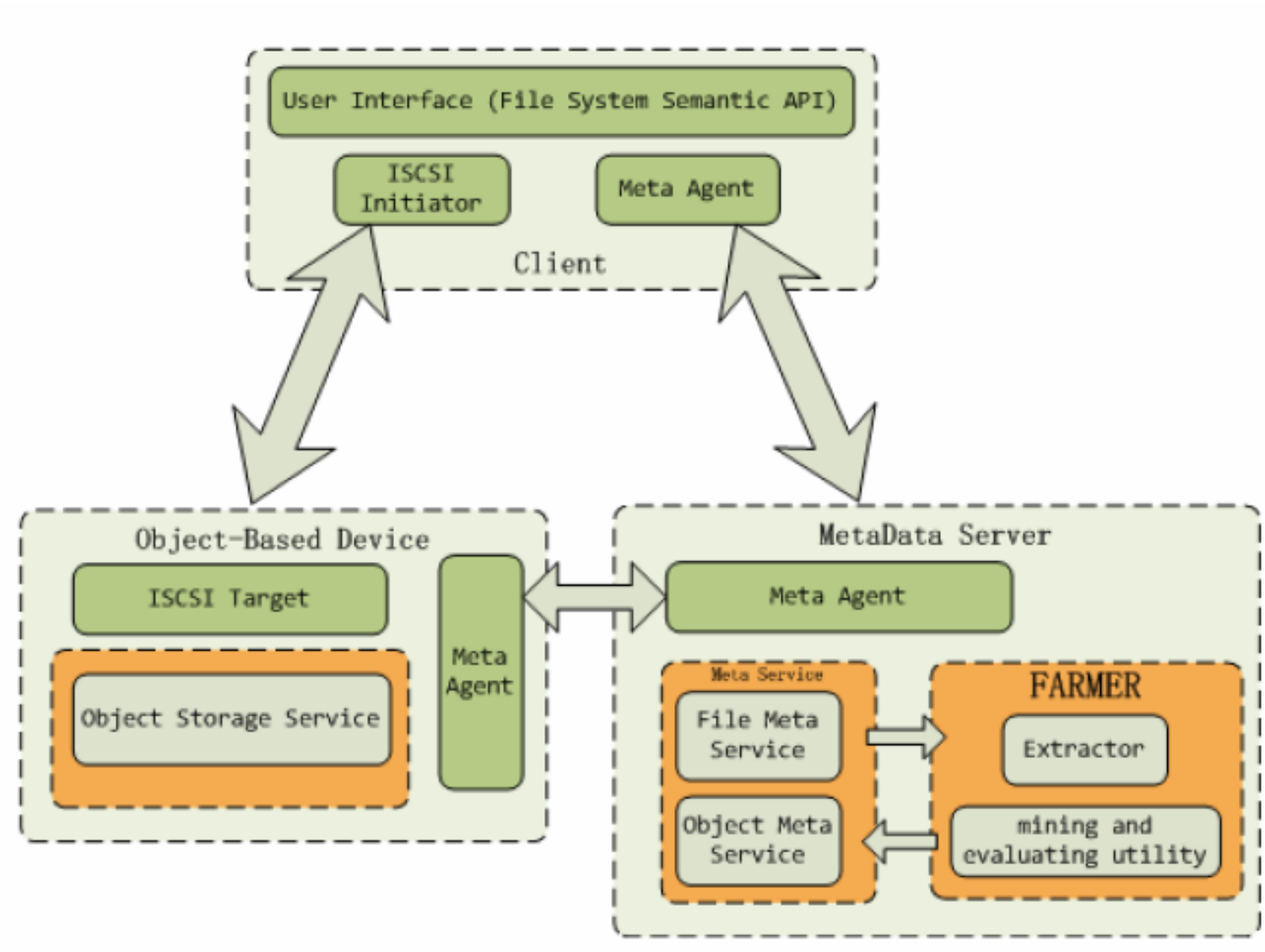
F(A,B): access frequency

- Four-stage processing



In this stage, FARMER collects the access sequence and evaluates the access sequence to compute file correlation degrees, and aggregates the correlation degrees into a file graph, the node represents an accessed file and the directed edge represents an access order, weight on each edge equals the value of correlation degree.

FARMER: Experiment Architecture



(HPDC'08) FARMER: File Access corRelation Mining and Evaluation Reference model for peta-scale file systems

Case Study – Prefetching Metadata

- Prefetching Accuracy
  - Compared to Nexus

    Improve about 20% (HP trace)
- Response time
  - Compared to Nexus

    Reduce by about 22.8% (LLNL), 12.3% (RES), 24.0% (HP trace)
  - Compared to LRU

    Reduce by about 27.6% (LLNL), 23.3% (RES), 36.1% (HP trace)
- Cache hit ratio
  - Compared to Nexus

    Improve about 10.6% (LLNL), 7.8% (RES), 13.1% (HP trace)
  - Compared to LRU

    Improve by about 18.4% (LLNL), 13.3% (RES), 25.3 % (HP trace)

(HPDC'08) FARMER: File Access corRelation Mining and Evaluation Reference model for peta-scale file systems

Metadata Prefetching Algorithm 2: Affinity-based Metadata Prefetching (AMP)

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)

(BCADEF)

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)

(BCADEF)

(CADEFA)

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)

(BCADEF)

(CADEFA)

(ADEFAB)

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)
(BCADEF)
(CADEFA)
(ADEFAB)
(DEFABE)

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)
(BCADEF)
(CADEFA)
(ADEFAB)
(DEFABE)
(EFABE)

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)
(BCADEF)
(CADEFA)
(ADEFAB)
(DEFABE)
(EFABE)
(FABE)

AMP: Scheme Description

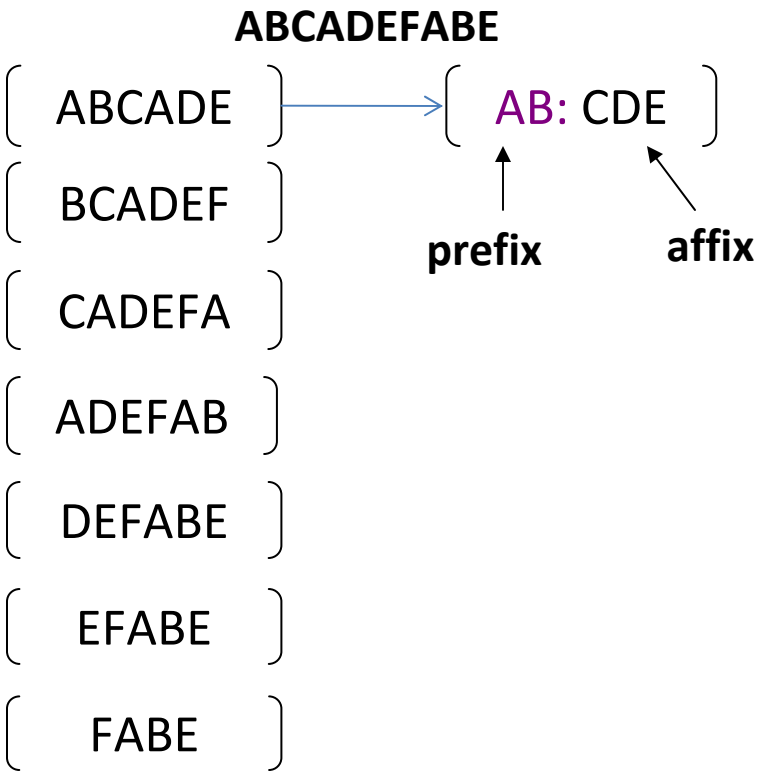
- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

ABCADEFABE

(ABCADE)
(BCADEF)
(CADEFA)
(ADEFAB)
(DEFABE)
(EFABE)
(FABE)

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:



(CCGrid'08) AMP: An Affinity-based Metadata Prefetching scheme for metadata servers

AMP: Scheme Description

- Step 1: According to the fixed group size, a slide window is adopted to divide the training trace into different groups. The first two items are fixed and the same items in each group are filtered.
  - Example training trace:

