Design and Performance of the Dawning Cluster File System

Jin Xiong (*xj@ncic.ac.cn*)

National Research Center for Intelligent Computing Systems

Institute of Computing Technology

Chinese Academy of Sciences

December 3, 2003

Outline

p Motivation & Background

p Design Issues

p Performance Evaluation

p Future Work

p Source

1. Motivation & Background

Why need cluster file systems?

- p Linux Clusters
  - n High performance computing and information services
- p Clustered applications impose new requirements on the file system
  - n Shared global file system with single file system image
  - n High parallel I/O bandwidth
  - n Intensive concurrent accesses from a large number of nodes

Key Issues

Cluster file system is aimed at provide global, shared, uniform, high-performance and scalable file service for applications on clusters.

- Single file system image
  - Global namespace, shared among nodes, uniform access method
- High performance
  - High parallel I/O bandwidth
  - High metadata performance, include file/directory creation, removal, lookup, etc
- Scalability
  - Very large number of nodes
  - Very large file system: large volume, millions of files
- Reliability and faulty tolerance
- Manageability

Related Work

- Academic research systems
  - xFS (UCB)
  - PVFS (Clemson University)
  - Open GFS (University of Minnesota)
- Industrial products or research systems
  - Frangipani (DEC)
  - GPFS (IBM)
  - CXFS (SGI)
  - Lustre (CFS Inc.)

Background

- NCIC is a research center that aims at developing high-performance computers

  http://www.ncic.ac.cn/ (Chinese)

- Supported by “863” High-Tech Program of China

  Dawning-1 (SMP, 1993)

  Dawning 1000 (MPP, 1995)

  Dawning 2000-I (Cluster, 1998)

  Dawning 2000-II (Cluster, 1999)

  Dawning 3000 (Cluster, 2000)

  Dawning 4000-L (Cluster, 2003)

  Dawning 4000-A (Cluster, 2004)

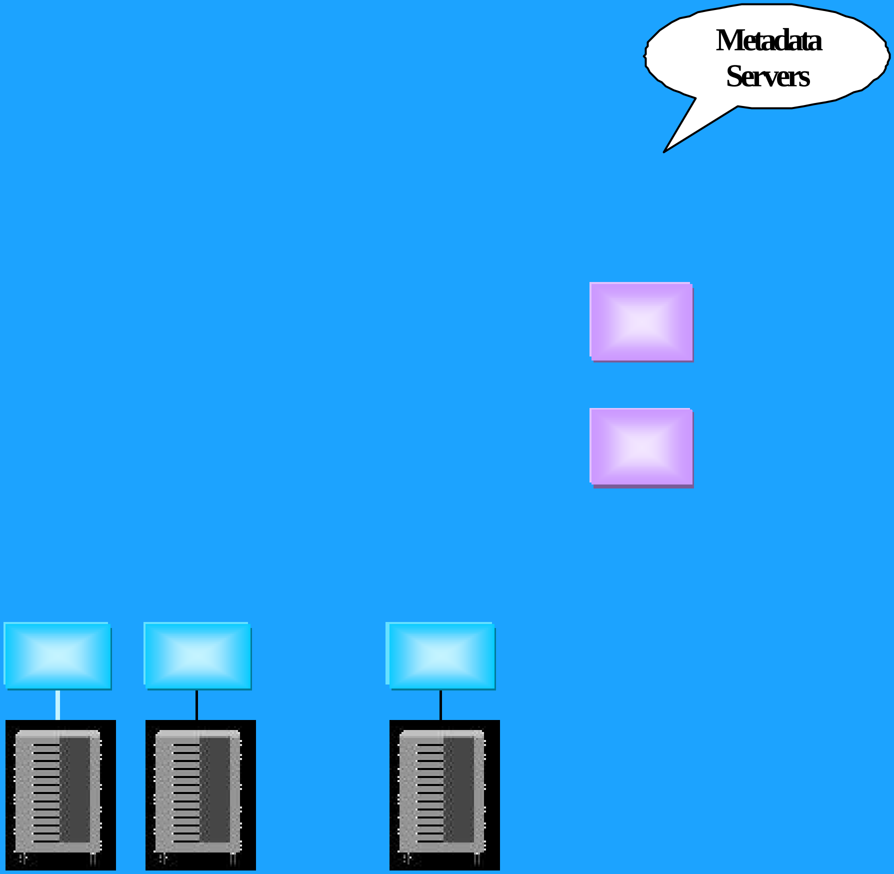
COSMOS

- COSMOS (1996-2000) (AIX)
  - A file-server based cluster file system for Dawning 2000 & 3000
  - Scalable architecture
    - Separate metadata handling from file data handling
    - Multiple file servers and multiple metadata servers, file data striping
  - Cooperative client-side cache, UNIX semantics, too complicated
- What DCFS improves
  - Metadata distribution policy
  - Striping policy
  - Communication mechanism
  - Caching policy
  - Management support

