TC Control of Docker Containers

Aldawsari Mohammed
Florida International University
malda021@fiu.edu

Nishant Maurya
Florida International University
nmaur009@fiu.edu

Shuai Xue
Florida International University
sxue004@fiu.edu

Yezehao Huai
Florida International University
yhuai001@fiu.edu

ABSTRACT

Docker platform is a technology that allows anyone to create, launch and destroy containers very easily as well as deploy applications in an isolated environment. It has been widely used by IT companies and government organizations. More importantly, docker containers make it easy to include and configure big data applications - a popular direction for such technologies. This virtualized technology provides flexibility and reduces more costs associated with the physical clusters. However, docker containers when deployed in big data applications pose several challenges such as optimal resource utilization, network configuration and performance. In a distributed environment like Hadoop, normally more than one job is transmitted to the cluster nodes or containers. Problems arise when these jobs compete for the system resources such as CPU, memory, disk I/O and network bandwidth. There may exist unfairness for some jobs increasing some of their execution times due to the strenuous competition. It is very important to monitor the behavior of such a big data tools on the shared environment

In this paper, we make use of this technology to compare the performance of two big data analysis systems, namely Hadoop and Spark. The comparison heavily relies on the impact of the bandwidth size change on the performance of a system. For each system we analyze, compare, and report the system's behavior with respect to the bandwidth change on workloads such as WordCount and PageRank. As of work thus far, we were able to report Hadoop performance on WordCount tasks with 10MB and 100MB bandwidth size as shown in the result section. The initial experiment of running WordCount task on Hadoop with 10MB and 100Mb bandwidth size shows that as the bandwidth increases, the running time decrease by approximately 24 seconds, along with it, the CPU load increases by 10%, as well as Memory load by 7%.

KEYWORDS

Docker, Containers, Big data, Bandwidth, Hadoop, Spark

ACM Reference Format:
Aldawsari Mohammed, Nishant Maurya, Shuai Xue, and Yezehao Huai. 2017. TC Control of Docker Containers. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Big data is a term that is used to represent datasets that are so complex and large that it is hard for traditional data processing systems to deal with. In general, big data is underutilized because big data technologies present many software/tools such as Spark Streaming, Hadoop MapReduce, Storm and Dryad Graph-related systems that are lauded for its ability to handle a huge amount of unstructured and structured data.

Apache Hadoop[1] is an open source implementation of MapReduce. MapReduce can't keep reprocessed data and state information during implementation. Spark[2] is a MapReduce-like cluster computing framework, but it can overcome the lacks of Hadoop in iterative tasks. It utilizes a data structure called resilient distributed datasets (RDDs), to allow data to be cached in memory.

As data is ever growing, more resources need to be allocated. Thus, network properties, such as the bandwidth, and their impact on the cluster performance must be carefully considered. In turn, and in order to test the impact of bandwidth alone on a cluster, one must account for and control the rest of the system infrastructure such that all the involved nodes in a cluster share the same specifications, namely, memory, CPUs, operating systems and I/O bandwidth.

Recently many big data IT companies utilize a new technology called Docker[3] to deploy their applications. Docker is a tool that can package an application and its dependencies in a virtual container, provide flexibility and portability to where an application can run. However, using docker containers when deploying big data tools poses several challenges such as optimal resource utilization, network configuration, and performance[5]. In this paper, we evaluate the performance of Hadoop and Spark based on Docker containers under different bandwidths. We report the performance loss and gain of these applications based on the bandwidth size as well as the individual container's behavior.

2 RELATED WORK

Felter et al.[7] use a suite of workloads that stress the CPU, memory, storage and networking resources. They show that containers result in equal or better performance than VM in almost all cases. Dusia and Yang[6] present an extension to the presently limited Docker's networks to guarantee the quality of service on the network.Their work ensures that critical and time-sensitive applications hosted in high-priority containers can receive a better share of a network bandwidth, without starving other Containers.Bhimani et al.[4]

show that Apache Spark using Docker can obtain speeds up to over 10 times when compared to VM. Several research groups have conducted an extensive study on the comparison of Hadoop and Spark[8]. However, with the use of big data tools in docker containers, which act as processes on the host machine, it is important to compare and analyze the container's behavior and usage of the system resources. This paper is structured as follows: first, we introduce the motivation behind this work followed by a description of the methodology. Furthermore, we detail the experiment setup and report the experimental results. Finally we end the paper with a discussion of the experiment and its results in the conclusion section.