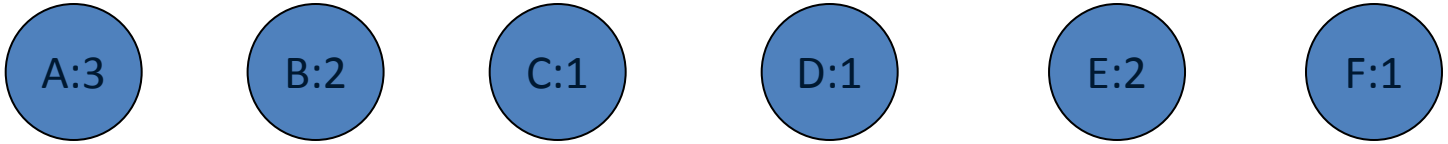
ABCADEFABE

ABCADE	AB: CDE
BCADEF	BC: ADEF
CADEFA	CA: DEF
ADEFAB	AD: EFB
DEFABE	DE: FAB
EFABE	EF: AB
FABE	FA: BE

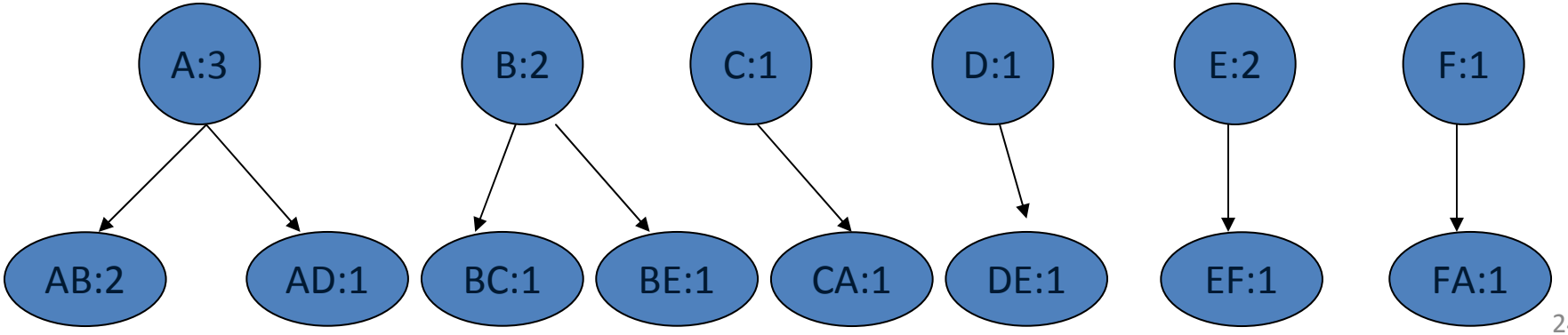
(CCGrid'08) AMP: An Affinity-based Metadata Prefetching scheme for metadata servers

AMP: Scheme Description (Cont')

- STEP 2: find out different items in the training trace and use these items as the root of the tree.
  - "A:3" indicates that the frequency of A is three.



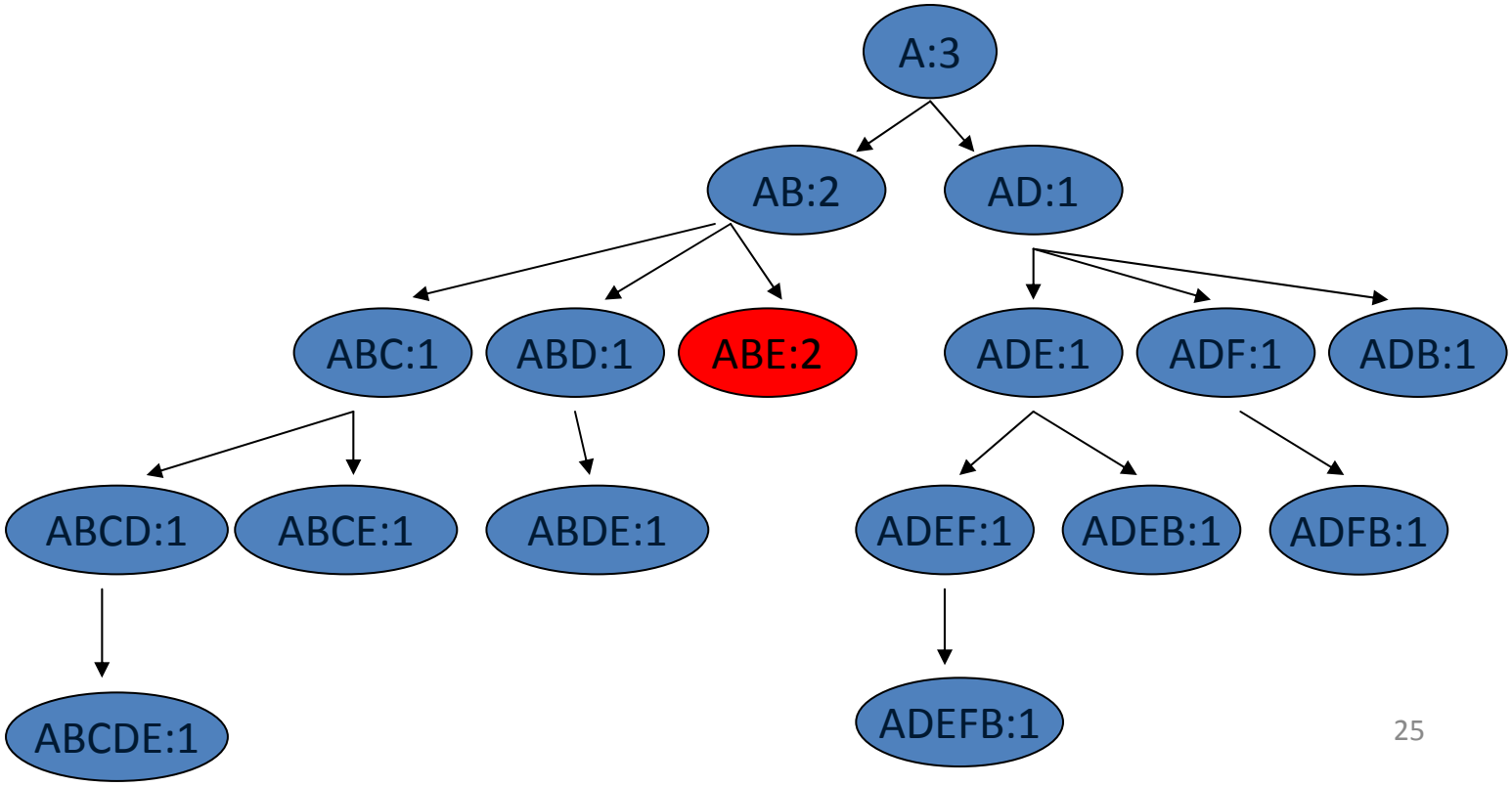
- STEP 3: allocate the first two items of each group to the corresponding root.



(CCGrid'08) AMP: An Affinity-based Metadata Prefetching scheme for metadata servers

AMP: Scheme Description (Cont')

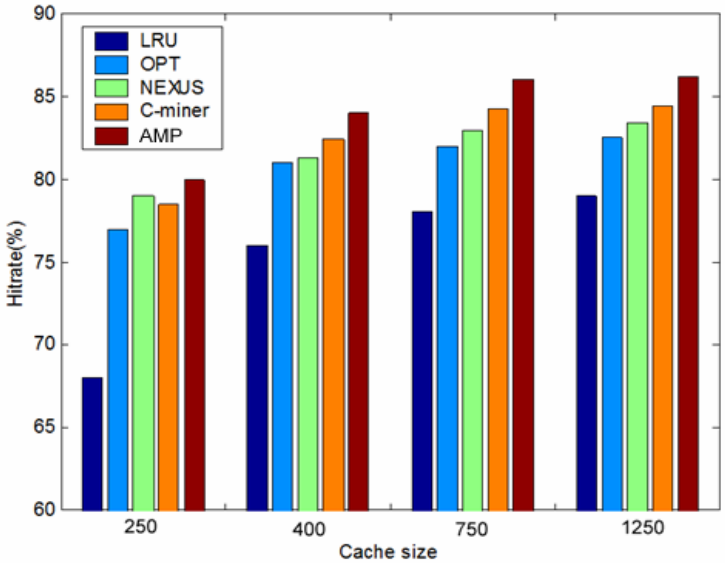
- STEP 4: allocate rest items of each group to the corresponding tree, and find out the prefetching candidates from the tree.