2. Design Issues

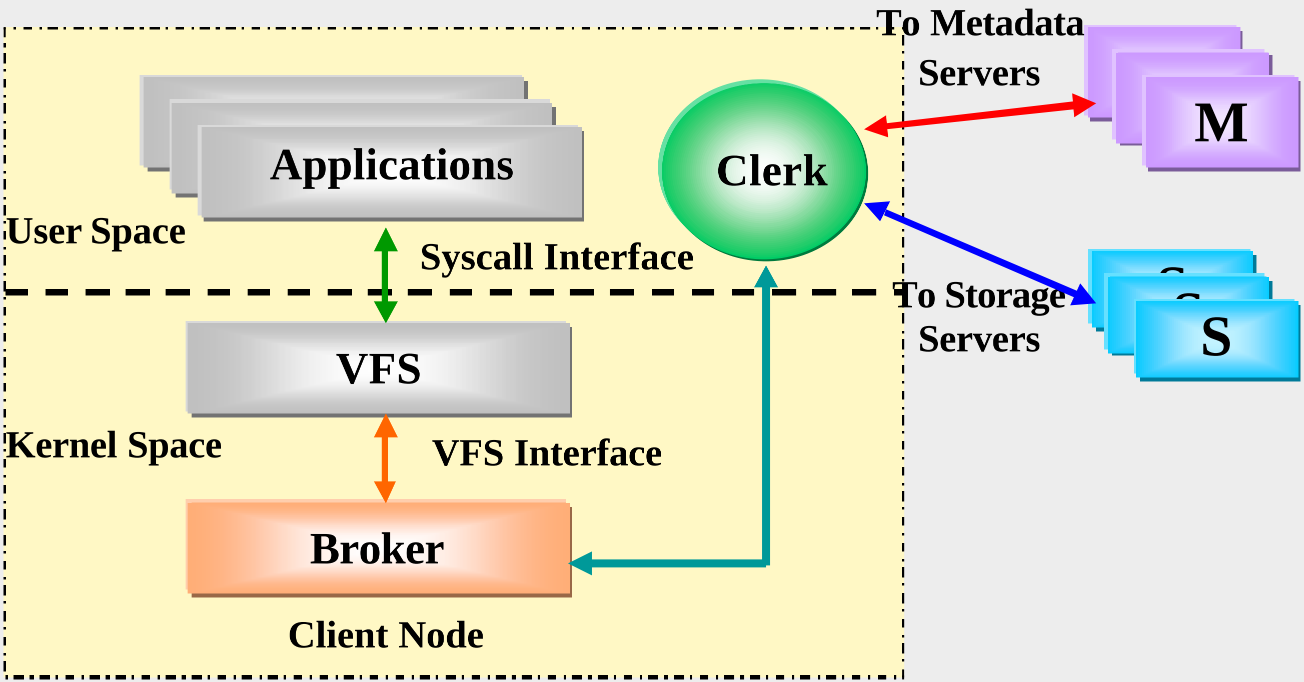
Overview

- DCFS (2001-2002)
  - A file-server based cluster file system for Linux clusters, especially for Dawning 4000-L
- Features
  - Shared global file system with single file system image
  - Standard interface: OS system calls and system commands
  - Scalable architecture
  - High performance
  - Flexible communication mechanism
  - Easy management

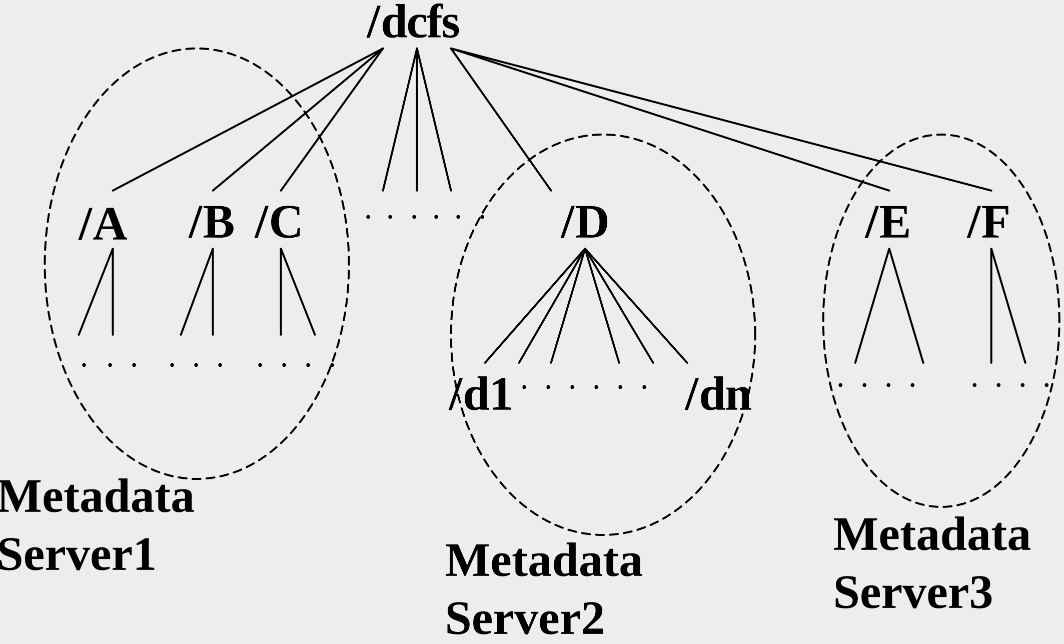


Metadata
Servers

Client-side Implementation



Metadata Management



- Supporting multiple DCFS file systems
- Each DCFS file system
  - A super-manager & a set of *MGR*s,
  - The super-mgr maintains the root directory
  - Each other *MGR* maintains one or more subtrees of the root directory
  - Retains parent –child relationships of objects
- Problems
  - Workload imbalance
  - Storage utilization imbalance