3 MOTIVATION

Currently, there are many cloud computing services based on Docker Containers. The Docker networking and the traffic control technology is essential[9]. Docker technology makes it easier to contain an application, as well as deploy and run containers. Native Docker has various options to manage container resources like CPU, memory and disk. Effective resource management will lead to better utilization of latent hardware and better performance isolation. However, using docker containers when deploying big data applications poses several challenges such as optimal resource utilization and network configuration. Hence, we suspect that one of the aspects that might cause performance loss is the change of the bandwidth size in such big data applications. Therefore, in this project, we focus on comparing the performance loss and gain of two big data analysis systems, namely Hadoop MapReduce and Spark, with respect to the change in the cluster's bandwidth size. Moreover, through this project, we aim to report the individual containers' behavior, computation processes, as well as identify the causes of decreased or poor performance in terms of the bandwidth size.

4 METHODOLOGY

We utilize two workloads, namely Word Count and PageRank tasks, to report performance comparison, when executed on Hadoop and Spark, with different bandwidths. Figure 1 shows the processes taking place in the experiment; Initially, four containers are set up as a cluster in a docker image, in a master and slave configuration, wherein, we install Hadoop and Spark in the Master and the slave containers. Another container also is created which has the reporting tools installed on it to record and display the performance measure of the cluster. Initially the WordCount is executed on Hadoop and spark, with cluster bandwidth as 10MB. After the performance result is generated which displays the performance measures, the cluster bandwidth is changed to 100MB for the whole cluster and the WordCount task is executed again on the whole cluster, generating the performance results for both systems. The cluster bandwidth is again changed to 1GB and WordCount is executed which generates performance result as well. Figure 2 shows the overall bandwidth change framework. After the WordCount task, PageRank task is executed on Hadoop and Spark. Similar to WordCount task, the experiment is performed with three different cluster bandwidths 10MB, 100MB and 1GB, which generates performance results in each time. Finally, the analysis is performed on all

