Parallel Data Lab Research Overview

Garth A. Gibson
http://www.cs.cmu.edu/Web/Groups/PDL/

Reliable, Parallel Storage Subsystems

- configurable architectures; rapid prototyping

Discovering and Managing Storage Parallelism

- cost-benefit exploitation of application disclosure

Parallel Filesystems for Parallel Programs

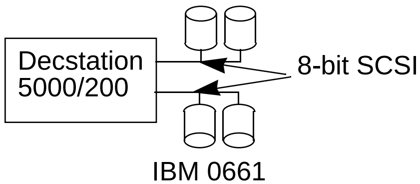
- application “controls”: hints, cache directives, redundancy

New Interfaces for Network-Attached Disks

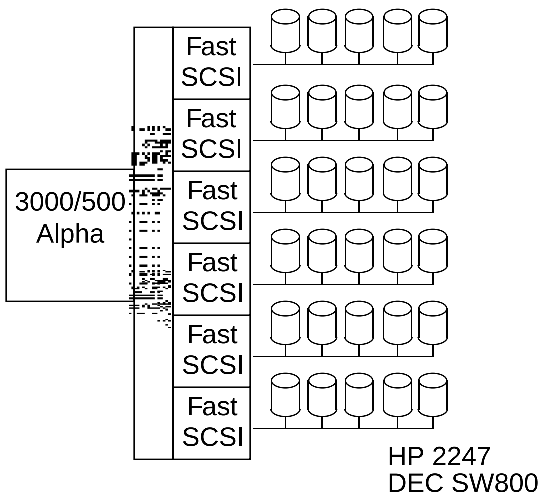
- scalable, secure, extensible storage systems