Storage Server Implementation

p Server-side Caching

n IOSes cache file data

p Multithreaded storage servers

n Disk accesses and network transfer can be processed simultaneously

p Files are striped

n RAID 0

n Striping info. is stored in each file’s inode

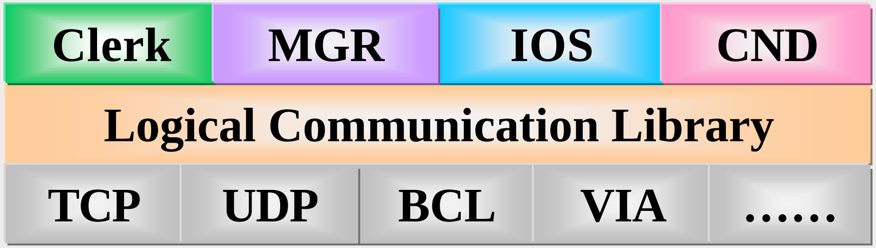
§ Start disk

§ Disks that form the stripe group

Communication Sublayer

A logical communication library

- Provides communication interface between DCFS components
- On top of physical communication protocols:
  - Either stream type protocols(TCP, UDP) or message-passing type protocols(BCL, VIA, …)
- Flexible communication mechanism



3. Performance Evaluation

Targets

- Peek performance: Peak(N)
  - The highest performance with N servers
- Server speedup: Speedup(N)
  - The performance enhancement with the increase of the number of servers
  - Speedup(N) = Peak(N)/ Peak(1)
- Efficiency: E(N)
  - Disk I/O utilization, when disk I/O is performance bottleneck
  - E(N) = BW(N) / (BW$_{disk}$ ´ N)
  - Protocol cost: C(N) = 1 - E(N)
- Sustainability
  - The maximum number of clients that can be supported simultaneously by N servers

Test Platform

- 32 compute nodes of Dawning 4000-L
  - 22 client nodes,
  - 1 MGR, 1/2/4/8 IOSes for bandwidth tests
  - 4 IOSes, 1/2/4/8 MGRs for metadata performance tests
- Node: Dawning Tiankuo R220XP server
  - 2 2.4GHz Intel Xeon Processors, 2GB memory
  - Redhat 7.2, Linux-2.4.18-3smp
- Network
  - Gigabit Ethernet:

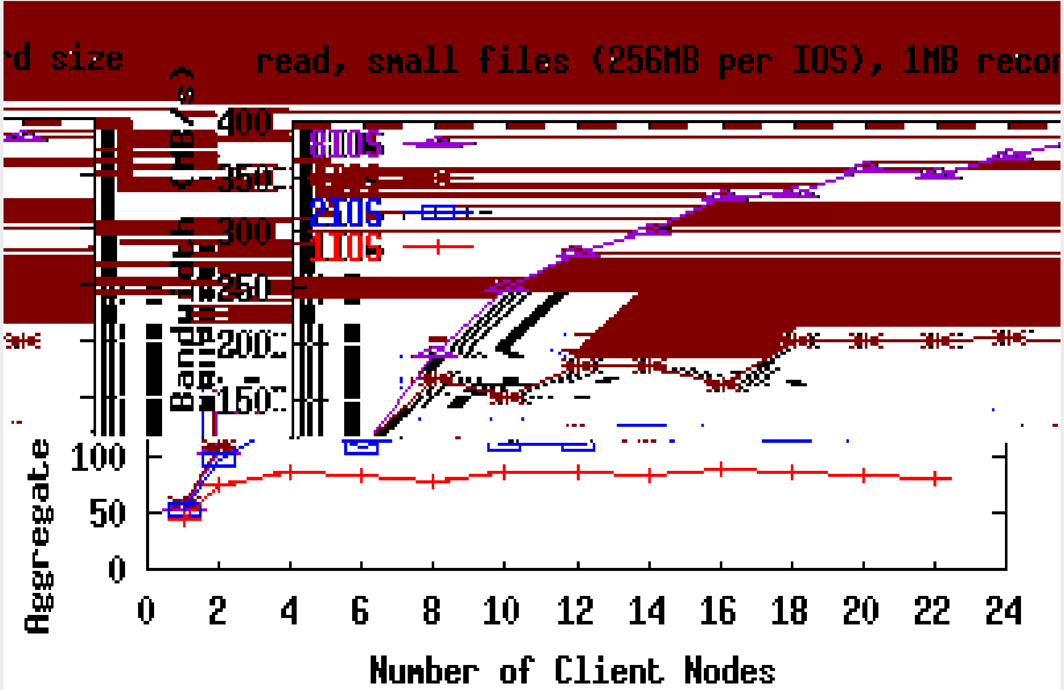
    106.2MB/sec, 97.1μsec latency, by netperf with 16KB message size
- Disks
  - Seagate Ultra320 SCSI disk:
    - 8MB data buffer, 2.99 msec average latency
    - 60MB/sec by iozone on EXT2 (for multiple read threads, 33MB/sec)
- Benchmarks
  - Bandwidth: *iozone*, http://www.iozone.org/
  - Metadata: *thput*, a self-written program

