Performance Measurements of a User-Space DAFS Server with a Database Workload

Samuel A. Fineberg
Don Wilson
NonStop Labs

Outline

- Background on DAFS and ODM
- Prototype client and server
- I/O tests performed
- Raw benchmark results
- Oracle TPC-H results
- Summary and conclusions

What is the Direct Access File System (DAFS)?

- Created by the DAFS Collaborative
  - Group consisting of over 80 members from industry, government, and academic institutions
  - DAFS 1.0 spec was approved in September 2001
- DAFS is a distributed file access protocol
  - Data requested from files, not blocks
  - Based loosely on NFSv4
- Optimized for local file sharing environments
  - Systems are in relatively close proximity
  - Connected by a high-speed low-latency network
- Built on top of direct-access transport networks
  - Initially targeted at Virtual Interface Architecture (VIA) networks
  - Direct Access Transport (DAT) API was later generalized and ported to other networks (e.g., Infiniband, iWarp)

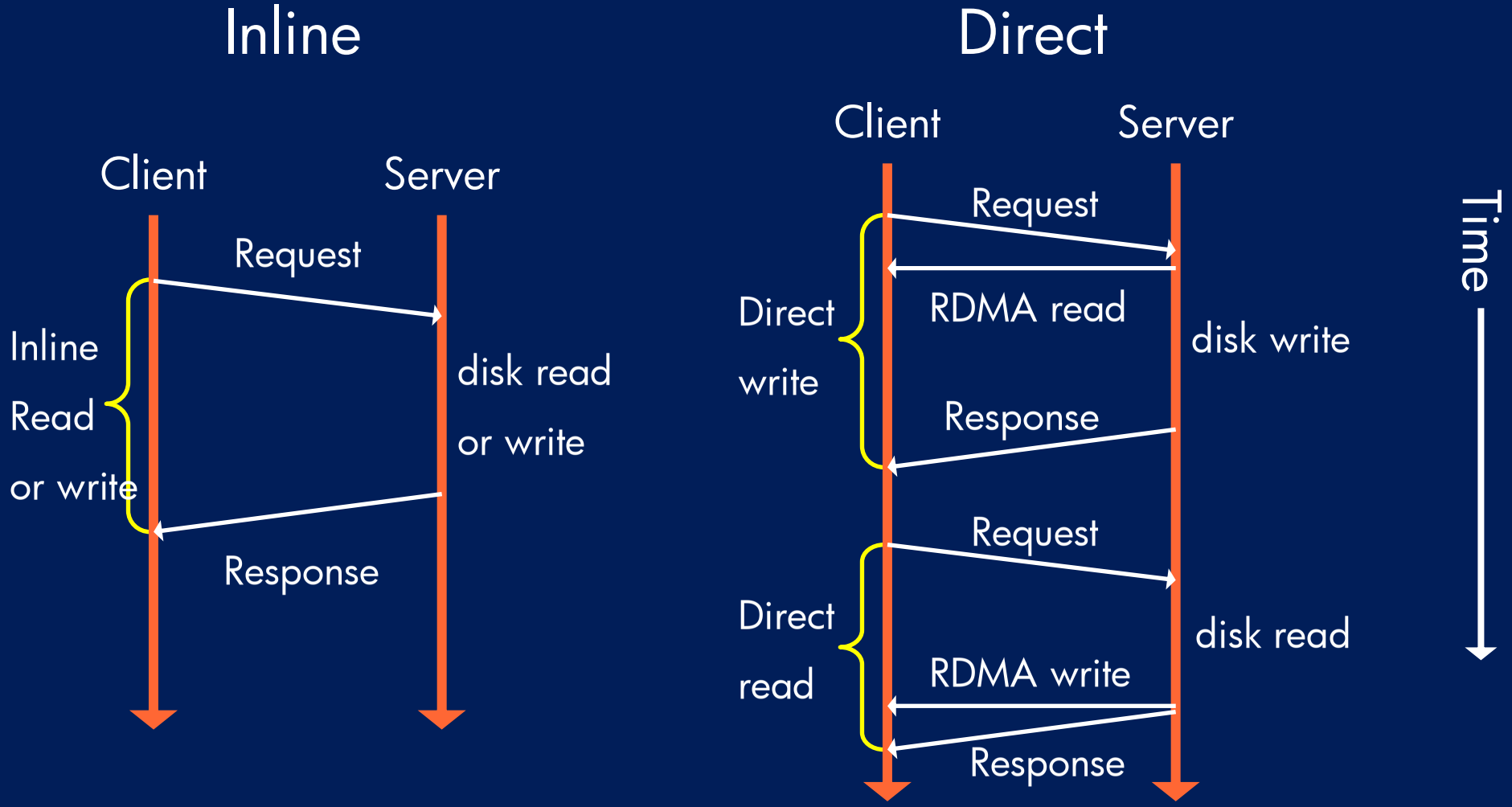
Characteristics of a Direct Access Transport