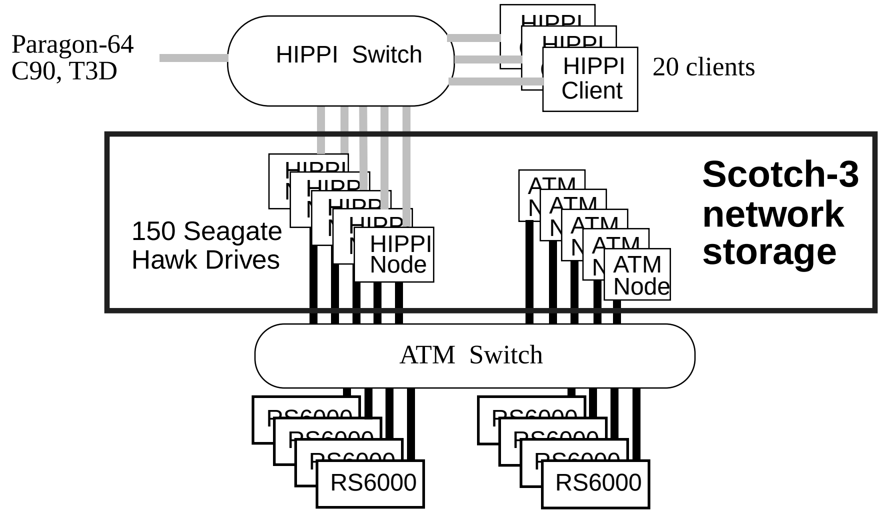
Scotch Parallel Storage Testbeds

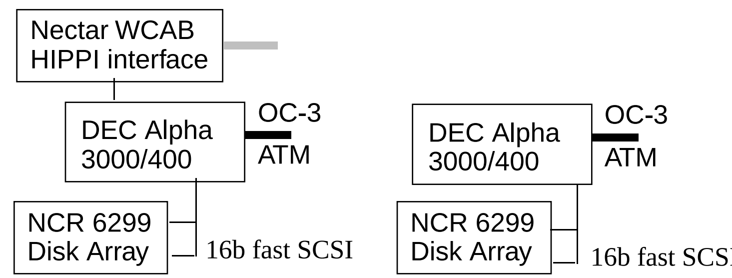
Scotch-1 direct-attach testbed



Scotch-2 direct-attach testbed







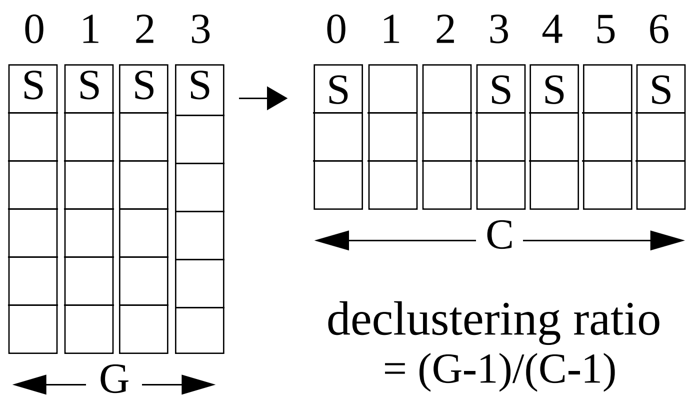
HIPPI Node Detail

ATM Node Detail

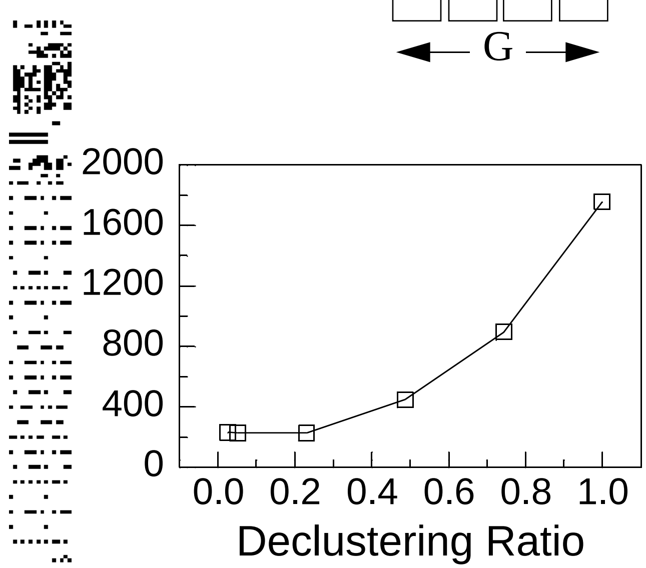
Scotch-1 decommisioned, Scotch-3 nets being debugged

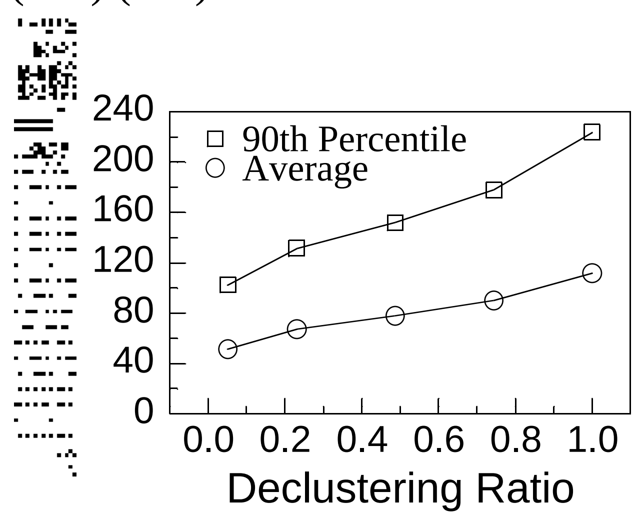
Beyond RAID 1-6 Example: Parity Declustering

- **Each parity block protects fewer than *N* data blocks**
- **Failure-induced workload balanced over all disks**



declustering ratio
= (G-1)/(C-1)



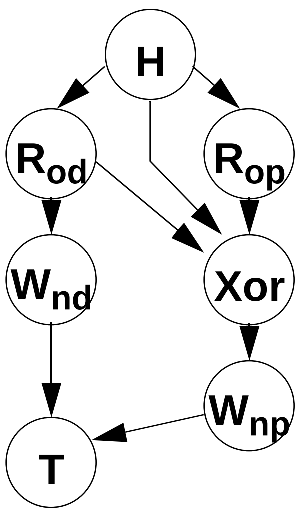


Rapid Prototyping and Evaluation for RAID

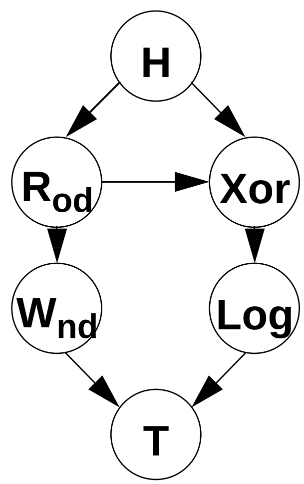
RAIDFrame: separate policy from mechanism

- Express RAID functions as Directed Acyclic Graph
- Execute DAGs on engine unaware of RAID architecture
- Distributable, portable “RAID N reference model”

