CS 494 - Cloud Data Center Systems

Final Project

Apache Solr on PEDs
(Performance Enhancing Dataflows)

University of Illinois at Chicago

Porter David

Montanari Claudio

Shinde Komal

1 Abstract

As large-scale data centers continue expand computing resources over the network, hardware accelerators and SDN methodologies have proven to optimize utility of these resources [3]. The best measure of the impact of these new technologies on resource utilization is through rigorous benchmarking exercises of production-grade cloud applications pre and post SDN and accelerator implementation. More specifically, we are most interested in the use case for RNICs in these distributed cloud environments. In our research, we propose deconstructing popular distributed applications like Apache SolrCloud to learn about their networking stack, and test it's efficacy (utility of resources) using a number of different workloads to get a clear performance benchmark. With these metrics, we determine whether new hardware accelerators or SDN methods could improve the performance of cloud applications in production. Therefore, we conclude our paper with a brief analysis of recent hardware accelerator and SDN research and make deductions on how to improve cloud application performance. In summary, our research shows porting new networking technologies to an RDMA framework is the best way to maximize performance of distributed search applications without sacrificing portability and usability of the underlying framework.

2 Problem Statement

It is undeniably imprudent to use previous studies as a benchmark of a cloud application pre augmentation. Without consistency between experiments, benchmarking is useless. If we intend to demonstrate the impact of accelerated hardware and SDN in HPC distributed environments, there needs to be a careful consideration to the hardware and applications used during the benchmarking phase pre and post system augmentation.

First step is selecting the most appropriate cloud application to benchmark. Since our goal for this project is establishing a benchmark for popular cloud applications that can benefit from RNICs, we need the selected application to consume resources affected by the RNICs. RNICs allow applications to 1) bypass the kernel, 2) offload processing and 3) eliminate data copying, they can offer lower latency, lower CPU utilization, and higher single-core throughput than traditional kernel-based networking. Furthermore, for our research to be validated by many people in the industry, the application must be widely-adopted and mature. Therefore, we looked for these properties in the application: wide-adoption, distributed, mature and well documented, CPU and Disk intensive. Apache SolrCloud was the best choice with respect to these constraints.

RNICs have been a computing conundrum for many years because the aforementioned capabilities of RNICs come at a cost when implementing at scale:

- Many RNIC frameworks are not sufficiently fault tolerant.
- RNICs exhibit highly variable performance.

- Current RNIC interfaces are too low level.

Therefore, this research must consider fault tolerance and the highly variable performance when benchmarking Apache Solr.

3 Project Description

SolrCloud is a system for fexible distributed search and indexing. We ported Apache SolrCloud on a commodity 4 node distributed cluster with the following specs:

- Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz with 40 hardware threads
- 125 Gb of RAM
- Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-47-generic x86_64)

Zookeeper is for load balancing and cluster management. In our project we configured three zookeeper servers externally for Solr cloud mode.
The main logical structure of Solr is a collection. A single collection can have multiple shards and shards can reside on multiple machines. In our project, we created a collection named as "reviews" and indexed dataset to it.

Once configured working with Solr-provided example sets, we will write scripts that execute different experiments. An experiment is defined as a task performed over large data sets. This experiment will be designed to selectively pressure test the application based on our benchmarking criteria identified as: CPU, Disk, and Network.

For the stretch goal, our proposed work believes that Apache Solr is a CPU-bound application, and that integrating a network stack utilizing the Mellenox VMA RNIC library will demonstrate enhanced performance. However, the difficulties with RDMA NICs highlight the need for a dynamic system that uses both RDMA NICs and other leading HPC technology. Our work intends to benchmark enhanced RPCs within the Solr environment to see how non-RDMA HPC technology impacts search applications like Solr. A stretch goal for us will be to see if we can port the application to both VMA and eRPC in effort to mitigate overloading the NIC. This could demonstrate that eRPC can underpin the work of the RDMA NICs while maintaining throughput.

4 Previous Work

eRPC

eRPC [2] provides performance out of the box with a library that requires no reworking of traditional network communication interfacing. The only tradeoff here is CPU utilization is very high, so this library would cause CPU throughput bottlenecks for search tasks [1].