- Connected model, i.e., VIs must be connected before communication can occur
- Two forms of data transport
  - Send/receive – two-sided
  - RDMA read and write – one sided
- Both transports support direct data placement
  - Receives must be pre-posted
- Memory regions must be "registered" before they can be transferred through a DAT
  - Pins data in physical memory
  - Establishes VM nslation tables for the NIC

DAFS Details

- Session based
  - DAFS clients initiate sessions with a server
  - DAT/VIA connection is associated with a session
- RPC-like Command format
  - Implemented with send/receive
  - Server "receives" requests "sent" from clients
  - Server "sends" responses to be "received" by client
- Open/Close
  - Unlike NFSv2, files must be open and closed (not stateless)
- Read/Write I/O "modes"
  - Inline: limited amount of data included in request/response
  - Direct: Server initiates RDMA read or write to move data

Inline vs. Direct I/O



Oracle Disk Manager (ODM)

- File access interface spec for the Oracle Database
  - Supported as a standard feature in Oracle 9i
  - Implemented as a vendor supplied DLL
  - Files that can not be opened using ODM use standard APIs
- Basic commands
  - Files are created and pre-allocated then committed
  - Files are then "identified" (open) and "unidentified" (closed)
  - All r/w I/O uses an asynchronous "odm_io" command
    - I/Os specified as descriptors, multiple per odm_io call
  - Multiple waiting mechanisms: wait for specific, wait for any
  - Other commands are synchronous, e.g., resizing

Prototype Client/Server

- DAFS Server
  - Implemented for Windows 2000 and Linux (all testing was on Windows)
  - Built on VIPL 1.0 using DAFS 1.0 SDK protocol stubs
  - All data buffers are pre-allocated and pre-registered
  - Data-driven multithreaded design
- ODM Client
  - Implemented as a Windows 2000 dll for Oracle 9i
  - Multithreaded to enable decoupling of asynchronous I/O from Oracle threads
  - Inline buffers are copied, direct buffers are registered/deregistered as part of the I/O
  - Inline/direct threshold (set when library is initialized)

Test System Configuration

- Goal was to compare local I/O with DAFS
- Local I/O configuration
  - Single system running Oracle on locally attached disks
- DAFS/ODM I/O configuration
  - One system running DAFS server software with locally attached disks
  - Second system running Oracle and ODM client, files on DAFS server accessed using ODM over a network
- 4-processor Windows 2000 server based systems
  - 500MHz Xeon, 3GB RAM, dual-bus PCI 64/33
  - ServerNet II (VIA 1.0 based) System Area Network
  - Disks were 15K RPM attached by two PCI RAID controllers, configured for RAID 1/0 (mirrored-striped)

