Key-Value-Links: A New Data Model for Developing Efficient RDMA-Based In-Memory Stores

Hai Duc NGUYEN

Faculty of Computer Science and Engineering
Bach Khoa University – VNU-HCM

Ho Chi Minh City, 17 May 2017
Motivation Applications

Low latency → must respond to request as soon as possible.
- Searching, auction apps, etc.

Big requests
- Facebook > 100 requests/HTML page
- Amazon: 100 – 200 request/page.

High data dependency:
- Sub-requests must perform are executed sequentially

Data to be processed are huge
Big Data Problem

- HCMC in 2015:
  - > 8 million people
  - > 7 million motorbikes
- Assume a GPS transmitter generates 1 signal for every 20 seconds → 20 thousand signals generated per second.
- Traditional in-memory stores require ~100 internal requests per signal
  → 35 million requests per second !!!
Motivation Applications: Examples

Facebook

Smart Traffic

Amazon
Why not key-value?

• The key value model is not a one-size-fit-all solution
  • Simple
  • Complicated dataset?
  • Large request?
• Potential performance issues
  • All operations are performed at the server side
  • Hash table is a hotspot
Outline

• Design Principles
• The KVL Data Model
• Performance Evaluation
• Conclusion
Design Principles: Technology trends

• Decreasing in DRAM price
  • 2015: 64GB per data node
• Remote Direct Memory Access (RDMA) is the key idea in recent high performance communication technologies
  • CPU bypass
  • Low-latency
  • Low cost
Design Principles: Key Ideas

• Keep data in RAM to speedup request execution.
  • Speedup request execution
  • Eliminate expensive disk operations

• Putting tracking information inside data content.
  • Avoid visiting hash table too many times
  • Enable direct fetching via RDMA
The KVL Data Model: An Example

Tracking information is represented as link, include \{physical address, data size\}
The KVL Data Model: Data Layout
The KVL Data Model

1. Get(C)
2. Thread-handoff
3. Lookup
4. Data transmit
5. Read(C → A)

Hash Table
Pre-allocated Memory Region
Memory
... ...
Listener
Worker thread pool
Data Node
Application
The KVL Data Model: KELI

The whole dataset is loaded into main memory in advance. Data are read from main memory

→ High performance

Data are write to disk. Changes are gradually got updated in main memory

→ Ensure durability
Performance Evaluation: Primitive Operation Latency
Performance Evaluation: Latency

Map matching

Web page construction

Graph Exploration
Conclusion

• KVL is a promising data model for developing high performance in-memory stores.
  • Avoid many sources of overhead.
  • Good resource utilization
  • Well fit real-life applications and technology trends

• Problems to be solved
  • Ensure data consistence
  • Handling write intensive workloads
The End

Thanks for listening