Ib-KV: Using Infiniband Effectively for Key-Value Services

Anuj Kalia (student author), David G. Andersen, Michael Kaminsky
{akalia, dga}@cs.cmu.edu, michael.e.kaminsky@intel.com
Carnegie Mellon University, Intel Labs

This poster will describe a new Infiniband-based key-value cache that uses the full processing power of Infiniband hardware. Unlike previous Infiniband-based key-value systems, Ib-KV focuses its design on reducing network round trips while using efficient Infiniband communication primitives. For small key-value items, Ib-KV delivers over 2X higher throughput than recent RDMA read-based key-value systems at significantly lower latency.

Current designs are based on RDMA reads (READs) and deliver only a fraction of the peak READ throughput. For `GET`s, Pilaf [3] uses 2.6 small READs on average. FaRM's [1] key-value store (FaRM-KV) uses 1 large READ, followed by 1 small READ if the values are of variable length. Our work is motivated by the contrast between the fundamental time requirements for cross-node traffic vs. CPU-to-memory lookups. Going between nodes takes roughly $1-3\mu$s, compared to $60-120ns$ for a memory lookup, suggesting that a multiple-RTT design like [1, 3] should be fundamentally slower than a single-RTT design.

Towards designing a single-RTT key-value system, we make two unconventional design decisions and arrive at a surprising request-reply mechanism. First, we use RDMA writes (WRITEs) instead of READs, despite the allure of bypassing the remote CPU entirely. A WRITE is cheaper than a READ because it is essentially a half round-trip operation. In Ib-KV, clients WRITE their requests to the server's memory; the server computes and sends the response back. Second, we embrace Infiniband's messaging verbs (SEND, RECV) despite conventional wisdom which says that messaging is slower than RDMA (READ, WRITE) [3, 1]. To avoid the scalability problems associated with connected transports, the Ib-KV server uses Infiniband's SEND messages over a datagram transport for responses. Incoming requests still use WRITEs—the server does not need to pre-post RECVs and therefore avoids the main overhead of messaging.

We optimize our Infiniband verbs by using unreliable transports (Unreliable Connection—UC and Unreliable Datagram—UD), payload inlining, and selective signalling. Using unreliable transports removes Infiniband ACK/NAK traffic, and the other two optimizations reduce one DMA operation each at the verb initiator. For 32 byte messages, our WRITE throughput (35 Mops) is over 75% higher than our READ throughput (19.6 Mops). Without any datastore operations, our hybrid, RDMA+messaging based request-reply communication achieves 19.8 Mops. *From a network-centric perspective, this is fortunate: it means that designs that use only one cross-datacenter RTT can potentially outperform multiple-RTT designs both in throughput and in latency.*

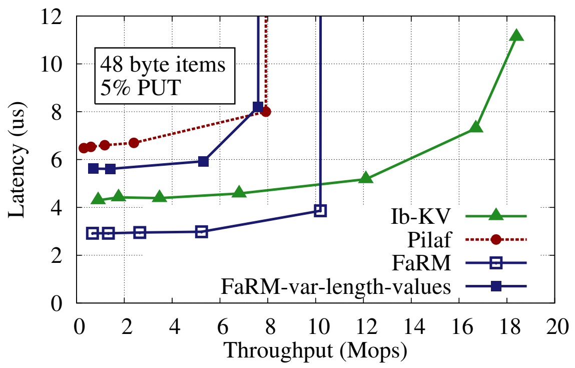

Figure 1: End-to-end performance comparison

Design overview: Ib-KV borrows its data structures (lossy index, circular log) and its partitioning scheme from MICA [2]. Clients use WRITEs over UC to place their requests on a memory region on the server machine. On detecting a new request for its partition (done via polling), a server process executes it using local data structures and sends the response via a SEND over UD.

Performance evaluation: To focus on understanding the effects of our network-related decisions, we compare our (working) Ib-KV implementation against simplified implementations of Pilaf and FaRM-KV that use the same communication methods as the originals, but *omit* the actual data-store operations (Ib-KV *does* store its data).

Figure 1 compares the single-server performance of the three systems for a read-intensive workload with 16 byte keys and 32 byte values. Ib-KV achieves up to 18.5 Mops—comparable to the throughput of native RDMA reads. Ib-KV's latency is over 40% lower than Pilaf and FaRM (in its variable length mode) at their peak throughput.

[1] A. Dragojevic, D. Narayanan, O. Hodson, and M. Castro. FaRM: Fast Remote Memory. In *Proc. 11th USENIX NSDI*, Berkeley, CA, 2014. USENIX. To Appear.

[2] H. Lim, D. Han, D. G. Andersen, and M. Kaminsky. MICA: A Holistic Approach to Near-Line-Rate In-Memory Key-Value Caching on General-Purpose Hardware. In *11th USENIX NSDI.*, Berkeley, CA, 2014. USENIX. To Appear.

[3] C. Mitchell, Y. Geng, and J. Li. Using One-Sided RDMA Reads to Build a Fast, CPU-Efficient Key-Value Store. In *Proc. USENIX ATC*, Berkeley, CA, 2013. USENIX.