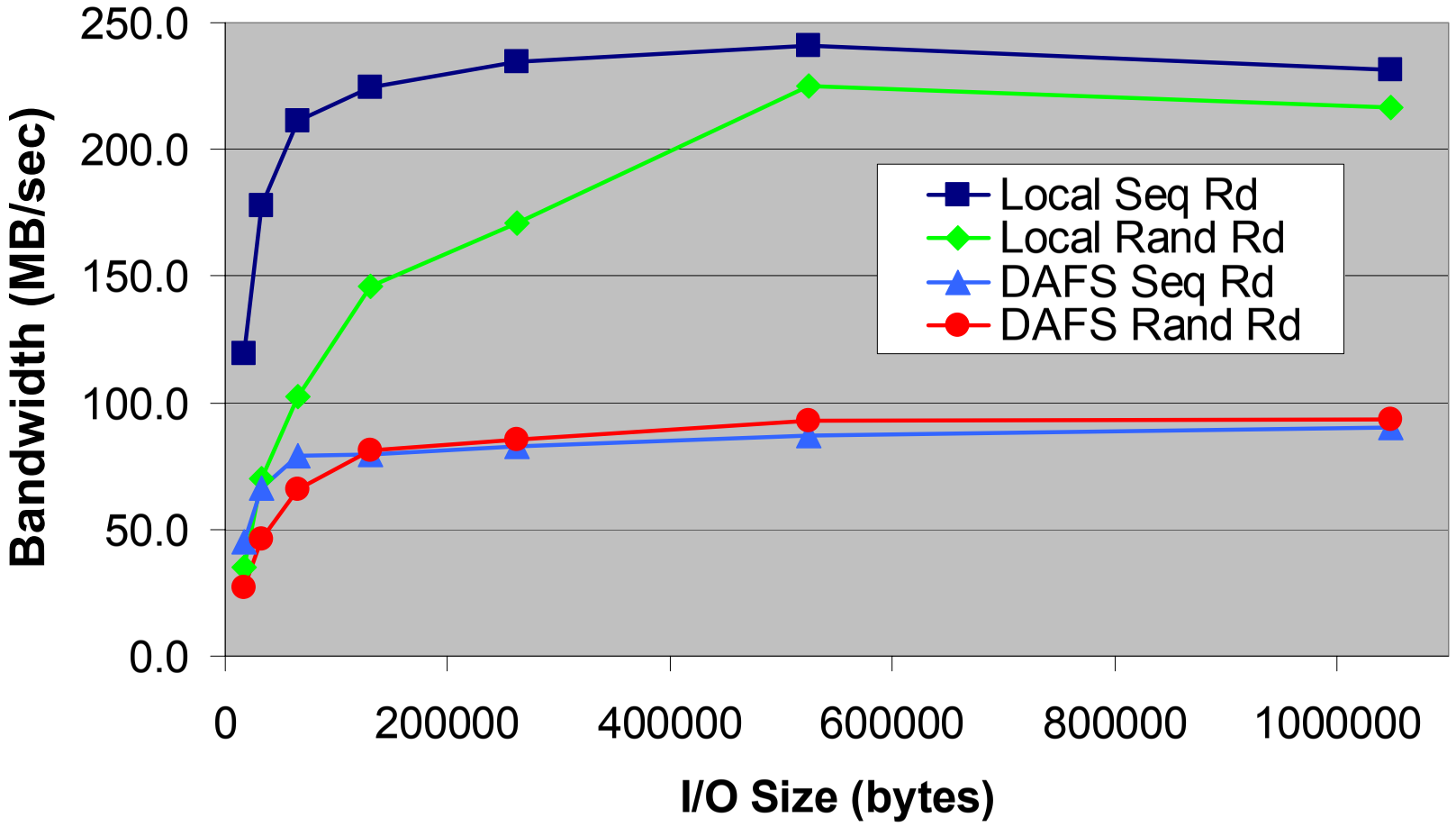
Experiments

- Raw I/O Tests
  - Odmblast – streaming I/O test
  - Odmlat – I/O latency test
  - DAFS tests used ODM dll to access files on DAFS server
  - Local tests used special local ODM library built on Windows unbuffered I/O
- Oracle database test
  - Standard TPC-H benchmark
  - SQL based decision support code
  - DAFS tests used ODM dll to access files on DAFS server
  - Local tests used ran without ODM (Oracle uses windows unbuffered I/O directly)