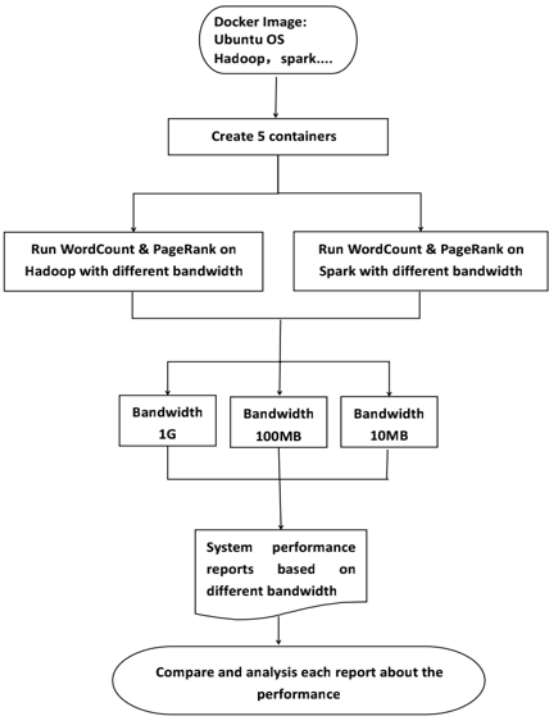

Figure 1: The work flow chart of the Comparison

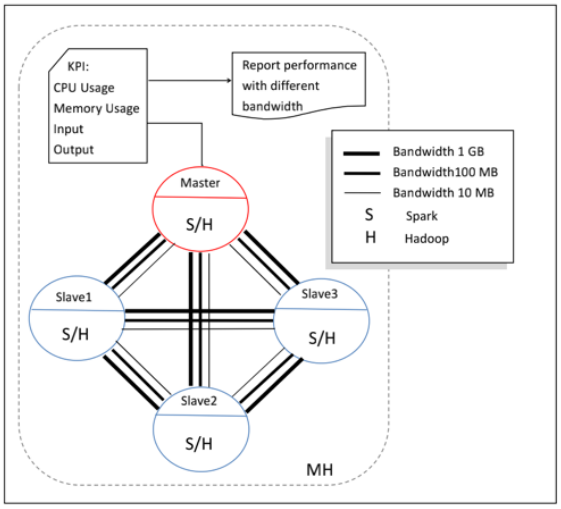

Figure 2: Bandwidth change framework

generated results to find the performance loss due to the changes in the bandwidth.

4.1 Experimental Setup

We run our experiment on a host machine with the following specs: 2.7 GHz Core i5 processor and 16GB memory. We first downloaded Shakespeare's books for the WordCount task. For PageRank task we

downloaded the Facebook dataset from the Stanford web graphs collection. The WordCountt task reads a text file and counts each time a word occurs, whereas, the PageRank task, first used by Google, ranks pages based on their importance and in-linked link. We make use of Docker technology in order to run our experiment. We downloaded a Ubuntu Docker image 14.04 including all required tools for the experiment - namely, Hadoop 2.7.2, Spark 1.2.0, openssh-server, wget, openjdk-7 and nano editor. Then we setup a docker network to allow containers join the cluster using the network name and can be identified by name in the cluster.

We initially construct four containers (one master and three slave nodes - This will be scaled in the final paper) with a memory limit of 2 GB for each container, network name, ports numbers, 'oom-kill-disable' parameter. The last option is set to avoid out of memory (OOM) error. We also configure Hadoop and Spark configuration in each container. Then we run Word Count and PageRank tasks on Hadoop and Spark while varying the cluster's bandwidth between 10MB, 100MB and 1GB in each run. This amounts to a total of 12 runs of the experiment. To capture the performance loss or gain, we use metrics collector tools such as cAdvisor, Grafana and Node Exporter [10].

Table 1: The performance result of running Word Count task on Hadoop for each run

App	Hadoop(WC)			Spark(WC)		
Bandwidth	10 MB	100 MB	1G	10 MB	100 MB	1G
Run Time	183440ms	158470 ms	-	-	-	-
CPU Load	61.77%	71.92%	-	-	-	-
Memory Load	75%	82%	-	-	-	-
Network Input MAX	12.53 KiB	26.33 KiB	-	-	-	-
Network Output MAX	18.21 KiB	33.23 KiB	-	-	-	-

4.2 Experimental Results

To compare the performance of Hadoop and Spark under different cluster bandwidth, we record and analyze the running time, CPU load, memory load, I/O network, overall throughput and throughput per container(yet to be included). As of the current stage, we have run only Word Count task on Shakespeare's books of size 1GB on Hadoop with 10MB and 100MB bandwidth size. Metrics collector cAdvisor and Grafana are used to monitor the performance. As Table 1 shows, when the bandwidth of the cluster is increased the running time decreases by approximately 24 seconds, along with 10% and 7% CPU load and a Memory load increases respectively. For each container, we also report the CPU usage as shown in figure 3 and figure 4. Figure 4 shows that the hadoop master container has low CPU utilization and the slave containers 1 and 2 utilize fairly close enough CPU under both bandwidths. Figure 5 and 6 shows the memory load for each container in the cluster. Apparently, one of the containers (hadoop-slave2) is consuming the memory more than the other containers under both 10MB and 100MB bandwidths.

5 FUTURE WORK

In the final paper, we will include the following: (1) Hadoop on WordCount with 1G bandwidth. (2) Hadoop on PageRank with the three bandwidth size. (3) Spark on WordCount with the three bandwidths. (4) Spark on PageRank with the three bandwidth. (5) Final Comparison.

