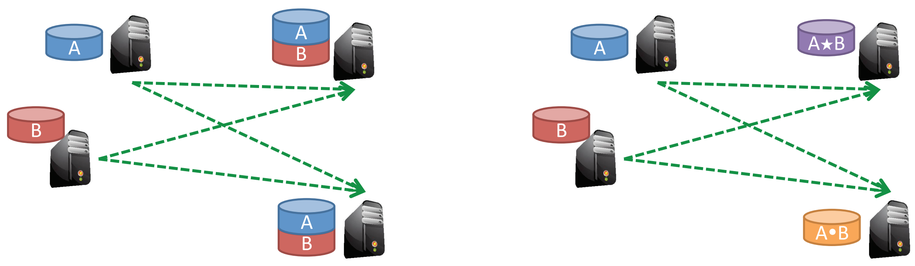
DiskReduce: RAIDing the Cloud - Grouping Choices

Bin Fan, Wittawat Tantisiriroj, Lin Xiao, Garth Gibson

Overview

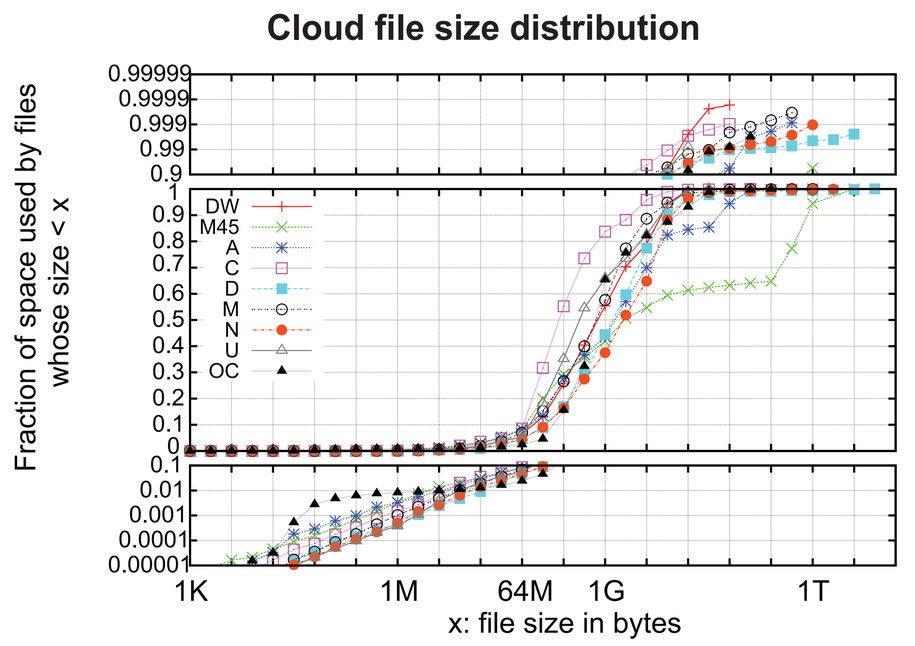
Google FS/ HDFS on Data Intensive Scalable Computers

- **Triplication can recover from 2 failures but it trades 200% extra storage for this redundancy**
- **Parity saves storage and tolerates the loss of any two nodes**

