SUORA: A Scalable and Uniform Data Distribution Algorithm for Heterogeneous Storage Systems

Jiang Zhou, Wei Xie, Jason Nobel, Mark Reyes, and Yong Chen
Department of Computer Science and Nimboxx, Inc.

Center for Cloud and Autonomic Computing

Abstract

- A novel data distribution algorithm SUORA (Scalable and Uniform storage via Optimally-adaptive and Random number Addressing)
- A Pseudo-random algorithm
- Fairly and uniformly distribute data cross a hybrid and tiered storage cluster
- Take full advantage of the characteristics of heterogeneous devices (capacity, throughput, latency and etc.)
- Achieve maximum read throughput while keeping load balance according to data hotness and threshold

Motivation and Goals

- Massive data requires effective methods to manage them for meeting new demands
- Most data centers use heterogeneous storage combining hard disk drives HDD with emerging storage class memory SCM (e.g., solid state drives SSD and phase change memory PCM)
- Data distribution is a key issue in big data storage
- Consider distinct characteristics and merits of different devices
- Load balance and optimally-adaptive placement among devices to improve read throughput
- Data hotness is an important factor considered

Methods and Techniques

Inspired by the SPOCA and ASURA algorithms

- Assign a portion of hash space proportional to server capacity for each device and use hash functions to map data to a point in a hash space.

Basic idea of SUORA

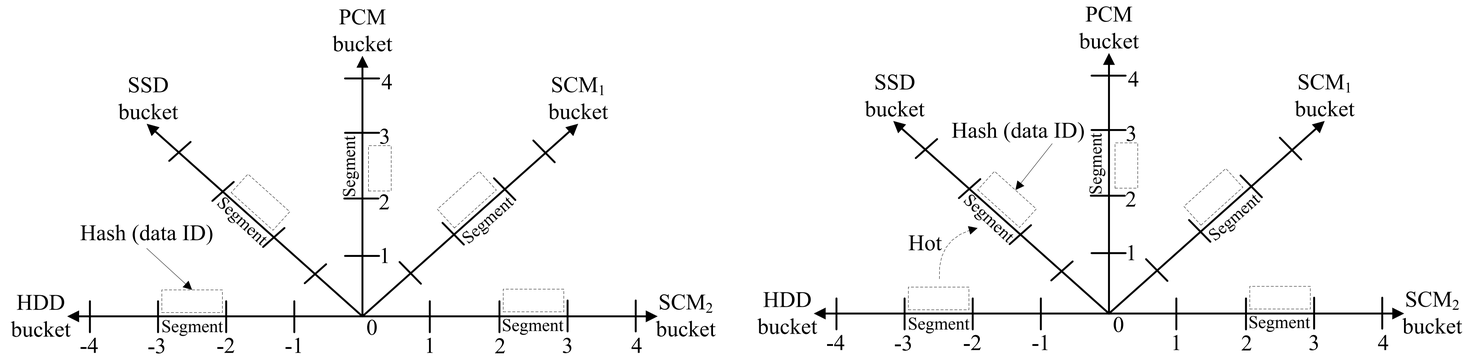
- Divide device sets into buckets with *throughput*
- Assign devices to segments with *capacity* in each bucket
- A sequence number for each data until mapping

Current Status

- Proposing three innovative algorithm models
- Completing the design details of the algorithm
- Conducting extensive analysis and evaluation
- Starting to collect I/O trace based on standard workload and benchmark
- Starting to explore further evaluation in more complex scenarios
- Plan to design an effective data I/O pattern detection and data caching strategy in an heterogeneous storage system
- Plan to implement the algorithm on practical storage system, such as Sheepdog