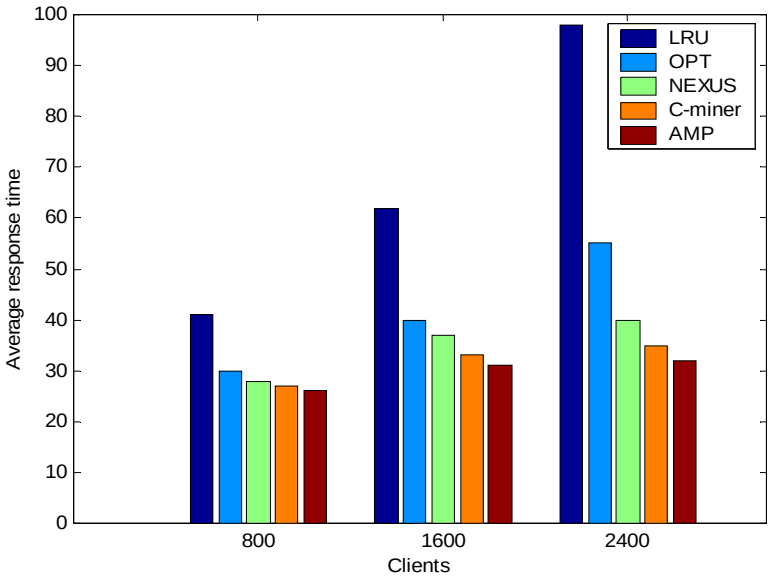
When A is missed, we prefetch B and E according to the prefetching candidate with the largest frequency.

AMP: Simulation Results

HP File Server Traces



Client local hit rate.

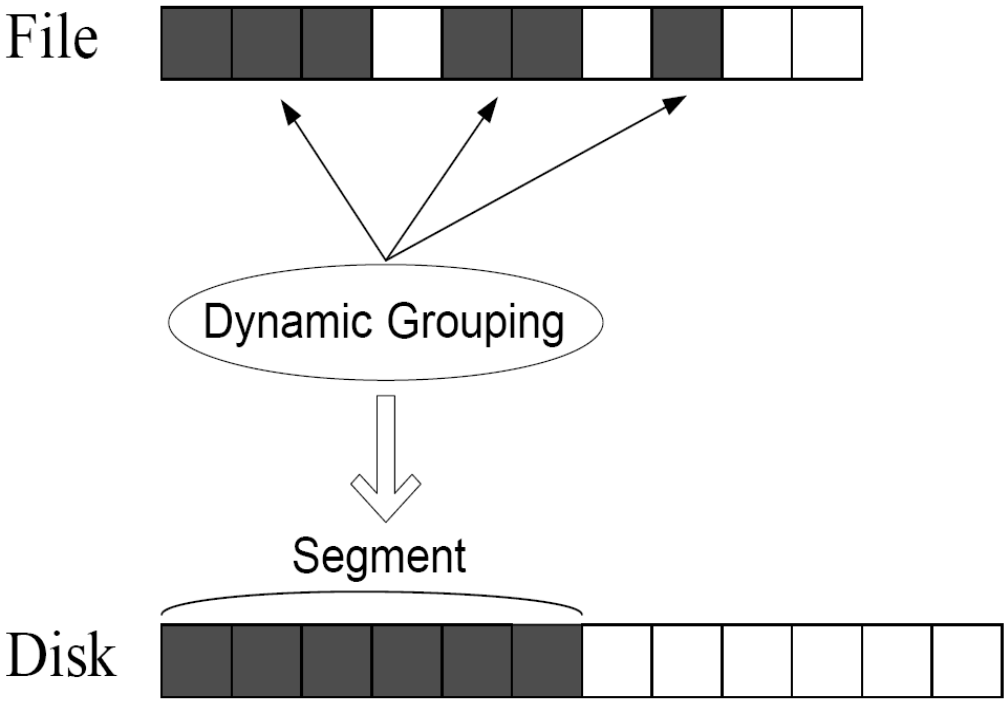


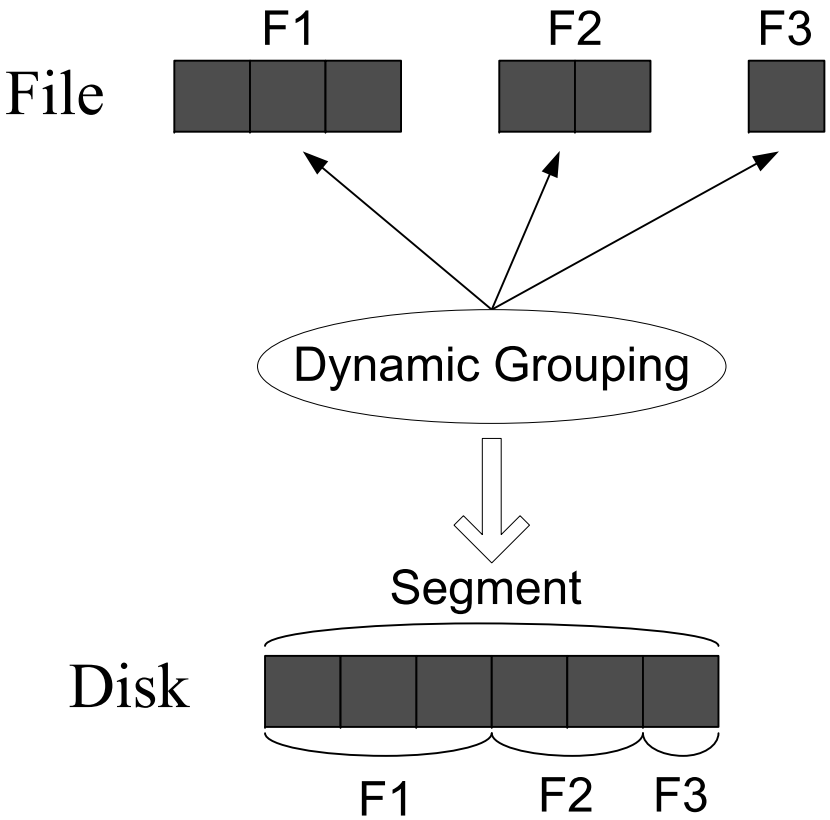
Average response time for 8 MDS, 800-2400 clients.

Compared with LRU, Nexus and C-miner, trace-driven simulations show that AMP improves the hit rates by up to 12%, 4.5% and 4%, while reduce the average response time by up to 60%, 12% and 8%, respectively.

SOGP Design: Segment I/O examples

- Reorganize on-disk data on the fly
- Make Future accesses become sequential accesses
- Recent popularity grouping algorithm: best-*j*-out-of-*k*





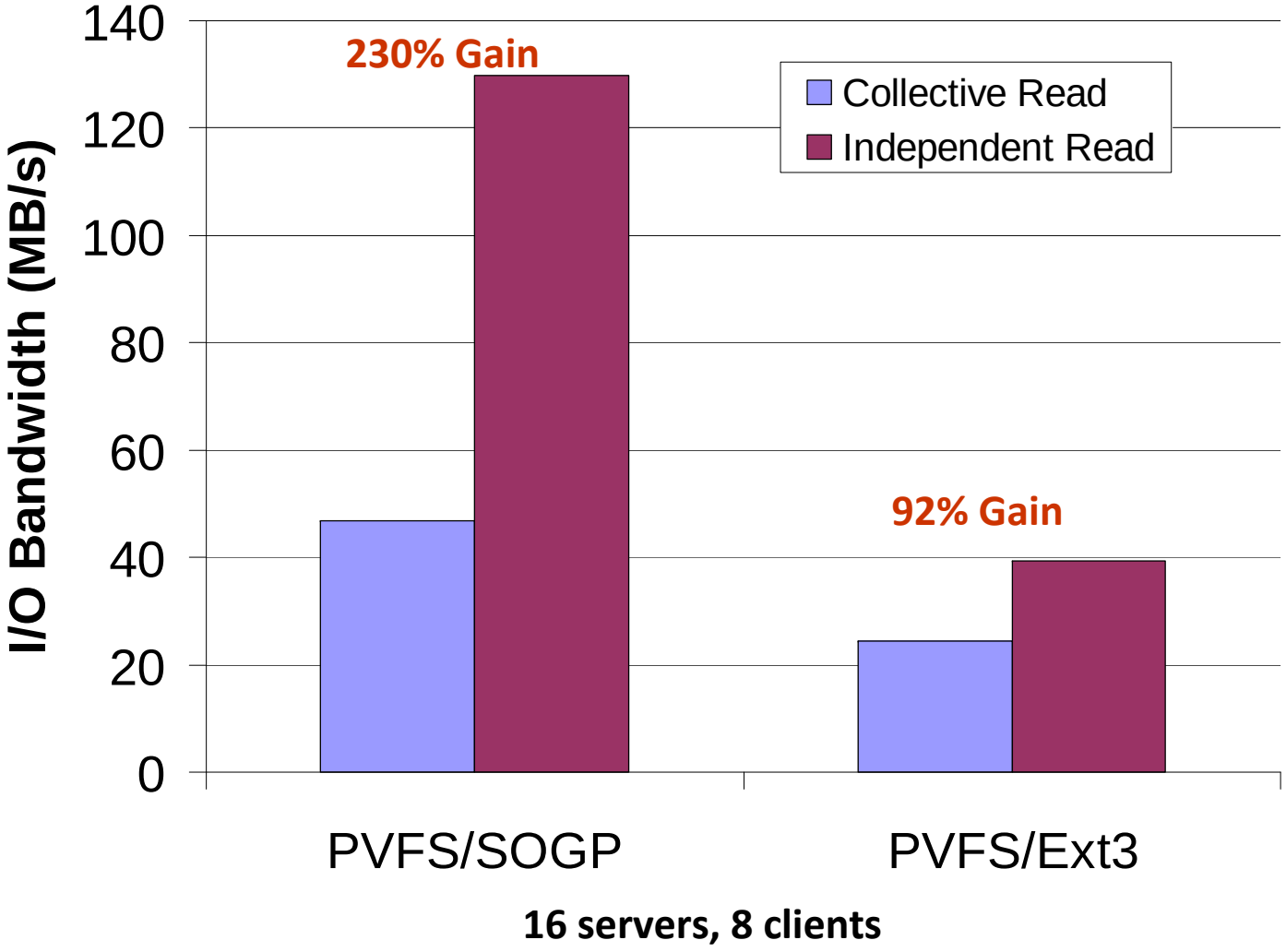
Evaluation Methodology: Hardware Platform Configuration

