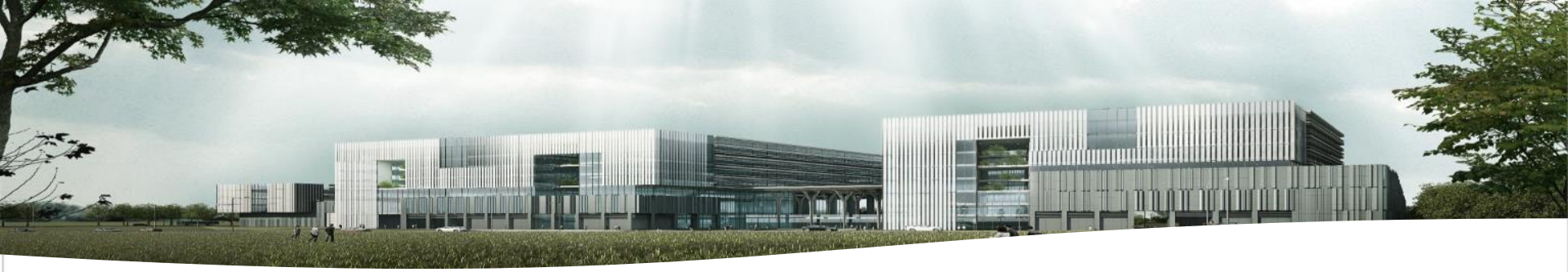


Datacenter Technology (Fall, 2018)

SHAstor: A Scalable HDFS-based Storage Framework for Small-Write Efficiency in Pervasive Computing

Lingfang Zeng (曾令仿)

Wuhan National Laboratory for Optoelectronics (WNLO)

Huazhong University of Science and Technology (HUST)

https://lingfangzeng.github.io/

With Contributions from

- Wei Shi @ **Carleton University**
- Fan Ni, Jiang Song@ **University of Texas, Arlington**
- Yang Wang, Xiaopeng Fan, Chengzhong Xu @ **Shenzhen Institutes of Advanced Technology，Chinese Academy of Sciences**

Outline

1. Introduction
2. Related Work
3. SHAstore Design
4. Performance Evaluation
5. Remarks and On-going Work

1 Introduction

Pervasive Computing (Ubiquitous Computing)

a computing paradigm where the information is processed by linking each object as encountered in environment.

- **Pysical Integration**

  a variety of connected electronic devices that communicate information between each other

- **Spontaneous Interoperation**

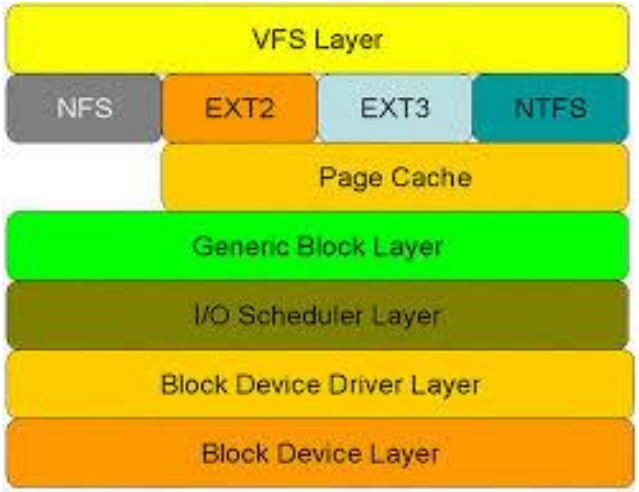
  constant availability and are completely connected to achieve everyware and ambient intelligence.

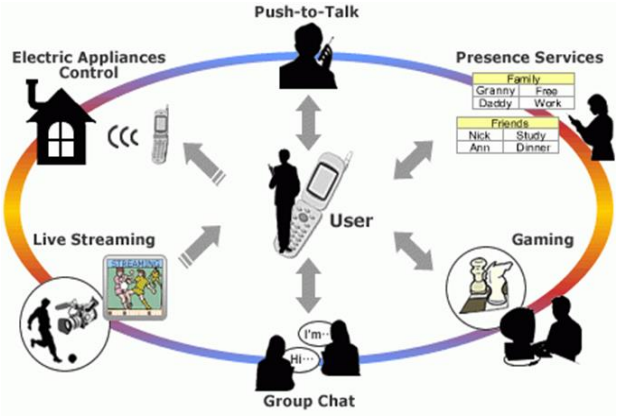
resources of diverse electronic devices are usually heterogeneous and constrained

1 Introduction

Popularity of Small Files

- Logic unit to access file system is block, the larger the block, the more efficient in data transfers.
- Small files refer to the data whose size is much smaller than block.