3.1 Aggregate I/O Bandwidth

Aggregate I/O Bandwidth for Small Files(1)

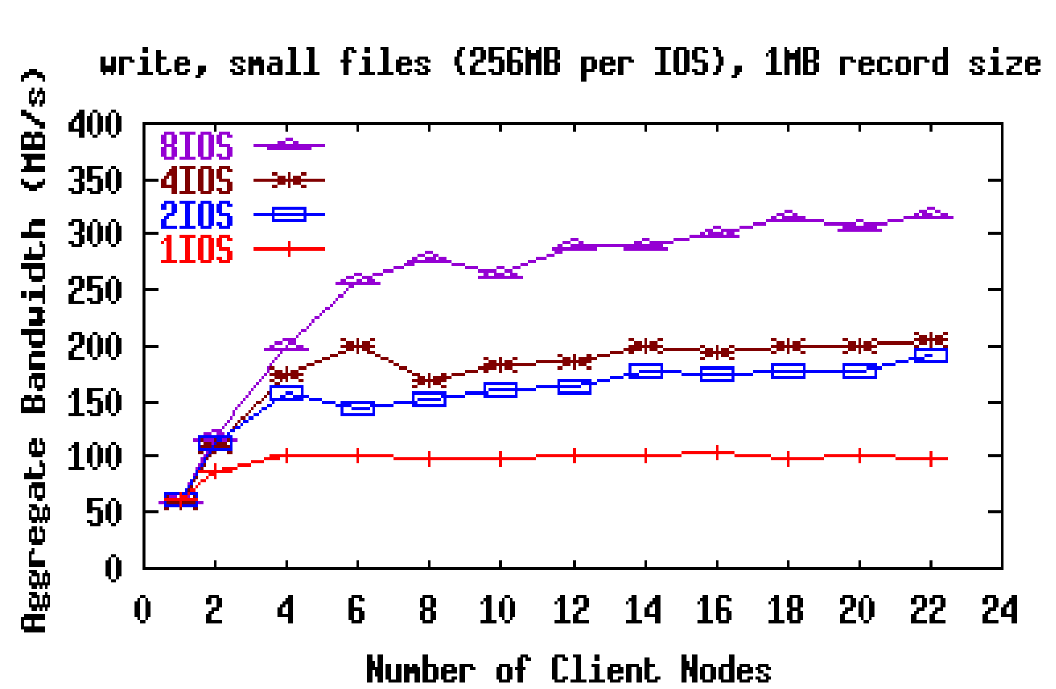
- All read/write data are in storage servers’ cache
- Total read/write size = 256MB X number of storage servers

Read



Aggregate I/O Bandwidth for Small Files(2)

Write



Aggregate I/O Bandwidth for Small Files(3)

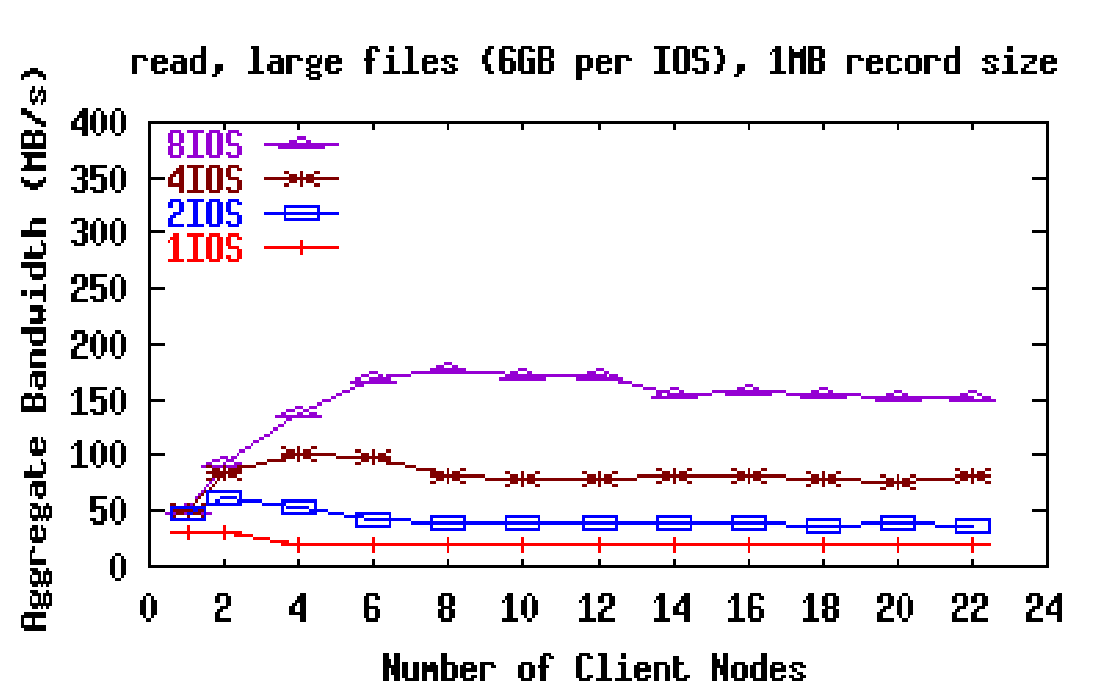
Num. of IOSes	Total RW Size (MB)	Read			Write		
		Num. of Clients	Peak Bandwidth (MB/s)	Speedup	Num. of Clients	Peak Bandwidth (MB/s)	Speedup
1	256	16	90.054	1	16	103.594	1
2	512	20	141.378	1.57	22	191.198	1.84
4	1024	20	305.922	3.40?	22	206.504	1.99?
8	2048	22	384.295	4.27?	22	319.717	3.09?