Overview		
	CASS @UCF	Chiba-City @ANL
Number of nodes	16	256
Interconnect	Nortel 5510-48T NonBlocking 48 port high speed network switch	64-bit Myrinet
Per node configuration		
Make and Model	Dell PowerEdge 1950	IBM Netfinity 7000s
CPU per node	2 x Intel Xeon Dual core 2.33 GHz	2 x Intel PIII 500 MHz
Memory	4GB DDR2 533 MHz	512 MB
Hard Drive	2 x 144GB SAS or 2 x 500GB SATA	9 GB

Evaluation Methodology: Software Environment

Noncontig, MPI-Tile-IO, IOR, CP2K, SIESTA etc.
HDF5, pNetCDF (optinal)
MPI/MPI-IO
PVFS client
PVFS server
SOGP
Ext3 file system

Experimental Results: Noncontig I/O Bandwidth



(ICPP'08) SOGP: Segment Structured On-disk Data Grouping and Prefetching

Main Research Results

- **Performance Optimization Through Semantic Mining & Grouping**
  - (ICDCS'08) GHBA: Scalable and Adaptive Metadata Management in Ultra Large-scale File Systems
  - (HPDC'08) FARMER: File Access corRelation Mining and Evaluation Reference model for peta-scale file systems
  - (CCGrid'08) AMP: An Affinity-based Metadata Prefetching scheme for metadata servers
  - (Ph.D. dissertation), Metadata and Data Management in High Performance File and Storage Systems
  - (ICPP'08) SOGP: Segment Structured On-disk Data Grouping and Prefetching
- **Workload Modeling**
  - **(MASCOTS'08)** A Novel and Generic Model for Synthesizing Disk I/O Traffic Based on The Alpha-stable Process
  - **(Cluster'08)** A Novel Model for Synthesizing Parallel I/O Workloads in Scientific Applications
- **Energy Efficiency Optimization**
  - **(MASCOTS'08)** GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability
  - **(MASCOTS'08)** Energy Efficient Buffer Cache Replacement
  - **(ICPP'08)** Impacts of Indirect Blocks on Buffer Cache Energy Efficiency
  - **(Cluster'07, Best Paper Award)**: Evaluating Memory Energy Efficiency in Parallel I/O Workloads

Complex I/O Behavors

- **IID vs Self-Similarity:**
  - IID: search engine I/O, tpc-d
  - Self-similar: cello96 (Gaussian) and cello99 (non-Gaussian)
  - Scientific application workloads: ior2, f1, m1 and MPI-IO
    - Non-IID for I/O arrival process on most computing nodes, and
    - IID for I/O arrival in a small number of remaining subtraces.

- **Gaussian self-similarity vs non-Gaussin self-similarity**
  - cello96 (Gaussian) and cello99 (non-Gaussian)

- **Accurately modeling disk I/Os remains an challenge issue due to the burstiness**

- **We propose an accurate and versatile I/O model based on alpha-stable process.**

(MASCOTS'08) A Novel and Generic Model for Synthesizing Disk I/O Traffic Based on The Alpha-stable Process
(Cluster'08) A Novel Model for Synthesizing Parallel I/O Workloads in Scientific Applications

Synthesizing I/O Workload

- The trimmed mean of errors
  - to evaluate the matching degrees between real workload and the synthetic ones.

Data Set	α-stable	Normal	Poisson	Lognorm
Search Engine I/O				
1	2.47	3.16	6.43	3.21
2	2.79	1.49	2.32	5.02
3	0.28	4.17	7.45	\
Tpc-d				
4	24.26	34.94	47.10	\
5	14.84	25.08	99.24	\
6	5.25	21.66	26.30	48.35
7	6.43	157.29	233.79	183.09
8	5.98	31.51	645.45	\

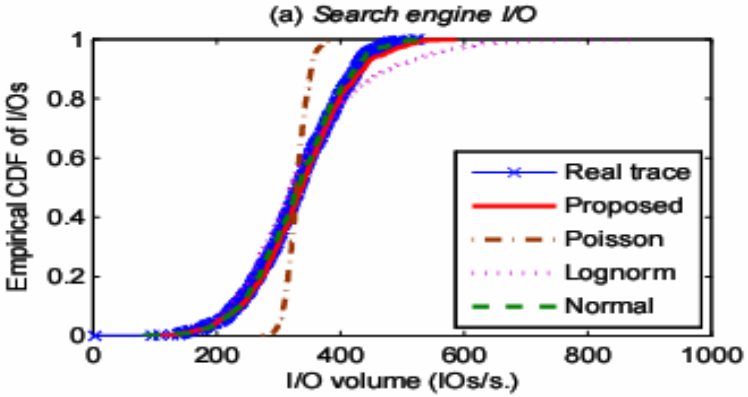
IID Workloads

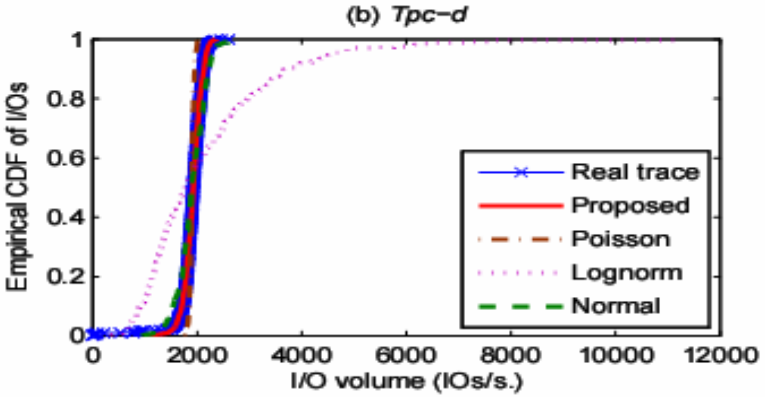
Data Set	α-stable	FBM	FARIMA	ON/OFF
Cello96				
9	12.82	2.94	15.92	16.22
10	21.02	17.48	17.17	17.73
11	8.88	112.01	61.22	3.02
12	18.00	75.00	61.18	20.04
13	41.83	96.44	38.71	24.98
Cello99				
14	0.41	126.68	20.24	5.60
15	1.45	49.71	1.15	5.07
16	3.28	187.13	5.61	7.25
17	0.69	74.92	0.45	3.15
18	1.76	75.67	0.85	7.26

Self-similar Workloads

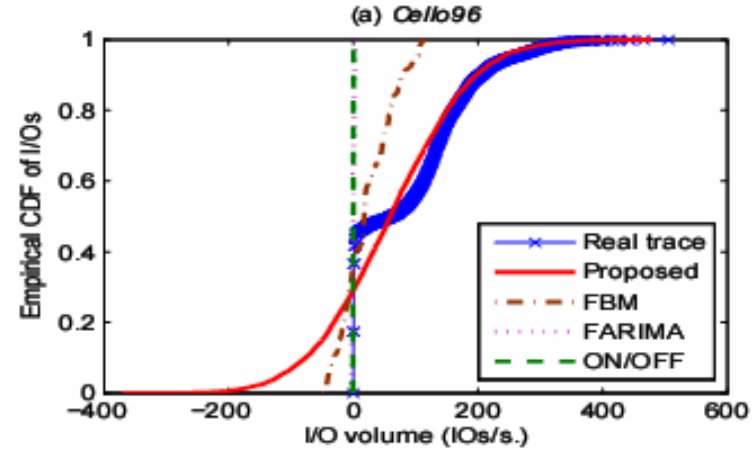
Empirical Study (Disk I/Os)

