Distributed File System

Chen Jin

Outline

- Motivation
- System overview
- System implementation
- Performance results

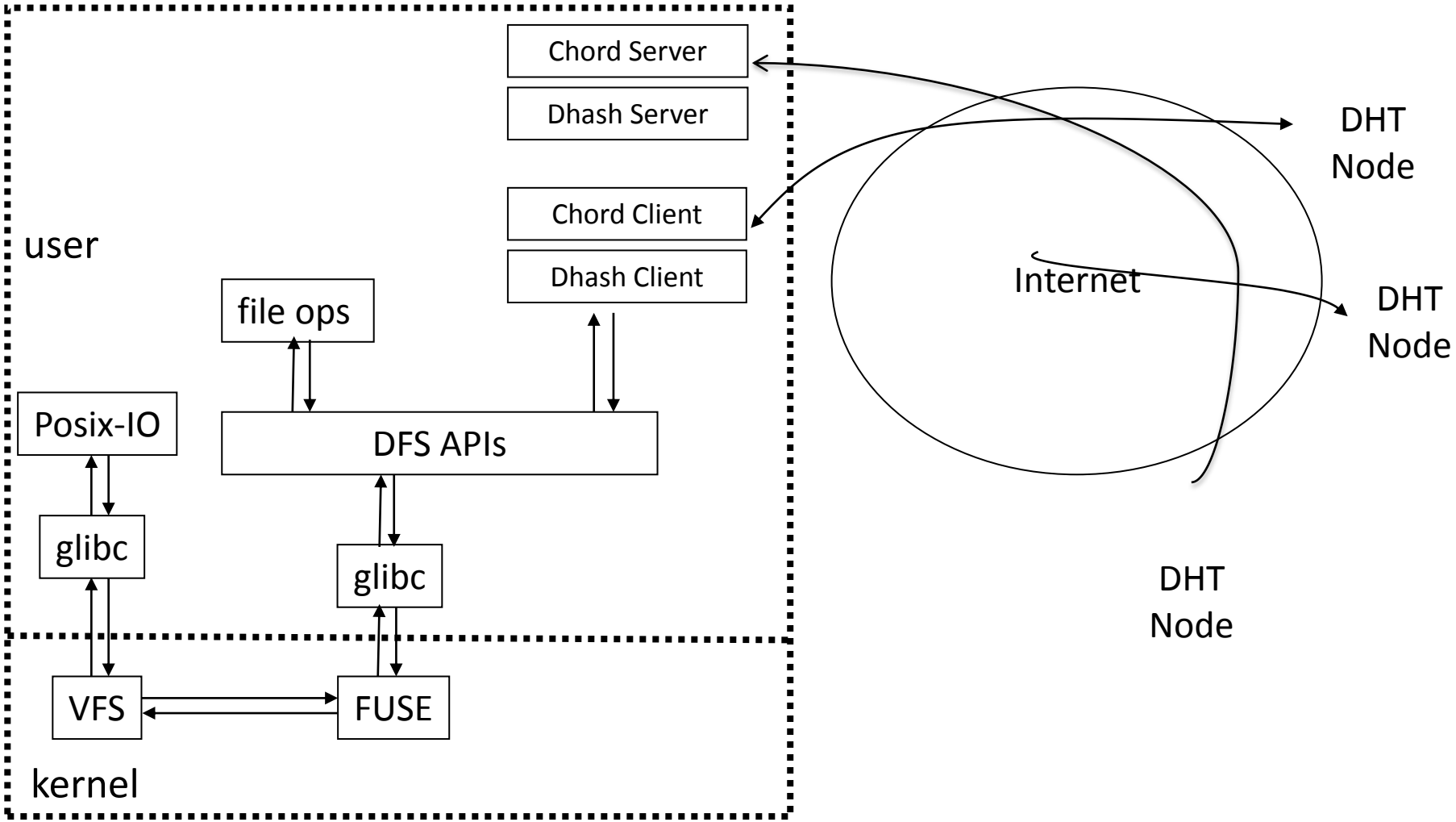
Motivation

- A single MDS is not enough
- P2p system concepts and scalability functions
- Propose a portable, scalable and high performance DFS

System Overview

- DHT-based metadata server cluster
  - Chord, Chimera, CAN, Pastry
- User-space local file system
  - FUSE