- Small files from multi-sources are pervasive for connected devices
- Performance critical in modern pervasive environments

1 Introduction

Motivation

Data are multi-sources, and small in sizes.

Compute resources are heterogeneous and fairly limited.

Application with environmental intelligence

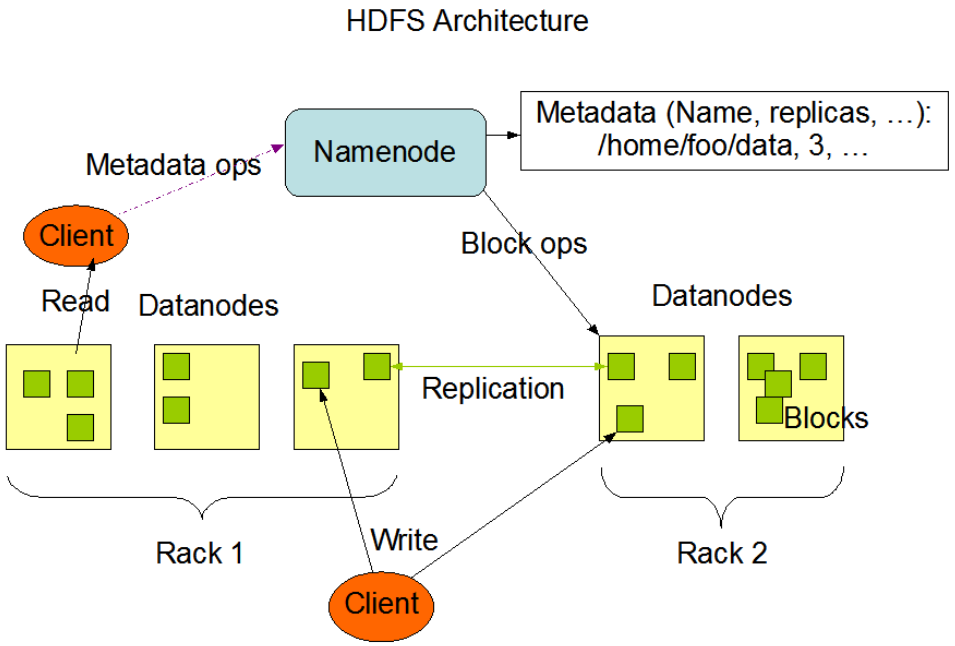
Deploy big-data frameworks to process gathered massive data from devices, then use results to direct devices to react with more intelligence for certain purpose.

Framework should have capability of efficiently dealing with enormous quantify of small files in different formats, and Hadoop is core enabling technology.

1 Introduction

Inefficiency for Small files in Hadoop

1. Additional system info. of small files may be larger, and cost more memory for r/w.
2. Basic file information is stored in system memory with more memory in namenode and datanode.
3. Reading through small files normally causes lots of seeks



1 Introduction

Brief on SHAstor

SHAstor based on Hadoop framework to improve throughput in process of small files for pervasive computing.

Basic Ideas

- Merge incoming small writes into large chunk of data.
- Store it as big target file in framework.
- Adds three extra modules to HDFS.
- A new ancillary namenode storing index table.
- Optimize small-writes, scale out with the number of datanodes

2 Related Work

Merge or Combine Small Files into Fewer Large Files

01

Optimzing HDFS in special and Hadoop in general for small file processing and storage.

02

Main idea is to merge or combine small files into fewer large files and develop strategies and mechanisms to accommodate their metadata and index files.

03

Efforts are roughly made from either inside Hadoop community or from outside.