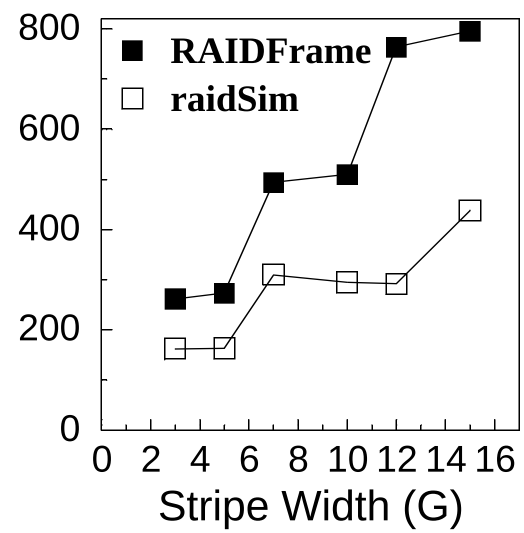
RAID Level 5 Small Write DAG



Parity Logging Small Write DAG



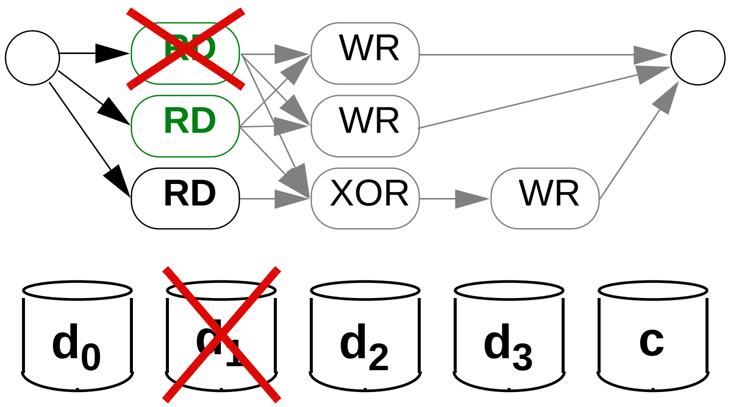
Example Evaluation: Parity Declustering



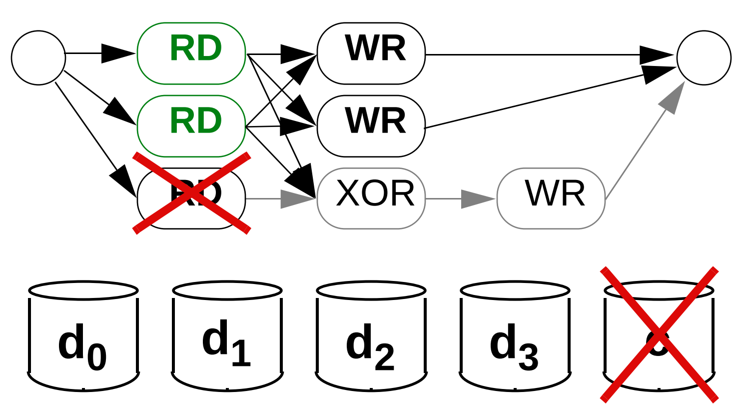
RAIDFrame as Research

Automating error recovery

- **DAG primitive handles individual error**
- **engine completes or cancels DAG and retries in new state**



unsuccessful commit, roll back
retry using new method



successful commit, roll forward
operation is complete

Automatic manipulation of DAGs

- **code simple DAGs, merge and optimize automtically**

Work To Be Done

Extensible caching for RAIDframe

- event-driven, composable triggers; write-deferring policies

Populate RAIDframe libraries

- log-structured, parity-logging, virtual striping, ...

Distribute RAIDframe widely

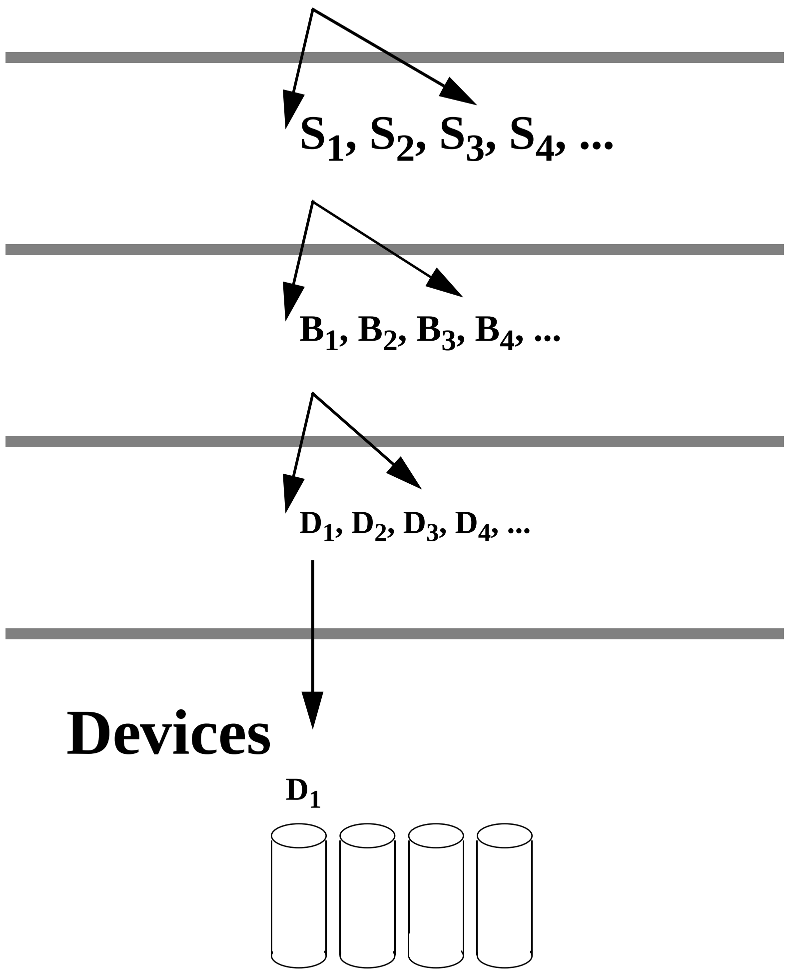
- hiring support staff; documentation underway