System Architecture



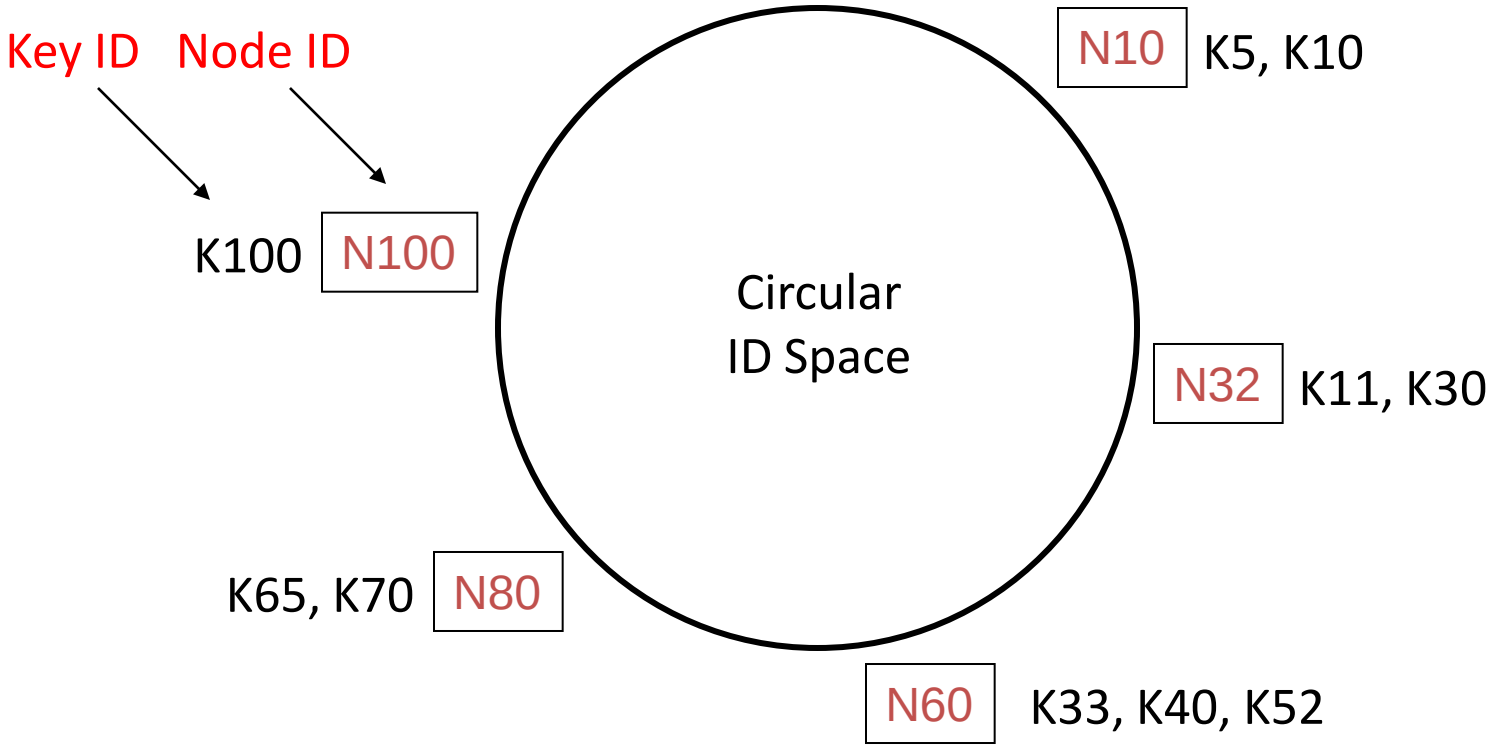
Lookup service

- Centralized
  - Napster (centralized Database, O(N))
- Flooded queries
  - Gnutella (worse case O(N))
- Routed queries
  - Chord (O(logN))

Chord

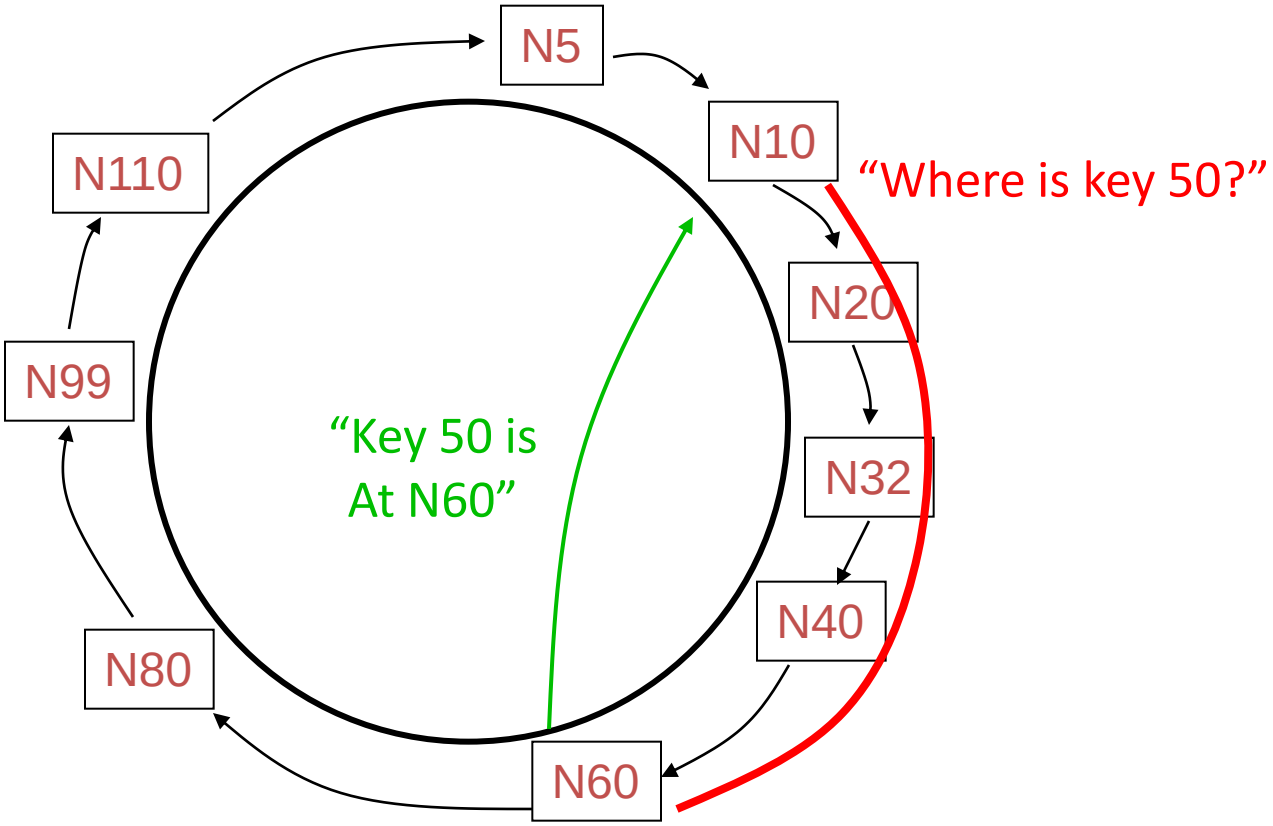
- Consistent hash
  - filename and IP address can be uniformly distributed in the ID space
  - Nodes join and leave the network without disrupting the network
- Keys and Nodes are assigned IDs from the same 160-bit id space
  - Node IDs = SHA-1(ip)
  - Keys = SHA-1(block content)
- How to map block keys to node IDs?

