NoVoHT: a Lightweight Dynamic Persistent NoSQL Key/Value Store

Kevin Brandstatter[1], Tonglin Li[1], Xiaobing Zhou[1], Ioan Raicu[1,2]

[1]Department of Computer Science, Illinois Institute of Technology, Chicago IL

[2]Mathematics and Computer Science Division, Argonne National Laboratory, Argonne IL

Abstract

With the increased scale of systems in use and the need to quickly store and retrieve information, key/value stores are becoming an important element in the design of large-scale storage systems. Key/value stores are well known for their simplistic interfaces, persistent nature, and excellent operational efficiency – they are also known as NoSQL databases. This paper presents the design and implementation of a non-volatile hash table (NoVoHT). NoVoHT was designed from the ground up to be lightweight, fast, and dependency-free. Our goal was to create a fast persistent key/value store that could be easily integrated and operated in lightweight Linux OS typically found on today's supercomputers. We also aimed to develop a system that performed as close as possible to an in-memory hash map, but with the added benefit of being persistent. We also extended the traditional key/value store interface (e.g. insert, lookup, remove) to include a novel operation (e.g. append) that has allowed NoVoHT to efficiently support lock-free concurrent write operations. NoVoHT is also dynamic, supporting live migration across node boundaries. We have run comparisons at significant scales against some of the more commonly used key value stores and have shown that NoVoHT can perform similarly or better than other systems such as Kyoto Cabinet, and BerkeleyDB. We observed up to 165K+ operations per second, up to 32X better performance than competing systems. We have evaluated NoVoHT with both traditional mechanical disks (HDD) as well as with solid state disks (SSD), and have deployed NoVoHT as the persistent back-end of a distributed hash table (ZHT) on an IBM BlueGene/P supercomputer at up to 32K-cores.

Experiment Design

Testbed

- AMD Opteron 6168
- 48 Cores
- 256 GB Ram
- SUSE Linux OS

Compared projects

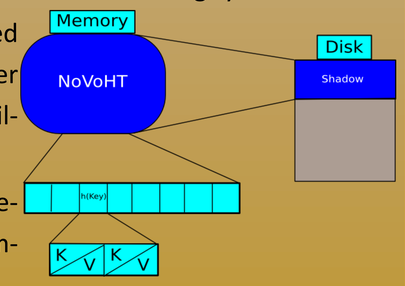
- NoVoHT
- KyotoCabinet
- BerkeleyDB
- LevelDB

Experiment

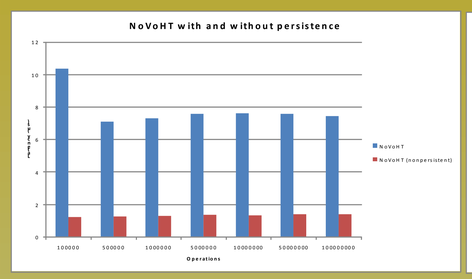
For each Key/Value store, we wrote a program that generates some number of key value pairs of a pre specified length. In our tests we used a key length of 48 bytes and values of 24 bytes. Then our program timed the cost of inserting, looking up, and deleting all of the pairs. From that, we calculated the operation latency.

Design

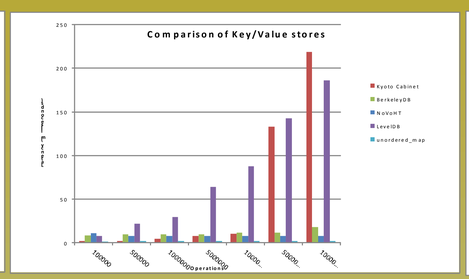
- Lock-Free: Support for unconventional operation such as append, allowing the efficiently support of lock-free concurrent modification operations
- Dynamic: Support live migration across physical nodes
- Lightweight: Micro-benchmarks delivering over 165K+ operations/second on single-node deployment up to 32X faster than existing systems
- Persistent: Combines memory mapped and disk mapped approaches to deliver both fast data access and high data resilience at the same time
- Real-System: Adopted by ZHT and deployed on IBM BlueGene/P supercomputer at 32K-core scales



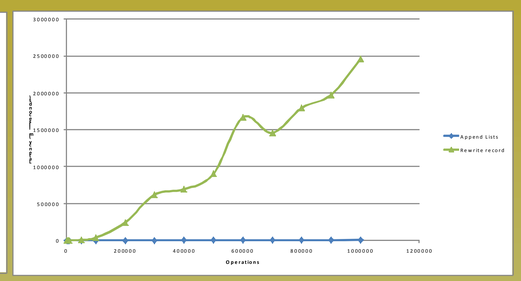
Performance Evaluation



We compared NoVoHT without persistence to the standard in memory hash table and found that NoVoHT performs as well if not better than the standard implementation.



NoVoHT maintains lower latency even for larger scales, whereas many of the alternatives begin to slow down significantly.



By using lists we show that we can achieve a constant time append operation that is independent of the number of operations.

Conclusion

NoVoHT is a persistent key/value storage system designed for fast access to data, while ensuring consistency and reliability. It is free from many extra dependencies, which allows it to be easily deployable on specialized systems where those dependencies may be difficult to satisfy.

In our tests we were able to maintain constant throughput at larger scales and sustain a throughput of 100K operations per second, which is more than 10X over its competition.

We have successfully deployed NoVoHT as the persistent storage mechanism in ZHT, a distributed hash table at more than 32K core scales.

We believe that NoVoHT is a prime solution to fast persistent data access and

Related Work and Acknowledgements

Tonglin Li, Xiaobing Zhou, Kevin Brandstatter, Dong fang Zhao, Ke Wang, Anupam Rajendran, Zhao Zhang, and Ioan Raicu. ZHT: A Light-weight Reliable Persistent Dynamic Scalable Zero-hop Distributed Hash Table. IEEE IPDPS, Boston, MA, 2013.

KyotoCabinet HashDB http://fallabs.com/kyotocabinet/api/classkyotocabinet_1_1HashDB.html, 2013

Margo Seltzer, Keith Bostic. "Berkeley DB", http://www.aosabook.org/en/bdb.html, 2013

LevelDB, https://code.google.com/p/leveldb/, 2013

FusionFS: Fusion distributed File System, http://datasys.cs.iit.edu/projects/FusionFS/index.html, 2013

MATRIX http://datasys.cs.iit.edu/projects/MATRIX/index.html, 2013

Future Work

Crash Recovery: Currently NoVoHT supports recovery from an unexpected shutdown, however this feature is still in testing and more work needs to be done to ensure reliability.

Limited Memory Usage: NoVoHT is currently entirely in memory, this means the memory usage grows with the amount of data. We would like to implement a mechanism to conserve memory and allow for a memory limit to allow for larger data sets.