Automatic manipulation of RAIDframe DAGs

- commit point insertion, static & dynamic optimization

Overcoming Disclosure Bottleneck: Informed Filesystems

Application $F_1, F_2, F_3, F_4, F_5, ...$

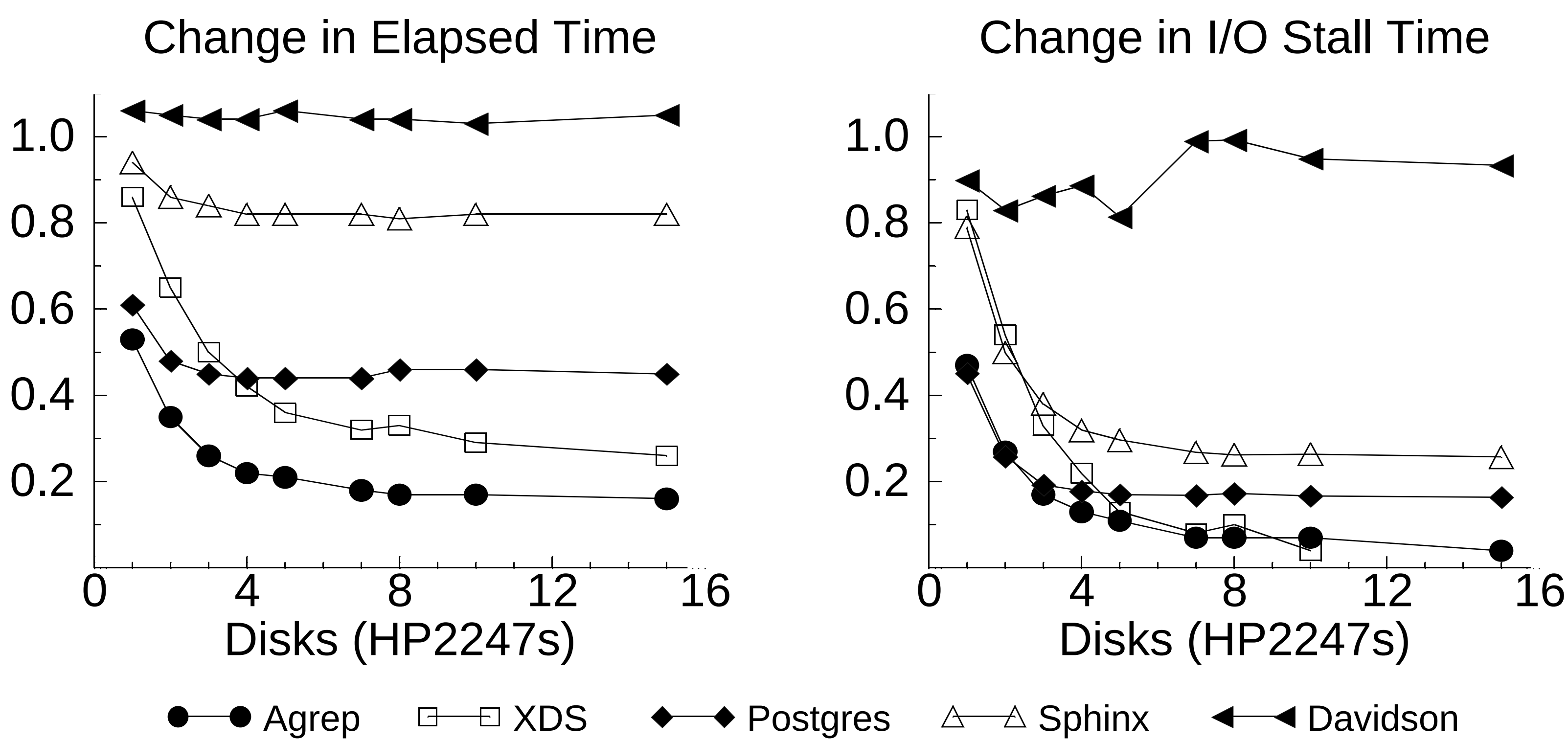


- **Expose concurrency**
  - **overlap I/O and computation**
  - **overlap I/O and think time**
  - **overlap I/O and I/O !!!!**
  - **I/O optimization**
    - seek scheduling
    - batch processing
- **Cache management**
  - **balance buffers between prefetch and demand**