2 Related Work

Inside Hadoop Community

Hadoop Archive (HAR)

- Reduce memory consumption by packing small files into data blocks.
- Not support appending operation
- Low access performance due to extra index file access.

Sequence File

- Stores data in form of binary key-value pair <file name, file content>, act as container for small files.
- Only supports appending operation, lacks of update and delete particular keys.
- Random read operation due to its unsorted files in key.

2 Related Work

Outside Hadoop Community

Extends existing Hadoop framework.

Develops new programming paradigms.

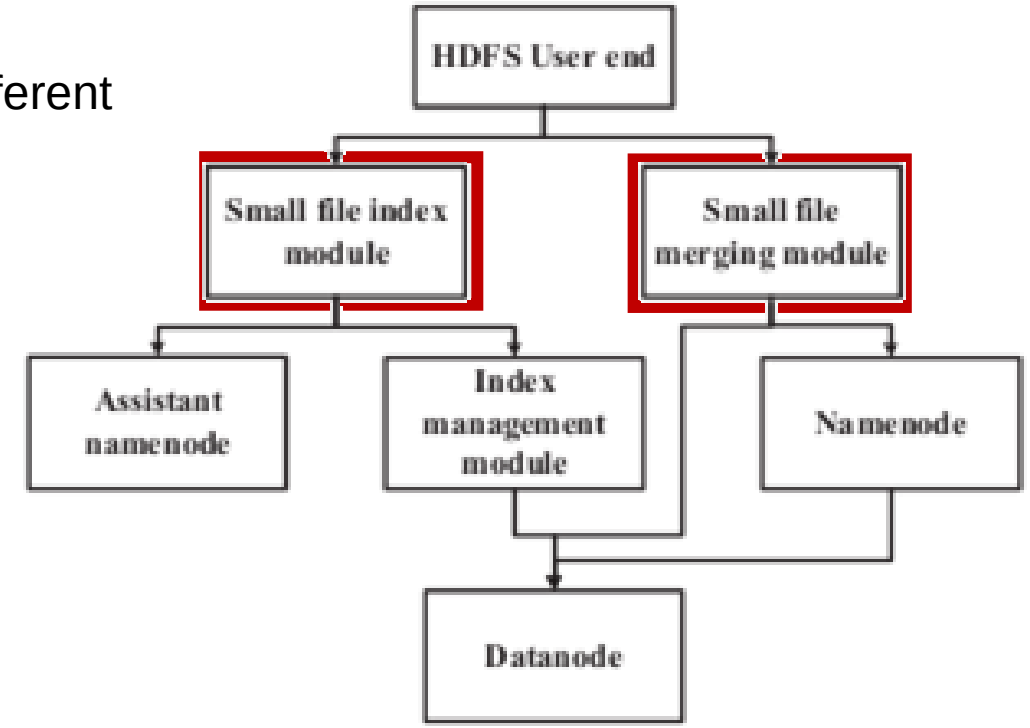
Optimized mapfile-based storage for small files，reducing internal fragmentation in data blocks, and memory consumption.

Generates a map record for each of small files in course of merging into large files, when prefetching and caching mechanisms are applied to enhance access efficiency.

3 SHAstor Design

SHAstor Structure

- Reduce the number of files by merging small size files.
- Heterogeneous data across different devices are unified.
- Index of merged files is stored in datanodes or namenode.



3 SHAstor Design

Small File Merging Module

Small-File Merging Module is further divided into two processes based on whether or not data requests by users are continuous.

A) File merging at client side

- **Merge**
  - Merge small files in client memory
- **Insert**
  - Index info inserted into index module.
- **Save**
  - Saved at datanodes or assistant namenode
- **Data Request**
  - When total number of small files or size of target file reaches threshold, client sends data requests to namenode.

B) File merging at server side

- **Upload**
  - User uploads many separated small files.
- **Data block**
  - Create new data block for merging or look for existing data block for appending.
- **Index info.**
  - Index info. of merged small files stored in the index module.
- **New data block**
  - Once total number of merged files or size of data block reaches threshold, new data block will be created for further small writes.

3 SHAstor Design

Small File Index Module

Two key issues in design of index module for efficient small writes.