Chord Hashes a Key to its *Successor*



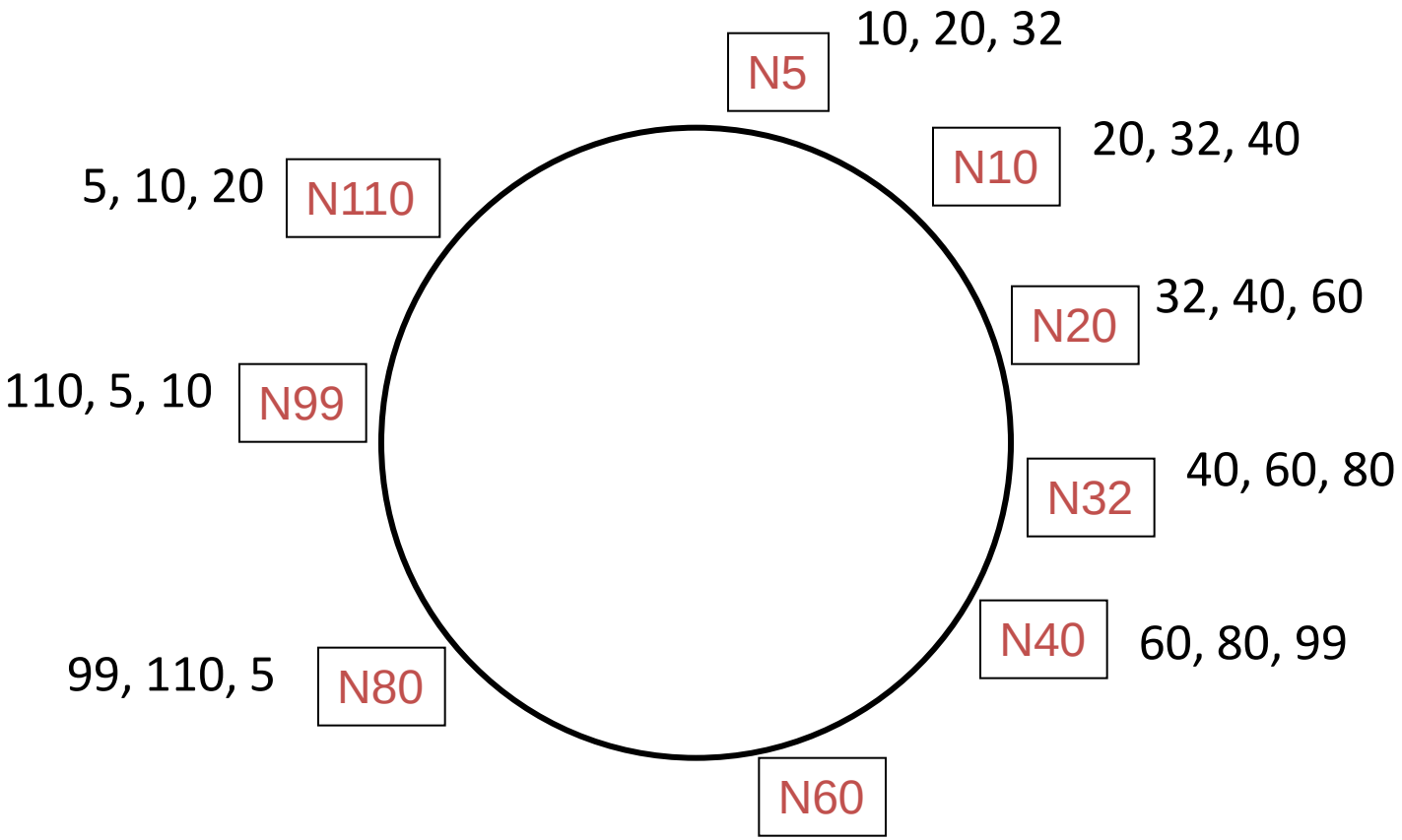
- Successor: node with next highest ID

Basic Lookup



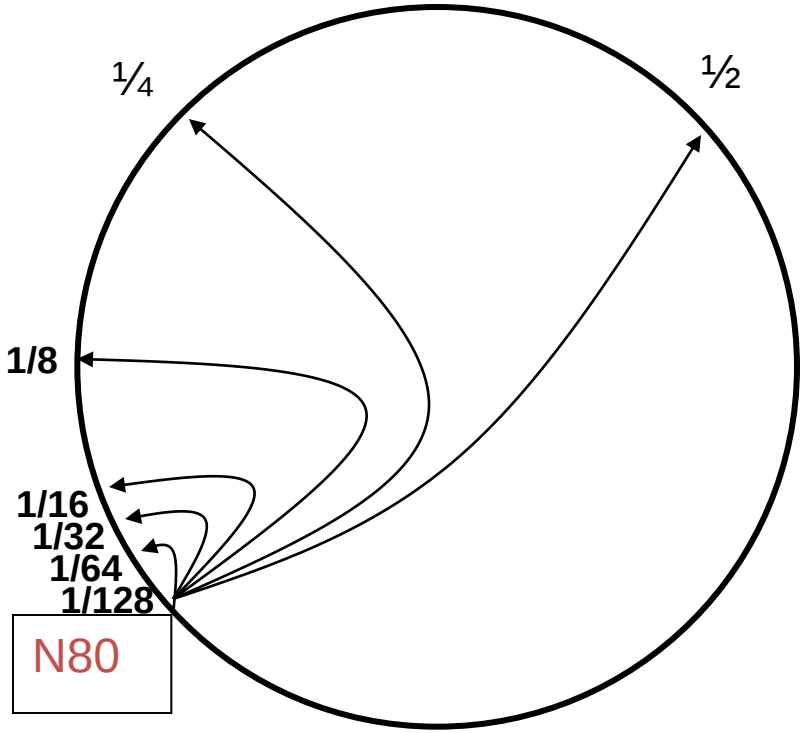
- Lookups find the ID's predecessor