Informed Prefetching Prototype Results

Annotated text search, 3D visualization, database join, speech recognition, computational physics

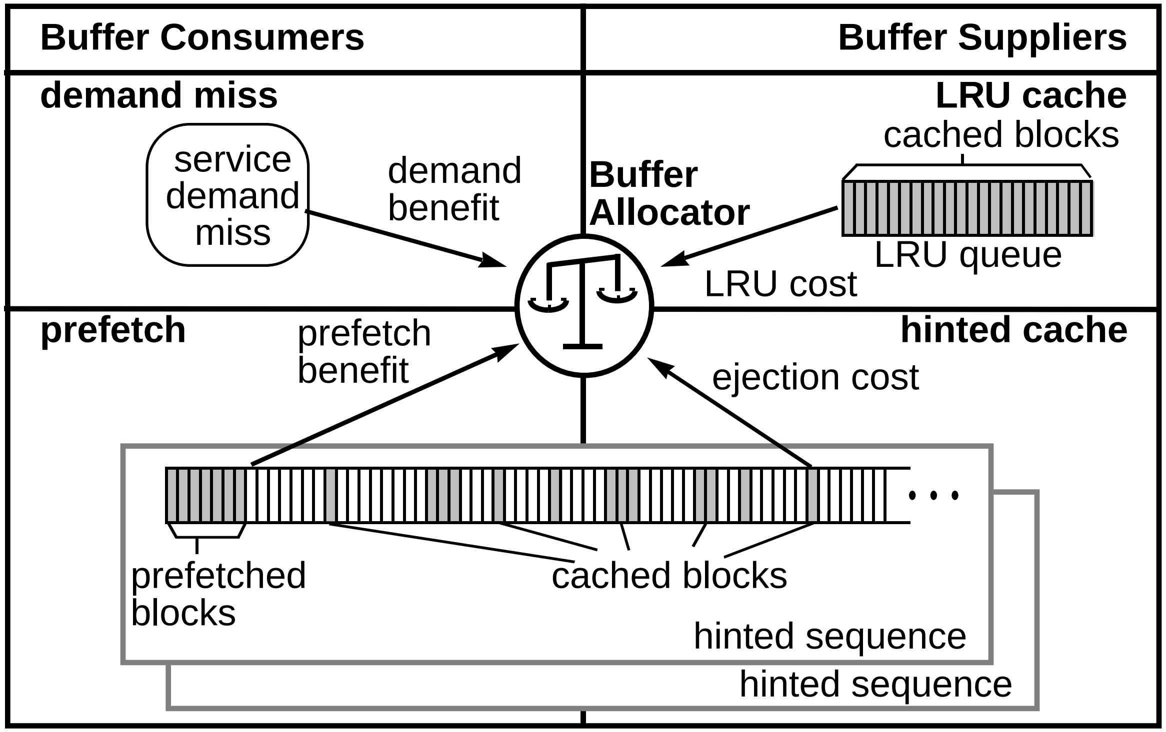
DEC Alpha (150 MHz), OSF/1, 12 MB LRU cache



Carnegie Mellon

Parallel Data Laboratory

Informed Cache Approach



Estimate:

- *benefit* of giving a buffer to a *consumer*
- *cost* of taking a buffer from a *supplier*

