Solid State Storage for File Systems

Roger Haskin
Senior Manager, File Systems
IBM Almaden Research Center

The Obligatory GPFS Plug

- GPFS parallel file system product originated as Tiger Shark prototype at IBM Almaden Research Laboratory
- Research continues to be involved in prototyping and developing new GPFS features and related technology
- 25 patents granted
  - 50 applied for
- 6 refereed publications
- … but this is not a GPFS talk

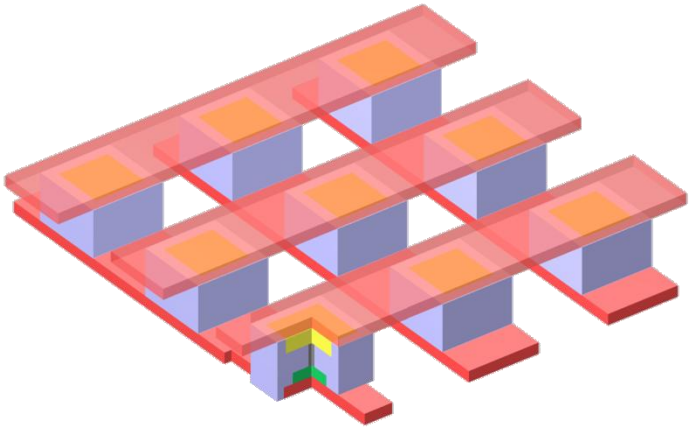
The Impact of Solid-State Technology on File Systems

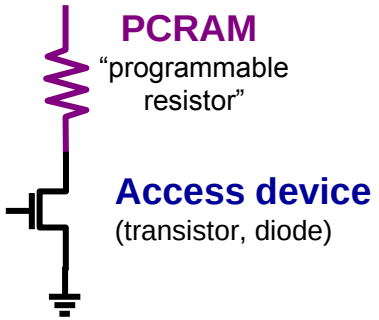
- Since 1957 (RAMAC), online storage has been predominantly magnetic disk
- The job of storage system software (file systems and databases)
  - To reliably store and retrieve data
  - While masking the latency and throughput limitations of disks
    - Sophisticated I/O scheduling
    - Careful layout of data on disk
    - Aggressive caching in memory
    - Parallelism
- Solid-state storage alters the whole design point of file and database system software
  - Smaller, fixed access latency
  - Potentially much higher throughput
  - Different, hopefully better, reliability

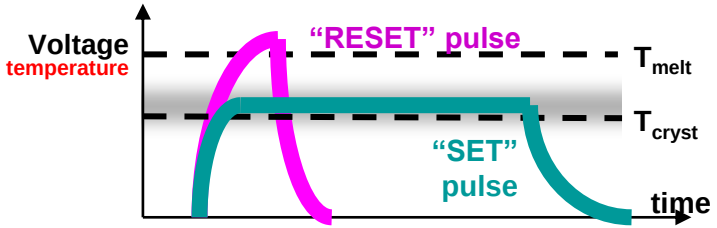
- How do we best take advantage of solid state?
- Does solid-state take all the sport out of storage system software?

Solid State Technologies

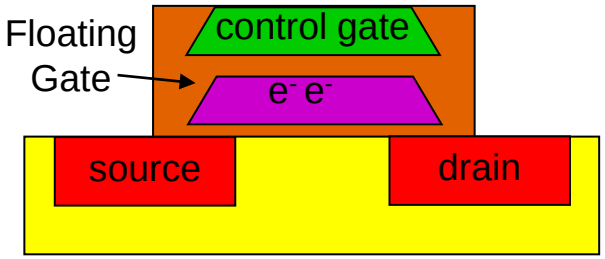
Phase-Change RAM (PCM)







