Dynamic Hashing: Adaptive Metadata Management for Petabyte-scale File Systems

Weijia Li, Wei Xue, Jiwu Shu, Weimin Zheng

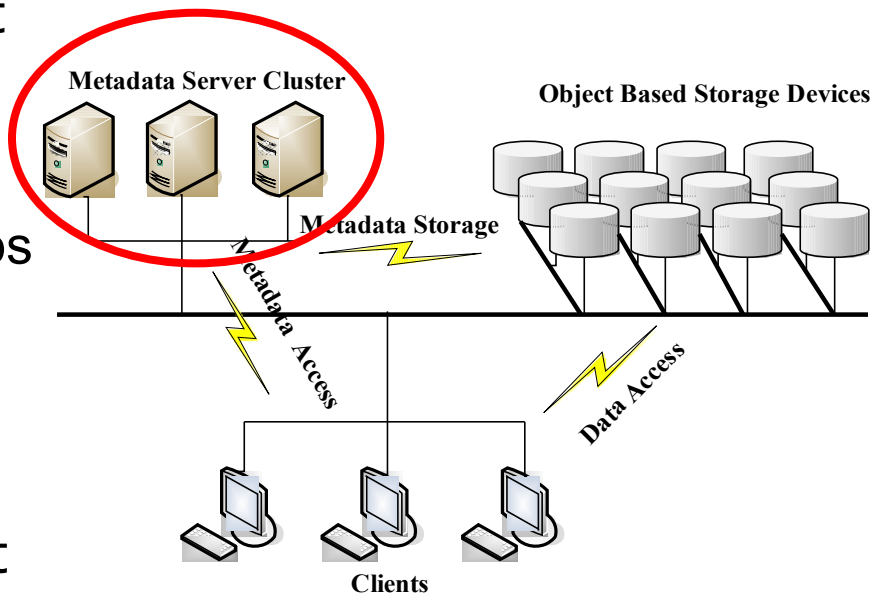
HPC Institute, Tsinghua University

2006-5-16

清華大學
Tsinghua University

Motivation

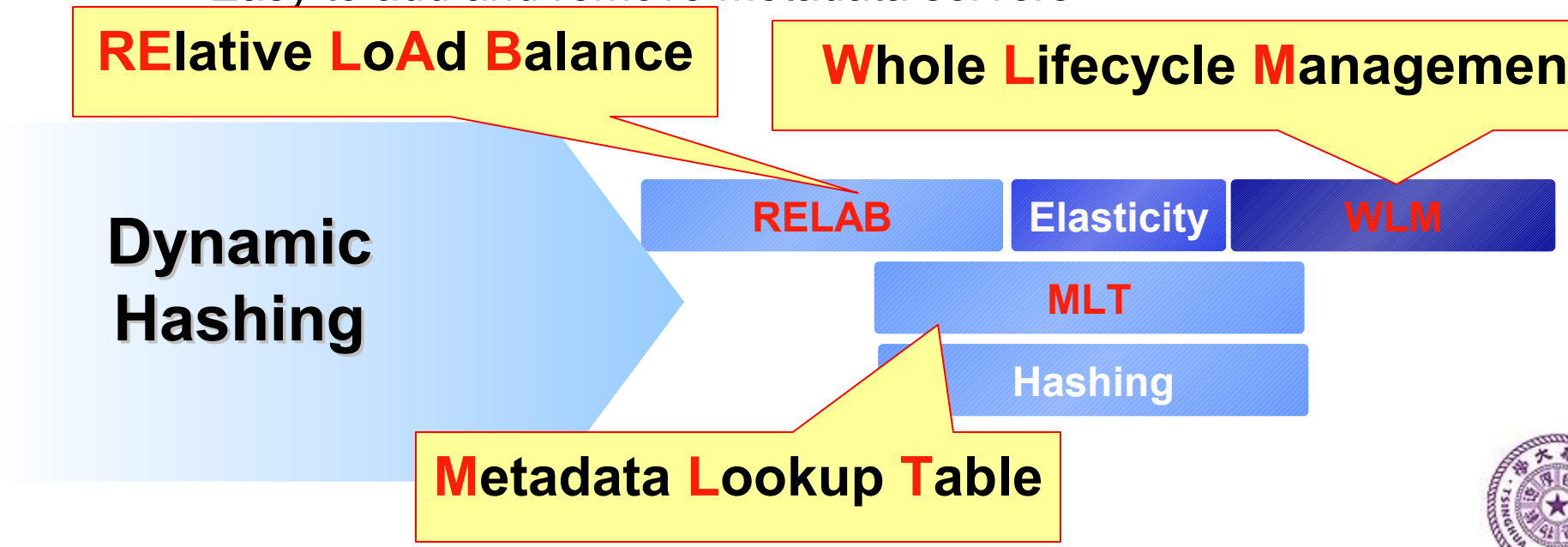
- Large scale File System is more and more popular
  - PB data (billions of files)
  - Many clients (such as 10,000 clients) access at the same time
  - Different Access Modes: different directories, same directory or even the same file
- Effective Metadata Management is Critical
  - Each File Op need access metadata
  - >50% Ops are only metadata Ops
  - Metadata Cluster makes thing more difficult
- goals for Metadata Management
  - Performance
  - Scalability
  - Reliability
  - ...