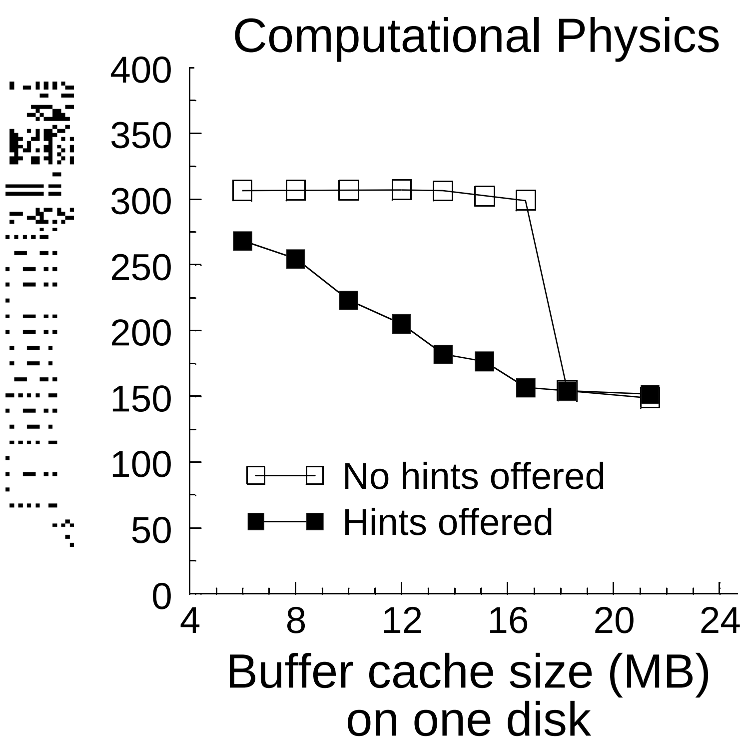
Reallocate when *benefit* > *cost*

Informed Caching Prototype Results

Re-examine computational physics (Davidson)

- **same DEC Alpha, one hp2247 disk**

Adapts cache replacement policy to workload



- **Better cache effectiveness**
  - **without hints, no benefit until data set fits in cache**
  - **with hints, MRU-like benefit**
- **Most effective where informed prefetching is least (limited bandwidth)**

Work To Be Done

Automatic extraction of disclosure

- context-dependent access pattern learning
- compiler extraction for out-of-core scientific codes