- Correct if successors are correct

Successor Lists Ensure Robust Lookup

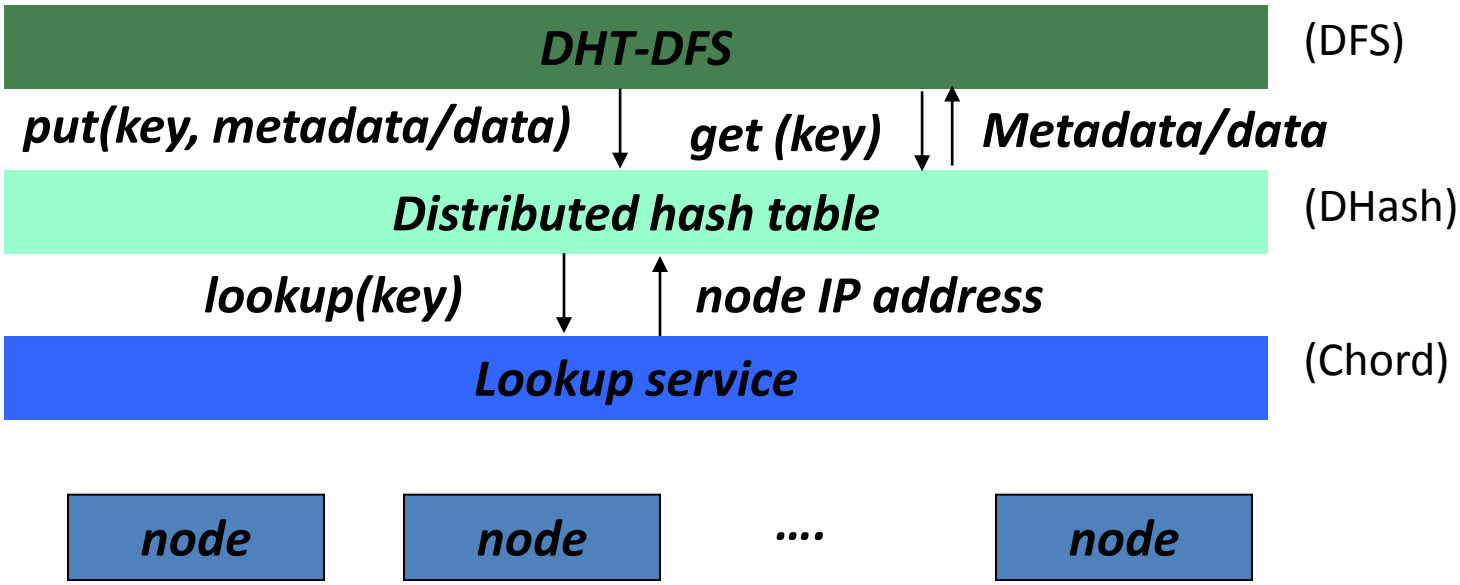


- Each node remembers *r* successors
- Lookup can skip over dead nodes to find blocks