- For 4 and 8 IOSes, these tests did not reach the peak bandwidth, limited by the total number of clients available (22).

Aggregate I/O Bandwidth for Large Files(1)

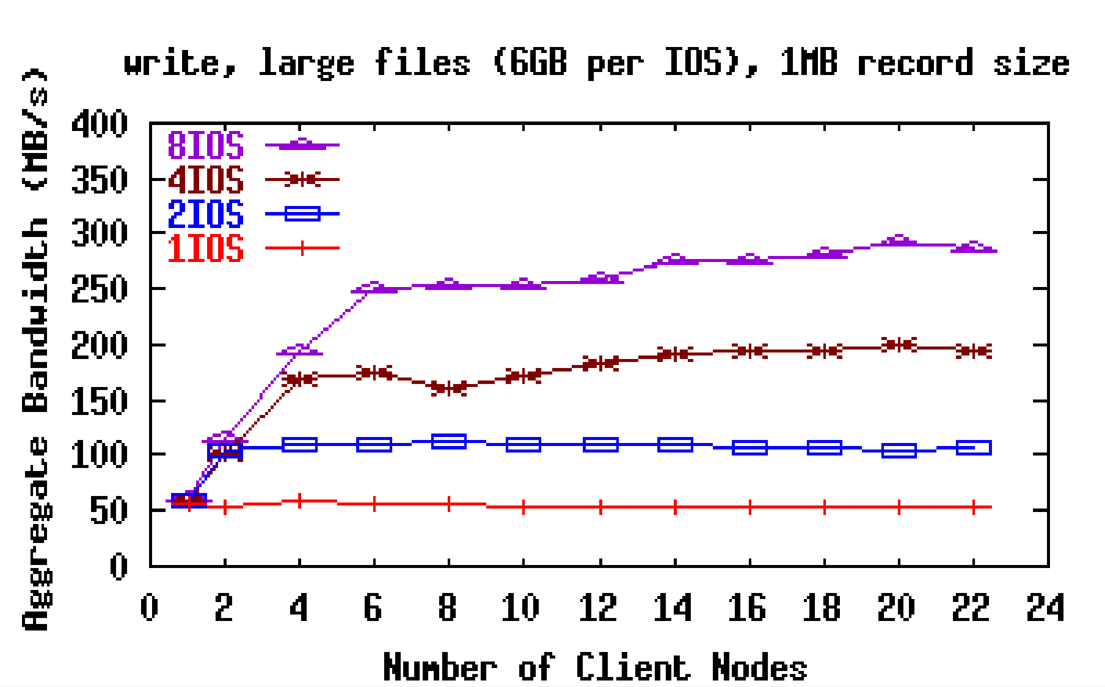
- Total read/write size is much larger than the total cache size
- Total read/write size = 6GB X number of storage servers

Read



Aggregate I/O Bandwidth for Large Files(2)

Write



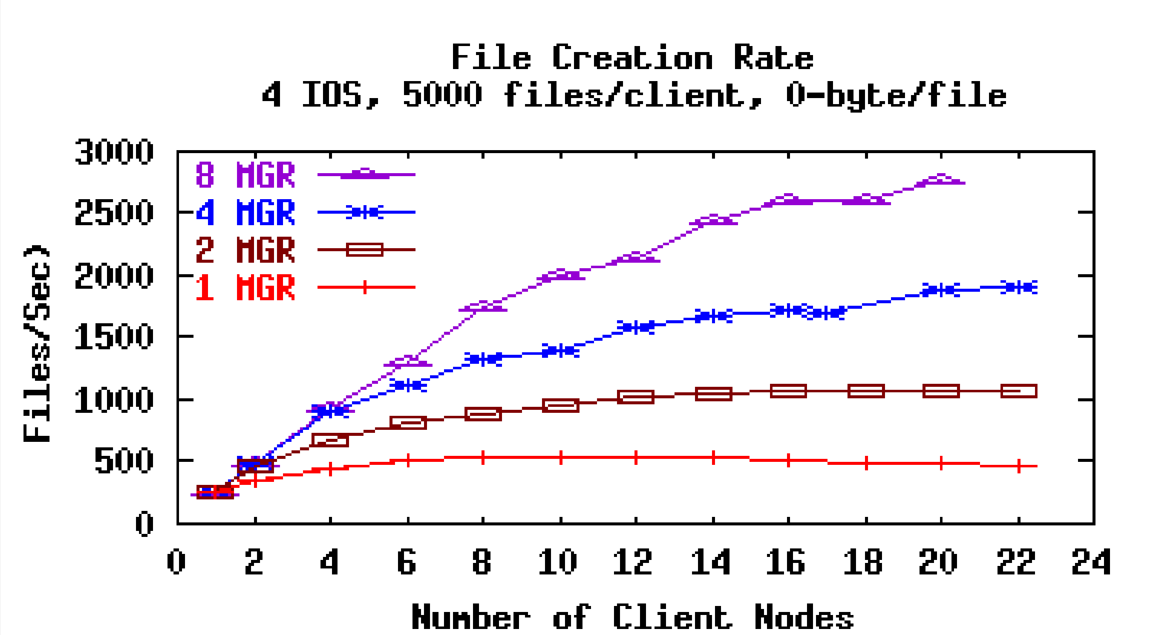
Aggregate I/O Bandwidth for Large Files(3)