Non-homogeneous and network devices

- non-uniform prefetch depth to avoid hot spots

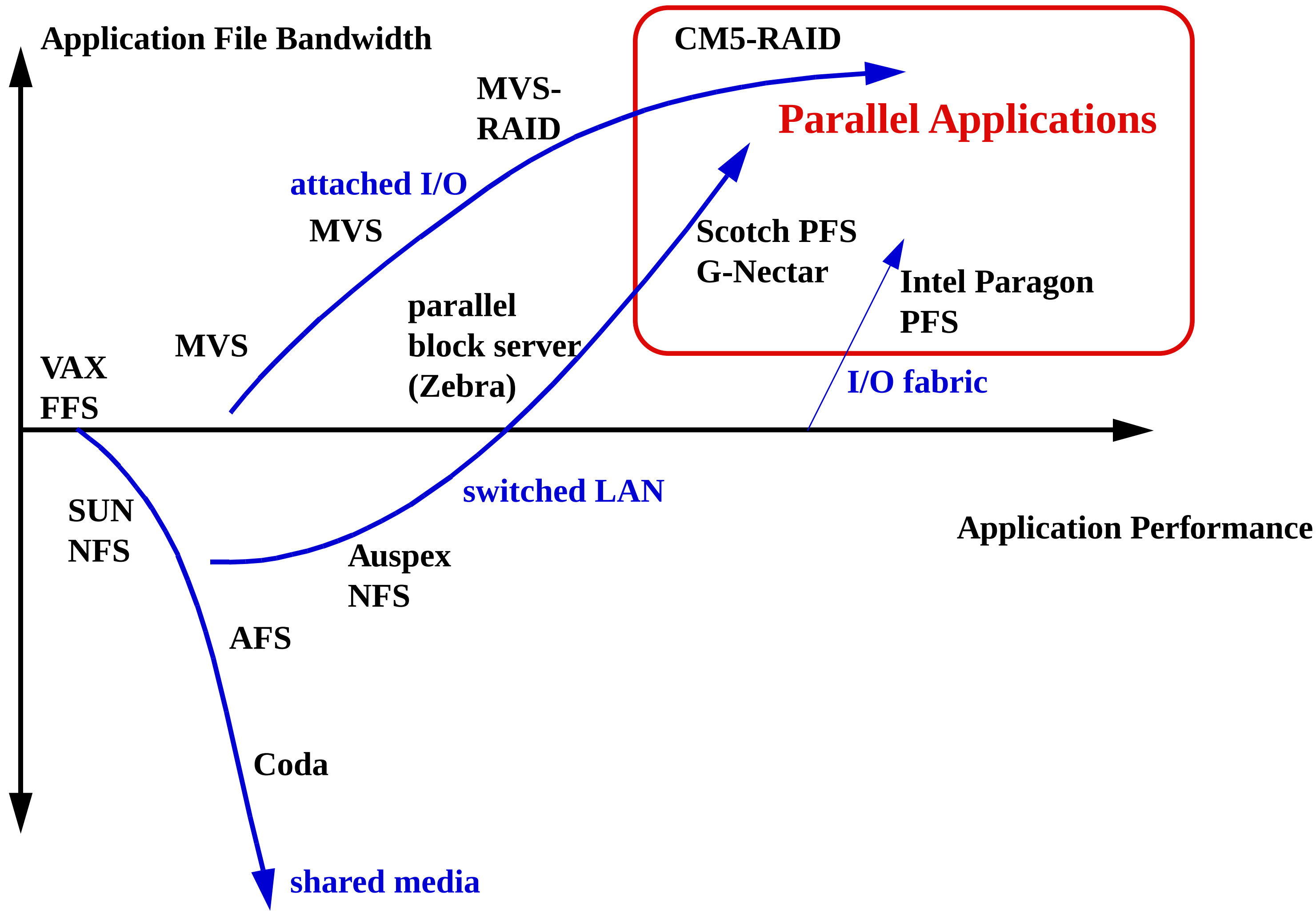
Integrate with VM management

- prepage predictably accessed memory objects

Integrate with parallel file system

- global management of server and client cache space

Support for I/O-intensive Multicomputer Apps



Efficient, scalable file access in heterogenous multicomputers

Scotch Parallel File System Approach

Resource management via informed prefetching and caching

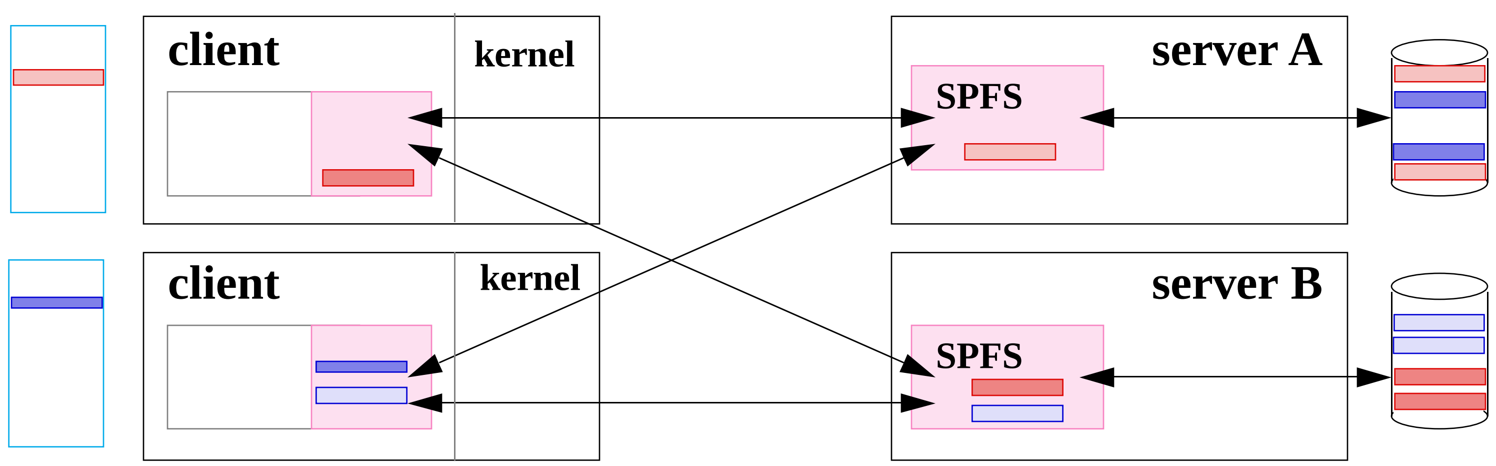
Optimistic client caching (like entry consistency)

- filesystem synch piggybacked on application synch

PFS semantics in library - no central mechanism

Per-file redundancy for dynamic, configurable availability

Co-developing PFS API for Scalable I/O (with IBM, Intel)



Network Support for Parallel Flows

Switch-style scalable storage transfers in multiple streams

- but networks deal in individual connections

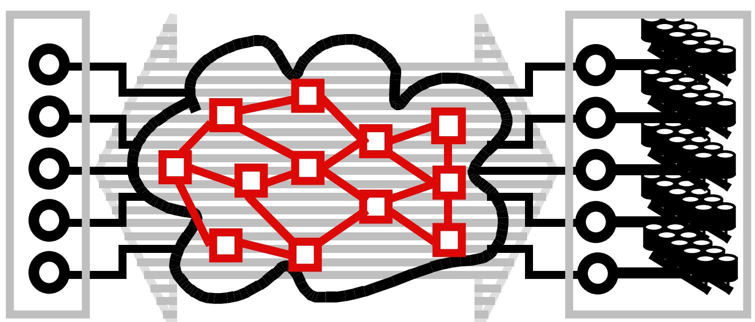
Network support for coordinated routing of multiple streams