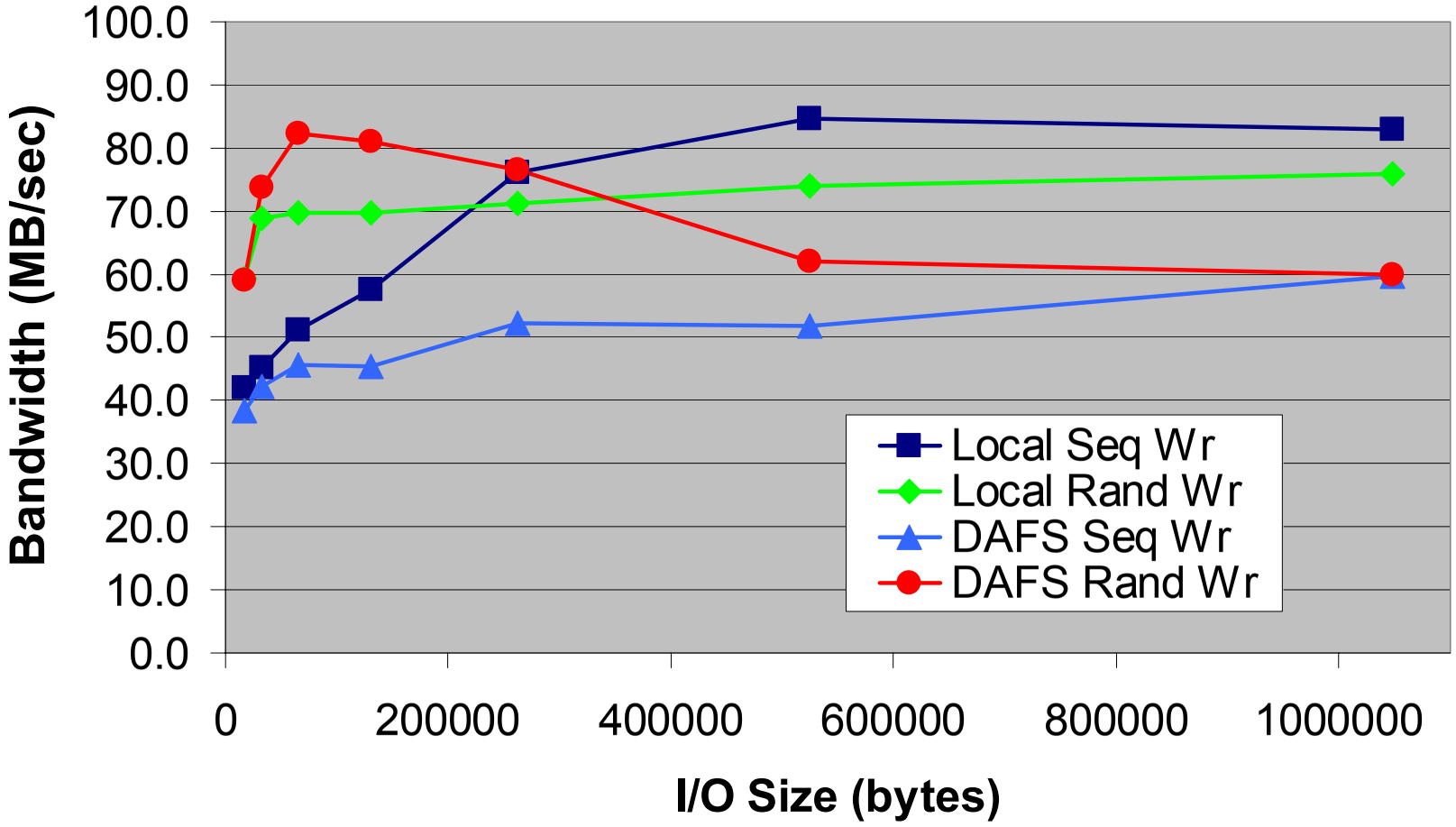
Odmblast

- ODM based I/O stress test
  - Intended to present a continuous load to the I/O system
  - Issues many simultaneous I/Os (to allow for pipelining)
- In our experiments, Odmblast streams 32 I/Os to server
  - 16 I/Os per odm_io call
  - wait for I/Os from the previous odm_io call
- I/Os can be reads, writes, or a random mix
- I/Os can be at sequential or random offsets

