Performance Virtualization for Distributed Storage Systems via Proportional Sharing

Yin Wang yinw@eecs.umich.edu, Arif Merchant arif@hpl.hp.com

Distributed storage systems

- A distributed storage system consists of storage bricks (e.g. NAS servers).
- Each brick runs the same set of software modules.
- A client may access its data through any brick—coordinator/gateway.
- A client’s logical volume may spread across several bricks. Different volumes may have different layouts.

The simple idea

Try to apply existing centralized scheduling algorithms.

Problem: no central scheduler

Solution: use centralized algorithm at the back-end of each brick; coordinators broadcast the info of processed requests to all back-end schedulers.

Question: how to reduce communication overhead?

Problem: resource competition

How to allocate resource fairly? What if clients have been assigned different data layouts?

Goal: proportional sharing—total service of a client is proportional to its weight.

Our contribution

Previous work: centralized scheduler and same data layout.

Our setting: clients may have different data layouts and access data via different coordinators.

The improved idea: Distributed Start-time Fair Queuing (DSFQ)

No need to broadcast. All we need is the total cost of the virtual requests, which is calculated by the coordinator and piggybacked onto the normal request.

Theoretical proofs and other experimental results are available.

Results

- The algorithm tries to make the total service of each client proportional to its weight, even if clients have different data layouts.
- Proportional total service may block the service on an individual brick. We can guarantee a minimum service on each brick for each client.
- Theoretical proofs and other experimental results are available.

Experiment 1: increasing stream

Two equally weighted clients

Heavy workloads that both saturate Brick A

Workload (thread) increases over time on Brick B

Total IOPs of clients f and g are equal, whatever the IOPs of client g is on Brick B.

Experiment 2: on/off stream

Three equally weighted clients

Heavy workloads that both saturate Brick A

Light on/off workload that is on every other 10 seconds.

Total IOPs of client f and g are equal, even the service capacity of Brick B varies because of a new on/off client.