FaRM

FaRM [1] demonstrates how important RDMA is for improving latency when dealing with high CPU load [Figure]. However, the usability of FaRM is not great. Their API is very low level and requires a great deal of effort and programmer reworking for optimization. Furthermore, at scale, rNICs suffer performance degradation when reading from many hosts, which typically outweigh the benefits that it provides.

- "Slim: OS Kernel Support for a Low-Overhead Container Overlay Network" https://www.usenix.org/presentation/zhuo

5 Expected Outcome

Our experiments will demonstrate which dimensions the application is bounded. We expect to clearly express each of our experiments along with the performance of each. These benchmarks will be used to gauge the performance delta of integrating RNIC frameworks into these experiments. We expect RNIC-enabled experiments to demonstrate higher throughput and lower latency due to lower CPU utilization.

6 Evaluation

In order to benchmark Solr we created a specific test stress. The test loads on the 3 nodes that are running Solr the Amazon Review dataset for the Electronics products (1.5 Gb approximately). Replication factor and sharding are set to 3. So the configuration will be as shown in Figure 1. Then a python script that generates random queries is run on the fourth node; in the meanwhile all the nodes are profiled using **dstat**.

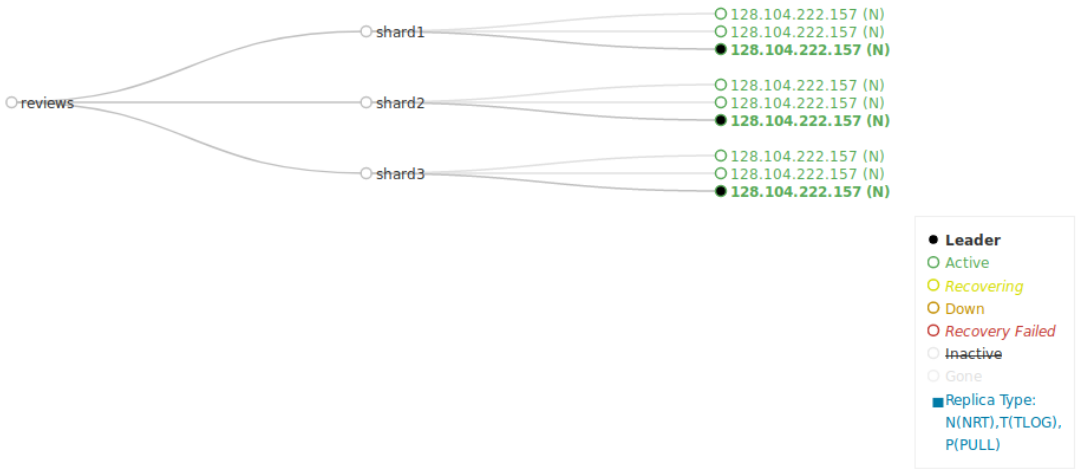

Figure 1: Solr configuration for the Amazon Review dataset; replication factor and sharding are set to 3.

In Figure 2 it's possible to see how latency and CPU utilization vary during time on one machine of the cluster running Solr. What we are plotting is CPU percentage of time spent in idle state, so the higher the better. The CPU utilization is not low very often, part of the reason is due to the reduced size of the dataset and of the query space. Indeed, Solr probably cached some of the queries results reducing the CPU load. For what concerns the latency spikes that sometimes appear, our hypothesis is that Solr estimate on which processor the data for a given query are cached; in this way when multiple similar query arrive at the same time such processors are over loaded and this results in high latency values.