Num. of IOSes	Total RW Size (MB)	Read			Write		
		Peak Bandwidth (MB/s)	Speedup	Disk I/O Utilization	Peak Bandwidth (MB/s)	Speedup	Disk I/O Utilization
1	6144	31.192	1	89.28%	59.578	1	94.13%
2	12288	60.790	1.95	86.98%	111.725	1.88	88.26%
4	24576	100.347	3.22	71.81%	198.840	3.34	78.54%
8	49152	178.361	5.72	64.82%	292.576	4.91	57.78%

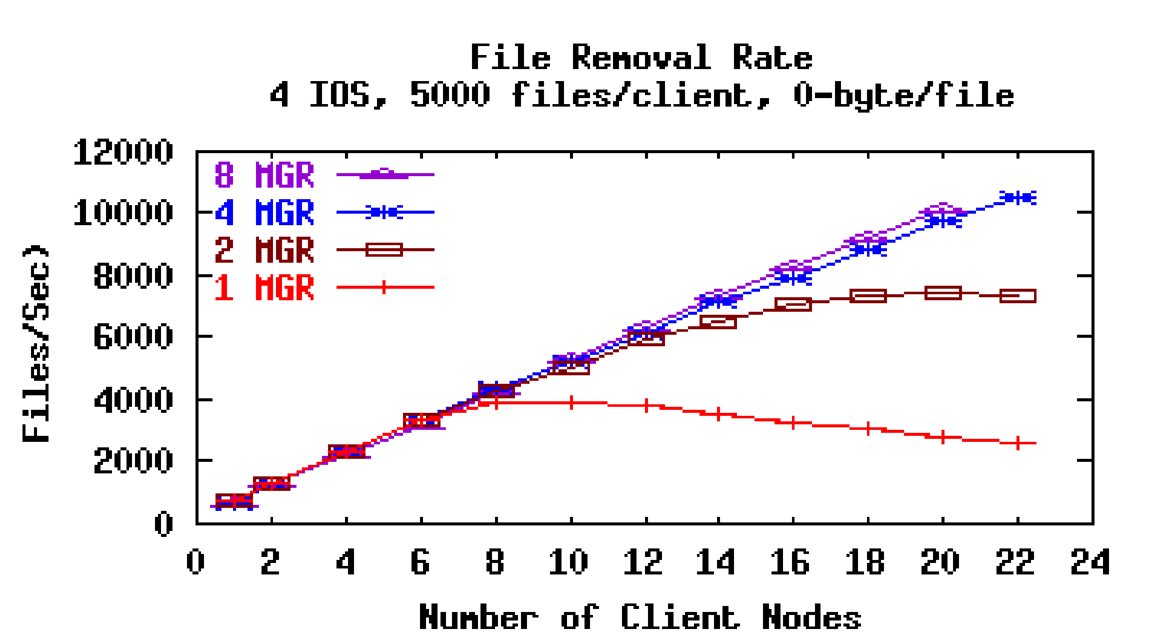
3.2 Metadata Performance

Aggregate File Creation Rate

- Each process created 5000 files, so total number of files = 5000 X number of clients
- File size is 0, only MGRs involved