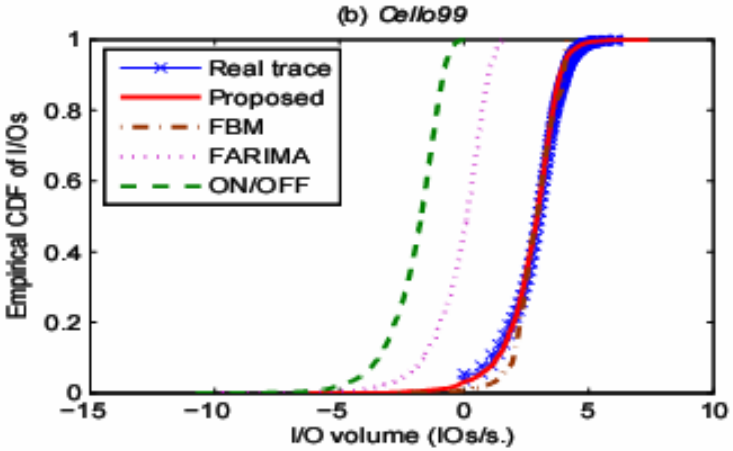
IID Workloads



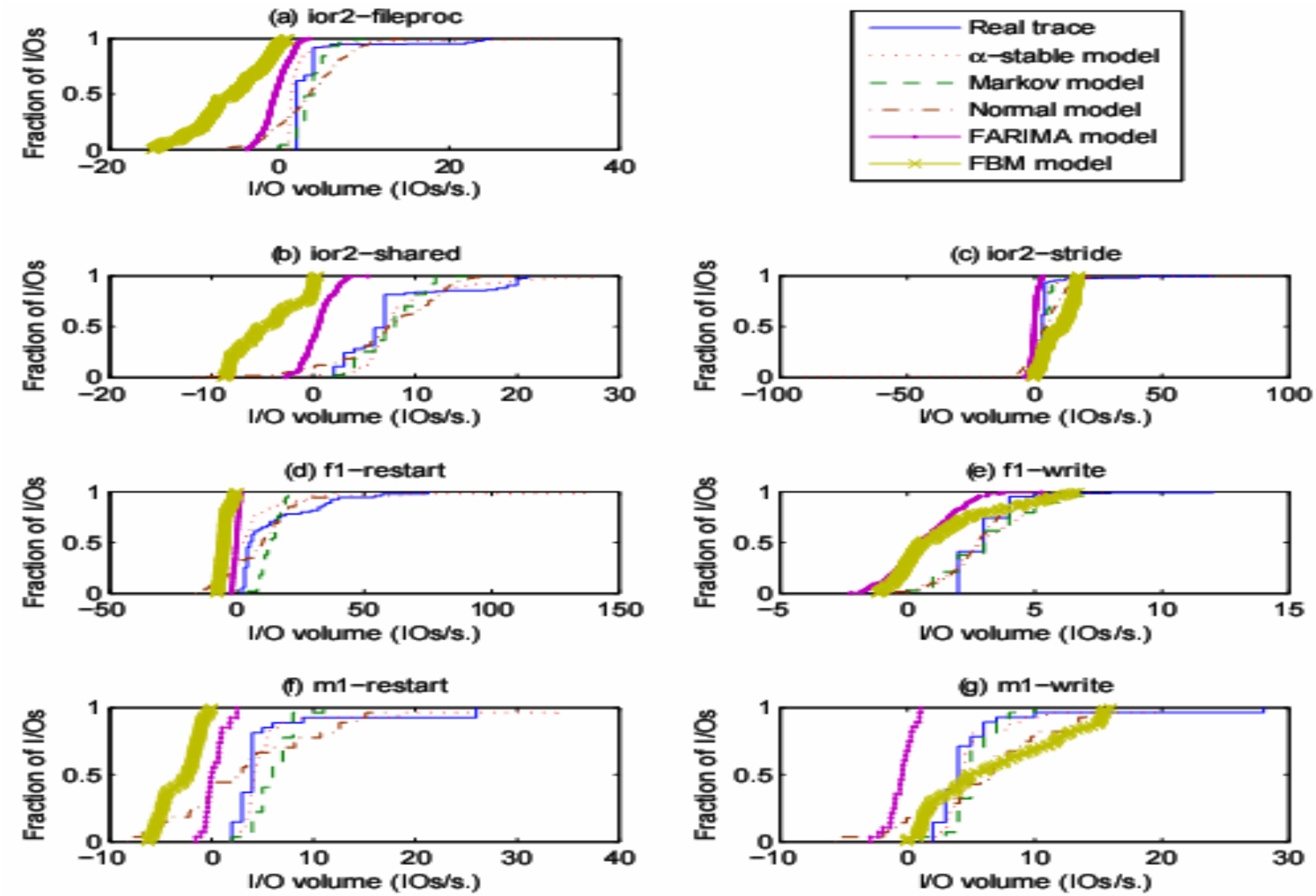


Self-similar Workloads

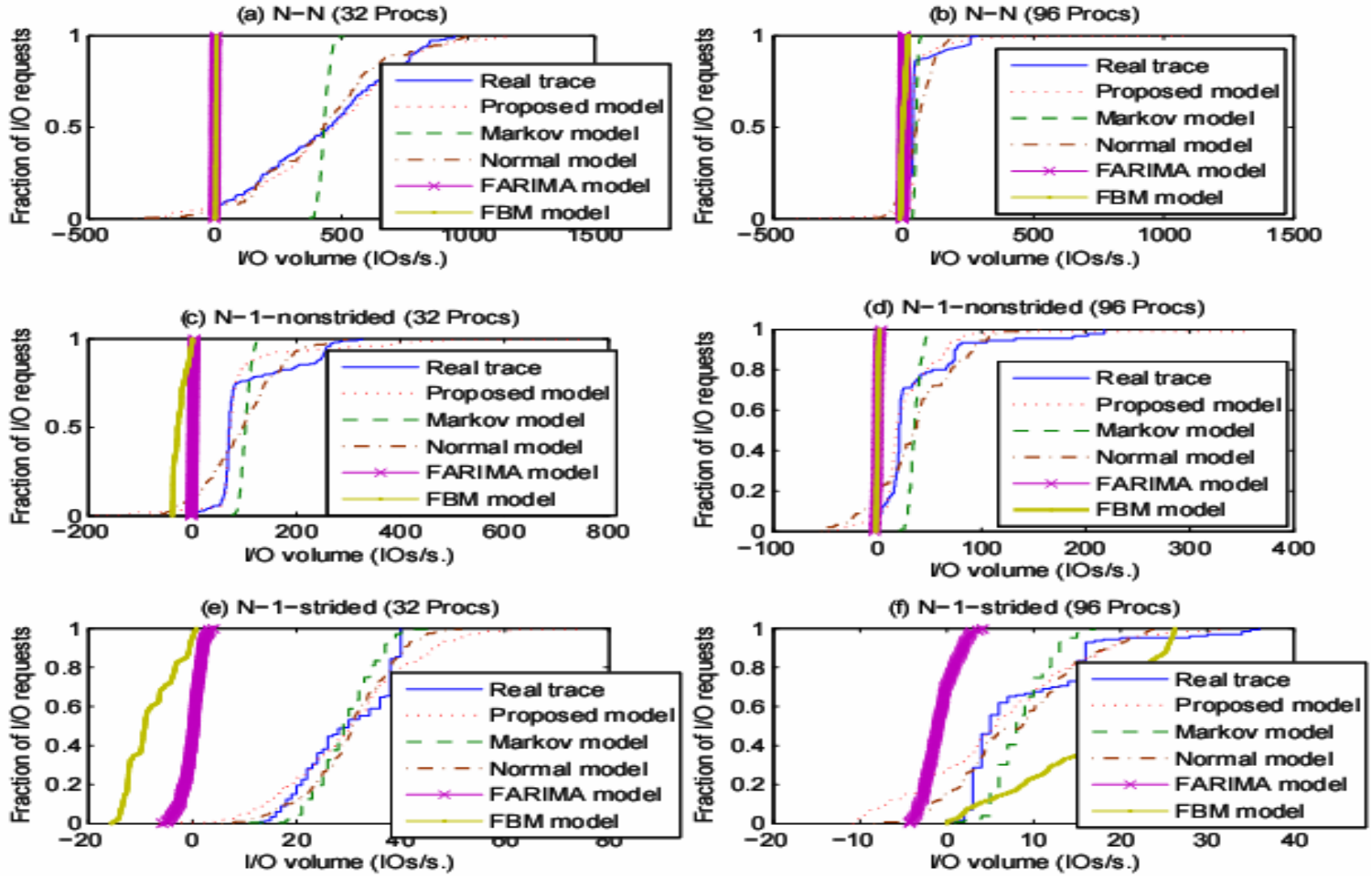




Empirical Study (Parallel I/Os - LLNL)