Flash Memory

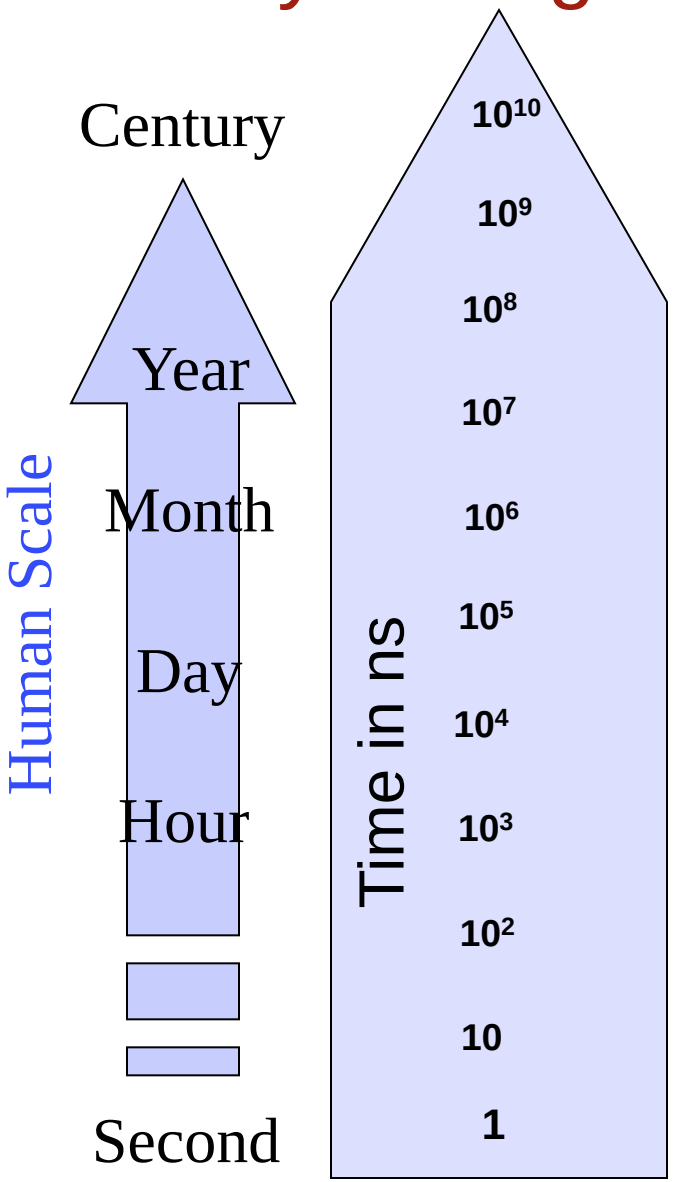


	Flash*	PCM*
Cell Size	2 F^2	5.8 F^2
Read	20 us.	1 us.
Write	200 us	3-5 us
Erase	2ms	n/a
Endurance	low	high

* estimates

R. Freitas, SustainIT'10

Memory/Storage Stack Latency Problem



TAPE access (40s)

DISK (5ms)

Storage

FLASH read (20 us)

Phase Change Memory (100 – 1000 ns)

DRAM (60ns)

L2 cache (10ns)

CPU operations (1ns)