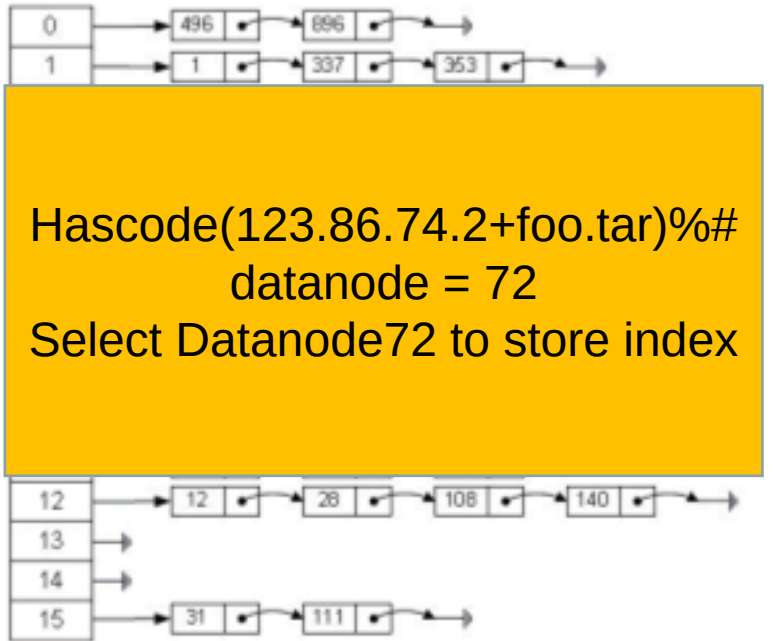
1. Determine index info for merged small files, and data structures to manage it.

2. Depending on workloads, select vantage location to store index table.

Module Container

SHAstor saves data structure of index module as key/value pairs in container, managed by **HashMap.**

```
private static int Hash(int h) {
        h+ = (h << 9);
        h^ (h >>> 14);
        h+ = (h << 4);
        h^ (h >>> 10);
        return h; }
```



3 SHAstor Design

Read, Search, Insertion and Deletion

Four major operations on :

Insertion process

- use name of small file as key value to calculate string type hashcode.
- use hash function and hashcode to calculate hash value as array index.
- store key/value pair of small file into array indexed by hash value.

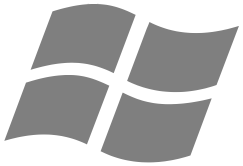
- **Implemented via access to index table.**
- **Retrieve original merged small files from target file.**

4 Performance Evaluation

Experimental Setups

Two physical machines
(100MB bandwidth)

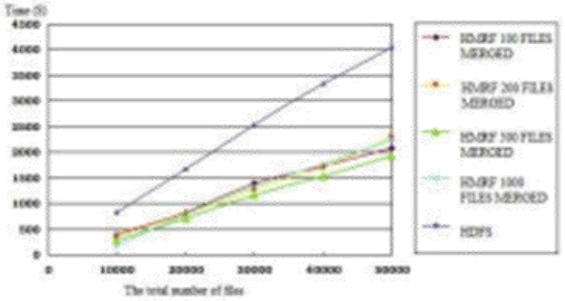
Two virtual machines
(VM)(CPU2.66GHz/mem
1GB/Disk 80GB)



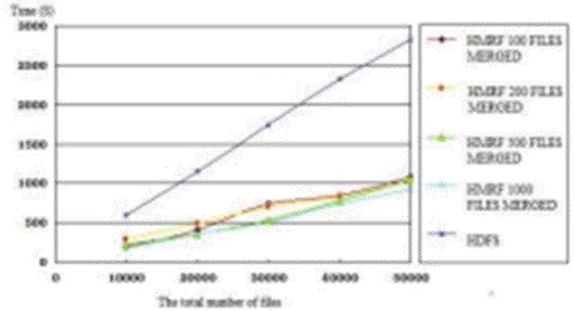
- Ubuntu 9.04
- Hadoop (0.20.2)
- Java Env (Java-6-openjdk)