Odmblast read comparison



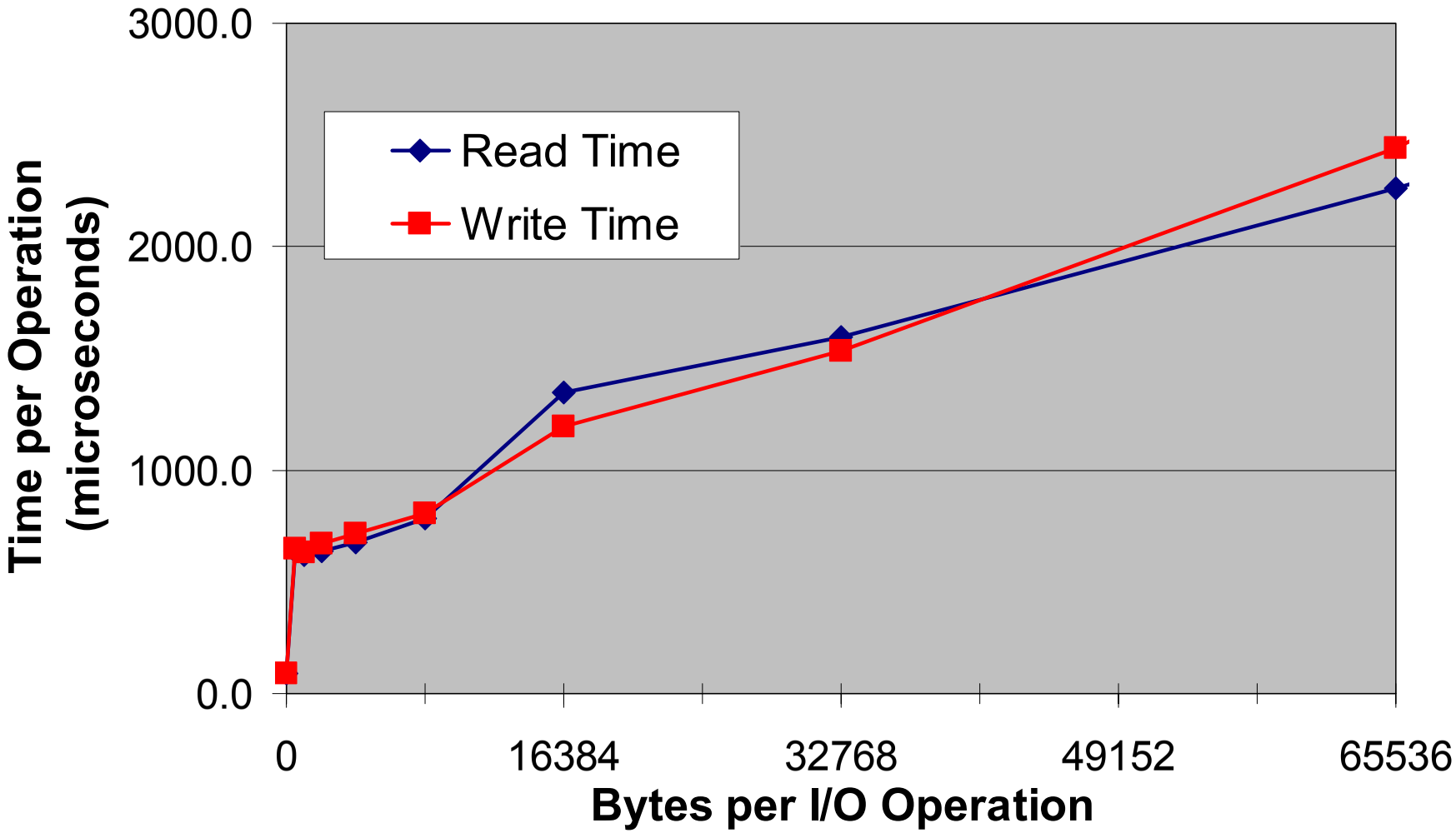
Odmblast write comparison



Odmlat

- I/O Latency test
  - How long does a single I/O take
    - (not necessarily related to aggregate I/O rate)
  - For these experiments, <16K = inline, ≥ 16K = direct
  - Derived the components that make up I/O time using linear regression
  - More details in paper