Memory

Freitas & Chu, FAST'10

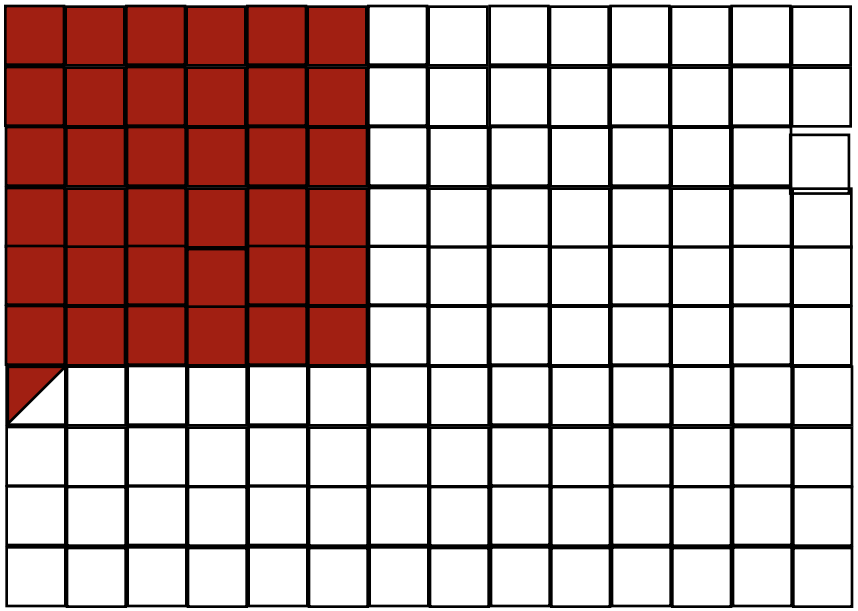
The Promise of Solid State Disk

- By 2020, Solid-state storage should revolutionize data centers

Bandwidth Driven Storage System: 400 TB/s

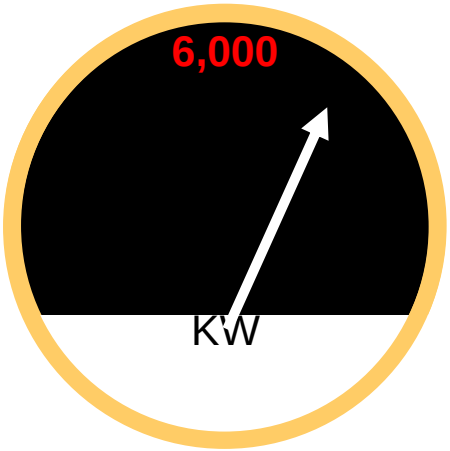
DISKS

Floor Space



6000 Square Feet

Power



R. Freitas, SustainIT'10

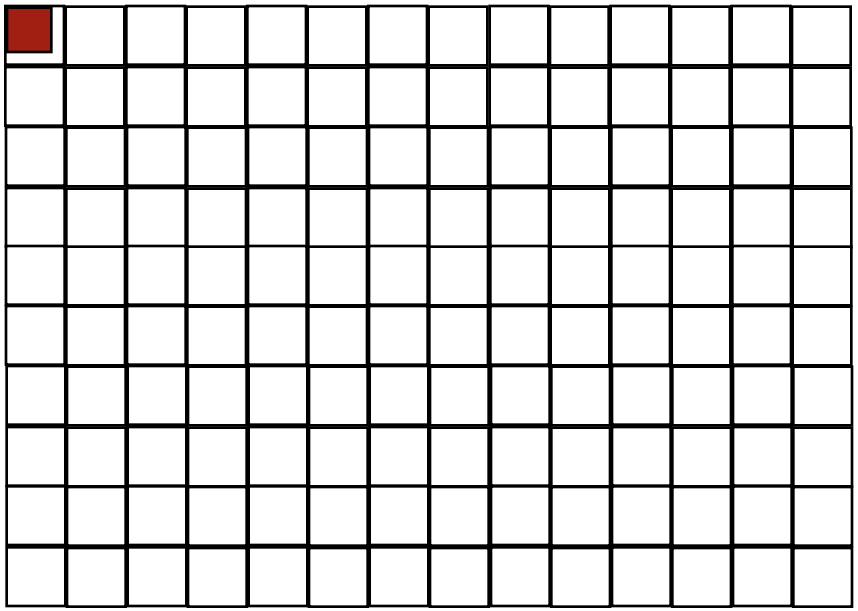
The Promise of Solid State Disk

- By 2020, Solid-state storage should revolutionize data centers