4 Performance Evaluation

File Writing & Reading

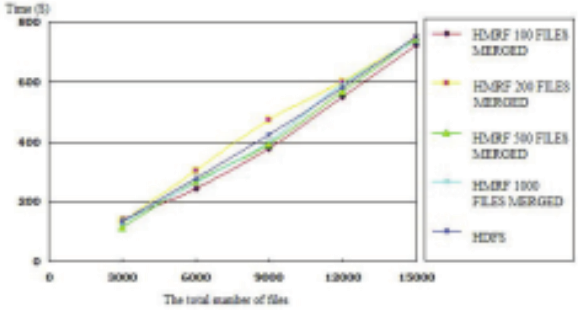


(a) $127KB$ files

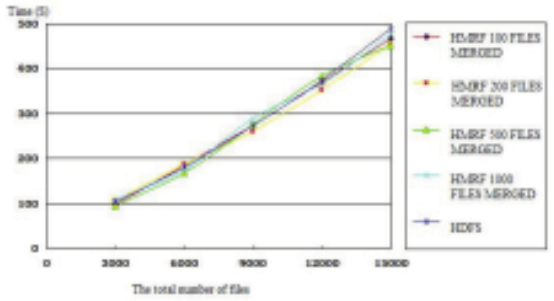


(b) $63KB$ files

	10000	20000	30000	40000	50000
Group1(GB)	1.2	2.4	3.6	4.8	6.0
Group2(GB)	0.6	1.2	1.8	2.4	3.0



(a) $127KB$ files

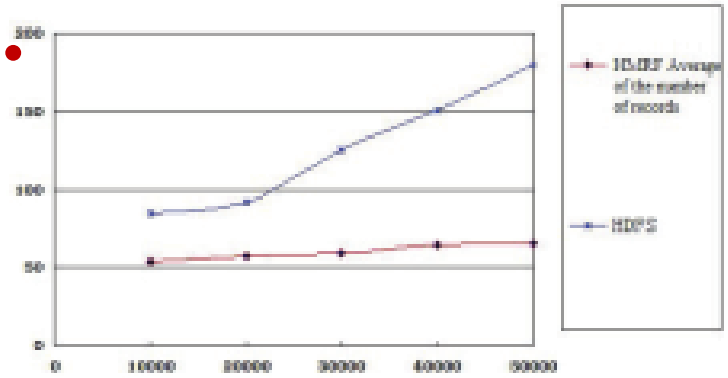


(b) $63KB$ files

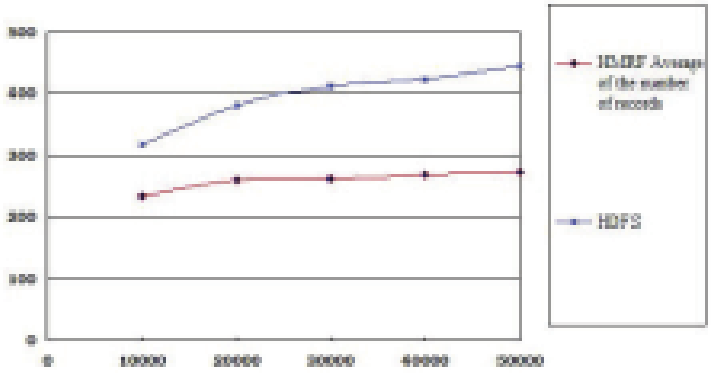
	3000	6000	9000	12000	15000
Group1(GB)	0.4	0.8	1.2	1.6	2.0
Group2(GB)	0.2	0.4	0.6	0.8	1.0

