Scaling and Understanding Large Data I/O Performance

Erez Zadok
Dept. of Computer Science
Stony Brook University
http://www.cs.stonybrook.edu/~ezk

Motivation
- RAM is fast but never big enough
- Eventually you hit I/O paths
- I/O (disks/networks) are slowest part
- 100–10,000x slower than RAM
- Bottleneck
- Must scale storage stack
  - From application, to OS, to hardware
  - Many intermediate software layers today

Large Data Analysis Systems

Client Requests
Tablet Server 0...N
Network Layer
Logical Layer
TSSL (Compression)
GTSSL
Layers
Examples

Multi-Tier Storage
- Future: heavy workloads
  - Massive data
  - Random read/writes
- SAMT: Sorted Array Merge Trees
  - Minimize random writes
  - Can merge efficiently
- Transactional KV store
  - Hot items percolate up
    - Colder ones trickle down
  - Scales 10–1000x better than alternatives

Insertion Performance

New Membership Filters
- Approximate Membership Queries (AMQ)
  - Approx. is member not in data set
  - Definite: is member in data set (False Positive rate)
- Bloom Filters (BF) and variants
  - Cannot scale beyond RAM, poor random access
- We use Quotient Filters (QF) instead of BF
  - QFs have better access locality
  - Can efficiently merge two QFs into a larger QF
- Cascade Filter (CF), built on QF
- Optimized for Flash/SSD
- Our performance
  - We do 670,000 inserts/sec (40x of other variants)
  - We do 530 lookups/sec (1/3x of other variants)
Insertion Throughput

Proposals
- Cascade Filter optimizations [3 months]
- Tradeoffs: lookup vs. insertion speeds
- Enhancements and Optimizations
- Tablet Server Storage Layer [6 months]
  - Optimize & Integrate with Cascade Filter
- TSSL File System [6 months]
  - POSIX access API
  - Works with unmodified applications
- Others? (storage, I/O path optimizations)

Scaling and Understanding Large Data I/O Performance

Q&A

Erez Zadok
Dept. of Computer Science
Stony Brook University
http://www.cs.stonybrook.edu/~ezk