Bandwidth Driven Storage System: 400 TB/s

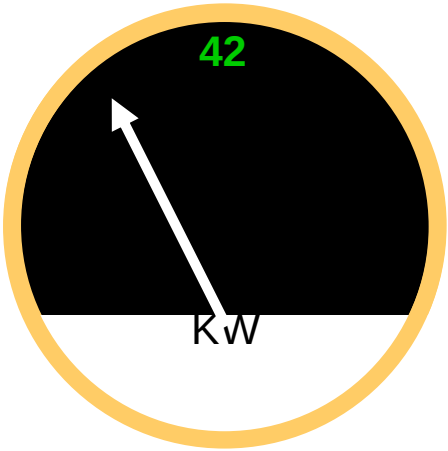
SCM

Floor Space



85 Square Feet

Power



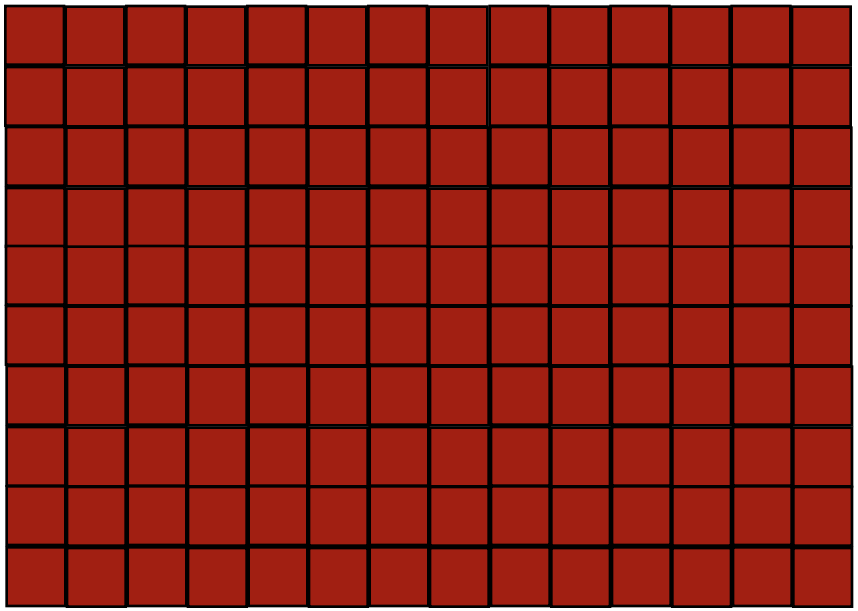
The Promise of Solid State Disk

- By 2020, Solid-state storage should revolutionize data centers

Transaction Rate Driven Storage System: 2000 MOP/s

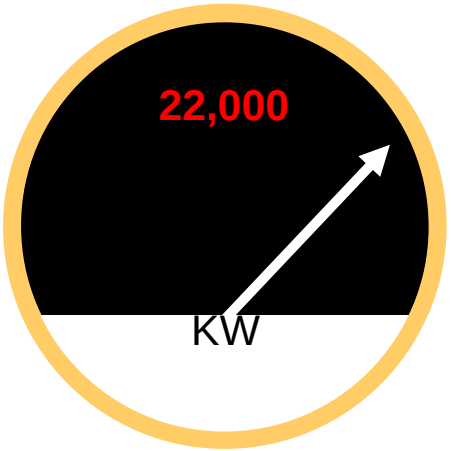
DISKS

Floor Space



23,000 Square Feet

Power



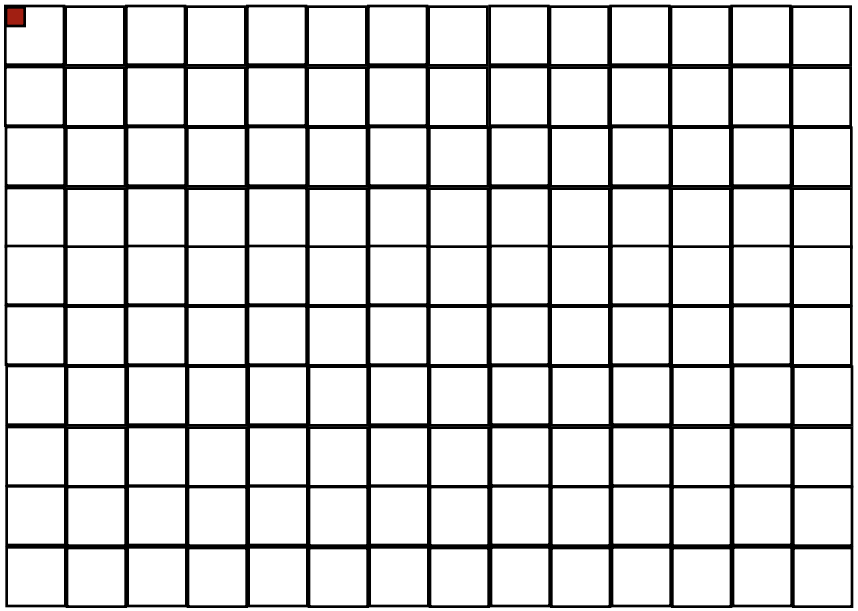
