DiskReduce: Making Room for More Data on DISCs
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

DiskReduce
- With parity, we can use a lot less storage and still tolerate the loss of any two nodes
- DiskReduce 1.0 uses a background process to search for optimal blocks to replace with their parities
- Search algorithm does not scale well to support more than one parity per data block

DiskReduce 2.0
- When generating blocks, the creator picks a RAID stripe pattern and sends consecutively created data blocks to corresponding nodes.
- Other nodes in the pattern picked at random
- Parallel reconstruction and load and capacity balancing
- Method is independent of RAID code
  e.g., apply RAID-DP in 3+2 disks
  \( f_1 = \text{row parity}, f_2 = \text{diagonal parity} \)
- Step0: \( N_0 \) picks a pattern \( \{ N_0, N_1, N_2, N_3, N_4 \} \)
- Step1: \( N_0 \) creates \( D_1 \) and sends to \( N_1, N_2 \)
- Step2: \( N_0 \) creates \( D_2 \) and sends to \( N_1, N_3 \)
- Step3: \( N_0 \) creates \( D_3 \) and sends to \( N_1, N_4 \)
- Step4: \( N_0 \) and \( N_1 \) compress \( D_1, D_2 \) and \( D_3 \) and replace with \( P_1 = f_1(D_1, D_2, D_3), P_2 = f_2(D_1, D_2, D_3) \)

Challenge
Complexity: Yahoo! & Cloudera nervous of cluster RAID
- In traditional RAID, encoding and reconstruction are inline with critical data path
- DiskReduce is based on HDFS approach
- Triplicate data blocks initially
- Use asynchronous background process to encode and to reconstruct
- This approach simplifies the implementation greatly

Deletion/Clean up
- Delete only one block either frees no space or needs parity to be recomputed
- Parity only internal to file (PanFS) not space effective
- Large percentage of small files (e.g. 50% of files in M45 HDFS clusters have 5 blocks or less)
- Parity computed on consecutively created blocks at one node
- Spatial locality: blocks of one file tend to be created together
- Temporal locality: files created at same time tend to be deleted together

DiskReduce 1.0 Experiments
- 16 nodes (PentiumD dual-core 3.00GHz, 4GB memory, 7200 rpm SATA 160GB disk, Gigabit Ethernet)
- DiskReduce 1.0 w/ single parity

Status and Plan
Status (DiskReduce version 1.0): stopped work
- A class project in Advanced Operating Systems (15-712)
- Extended HDFS to support a single parity (both encoding and recovery path)

Plan (DiskReduce version 2.0): designing now
- Retain most of components used in version 1.0
- New block-to-code scheme that can scale well to support more than one parity per data block
- Support deletion space recovery (cleaning)
- Developing analytical model of MTTDL, rate of data loss, expected amount lost per loss event