Project Results or Plans

- Algorithm model



(a) Data placement (b) Data movement for hotness and threshold

- Data distribution

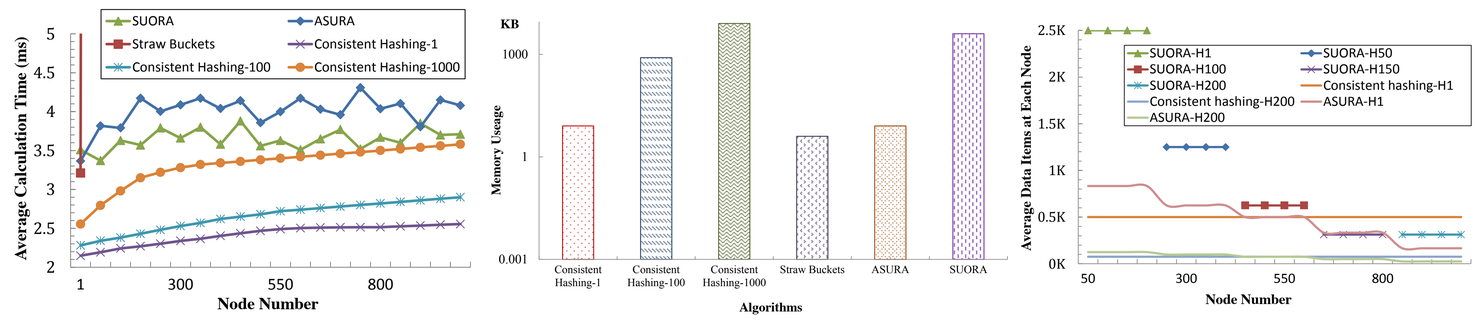
Step 1: data nodes are divided into two buckets, HDD bucket and SSD bucket

Step 2: each node in the bucket is assigned to segments in a number line

Step 3: all data are placed in the HDD bucket at initial placement and a random number sequence (RNS) is generated until the data fits one segment in the number line

Step 4: data distribution is automatically adjusted between HDDs and SSDs according to the hotness (read counts) and threshold

- Evaluation

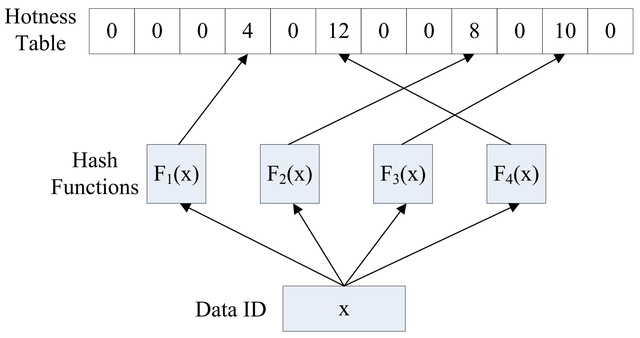


(a) Calculation time (b) Memory consumption (c) Hot data distribution

Discussion

Hotness table

- Multiple hash functions for reducing hash collision
- Each bucket has an associated threshold
- Data identified as hot if counters exceed threshold
- Automatically locate data with hotness table and the random number sequence
- Reset the hotness counter periodically



Comparison: analysis evaluation of different algorithms*

Algorithm	Computation Time		Memory usage	Uniform Distribution	
	Device initialization	Data distribution		Homogeneous	Heterogeneous
Consistent Hashing	*Poor* $O((n+v)\times log(n+v))$	*Fair* $O(log(n+v))$	*Poor* $O(n+v)$	*Poor* Hash both values	*Poor* By near capacity
Straw Buckets in CRUSH	*Excellent* Negligible	*Poor* $O(n)$	*Good* $O(n)$	*Good* Hash device values	*Poor* By near capacity
ASURA	*Excellent* Negligible	*Good* $O(1)$	*Good* $O(n)$	*Good* Random number sequence	*Fair* By capacity
SUORA	*Excellent* Negligible	*Good* $O(1)$	*Good* $O(n)$	*Good* Random number sequence	*Excellent* By capacity, throughput, cost, etc.

* *n* means node number and *v* means virtual node number (for consistent hashing)

Acknowledgements

We are grateful to the Nimboxx and the Cloud and Autonomic Computing site at Texas Tech University for the valuable support for this project. Thank High Performance Computing Center at Texas Tech University for providing the computing resources and support for this project.

NSF CAC Semi-Annual Meeting, October 1-2, 2015
The Center for Cloud and Autonomic Computing is supported by the National Science Foundation under Grant No. 1362134.

TEXAS TECH UNIVERSITY