Aggregate File Removal Rate



Aggregate File Creation & Removal Rate

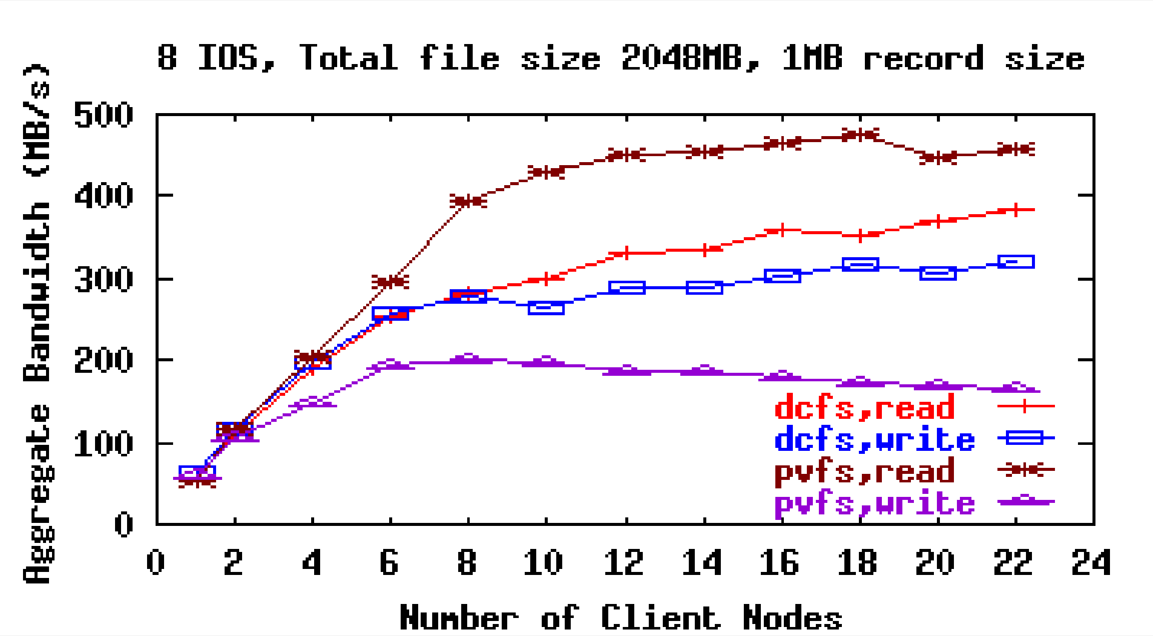
Num. of MGRs	File Creation			File Removal		
	Num. of Clients	Peak Creation Rate (Files/s)	Speedup	Num. of Clients	Peak Removal Rate (Files/s)	Speedup
1	10	545.07	1	10	3903.97	1
2	18	1065.37	1.95	20	7455.02	1.91
4	22	1898.54	3.84 ?	22	10523.67	3.62?
8	22	2765.00	?		?	?

p For 4 and 8 MGRs, these tests did not reach the peak rates, limited by the total number of clients available (22).

3.3 DCFS vs. PVFS

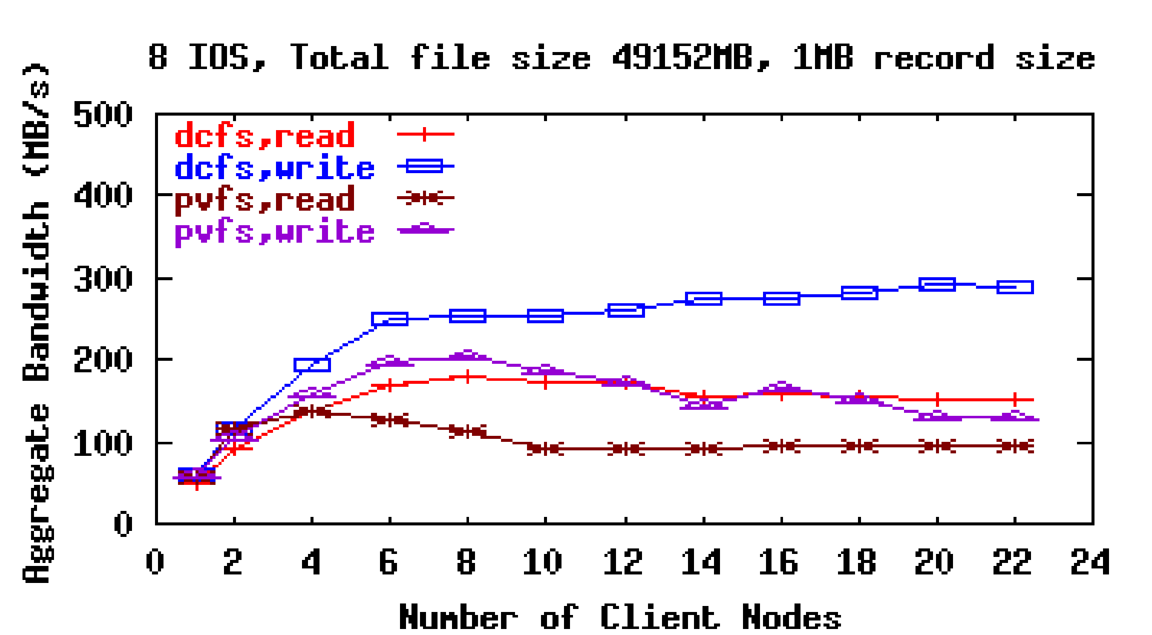
Aggregate I/O Bandwidth for Small Files

- DCFS: 22 client nodes, 1 MGR, 8 IOSes
- PVFS: 22 client nodes, 1 MGR, 8 IODs



- Aggregate read bandwidth: DCFS underperformed PVFS
- Aggregate write bandwidth: DCFS outperformed PVFS