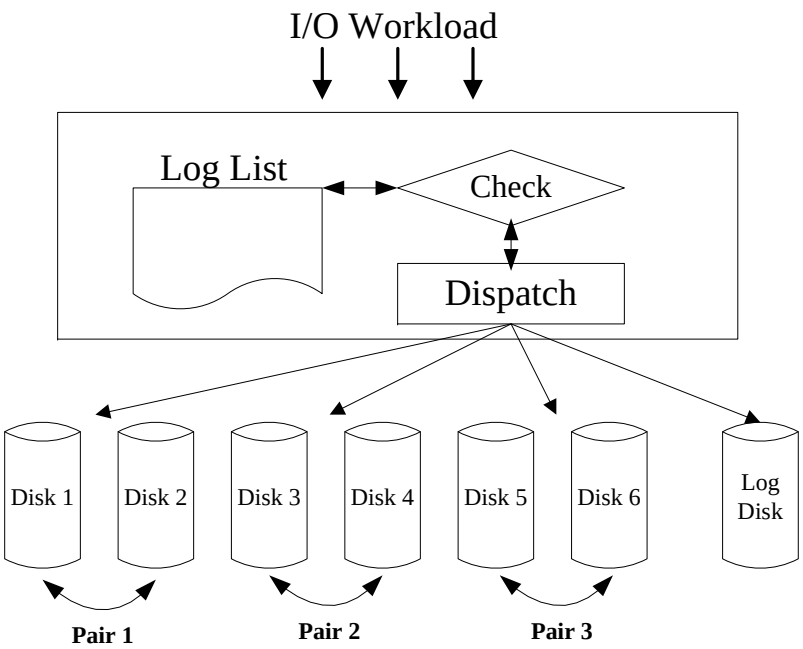
Empirical Study (Parallel I/Os -- LANL)



Main Research Results

- **Optimizing Metadata Management with Semantic Mining & Grouping**
  - (ICDCS'08) GHBA: Scalable and Adaptive Metadata Management in Ultra Large-scale File Systems
  - (HPDC'08) FARMER: File Access corRelation Mining and Evaluation Reference model for peta-scale file systems
  - (CCGrid'08) AMP: An Affinity-based Metadata Prefetching scheme for metadata servers
  - (Ph.D. dissertation), Metadata and Data Management in High Performance File and Storage Systems
  - (ICPP'08) SOGP: Segment Structured On-disk Data Grouping and Prefetching
- **Workload modeling**
  - (MASCOTS'08) A Novel and Generic Model for Synthesizing Disk I/O Traffic Based on The Alpha-stable Process
  - (Cluster'08) A Novel Model for Synthesizing Parallel I/O Workloads in Scientific Applications
- **Energy Efficiency Optimization**
  - **(MASCOTS'08)** GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability
  - **(MASCOTS'08)** Energy Efficient Buffer Cache Replacement
  - **(ICPP'08)** Impacts of Indirect Blocks on Buffer Cache Energy Efficiency
  - **(Cluster'07, Best Paper Award)**: Evaluating Memory Energy Efficiency in Parallel I/O Workloads

GRAID: Improving Power-Efficiency of RAID10



Architecture of GRAID

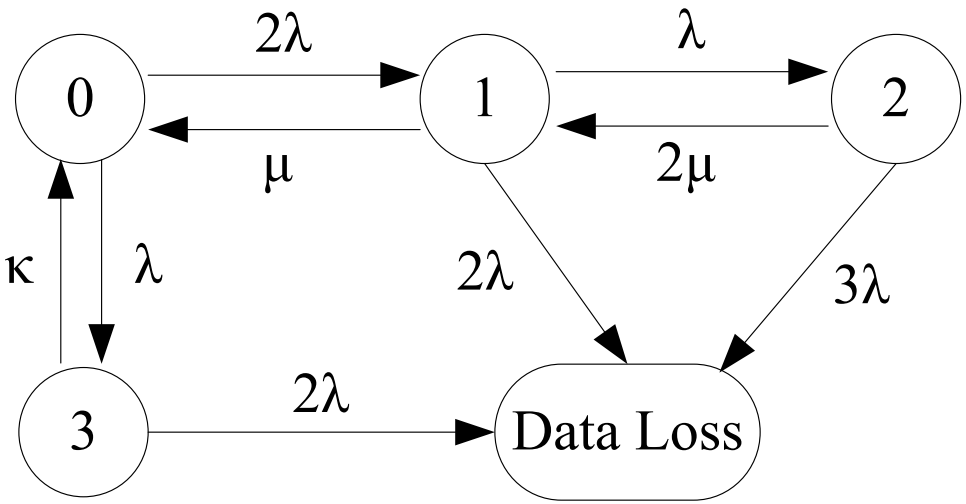
Basic Idea:

1. Update the mirroring disks only periodically
2. Store all updates since the last mirror-disk update in a log disk

Objective: spin down all the mirroring disks (or half of the total disks) most of the time to a lower power mode to save energy without sacrificing reliability.

GRAID: GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability (MASCOTS'08)

Reliability

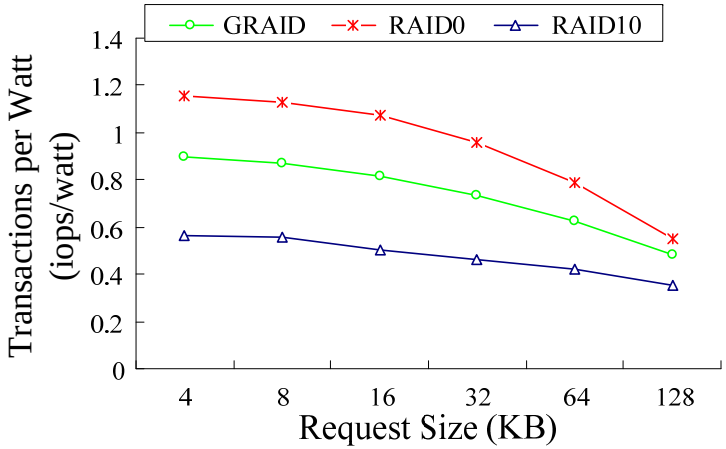


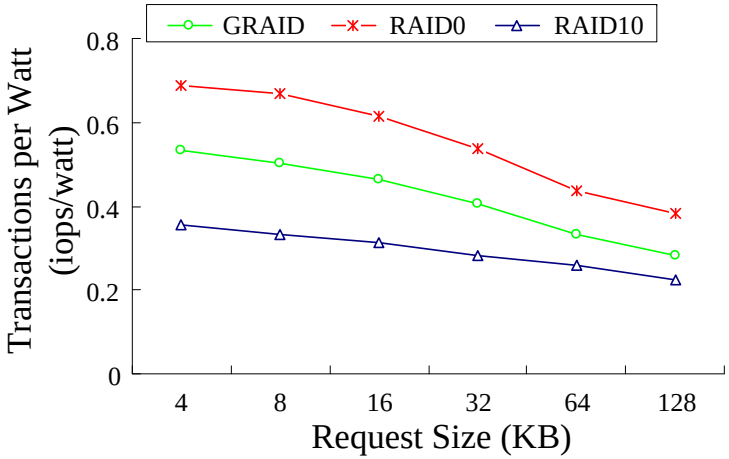
State transition probability diagram for a GRAID disk array consisting of 4 data disks and 1 log disk. Reliability analysis shows that the reliability of GRAID, in terms of MTTDL (Mean Time To Data Loss), is only slightly worse than RAID10.