Chord “Finger Table” Accelerates Lookups

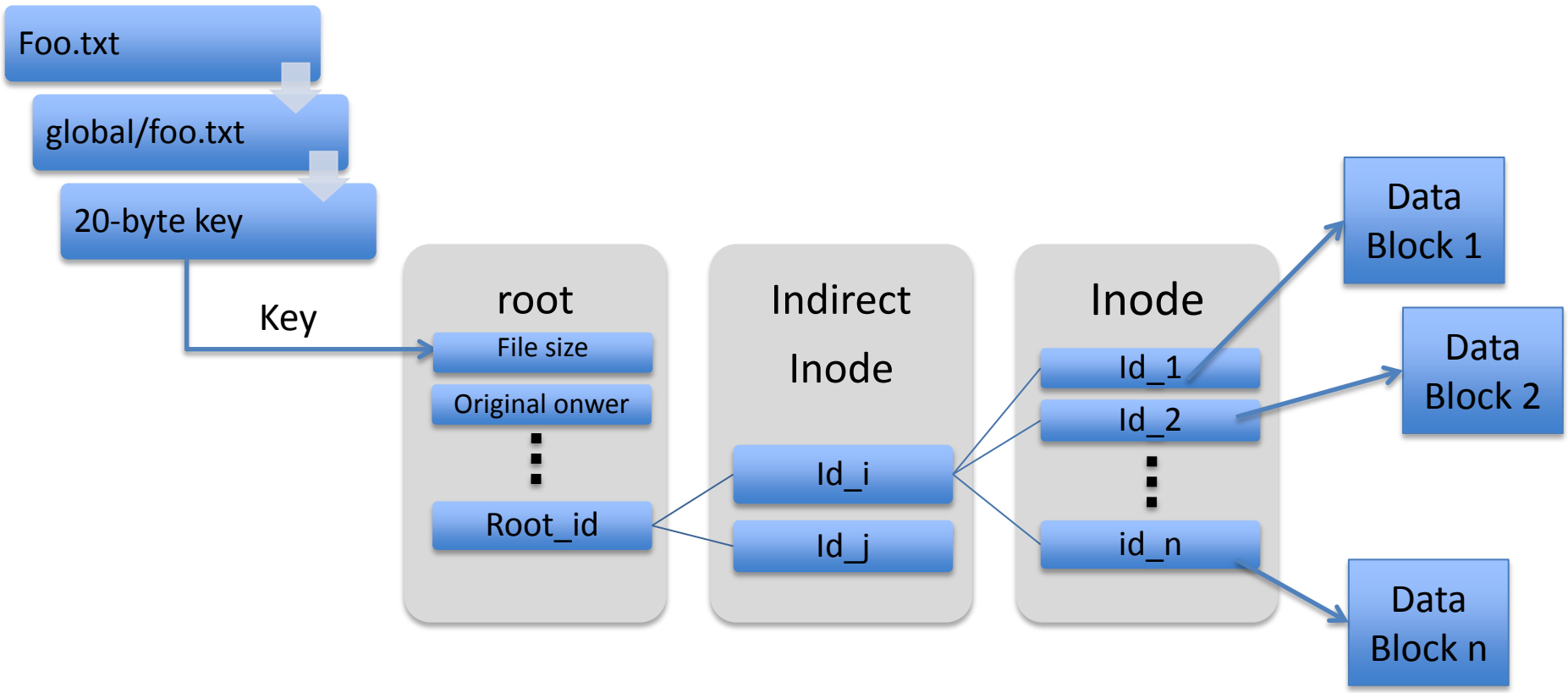


Software Stack



- DHT distributes metadata storage over many nodes

File Data structure



File Size = (BLOCKSIZE/20)^2 * BLOCKSIZE
If BLOCKSIZE = 16k, file size = 10G

DFS APIs

- Dht_init/finalize
- Dht_open/close
- Dht_read/write
- Dependency
  - Sfslite, berkeley database, Chord/dhash

dht_init/finalize

- Dht_init
  - Initialize the DFS client
  - Set configuration parameters
- Dht_finalize
  - Release the resources allocated by dht_init

dht_open/close

- Dht_open
  - Name mapping
    - Global name to Chord Key
  - Fetch data if the file mode is read open
- Dht_close
  - Commit the store if file mode is write open