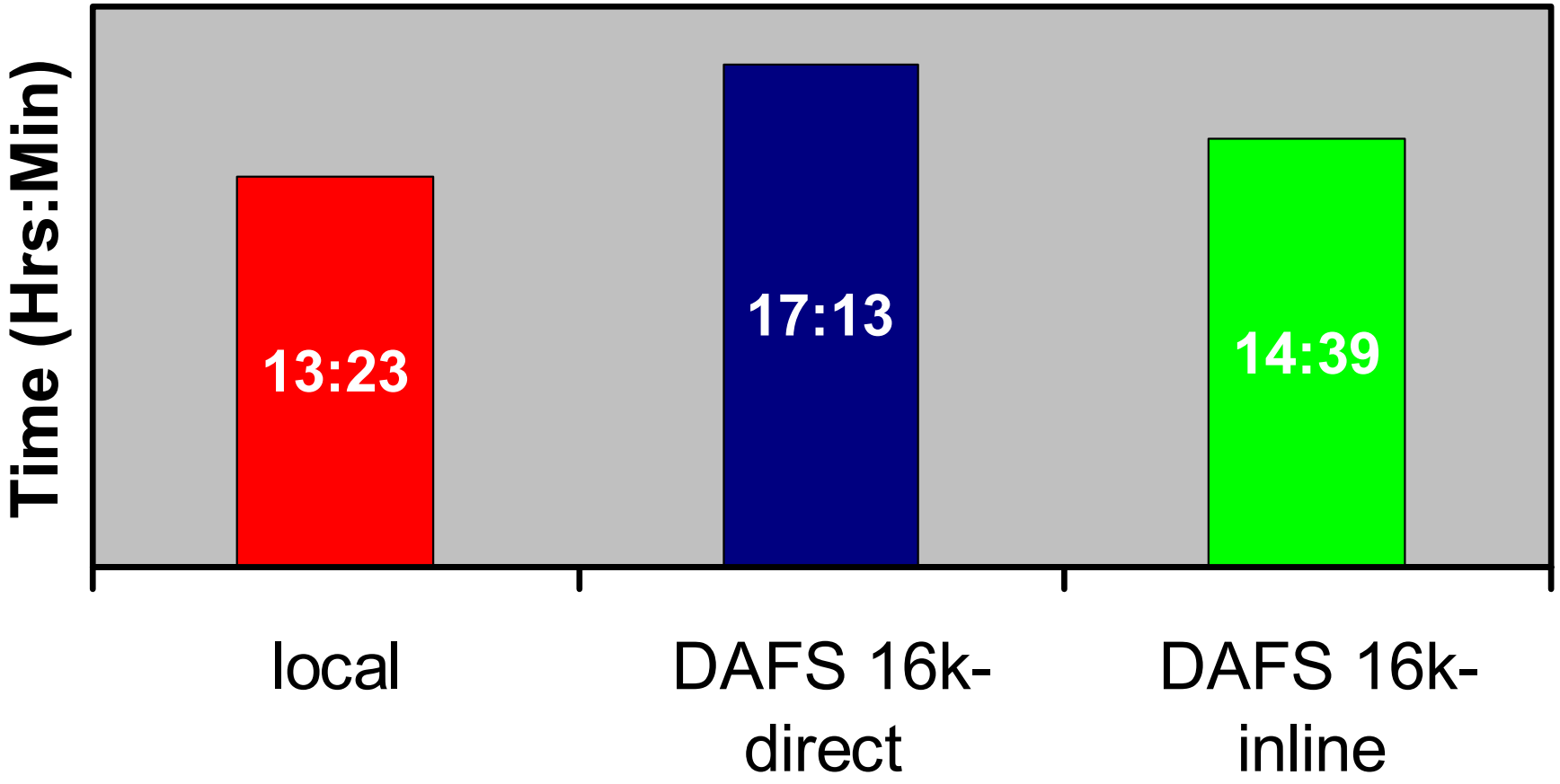
Odmlat performance



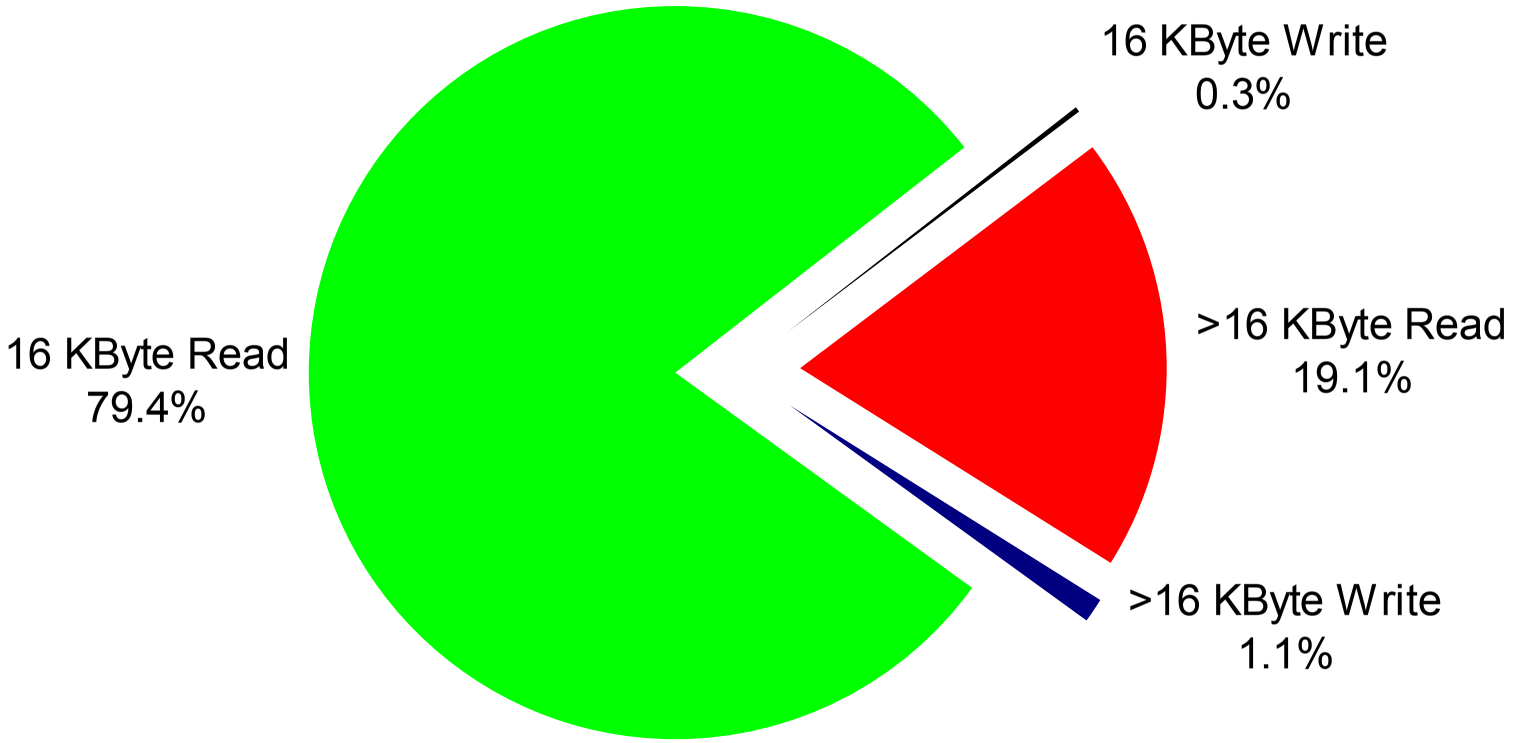
Oracle-based results

- Standard Database Benchmark - TPC-H
  - Written in SQL
  - Decision support benchmark
  - Multiple ad-hoc query streams with an “update thread”
  - 30GB database size
- Oracle configuration
  - All I/Os 512-byte aligned (required for unbuffered I/O)
  - 16K database block size
  - Database files distributed across two NTFS file systems
- Measurements
  - Compared average runtime for local vs. DAFS based I/O
  - Skipped official “TPC-H power” metric
  - Varied inline/direct threshold for DAFS based I/O