Storage system architecture

Dynamic Hashing Metadata Management

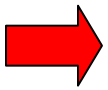
- Dynamic Hashing (DH)
  - provide high-performance and scalable metadata management, especially for metadata cluster
- High-performance
  - Adaptive to workload changing
  - Avoid bottlenecks due to hotspots
- Scalability
  - Easy to add and remove metadata servers



Metadata Lookup Table (MLT)

- Mapping hash value to MDS ID
- The version field indicates if the corresponding entry is out of date
- **Entry** is the minimum unit of metadata redistribution
- All MDSs and clients keep a copy of MLT
  - Broadcast between MDSs when update
  - Lazy update policy for clients

Entry



Range of Hash Values	Metadata Server ID	Version
0-001F	0	1000
0020-003F	1	1000
0040-005F	2	1111
0060-007F	3	1101
……	……	……
……	……	……
FF70-FF8F	0	1000
FF90-FFAF	1	1011
FFB0-FFCF	2	1100
FFD0-FFFF	3	1111

Relative load balance strategy (RELAB)

- Abstract Load vs Relative Load of an MDS
  - Abstract load : Sum of access frequencies of active MLT entries
  - Relative load : Abstract load / power of the MDS
- Goal: to keep the relative load balanced
- Method
  - Busy MDSs move entries of metadata to non-busy MDSs periodically
- Procedure
  - Record access frequency for each active entry in the MLT
  - Calculate the relative load for each MDS
  - For each MDS, broadcast the relative load to all other MDSs
  - Calculate the ideal relative load on each MDS
  - Decide server pairs of busy and non-busy MDS
  - Transfer metadata from busy MDSs to non-busy MDSs
- Elasticity has similar idea to RELAB

Whole lifecycle management (WLM)

- Goal: to manage the whole lifecycles for all hot-spots

Finding hot-spots

- **Keeping files with high access frequency as hotspots**
- **Utilizing cache mechanism**
- **Each MDS finds hot-spots without introducing communications**

Eliminating hot-spots

- **Replicating hot-spots and storing replicas in different MDSs to avoid bottlenecks**
- **Client maintain the list of metadata servers with the same file and select metadata server randomly**

Reclaiming replicas

- **MDS manages all replicas on itself**
- **When the hot-spot is not hot any more, reclaiming the replicas individually**

Comparison with Dynamic Subtree Partitioning

- Pros
  - Easy to add and remove metadata servers
    - Move metadata in parallel
    - Load balancing is still kept after the metadata movement
  - Detailed algorithm to find hot-spots and reclaim replicas
  - Much fewer forwarded requests
    - Client maintain the list of metadata servers and can access the correct metadata server directly
- Cons
  - A little more memory overhead
    - MLT
    - Hotspots info
  - A little more computation overhead

Thank You !

http://storage.cs.tsinghua.edu.cn

清華大學
Tsinghua University