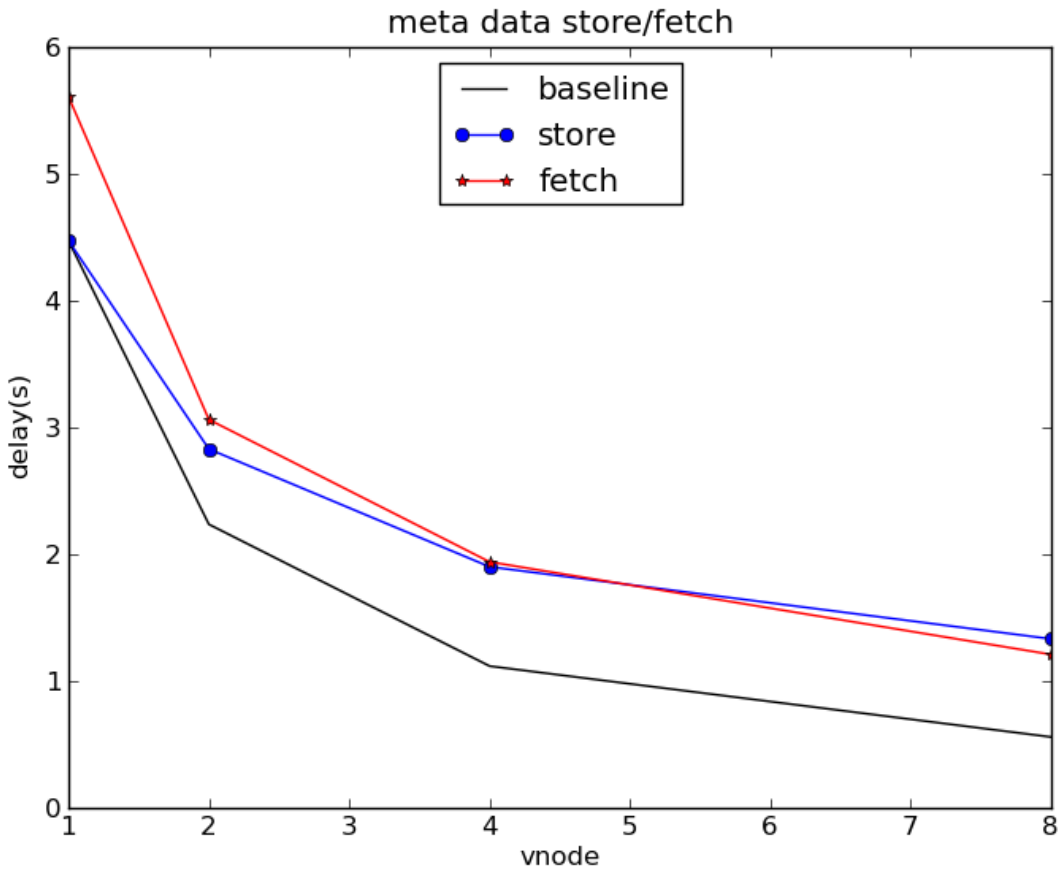
dht_read/write

- Local memory copy

Performance Evaluation

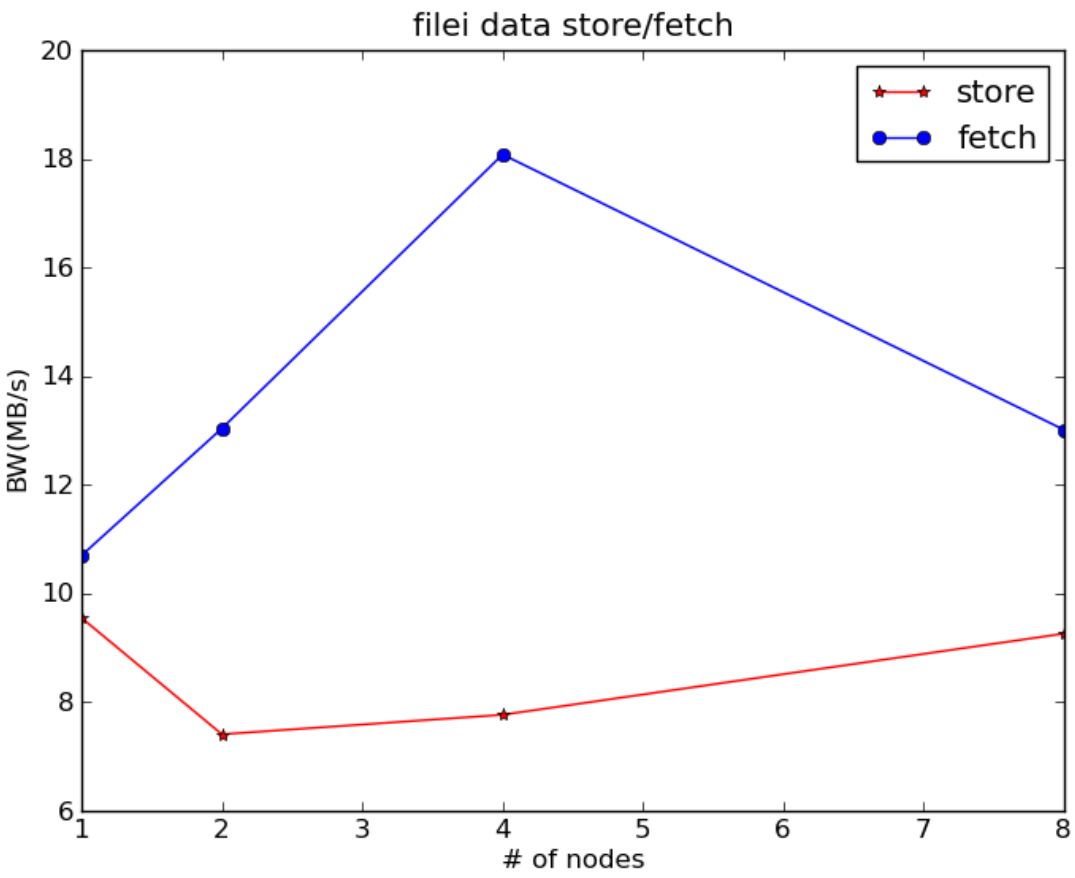
- Experiments setup
  - 1-8 Chord nodes at Falkon
  - 16 virtual nodes total
  - No block replication
  - The network is static, no node join or leave during file operations