The Promise of Solid State Disk

- By 2020, Solid-state storage should revolutionize data centers

Transaction Rate Driven Storage System: 2000 MOP/s

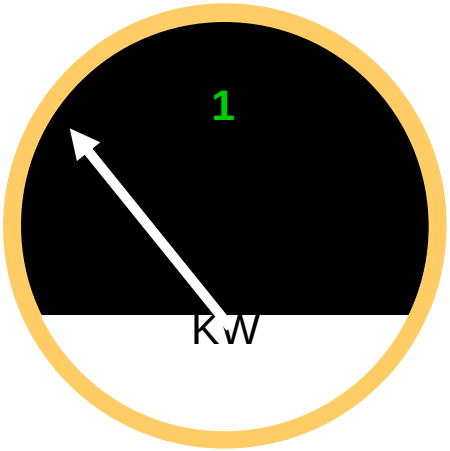
SCM

Floor Space

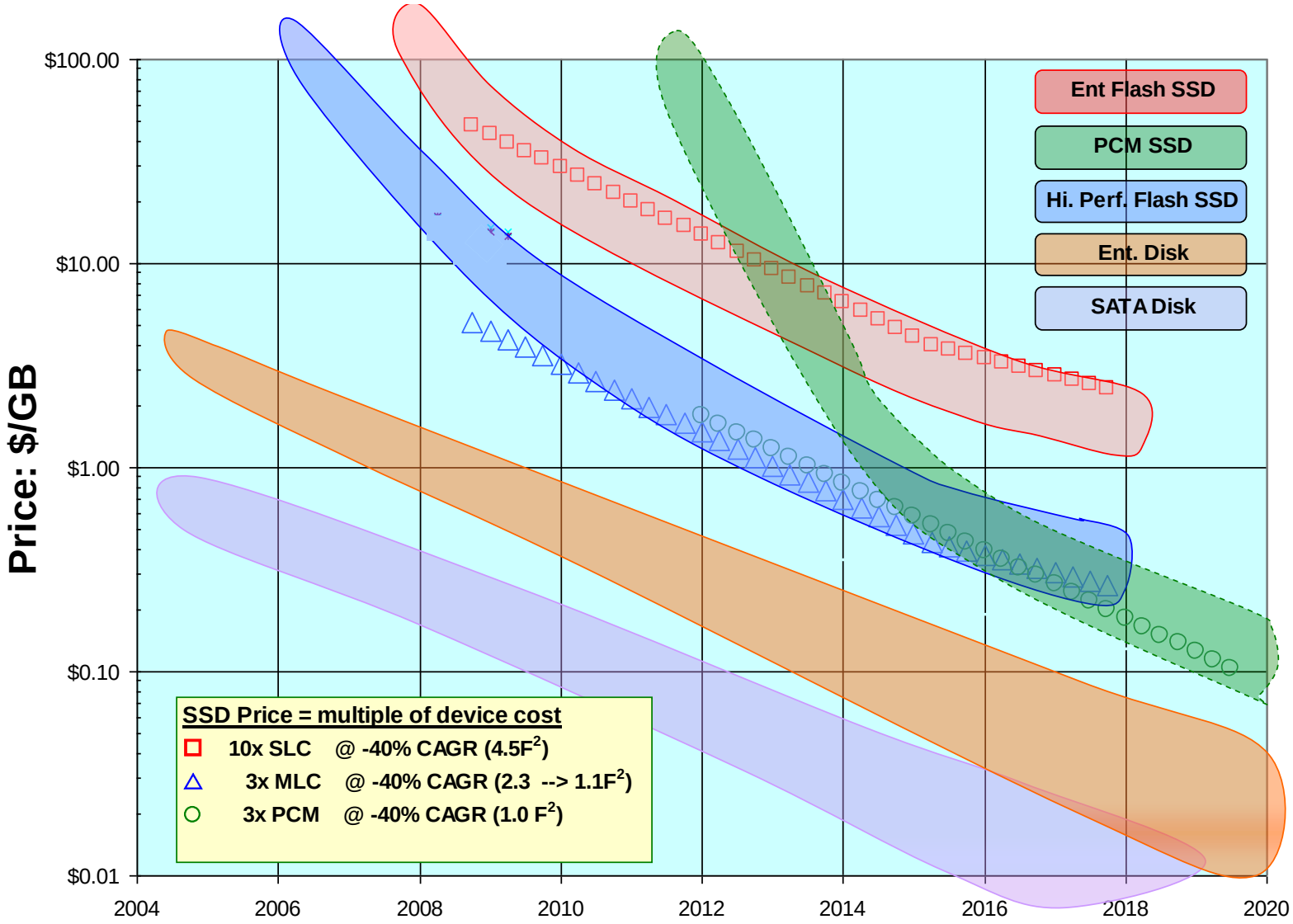


11 Square Feet

Power



Subsystem Price Crystal Ball



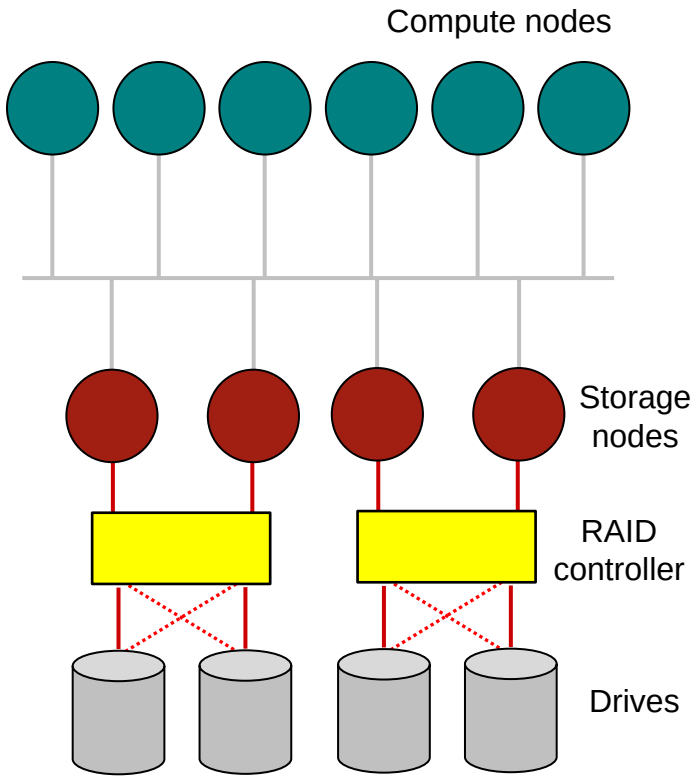
R. Freitas, SustainIT'10

Implications for File Systems

- Solid-state 1000x-10000x faster than disk, but still 10x as expensive per bit
  - Barring unforseen circumstances, disks will still be around for a while
  - The rich may be able to replace disk completely
  - The great unwashed will need to get by augmenting disk with solid-state
- The obvious uses
  - Metadata – the low-hanging fruit (but you still need a ladder)
  - Data – Which? Mine or yours?
- Are we even going about this right?
  - Is solid-state a fast replacement for disk, or a persistent replacement for DRAM?