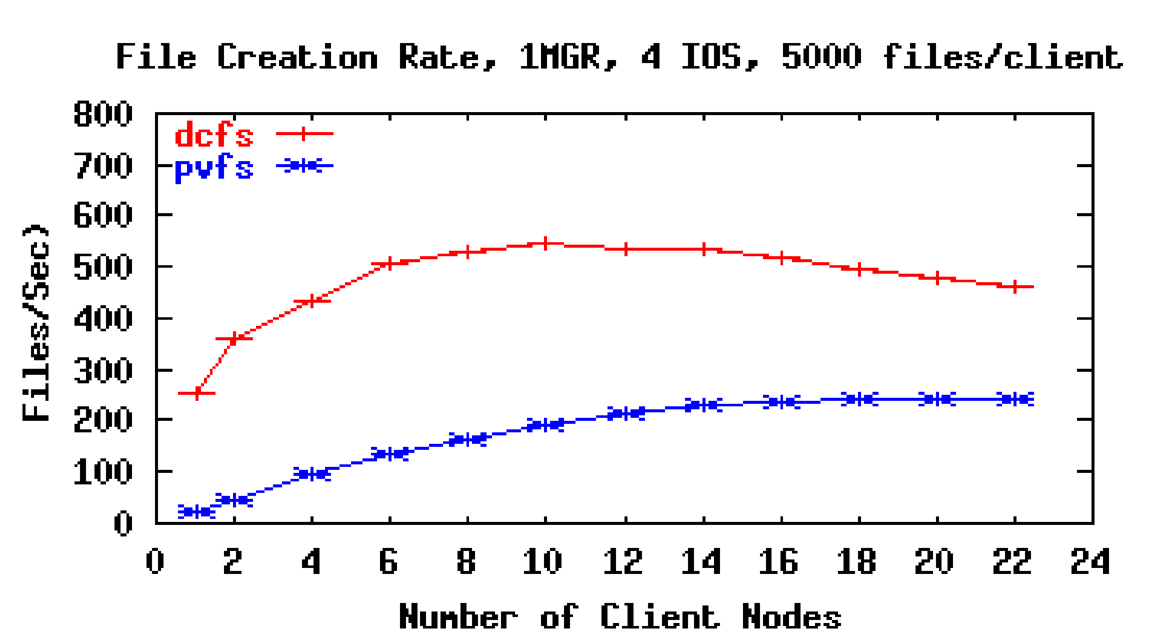
Aggregate I/O Bandwidth for Large Files



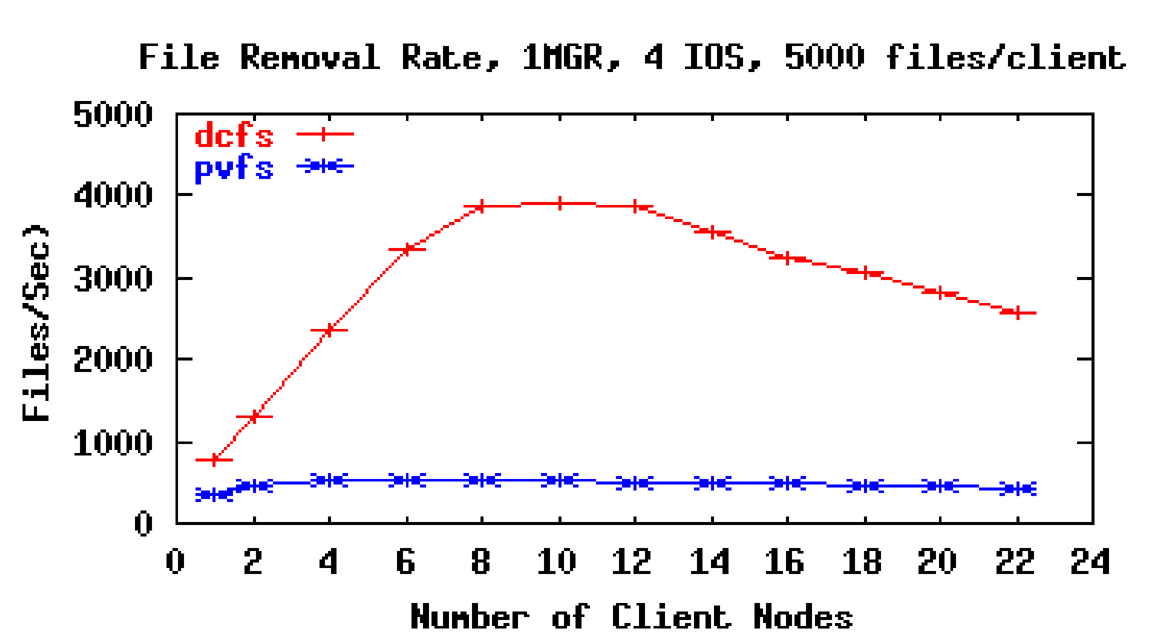
- Aggregate read bandwidth: DCFS outperformed PVFS
- Aggregate write bandwidth: DCFS outperformed PVFS

Metadata Performance – File Creation Rate

- DCFS: 22 client nodes, 1 MGR, 4 IOSes
- PVFS: 22 client nodes, 1 MGR, 4 IODs



Metadata Performance – File Removal Rate



- Aggregate **creation** rate: DCFS outperformed PVFS
- Aggregate **removal** rate: DCFS outperformed PVFS

4. Future Work

Future Work

- Performance analysis on larger scale platforms and real applications
- Reliability and fault recovery
- Metadata distribution policies that can eliminate imbalance
- Client-side caching

- DCFS2
  - In progress

Source

p DCFS source code is available. Please contact us for the source code.

p DCFS Web Site

http://www.ncic.ac.cn/dcfs/

p Contact

dcfs@ncic.ac.cn

xj@ncic.ac.cn

The End

Thank you!