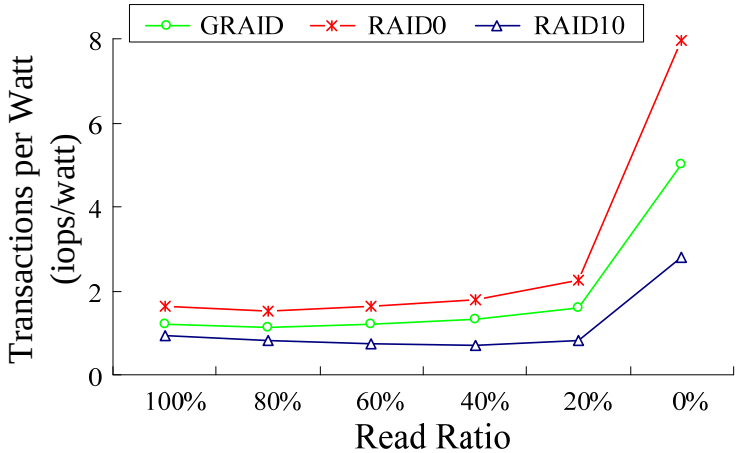
GRAID: GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability (MASCOTS'08)

Performance

Prototype Implemented based on Linux Software RAID





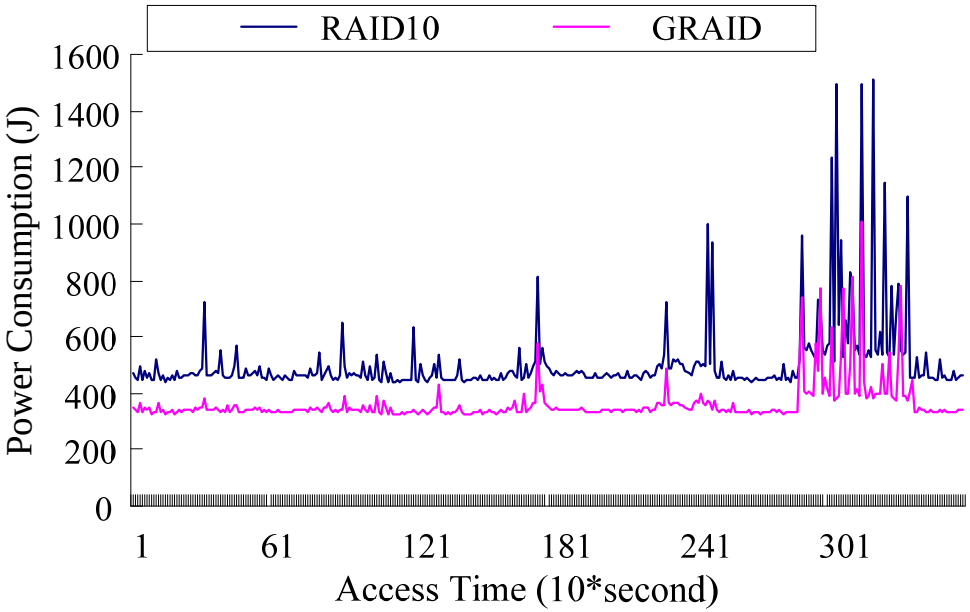


Iometer benchmark shows that GRAID's energy efficiency is significantly better than that of RAID10, but worse than RAID0 which is without redundancy

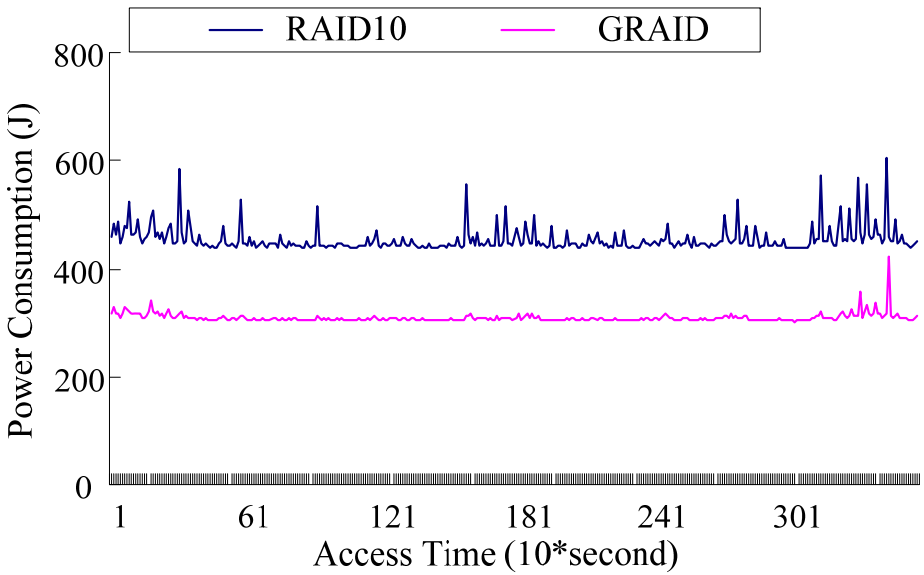
GRAID: GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability (MASCOTS'08)

Energy Comparisons

- Two financial traces from Storage Performance Council
- Use a block-level trace replay software RAIDmeter



(a) Financial1.spc



(b) Financial2.spc

GRAID's energy efficiency is better than RAID10 by 25.4%.

GRAID: GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability (MASCOTS'08)

Motivations for Memory Energy

- Memory can consume 50% more power than processors on data servers [*IBM*]

Table 1. Power consumption breakdown for an IBM p670.

IBM p670 server	Processors	Memory	I/O and other	Processor and memory fans	I/O component fans	Total watts
Small configuration (watts)	384	318	90	676	144	1,614
Large configuration (watts)	840	1,223	90	676	144	2,972

Small: single MCM with four single-core chips with a 128-Mbyte L3 cache and a 16-Gbyte memory

Large: dual MCM with four dual-core chips) with a 256-Mbyte L3 cache and a 128-Gbyte memory

- The memory capacity continues to increases rapidly

  IBM Bluegene @ LLNL: 32TB memory

  EMC Symmetrix DMX3000: up to 256GB/server

- Network DMA and disk DMA are dominant memory operations on data servers

Energy-aware Buffer Caching

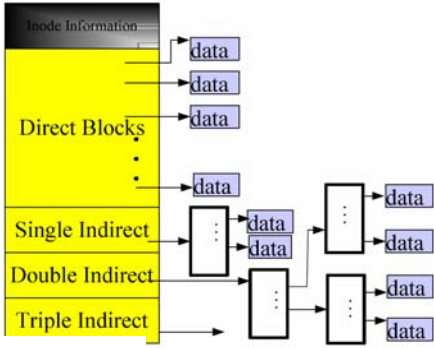
- Quantify the energy impact of eight different cache replacement algorithms including ARC, Belady, LRU, LIRS, LRFU, MQ, LRU2, and 2Q.

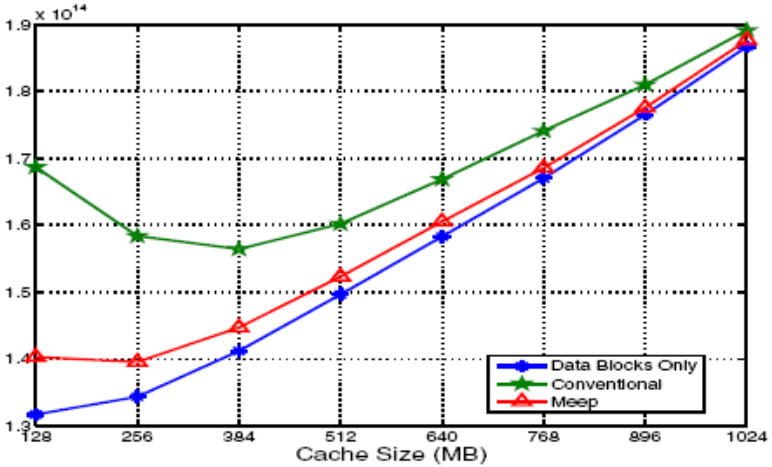
- Interplay among the following three factors are the most important
  - cache algorithm's hit rate,
  - clustering hot block ability,
  - cache populating schemes

- Sequential populating is better than random populating, especially under sequential workloads

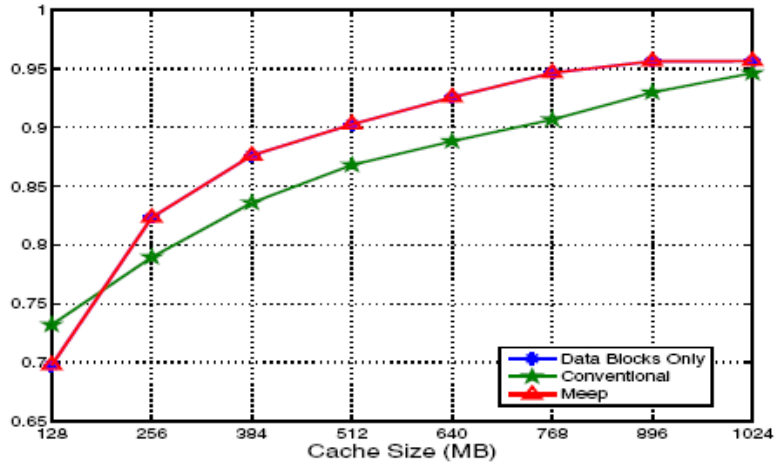
(MASCOTS'08) Energy Efficient Buffer Cache Replacement
(ICPP'08) Impacts of Indirect Blocks on Buffer Cache Energy Efficiency
(Cluster'07, Best Paper Award): Evaluating Memory Energy Efficiency in Parallel I/O Workloads

