

GIGA+: Scalable Directories for Shared File Systems

Swapnil V. Patil, Garth Gibson, Milo Polte, Sam Lang (Argonne National Lab)

Problem: Scalable Directories

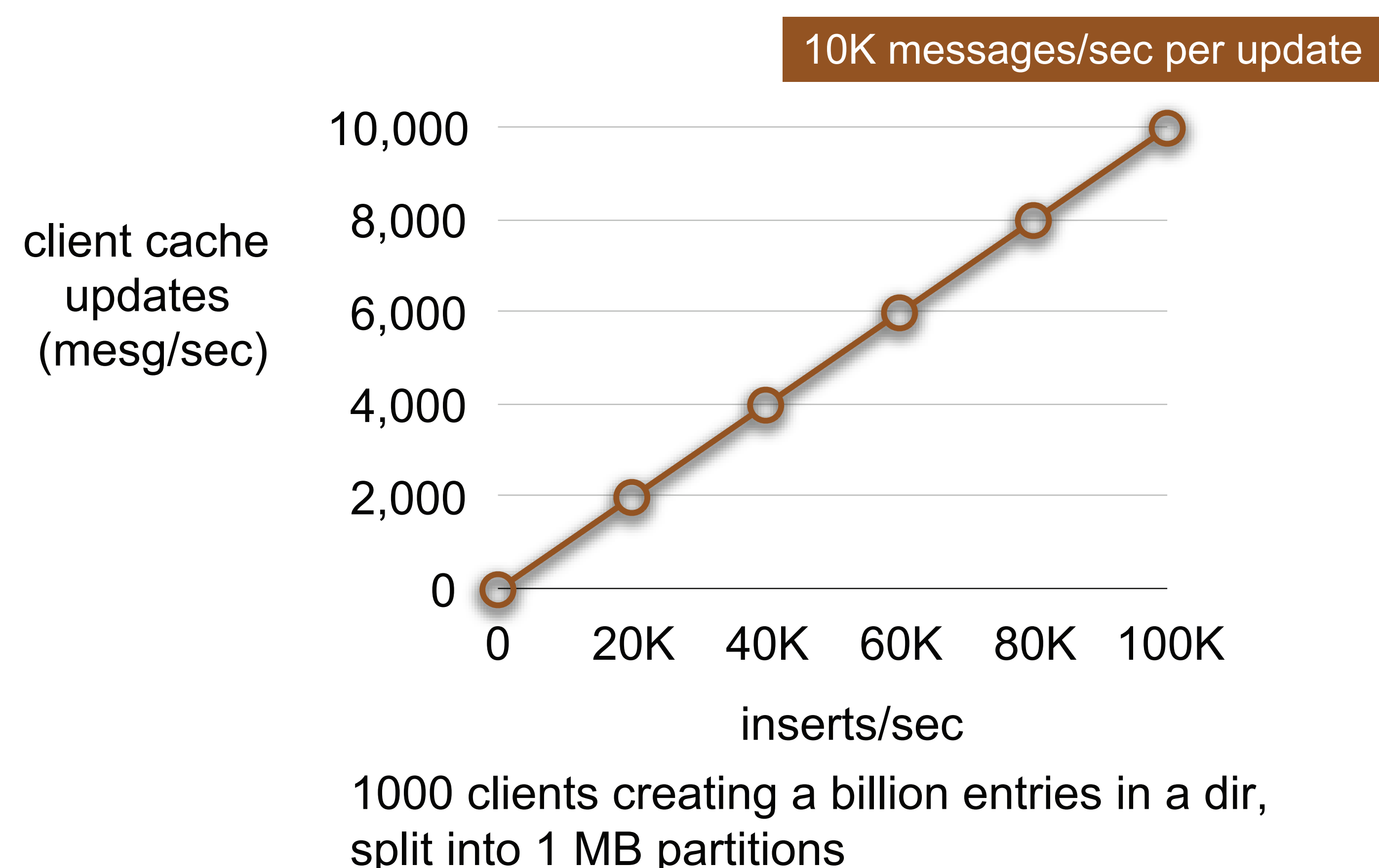
- Push the limits of scalable file-systems
 - Directories with billion to trillion entries, striped on many servers
 - Highly concurrent access to handle 100K+ inserts/sec
- Why? Applications sometimes use FS as a fast, lightweight “database” of small files
 - E.g., phone logs, large checkpoints, scientific experiments (like genomics, high-energy physics)
 - Apps like to maintain UNIX file system semantics/interface
 - Apps running on compute clusters, highly concurrent

Scale and Performance in GIGA+

- Prior work: Using fast, dynamic indexing structures
 - Boxwood [MacCormick04], GPFS [Schmuck02] synchronize servers and update clients after every change
 - Linear Hashing [Litwin81+] uses “coordinator” for growth
- In contrast, GIGA+ indexing uses *less synchronized, more parallel* growth
- Goal: High concurrency through *minimal synchronization*