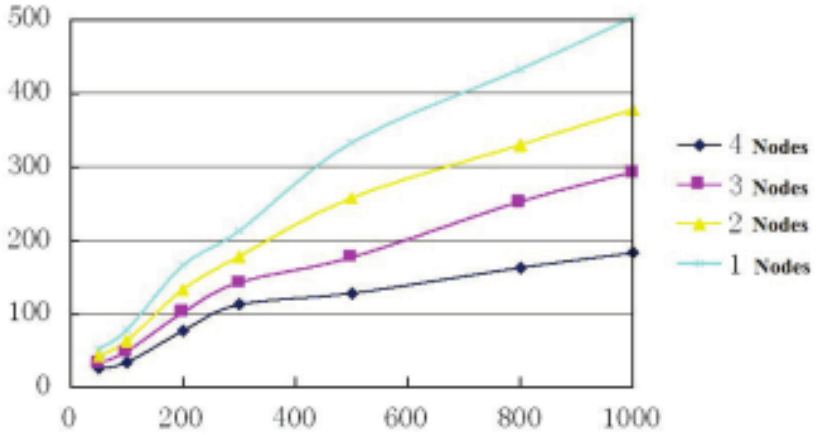
4 Performance Evaluation

Memory Footprints & Scalability



(a) Namenode

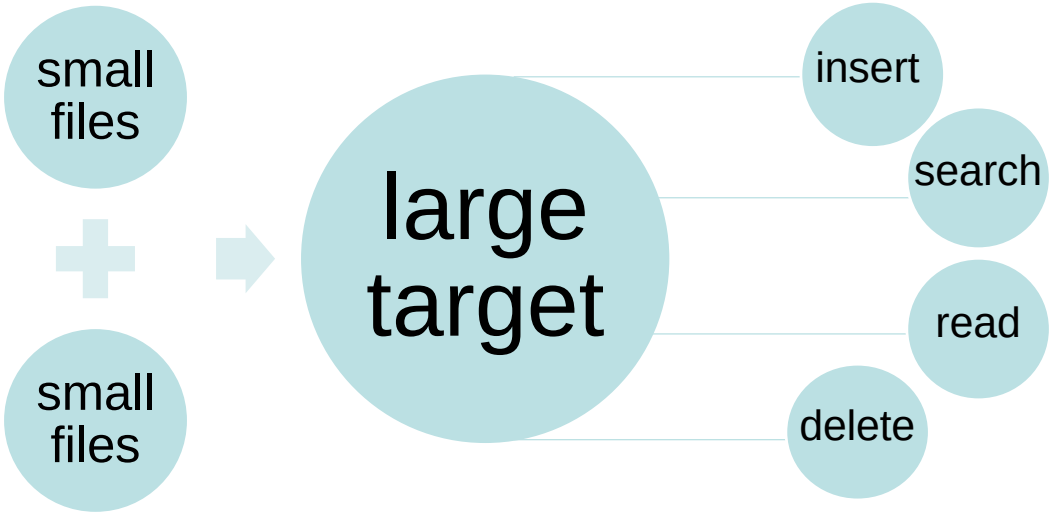




#. of rec.(M)	50	100	200	300	500	800	1000
Size(MB)	46	92	184	276	460	736	920

5 Remarks and On-going Work

SHAstor, for small-write efficiency in pervasive computing.

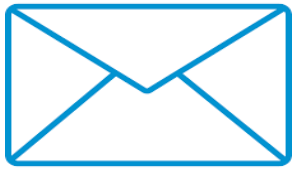


SHAstor is not only efficient but also scalable for small writes, having potentials to realize ambient intelligence.

5 Remarks and On-going Work

Data heterogeneity	Algorithms and mechanisms
Merged Files	Update operation
Writeback for small synchronous writes	Combine SHAstor with Hitchhike I/O scheduler
Real pervasive computing platform	Provide diverse electronic devices with efficient remote control for ambient intelligence

Thanks!



lfzeng@hust.edu.cn

Best Paper Award - UIC 2018

Best Paper Award

Lingfang Zeng, Wei Shi, Fan Ni, Jiang Song, Xiaopeng Fan, Chengzhong Xu, and Yang Wang

SHAstor: A Scalable HDFS-based Storage Framework for Small-Write Efficiency in Pervasive Computing

Of the 15th IEEE International Conference on Ubiquitous Intelligence and Computing (UIC 2018), in the Work-in-Progress session, held in Guangzhou, China, October 7-11, 2018.

Prof. Guanling Chen
University of Massachusetts Lowell, USA
General Chair

Prof. Guojun Wang
Guangzhou University, China
Executive General Chair

GUANGZHOU UNIVERSITY · IEEE · IEEE Computational Intelligence Society · IEEE CIS Smart World TC · IEEE computer society · TCSC IEEE