File System Metadata on Solid-State Disk

- For many workloads, file system performance is dominated by metadata update latency
- Several modern file systems allow metadata (directories, inodes, allocation maps, etc.) to be stored on separate disks
- Put metadata on solid-state drives
- Experiments with GPFS show up to 3x performance for metadata intensive workloads when metadata stored on solid-state storage (YMMV).
- *But 3x is not 1000x!*
  - Still have locking overhead (GPFS)
  - … or metadata server overhead (others)
  - … plus network and I/O overhead



Data on Solid-State

- If solid state storage remains 10x-30x cost of disk it will remain too expensive to put all user data on solid-state
- Optimizing data placement onto the various types of storage will be increasingly important
- E.g. GPFS policy-based ILM
  - Multiple storage pools, e.g. SSD, SAS, SATA
  - Declarative Placement, migration, deletion policies based on file attributes
  - … age, access time, file type (database vs. media data), project, …
  - Position in name space independent of placement on physical storage
- The challenge: policies that work at scale

– **Placement policies**, evaluated at file creation, example

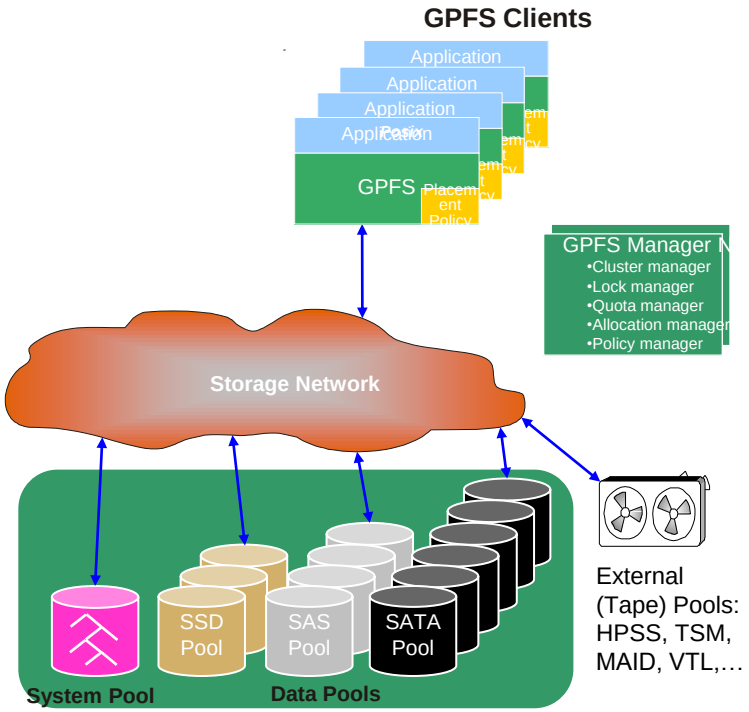
```
rule rogersfiles set pool SSD for fileset
  rogersfileset
rule otherfiles set pool SAS
```

– **Migration policies**, evaluated periodically

```
rule cleanssd migrate from pool SSD threshold
  (90,70) to pool SAS
rule cleansas when day_of_week() = monday migrate
  from pool SAS to pool SATA where access_age > 30
  days
```

– **Deletion policies**, evaluated periodically

```
rule purgesata when day_of_month() = 1 delete from
  pool sata where access_age > 365 days
```



Is SSD the best packaging for solid-state storage?

- SSD write latency 50-100 usec, RAID controller write to cache around the same
- Counting I/O setup and queuing, write latency around 1 msec.

… on the other hand …

- Phase Change Memory write latency around .1 - 1 usec
- Blue Waters switch latency around 1 usec.

Does solid-state obsolete the I/O model of the last 50 years?

More Radical Approaches

- Move data processing to the storage (Active Storage, Reidel et. al.)
  - Advantage: can minimize latency, e.g. by packaging persistent storage as memory rather than as a disk drive
  - Disadvantages:
    - Difficult to provide a secure, high-performance application environment
    - System imbalance : active storage devices may have too much or not enough CPU, memory
    - What about write-intensive workloads (like HPC)? The data has to come from somewhere!
- Persistent Global Memory
  - Storage software often uses locks to serialize access to shared storage (Oracle RAC, GPFS)
  - If you have a lock, it's easier to access storage as memory than as disk
    - Memcpy() – no context switch, no interrupt, no pin, no block boundaries
    - Local DRAM is still 20x faster than global storage, still use it for buffer cache

Questions?