Energy-aware Buffer Caching: Meep

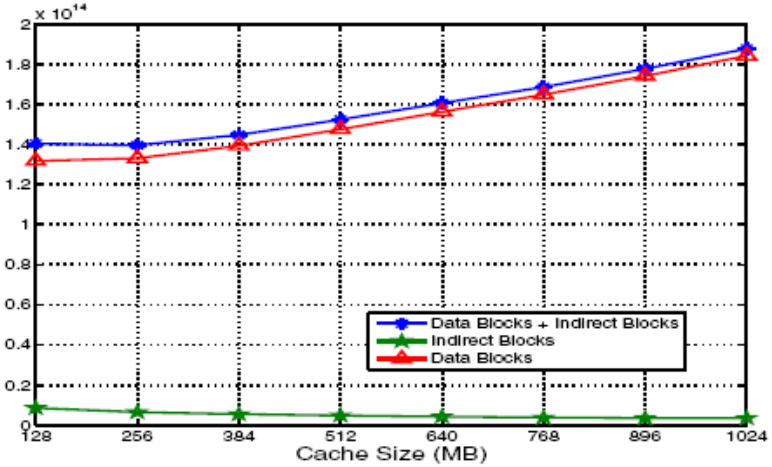




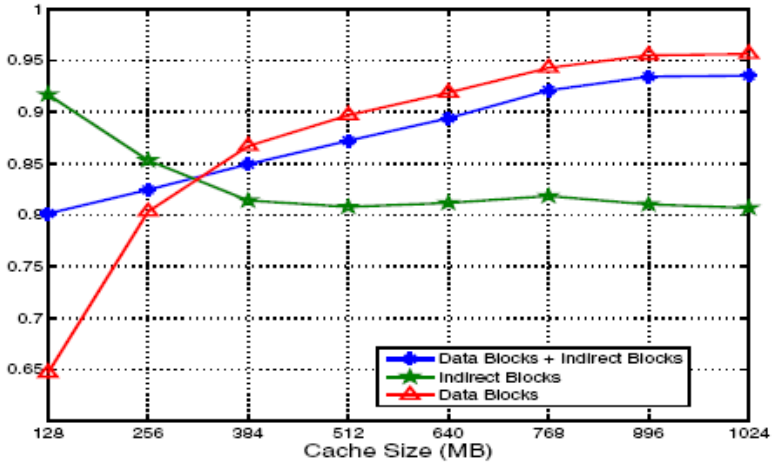
(a) Energy (unit ns*mW)



(b) Hit Rate



(c) Energy Decompositon (unit ns*mW)



(d) Hit Rate Decomposition

Thank you!

Definition 4.1 *A random variable X is said to have a stable distribution if there exist parameters* $0 < \alpha \leq 2$, $\sigma > 0$, $-1 \leq \beta \leq 1$, *and* $\mu \in R$ *such that its characteristic function has the following form [21]:*

$$E[e^{i\theta X}] = \begin{cases} e^{-\sigma^{\alpha}|\theta|^{\alpha}(1-i\beta sign\theta \tan\frac{\pi\alpha}{2})+i\mu\theta}, & \alpha \neq 1 \\ e^{-\sigma|\theta|(1+i\beta sign\theta \ln|\theta|)+i\mu\theta}, & \alpha = 1 \end{cases} \quad (3)$$

where $sign\theta = \begin{cases} 1, & \theta > 0 \\ 0, & \theta = 0 \\ -1, & \theta < 0 \end{cases}$, α, β, σ *and* μ *are characteristic exponent, skewness parameter, scale and location parameters, respectively.*

The characteristic exponent a represents the level of burstiness.

Data Center Energy Use

In 2005 data centers used:

- About 45 billion kWh - roughly 1.2% of all U.S. electricity consumption (comparable to the amount used for color TVs)
- About 2.6 million kW (power demand)

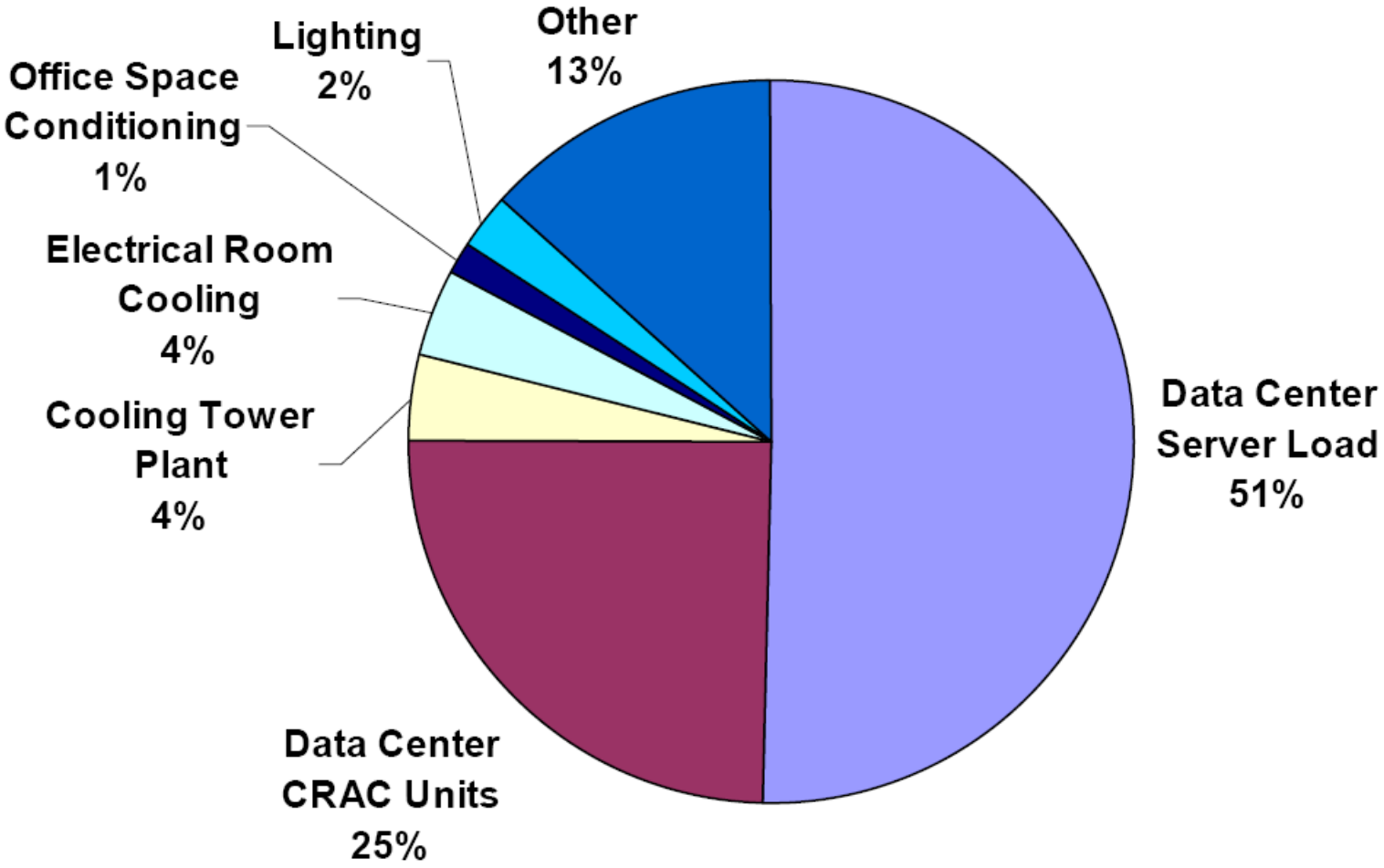
Data Center Energy Forecast

It is projected that DC & server electricity use will more than double from 2006 to 2011:

- From 59 billion kWh to 119 billion kWh (2.8% of electric use)
- If current trends continue could slow the rise to 103 billion kWh

Denise Rouleau, report “work together”.

Energy Use of Data Centers



Source: Alliance to Save Energy 1/2007