Large-scale Evaluation of GIGA+ Scalable Directories
(or, How to build directories with trillions of files)

Swapnil Patil and Garth Gibson (Carnegie Mellon University)

Problem: Scalable Directories

Need high performance metadata services
- Most file systems store a directory on a single MDS
- Apps using file systems in new ways, like a simple DB
  - Apps generate millions of small files in one directory
  - Large apps run in parallel on clusters of 100,000s of CPUs

Build scalable directories for shared file-systems
- POSIX-compliant, maintain UNIX file system semantics
- GIGA+ indexing divides a directory into partitions, growing incrementally over multiple servers in parallel
- Eliminates serialization and system-wide synchronization

GIGA+ Optimizations

Power-of-2 optimization (when number of servers = 2^D)
- Below tree depth D, all split operations create partitions on the same server
- Splitting network traffic becomes zero
- Client bitmap errors go to zero (client bitmap only needs to represent first D rows of split tree)

Addition of servers with minimal redistribution
- If the number of servers is doubled, half the partitions of every current server move to the new servers

Experimental Evaluation

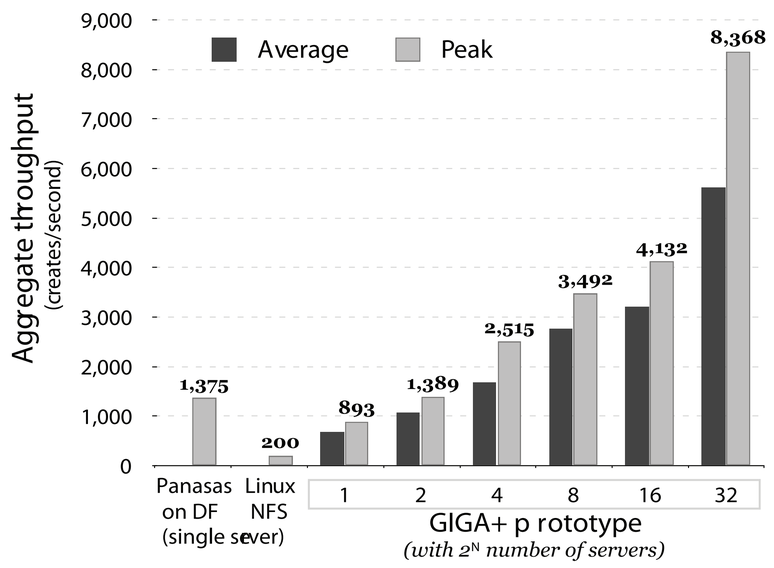
FUSE-based user-level implementation evaluated on a 100-node cluster at Sandia National Labs
(thanks to Ruth Klundt and Lee Ward)
- Two dual-core 2.8GHz AMD Opteron processors with 8GB memory and 7200rpm 80GB disk
- GigE backplane with a HP Procurve 2824 switch

UCAR Metarates benchmark
- MPI application that manages multiple clients creating files in a single directory
- Once all files are created, performs stat() and utime() on each file

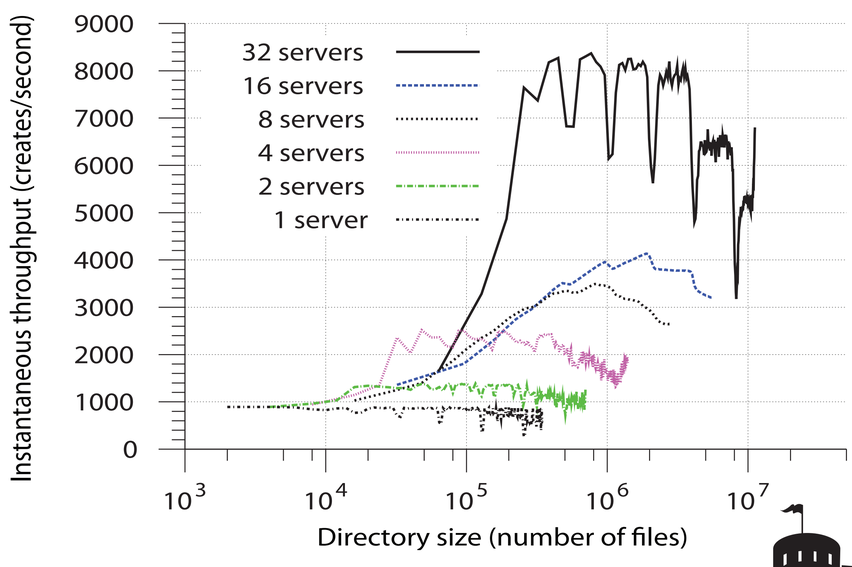
Experiment
- Each client creates 375,000 files one each server, in a common directory striped over many servers

Scale and Performance of GIGA+

Peak performance of more than 8,300 file creates/second
- Scales by 55-60% with the addition of 2X more servers
- Copies updated lazily, on addressing an incorrect server



Understanding GIGA+ scaling
- Initial "step up" as directory grows to use all servers
- "Spikes" due to partition splits, when all servers split at the same time
  - Servers can stagger to avoid splitting simultaneously



Cost of Splitting

Servers perform redundant work by repeatedly rehashing and moving entries to new partitions
- Measure the number of redundant creates as a fraction increase on the number of requested creates
- Sensitive to the partition size and the number of files created on each server

Entries per partition	8 servers, each creating .. 10^3 files	8 servers, each creating .. 10^4 files	16 servers, each creating .. 10^3 files	16 servers, each creating .. 10^4 files	32 servers, each creating .. 10^3 files	32 servers, each creating .. 10^4 files
1K	78.2%	64.4%	80.4%	64.8%	75.8%	64.2%
8K	75.2%	61.8%	76.8%	63.8%	71.1%	62.9%
24K	46.2%	91.1%	51.2%	93.4%	56.2%	92.1%

Carnegie Mellon

Parallel Data Laboratory