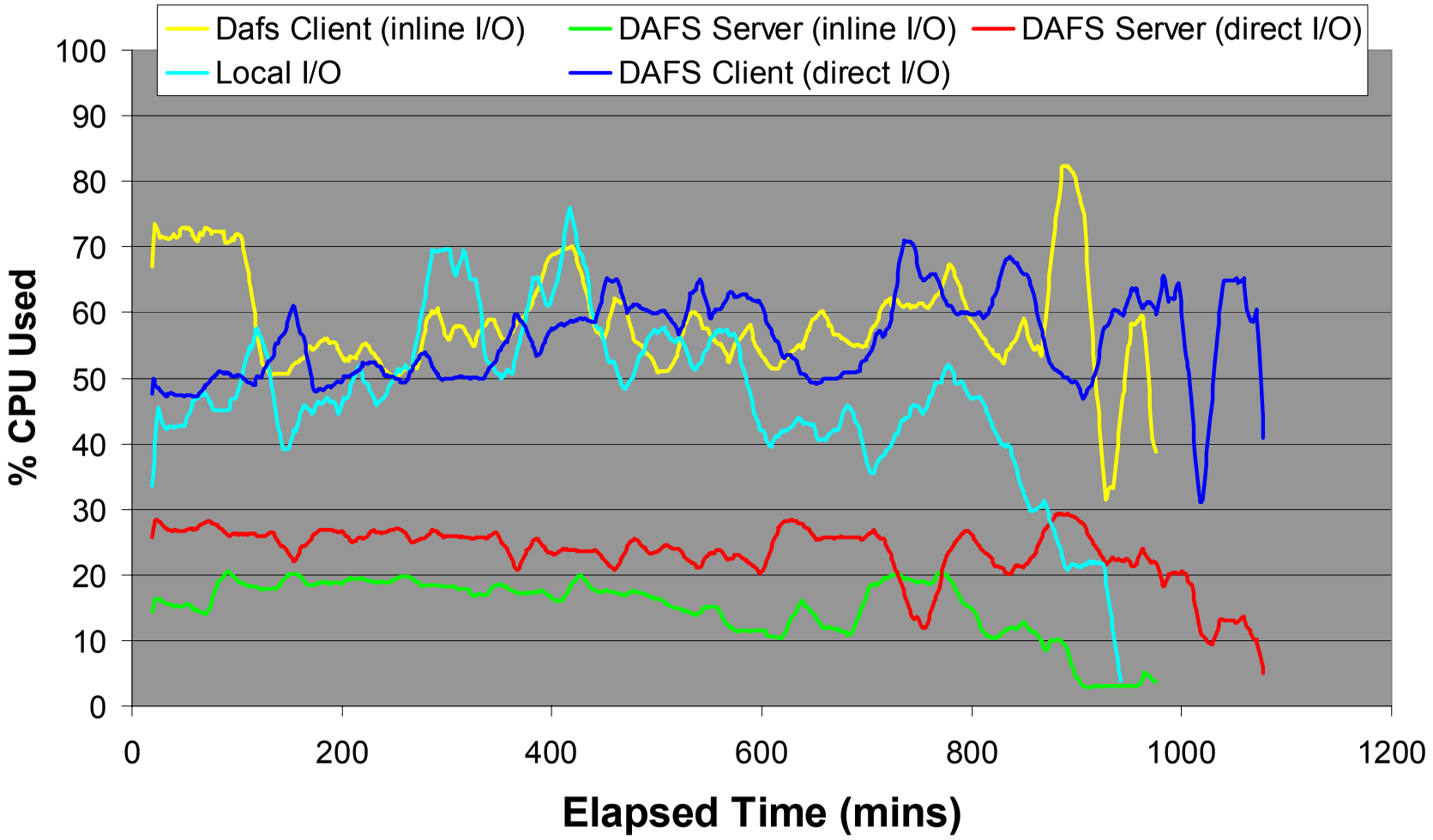
Oracle TPC-H Performance



Oracle TPC-H Operation Distribution



Oracle TPC-H CPU Utilization



TPC-H Summary

- Local I/O still faster
  - Limited ServerNet II bandwidth
  - Memory registration or copying overhead
  - Windows unbuffered I/O is already very efficient
- DAFS still has more capabilities than local I/O
  - Capable of cluster I/O (RAC)
- Memory registration is still a problem with DATs
  - Registration caching can be problematic
    - Can not guarantee address mappings will not change
    - ODM has no means for notifying NIC of mapping changes
  - Need either better integration of I/O library with Oracle or better integration of OS with DAT
    - Transparency is expensive!

Conclusions

- DAFS Server/ODM Client achieved performance close to the limits of our network
  - Local SCSI I/O was still faster
- Running a database benchmark, DAFS TPC-H performance was within 10% of local I/O
  - Also provides advantages of a network file system (i.e., clustering support)
- Limitations of our tests
  - ServerNet II bandwidth was inadequate – no support for multiple NICs
  - Needed to do client-side registration for all direct I/Os
- TPC-H benchmark was not optimally tuned
  - Needed to bring client CPU closer to 100%
    - More disks, less CPUs, other tuning
  - CPU offload is not a benefit if I/O is the bottleneck


hp
invent