On the Performance Impact of NUMA on One-sided RDMA Interactions

Jacob Nelson
CSE, Lehigh University
Bethlehem, PA, USA
jjn217@lehigh.edu

Roberto Palmieri
CSE, Lehigh University
Bethlehem, PA, USA
palmieri@lehigh.edu

***Abstract*—One of the consequences of ultra-fast networks like InfiniBand is that known implications of Non-uniform Memory Access (NUMA) locality now constitute a higher percentage of execution time for distributed systems employing Remote Direct Memory Access (RDMA). Our findings quantify the role NUMA plays in RDMA operation performance and uncovers unexpected behavior.**

***Keywords*-RDMA, NUMA, Performance, Locality**

I. Introduction

Remote Direct Memory Access (RDMA) and Non-uniform Memory Access (NUMA) are common in high-performance computing testbeds and infrastructures. NUMA is widely recognized in industry and academia [4], [2] and RDMA's capabilities are being rapidly explored in a multitude of settings [1]. As the real innovation underlying RDMA, *one-sided operations* have the unique characteristic of being blind to local server computation. Similarly, the local server also has no control or visibility of the RDMA operation, and therefore runtime optimizations are disallowed (e.g., NUMA balancing).

This paper overviews the findings of our evaluation study showing the performance implications arising from deploying RDMA one-sided operations and NUMA. We develop and test a microbenchmark and the well-known Memcached application, ported to use RDMA [3].

Our experiments use nodes consisting of two Intel Xeon E5-2670 v3 processors, totaling 48 cores and two NUMA zones on each machine. All nodes are equipped with one Mellanox ConnectX-3 single port RDMA adapter connected over a 56 Gbps InfiniBand network. Our experiments are implemented in C++ using version 4.5 of the Mellanox driver for Linux. In each experiment, client connections issue one-million operations to the server.

Terminology. A *queue pair* (QP) defines an RDMA connection. *RDMA-local* references the NUMA zone where the RNIC is homed. *RDMA-remote* references the NUMA zone where the RNIC is not homed (i.e., RDMA operations to RDMA-remote memory must cross the processor interconnect on the server machine).

II. Microbenchmark

Individual client latency and throughput are recorded. Latency is measured by immediately polling for completions after enqueuing an RDMA request. Throughput is measured as the average of 10ms-long instantaneous throughput taken during execution. Connections write to disjoint but adjacent addresses in a cache aligned buffer to ensure that clients do not interfere. For smaller access sizes, false sharing is possible. Our reported findings focus on RDMA-reads, except where noted, since RDMA-writes are acknowledged once the request is processed by the RNIC, not when it becomes available on the remote node's memory subsystem.

A. NUMA impact in isolation

Latency is measured over one-million client requests of 64 bytes, corresponding to the maximum individual write by a cache-coherent I/O operation and avoiding false sharing.

Findings. Measuring the speedup of RDMA-local read latency over RDMA-remote read latency for fewer than 16 QPs reveals a consistent 3-5% latency degradation when the accessed memory resides in the RDMA-remote NUMA zone. After 16 QPs, the system becomes bottlenecked by inherent limitations of the RNIC, such as its number of execution units and its small cache.

We estimate the contribution of memory hierarchy to latency by comparing local access times using Intel's Memory Latency Checker (MLC) tool. Due to Intel's DDIO implementation, I/O operations bypass the cache when memory does not reside on the same NUMA zone as the origin of the I/O. Local L3 access within a NUMA zone is $38ns$ on our machines. Access to NUMA-remote memory is $122ns$. Concerning RDMA, the difference of $84ns$ is 3.6% of the $2.3\mu s$ baseline RDMA-read, corresponding to the observed 3-5% difference in latency. In an open-loop configuration, this translates to a 5-7% reduction in throughput because of the increased contention on hardware components.

B. Adding an independent load

An independent process allocates a buffer in memory and spawns threads accessing the buffer randomly. We specify the NUMA locality of the buffer and threads to examine their impacts on RDMA performance.

Findings. We first measure the throughput of one-sided RDMA operations as server-local threads make random accesses on a distinct memory buffer. In the initial configuration, the independent workload is NUMA-agnostic meaning

its threads and memory are not bound to a specific NUMA zone on the server. With an RDMA workload consisting of 8 QPs and 64-byte accesses, we were surprised to see no impact on performance. Reducing the remote access size to 8 bytes led to as much as a 20% drop in performance compared to the no-load baseline.

Pinning the independent workload to a single NUMA zone eliminates the performance hit seen for the NUMA-agnostic workload, indicating that contention on the processor interconnect is the primary contributor to lower throughput.