Synchronization is Expensive

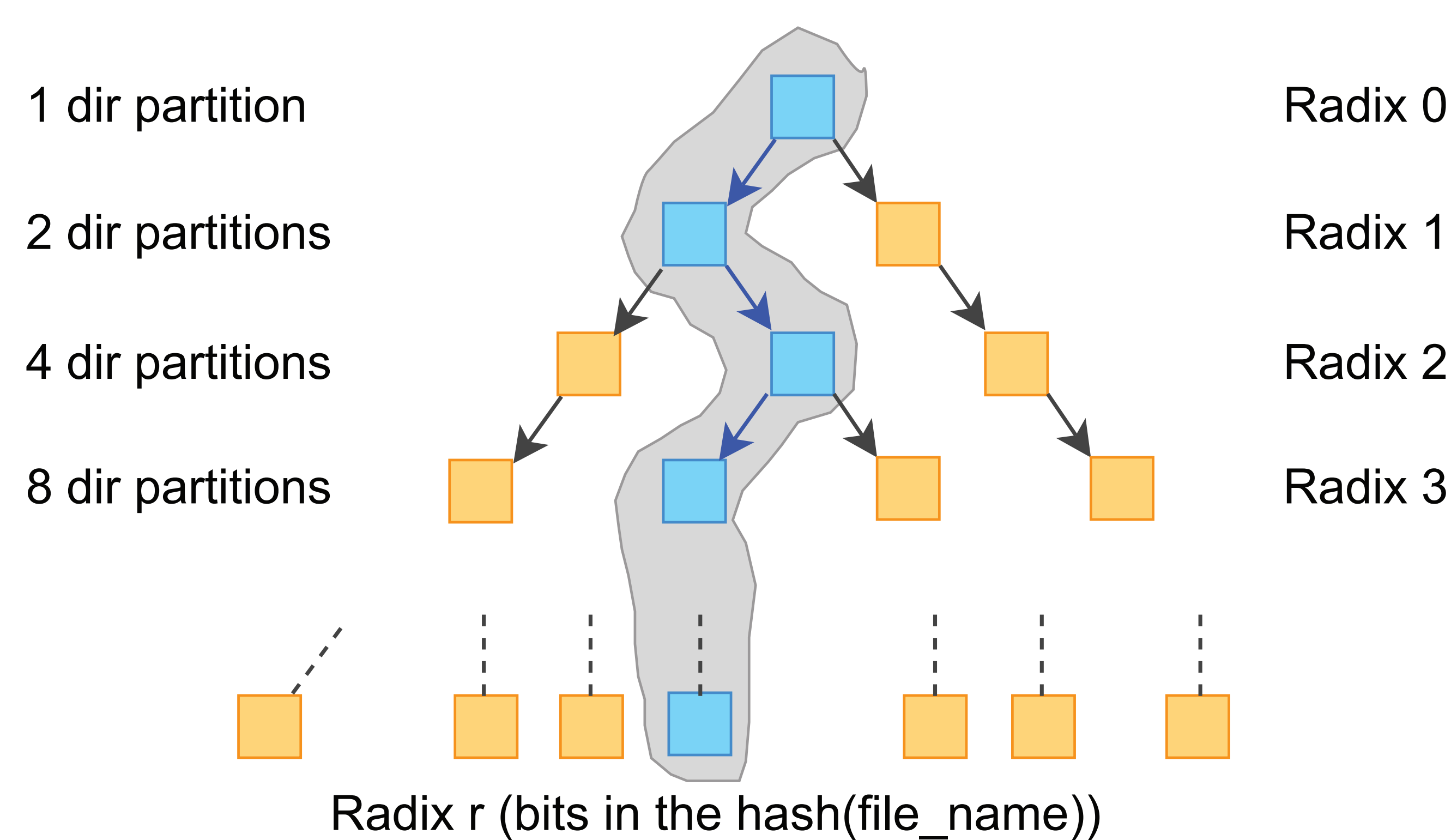
- Metadata mapping: name to <server, partition>
 - Mapping changes frequently, so client cache consistency is expensive



- GIGA+ clients use stale partition-to-server maps, without affecting the correctness of operations

Indexing: Decentralized & Concurrent

- Servers “split” partitions independently, to grow
 - No synchronization with any server (except new partition)
- Clients use stale radix for each path to a leaf node
 - Clients learn about correct radix with “stale” probes



Filename “foo” hash (“foo”) maps to a leaf, or one of its parents, based on total directory size (*worst case* $\log N$ probes down or up)

Summary

- Scale and performance in GIGA+
 - Store billions to trillions of files in a directory and handle 100K+ operations/second
 - High concurrency through minimal synchronization
 - Servers split partitions independently, in parallel
 - Use stale metadata mapping state at the clients
- Prototype in PVFS (Parallel Virtual File System)
 - Open-source cluster FS that stores directories on a single server
 - Based on state-machine
 - Add state-machines for splitting partitions, update client state
 - Split history stored as attributes of a partition (forms tree structure, unlike i-node structure of GPFS)