- multi-path connections, source routing, load-sensitive

API for negotiating parallel flow service

- enable applications to adapt to bandwidth availability

Switched Networks



Parallel client

Parallel file server

Work To Be Done

Scotch parallel file system evaluation

- first prototype fighting ATM; second in design
- integrate coordinated routing for parallel flow

Integrate network, file system, programming tools

- parallel flow service, SPFS and PVM/Dome/Pyxis

Application evaluation

- computational physics (Hartree-fock)
- seismology (Dip Moveout)
- spatial, geographic databases (Census maps)
- distribution to Scalable I/O application groups

Storage Architecture Trends

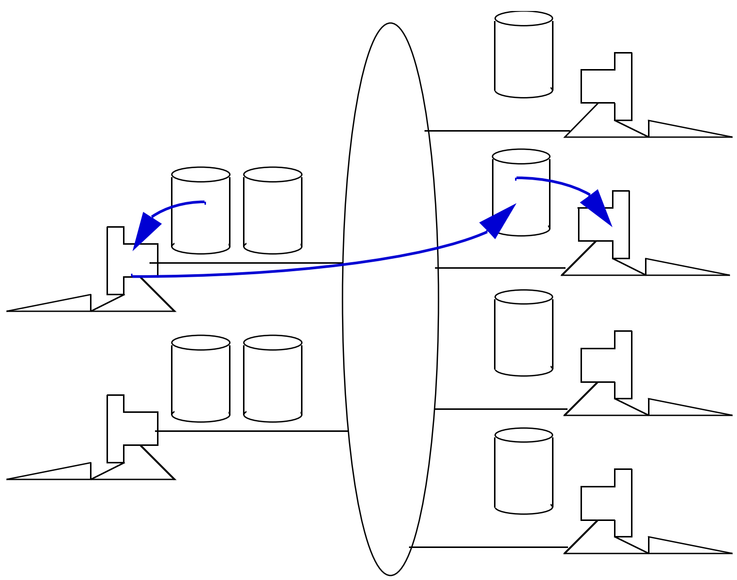
Growth in drive-embedded functionality

- disk scheduling, readahead/writebehind, RAID support

Migration to serial, many-ported drive interface

- faster drives, multi-drive bandwidth, drive-to-drive transfers

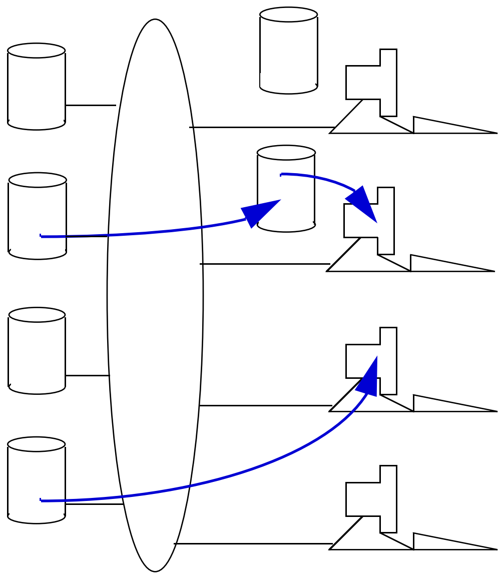
Data bytes travel over LANs to real consumer



Workstation a poor (costly) server

- designed around caching for processor-local work
- network bandwidth limited to little more than single disk bandwidth
- induces extra copying

Network Attached Secure Disks



Attach drives directly to network

- fewer copies and appropriate bandwidth
- addressability for drive-to-drive transfer

External filesystem personality

- “traffic cop” DMA management

Raise drive functional interface to file system level

- < file, offset, length > for better readahead, remapping, ...

Integrate drive into LAN security protocol

- tamper-resistant encryption for authentication check
- user configurable encryption over net or on media

Work To Be Done

File system

- SPFS/SIO-API strawman; flexible access control; parent FS

Security

- lightweight protocol; infrequent server interaction; bootstrap

Networking

- SCSI-like local efficiency; wide-area access interoperability

Device architecture

- cost-effective drive microarchitecture; encryption interface

Embedded apps - decentralized video

- self-scheduled to deadlines; drive execution model
- target video service for information on demand apps

Summary: Evolving Parallel Storage Requires ...

Rapid prototyping for RAID: RAIDframe

- flexible, architecture-rich, automated recovery

File system support for storage parallelism

- informed prefetching and caching

Parallel file systems for parallel applications

- highly available, highly scalable, global resource management

Network-Attached, Secure Disks (NASD)

- eliminate workstation as DMA device and raise interface level

Industrial interaction and support

- HP, Symbios, IBM, Seagate, DEC, DG, EMC, STK

Points of Leverage