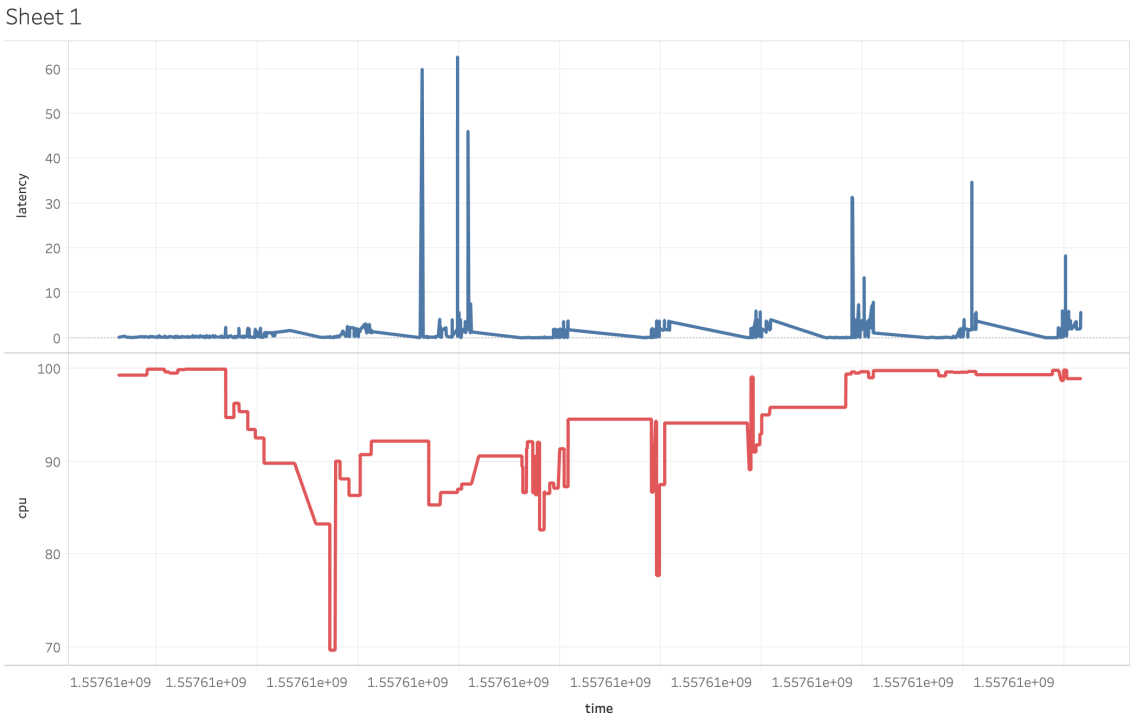

Figure 2: Query latency (seconds) and CPU Utilization on one of the machines in the cluster serving requests. *Note CPU axis shows time spent in idle state which range 70% to 100%.

In Figure 3 it's possible to compare the CPU utilization of the four machines, the first three running Solr while the fourth generating the traffic. Looking at the graph one might think that we could have generated more traffic on the fourth server, but the bottleneck was the solr server because the responses weren't fast enough for the server to fully utilize its CPUs. More specifically, the servers running Solr had low network and disk utilization while some of their cores where highly utilized thus, we can conlude that in this case the bottleneck was on some of the CPUs that were running Solr.

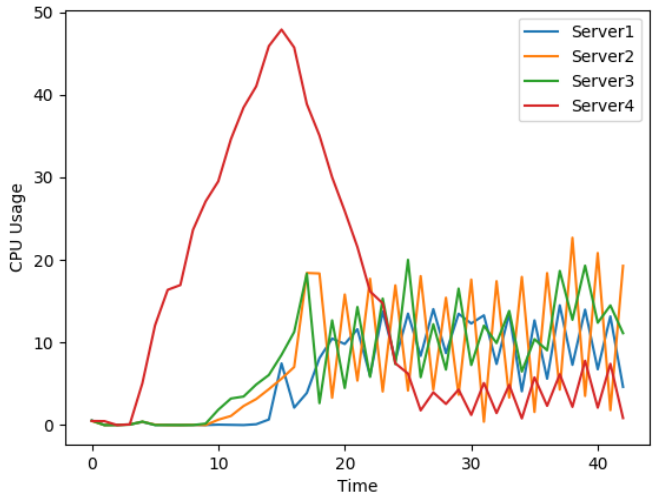

Figure 3: CPU utilization during time for all the servers: servers from 1 to 3 running Solr and server 4 generating the traffic.