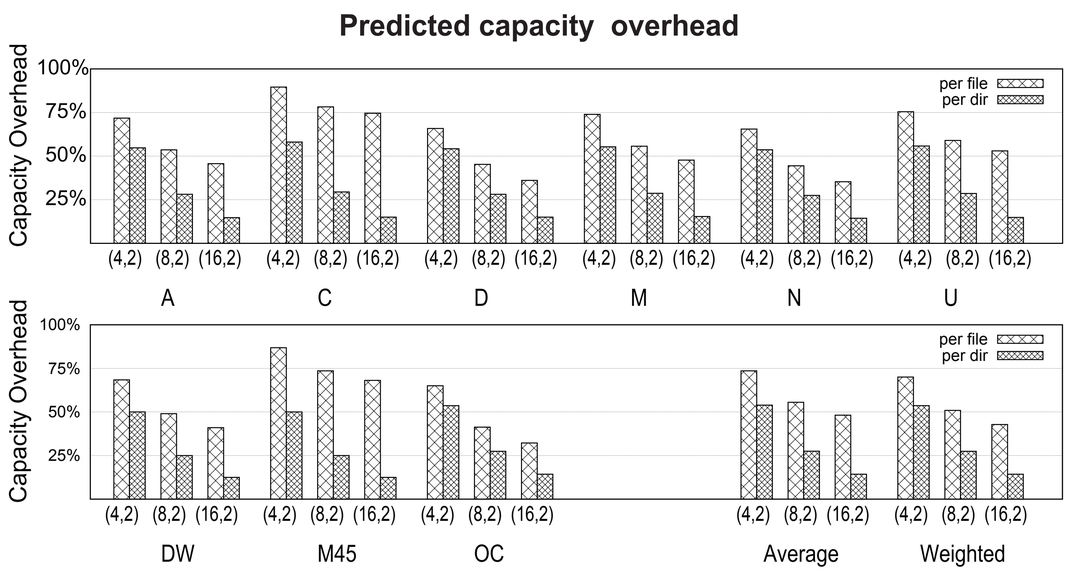


GFS / HDFS DiskReduce

Capacity Overhead



- **Across 9 file systems (1.5 - 21 PB), 30 - 80% of the storage used by files smaller than 1 GB (size of 16 blocks, 64 MB each)**
- **Since each block is large (64 MB by default), there will be few blocks in per-file RAID sets**



- **With 8 data blocks in a RAID set, per-file RAID 6 requires about 56% capacity overhead while per-dir RAID 6 requires only 28% overhead (25% is minimal)**

RAID Set Definitions

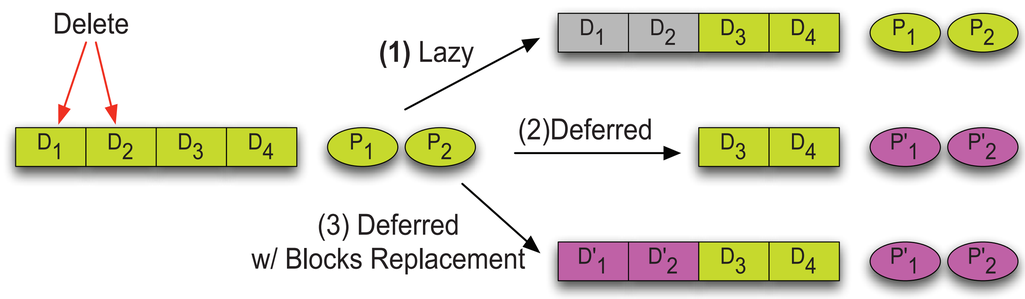
RAID Per-file: blocks in a RAID set are from the same file

- **+ Simple**
- **- Too much overhead**

RAID Across-files: blocks in a RAID set can be from different files

- **+ Per-directory RAID 6 can achieve lower overhead**
- **- Small write problem - potential read-modify-write to update parity blocks on file deletion**

Handling Deleted Blocks

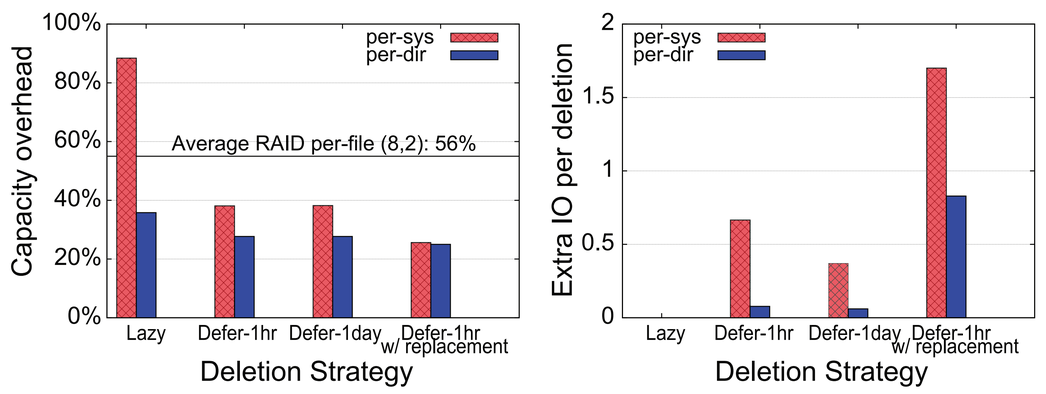


3 ways to handle deletion:

1. **Lazy Deletion**
   - **Never recalculate RAID equation so only recover space when all blocks in a RAID set are deleted**
2. **Deferred Deletion**
   - **After a timeout to allow all blocks to be deleted naturally, recalculate RAID over smaller data set**
3. **Deferred Deletion with Block Replacement**
   - **When recalculating check blocks, add new data blocks into existing RAID set to prevent RAID sets from becoming shorter**

Trace analysis

Yahoo! M45 trace over 2000 hours (80 days)



- **Per-system is not recommended, either high capacity overhead or high I/O cost**
- **Lazy per-dir needs no extra I/O, but its capacity overhead (36%) is significantly larger than minimal (25%)**
- **Defer 1 day per-dir reduces capacity overhead to near minimal (28%), paying only 0.06 (64MB) I/O per block created and deleted (which at least 3 I/Os)**

Carnegie Mellon YAHOO! facebook Parallel Data Laboratory PDLR11

DiskReduce: RAIDing the Cloud - Encoding Options and Reliability

Bin Fan, Wittawat Tantisiriroj, Lin Xiao, Garth Gibson

Immediate vs. Background Encoding

Immediate encoding:

+ Efficient

- Complex: Handling failures on critical path

Background encoding:

+ Simple & no change in client code

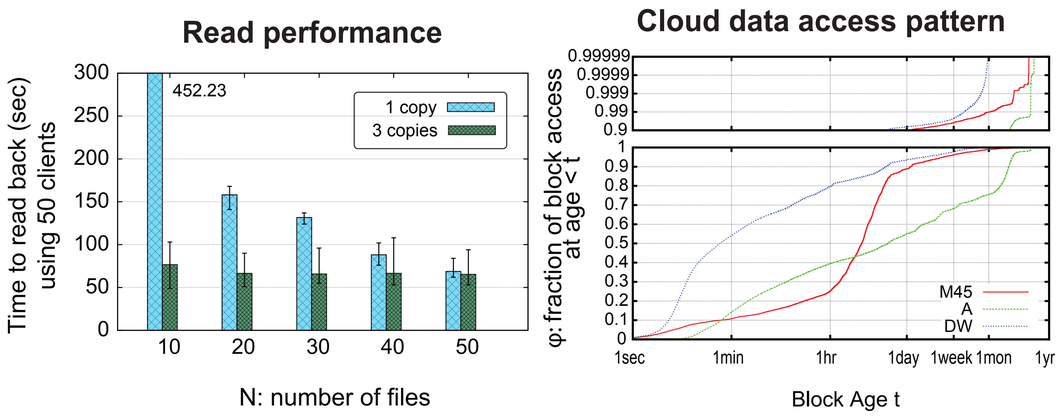
+ Cache young data for higher read bandwidth

- Less efficient

Performance Implication

75GB dataset written by N clients (1 file each) with 1 or 3 copies

- **Note first copy stays in writing node's disk**
- **Hotspots benefit from replication**

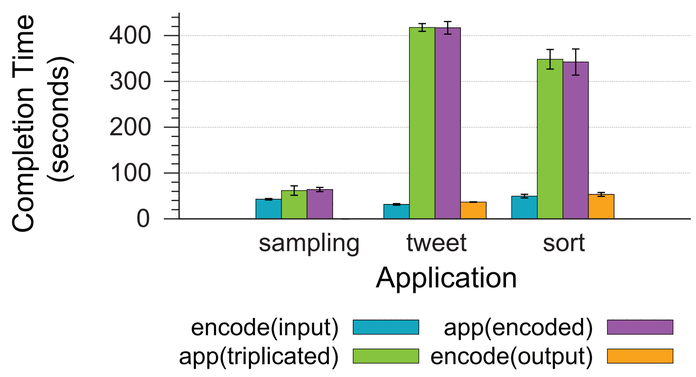


Treat triplicated data as a cache:

- **Locality: about 90% of data blocks are accessed within 1st day after creation in M45 & DW and 50% in cluster A**
- **Even with hotspot, most accesses can get benefit from replication if encoding deferred by 1 day**