Metadata store/fetch



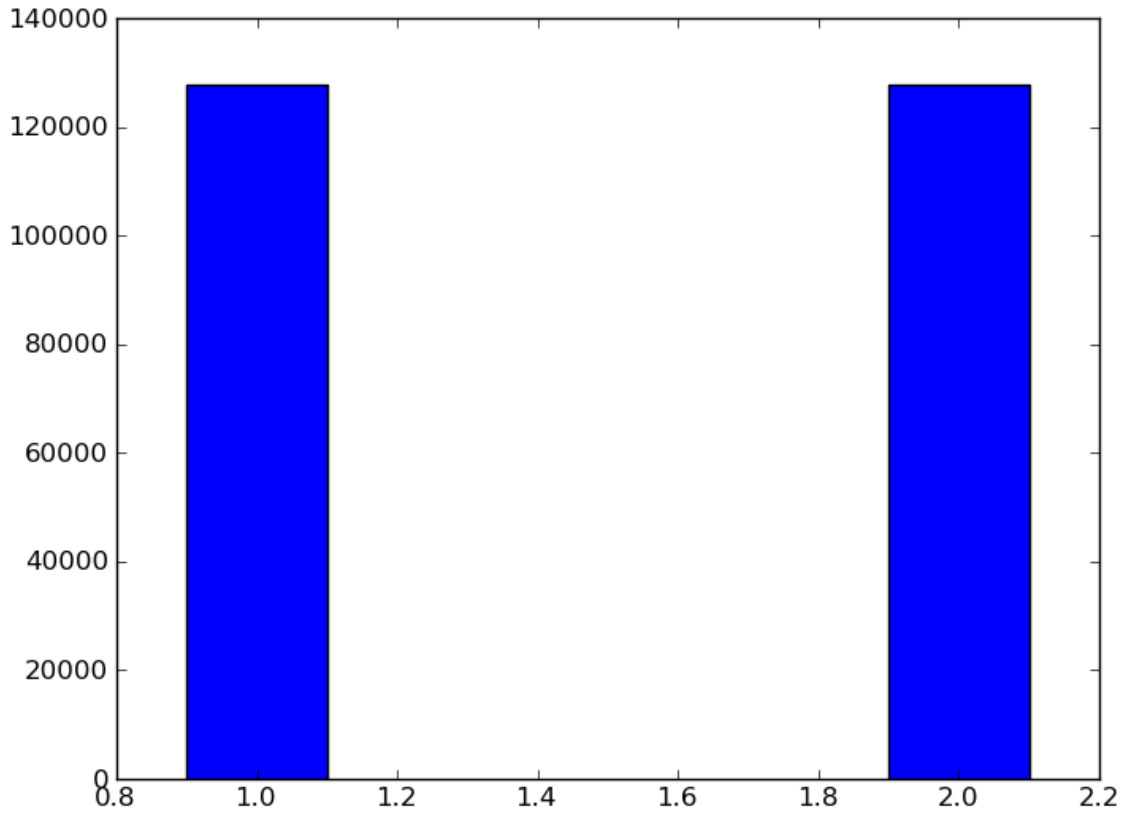
2000 metadata ops per node

File data store/fetch BW



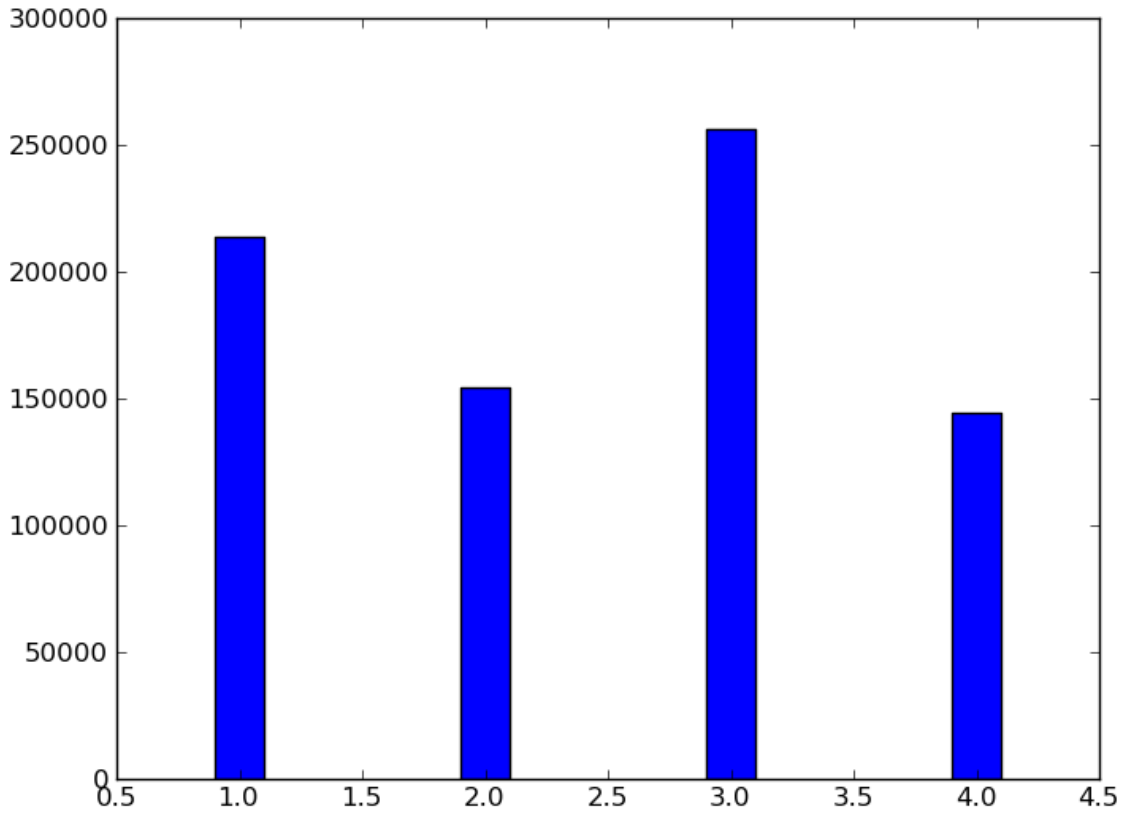
100 files per node, 10MB per file

Load balance



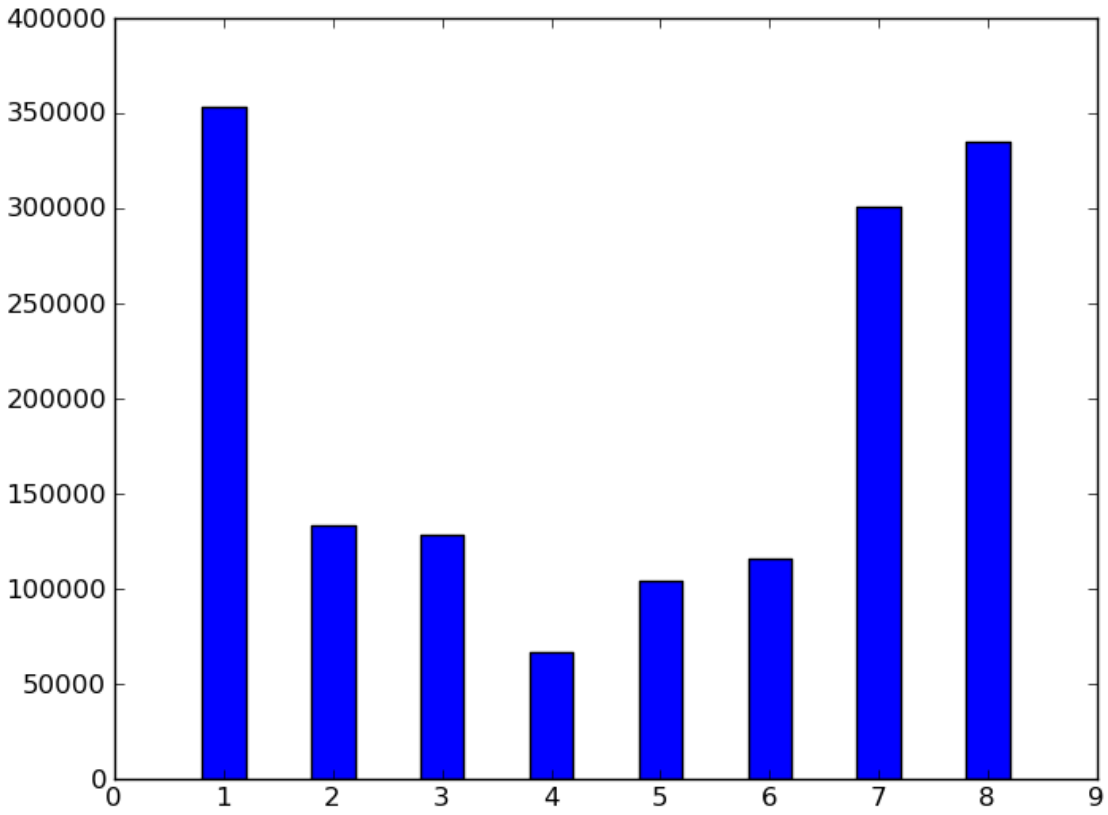
N=4, 100 file writes per node, 10MB per file

Load balance (n=4)



N=4, 100 file writes per node, 10MB per file

Load balance (n=8)



N=8, 100 file writes per node, 10MB per file

Future work

- Integrate the DHT APIs into fuse
- Refine the APIs to support directory, permission
- replicates, data consistency, data caching and prefetching
- Replace the Berkeley DB as backend storage