Investigating Hardware Offloading for Reed-Solomon Encoding

GUSTAVO DE LEON, University of California Berkeley

JOHN DERMER, Eastern New Mexico University

TYLER RAU, Dakota State University

CCS Concepts: • **Computer File System → Encoded Data**

KEYWORDS

Reed-Solomon

1 INTRODUCTION

Reed-Solomon Erasure Encoding is a storage scheme that offers the same level of safety as making redundant copies of the data, but with lower overhead. For example, to protect against data loss of a disk, you could write an extra copy of everything written to one disk onto a different disk. This would use 100% more storage than the size of your data. With RS, the same level of protection can be achieved with a smaller amount of overhead. The trade-off to this is that RS requires heavier computation, than would otherwise be required when storing data, in order to calculate the parity blocks. This presents a problem as it requires users to purchase brawnier CPUs than would be otherwise necessitated.

However, Mellanox's ConnectX-4 and ConnectX-5 Infiniband cards have the capability to offload RS encoding from the CPU to the HCA; removing the need for powerful CPUs and significantly decreasing the cost of safe data storage. Currently, the lab uses Intel's Intelligent Storage Acceleration Library (ISA-L) for their RS encoding; therefore we chose it as our point of comparison. Through this research, we set out to test two things on these cards: the speed at which they encode data, and their ability to rebuild lost data encoded through other means. If the cards proved comparably fast to ISA-L, and are capable of rebuilding its data, then these cards could significantly decrease the cost of using RS erasure encoding for data storage.

2 EXPERIMENTAL AND COMPUTATIONAL DETAILS

We looked to compare the performance of RS computations on the CPU, via ISA-L, and the HCA, via offloading, in many ways. Both ISA-L and offloading were tested to find the effect that the block size scaling and concurrency scaling had on the throughput of each. Throughput was measured by counting how many times the encoding was completed within 60 seconds. Block size scaling tests covered block sizes of $1024 * 2^X$ bytes where X is in the range 0 to 14 on a single thread. Concurrency scaling tests consisted of a constant block size of 1048576 bytes and involved between 2 and 20 threads being added to the mix with minimal delay via a bash script. For concurrency scaling tests we measured total throughput, defined as the summation of the throughput of all threads in a given test.

LA-UR-17-26974

We also tested concurrency scaling on a machine with 6 ConnectX-4 cards inside a single machine. These scaling tests measured total throughput on concurrency scaling tests with 1 to 20 processes per card in the mix at once, total number of process: 6 * X where X is between 1 and 20. Additionally, we performed scaling tests that involved scaling concurrency and the number of cards in use. In these scaling tests, we started with 1 card in use running, 1, 2, 3, 4 processes concurrently. Then we added another card to the mix and ran 1, 2, 3, 4 processes tied to each card.

3 RESULTS AND DISCUSSION

We found that the ConnectX-5 cards performed better, on a single thread at all tested block sizes, then the CPU. The ConnectX-5 graph shows diminishing returns for increasing block sizes, flat lining around a block size of 256KB. ISA-L appeared to do the same but there was a slight down ward dip between block sizes 131072 bytes and 524288 bytes.

We also found that total throughput performance didn't have a clear better performer. The ConnectX-5 flat lined quickly at 3 processes. While ISA-L's performance continued to improve until reaching 14 processes running. Afterward, ISA-L's performance degraded slowly. Early on in the results the ConnectX-5 performs better; however ISA-L does pass it in performance, but it does dip down again. It appears that if further testing were done ISA-L might fall below the ConnectX-5. With a 1MB block sizes, the ConnectX5 maxes out its throughput at 3 processes, while it takes 14 processes to max out the 2x 8 core CPUs in these machines. However, when they reach these points, they both perform roughly the same at about 10 GB/S.

Lastly, the results from the machine equipped with 6 ConnectX-4 cards showed an increase in performance that was roughly 3 times the performance of a single ConnectX-4 card. The performance peaked at 3 concurrent processes per card and then performance degraded slowly. When scaling the number of ConnectX-4 cards and processes, on a single Dell PowerEdge R730, we found that additional cards always adds some additional performance; however the performance increases begin to diminish after 3 cards.

4 CONCLUSIONS

Our two research questions were if the ConnectX-4 and ConnectX-5 cards were fast enough to be suitable replacements for current methods of RS encoding, and if they could be used in a backwards compatible manner to avoid having to re-encode already encoded data with other libraries or methods.

To the question of compatibility, we can confirm that while the ConnectX-4 cards are only compatible with any data encoded using a Galois Field of up to 2^4, the ConnectX-5 cards are compatible with up to 2^8, making the cards are fully compatible with data encoded with Intel's ISA-Library which uses 2^8 exclusively.

To the question of speed, we can confirm that the ConnectX-5 cards enable a substantial increase of encoding throughput over Intel's ISA-L. Regretfully, a direct comparison with the ConnectX-4s is impossible since they are not able to encode in a GF of 2^8. However, we can confirm that multiple ConnectX-4 cards are able to work together to significantly increase the amount of data a single server is able to encode, and there is no reason to believe that would not hold for ConnectX-5 cards as well. Furthermore, the ConnectX-5 cards were faster than the 4s, meaning the speedup from using multiple cards on a single server could be even more dramatic than currently observed.