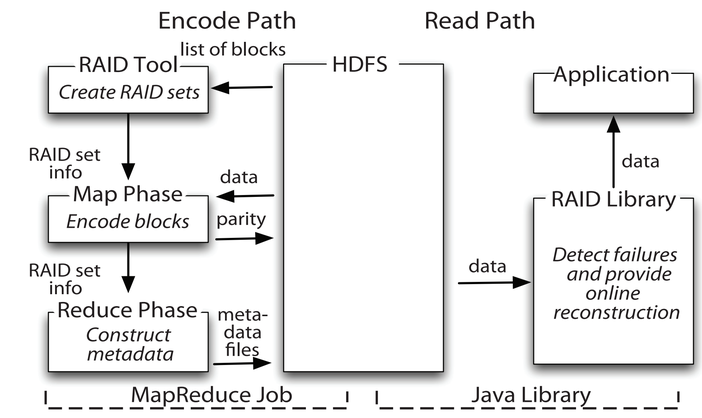
Application Performance

- **Sampling (B. Fu): Read 150GB astronomy data-set**
- **Twitter (B. Meeder): Reformat 24GB dataset to 56GB**
- **Sort: sort 128GB dataset**



- **The time each application takes when input dataset are triplicated or encoded are comparable.**
- **20% slow down if encoding is done during busy time**
- **Or need at least 20% of idle time**

Prototype



- **The prototype is built as a tool and a client library**
  - **Tool (Mapreduce): encode a directory into RAID sets or repair corrupted files**
  - **Library: detect and correct missing data while reading**
- **Released as Mapreduce-2036 patch for HDFS 0.22.0 @ http://issues.apache.org/jira/browse/MAPREDUCE-2036**
- **60 nodes (two quad-core 2.83GHz Xeon, 16GB memory, four 7200 rpm SATA 1TB disks, 10 Gigabit Ethernet)**
- **Dataset: 240GB (3,840 files, each 64MB in size)**

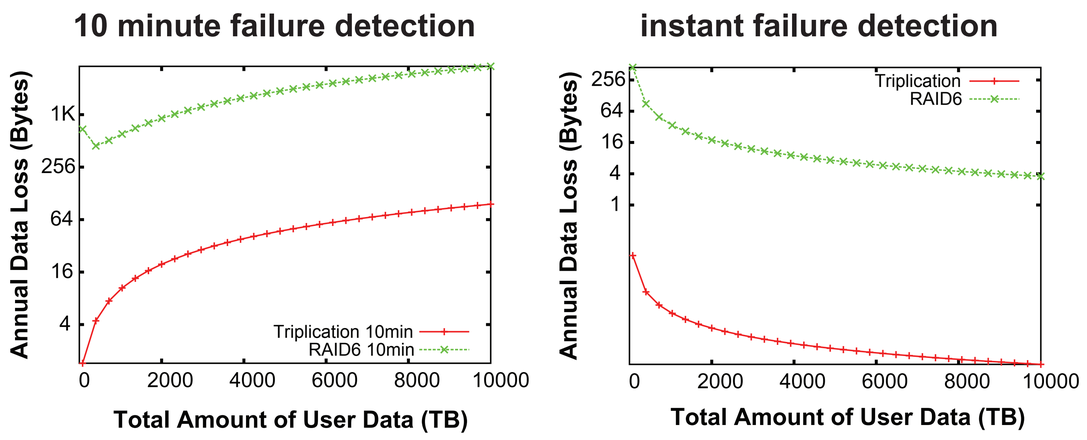
Operation	Throughput GB/s(stdev)	Disk I/O GB/s (stdev)
Write(Triplication)	1.93(0.06)	5.80(0.18)
Encode(RAID6 8+2)	3.69(0.34)	4.61(0.43)
Repair	0.23(0.02)	2.09(0.19)

- **Encoding is fast but reconstruction needs improvement**

Reliability Modeling

We expect Triplication to be more reliable than RAID6

- **Time to fail/repair model is exponentially distributed, 2% annual failure rate**
- **1TB/disk 80% full, 64MB/chunk, 8+2, 25MB/s/disk repair**
- **Compare bytes lost per year as a function of total sizes**
- **Orders of magnitude difference is still only a few bytes/year**



Interaction of parallel repair and detection delay

- **With instant detection, surprisingly scalable repair improves reliability with scale**
- **With more common delayed detection, unsurprisingly scalable repair less effective**

Carnegie Mellon YAHOO! facebook Parallel Data Laboratory PDLR10