In Figure 4 it's possible to compare resource (CPU, Memory, Network) utilization on one of the servers running Solr. It's interesting to notice how disk accesses and network traffic are both low, which is in line with our hypothesis. When looking at CPU utilization though, we still have low values which is not in line with our hypothesis. The interesting insight is that the graph shows only aggregate results and if we look at CPU utilization of the single cores we have almost constant 100% CPU utilization. It wasn't possible to properly record such data because we are running on a cluster where each machine has 40 hardware threads and the one subject to this heavy load where changing between different experiment runs. So we think that developing a system that properly locate such threads and redirecting the load thanks to RNICs would be beneficial, even if maybe not necessary.

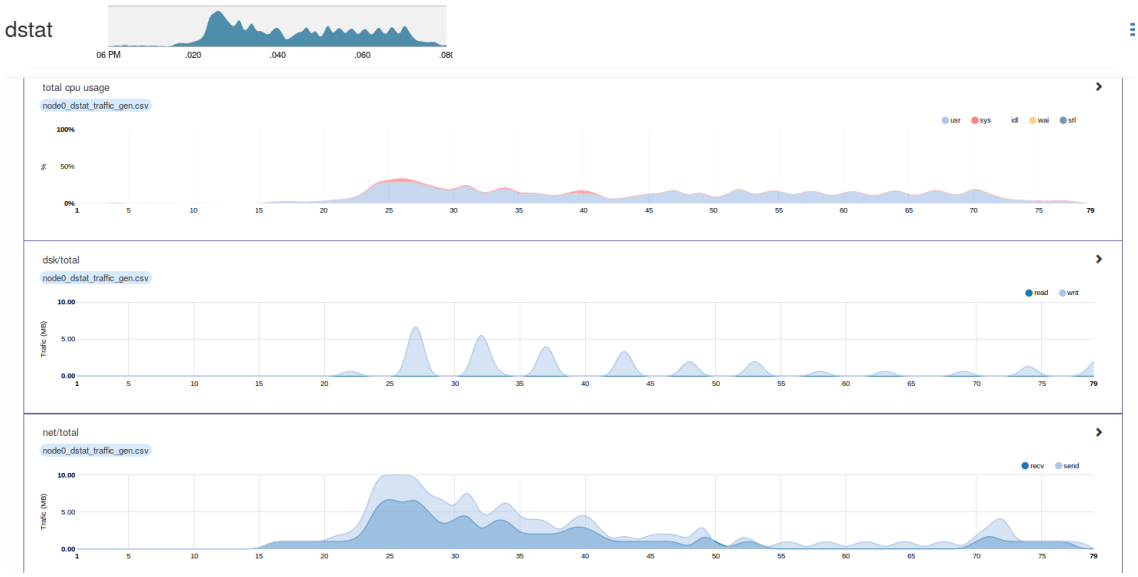

Figure 4: CPU utilization, Disk accesses and Network traffic during time for one of the servers running Solr.

7 Conclusion

Our work shows that Apache SolrCloud can be a CPU-bound application, and that integrating a network stack utilizing the Mellenox VMA RNIC library will demonstrate enhanced performance given the analysis from our research of FaRM [1]. However, the difficulties with RDMA NICs highlight the need for a dynamic system that uses both RDMA NICs and other leading HPC technology. Given the recent usability and performance success of eRPC, we believe client applications should interface with this library, but also utilize RDMA to offset the increased load on the CPU. We propose improving designed an opportunistic RDMA framework that wiRDMA over non-RDMA enhanced protocols such as eRPCs using opportunistic RDMA leasing.

8 Future Works

Next Steps will be to see if we can port the application to both VMA and eRPC in effort to mitigate overloading the NIC. This could demonstrate that eRPC can underpin the work of the RDMA NICs while maintaining throughput.

References

[1] A. Dragojević, D. Narayanan, M. Castro, and O. Hodson. Farm: Fast remote memory. In *11th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 14)*, pages 401–414, 2014.

[2] A. Kalia, M. Kaminsky, and D. G. Andersen. Datacenter rpcs can be general and fast. *CoRR abs/1806.00680*, 2018.

[3] Y. Xu, Z. Sun, and Z. Sun. Sdn-based architecture for big data network. In *2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pages 513–516. IEEE, 2017.