C. Load accessing remotely accessible memory

Symmetric distributed transactional systems, graph engines and distributed shared memory are all classes of workloads where a local thread may operate on remotely accessed memory. We model them by introducing server load threads in our microbenchmark. Figure 1 shows the improvement of RDMA-local over RDMA-remote operations. Remote accesses are 8-bytes and the remote workload is either 100% reads or 100% writes. Meanwhile, server-local threads randomly access the remotely accessible memory with a workload mix of 90% writes and 10% reads.

Findings. When the local workload is moderate (i.e., more than 8 threads) one-sided RDMA writes to RDMA-remote memory outperform their RDMA-local counterparts by 70-80%. That is, both memory and worker threads are in the RDMA-remote NUMA zone. Under the same circumstances, RDMA-remote reads have 35% better throughput than the RDMA-local operations. This behavior contradicts the expectation that RDMA operations should target memory on the NUMA zone local to the RNIC. Note that at a 50%/50% local workload mix we still observe remote writes to RDMA-remote memory with 50% improvement over RDMA-local.

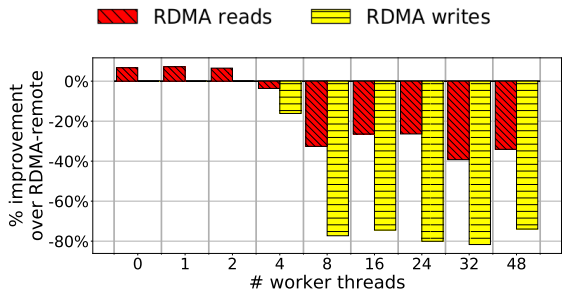

Figure 1. Percentage improvement in throughput of RDMA-local operations over RDMA-remote under our local server load configuration.

We corroborate these findings with the newer generation Mellanox ConnectX-4 network card in a similar testbed, where we observe compounded performance impacts. RDMA write to memory in the RDMA-remote NUMA zone can be up to 300% faster than to memory in the NUMA zone homing the server's RNIC.

III. RDMA-Memcached

In our experiments, clients issue 90% reads and 10% writes to RDMA-based Memcached; each key-value pair is a total of 67 bytes, with 64-byte values. RDMA-based Memcached uses active messaging with one-sided RDMA reads for all data transfers [3]. We record the average throughput of 10 trials of 100,000 operations for each client under three scenarios: *i*) both server and clients are pinned to the RDMA-local or *ii*) to the RDMA-remote zone in their respective nodes; or *iii*) we bind the client to the RDMA-local zone and the server to the RDMA-remote zone.

Under no additional load, performance is best when both the client and the server are pinned to the RDMA-local NUMA zone (up to 20% better than both RDMA-remote).

Next we add a NUMA-agnostic independent workload, with threads on the server accessing a large buffer with an equal percentage of reads and writes. This configuration restricts performance to 50% of the unladen baseline for all configurations due to processor interconnect contention.

When the above load is pinned to a specific NUMA zone, say RDMA-local, RDMA-remote throughput is not impacted. The same holds in the reverse configuration. For both cases, the workload reduces the throughput of remote operations targeting the same NUMA zone by 40-50%. In other words, up to 2x speedup can be achieved by directing RDMA operations to an unburdened NUMA zone, regardless of its locality with respect to the RNIC.

IV. Conclusion

This work demonstrates that NUMA architectures play an important role in RDMA performance, sometimes contradicting the presumed behavior. As RDMA access latency continues its downward trajectory, NUMA will play an increasingly important role in end-to-end performance.

Acknowledgment

Authors thank Haggai Eran for his comments at the beginning of this work. This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-17-1-0367 and by the National Science Foundation under Grant No. CNS-1814974.

References

[1] C. Binnig, A. Crotty, A. Galakatos, T. Kraska, and E. Zamanian. The end of slow networks: it's time for a redesign. *Proceedings of the VLDB Endowment*, 9(7):528–539, 2016.

[2] H. Daly, A. Hassan, M. F. Spear, and R. Palmieri. NUMASK: high performance scalable skip list for NUMA. In *DISC*, volume 121 of *LIPIcs*, pages 18:1–18:19, 2018.

[3] J. Jose, H. Subramoni, M. Luo, M. Zhang, J. Huang, M. Wasiur Rahman, N. S. Islam, X. Ouyang, H. Wang, S. Sur, et al. Memcached design on high performance rdma capable interconnects. In *ICPP*, pages 743–752. IEEE, 2011.

[4] C. Lameter. Numa (non-uniform memory access): An overview. *Queue*, 11(7):40:40–40:51, July 2013.