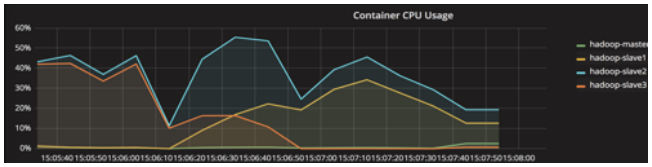

Figure 3: Containers CPU utilization with 100MB network bandwidth

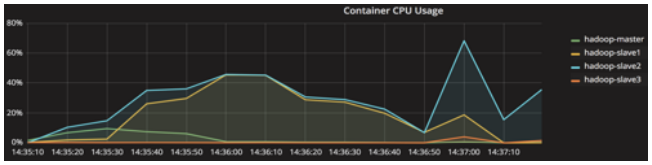

Figure 4: Containers CPU utilization with 10MB network bandwidth

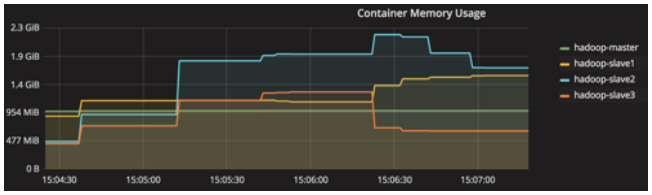

Figure 5: Containers Memory usage with 100MB network bandwidth

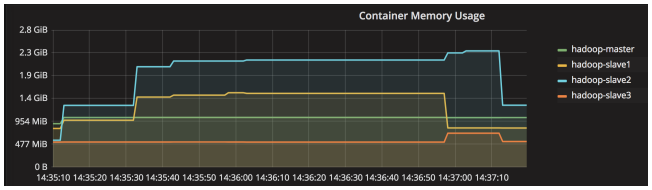

Figure 6: Containers Memory usage with 10MB network bandwidth

6 CONCLUSION

With Docker containers, one can quickly deploy and scale applications into any environment and with as many containers as needed. More importantly, this virtualized technology provides flexibility and reduces costs associated with the physical clusters. However, using docker containers when deploying big data tools poses several challenges including optimal resource utilization, network configuration, and system performance. Therefore, it is important to monitor the behavior of distributed processing systems in such environments. This work highlights an extensive comparison of two big data analysis tools: Hadoop and Spark. The comparison between these two systems is based on recording and analyzing the running time, CPU load, Memory load, I/O network, overall throughput

and throughput per container on a two different workload namely WordCount and PageRank tasks. The initial experiment of running WordCount task on Hadoop with 10MB and 100Mb bandwidth size shows that as the bandwidth increases applications' running time decrease by approximately 24 seconds, along with an increase in the CPU and memory loads by 10% and 7% respectively.

REFERENCES

[1] [n. d.]. Apache Hadoop. ([n. d.]). http://hadoop.apache.org/

[2] [n. d.]. Apache Spark. ([n. d.]). https://spark.apache.org/

[3] [n. d.]. Docker. ([n. d.]). https://www.docker.com/

[4] Janki Bhimani, Zhengyu Yang, Miriam Leeser, and Ningfang Mi. 2017. Accelerating big data applications using lightweight virtualization framework on enterprise cloud. In *High Performance Extreme Computing Conference (HPEC), 2017 IEEE*. IEEE, 1–7.

[5] Minh Thanh Chung, Nguyen Quang-Hung, Manh-Thin Nguyen, and Nam Thoai. 2016. Using docker in high performance computing applications. In *Communications and Electronics (ICCE), 2016 IEEE Sixth International Conference on*. IEEE, 52–57.

[6] Ayush Dusia, Yang Yang, and Michela Taufer. 2015. Network quality of service in docker containers. In *Cluster Computing (CLUSTER), 2015 IEEE International Conference on*. IEEE, 527–528.

[7] Wes Felter, Alexandre Ferreira, Ram Rajamony, and Juan Rubio. 2015. An updated performance comparison of virtual machines and linux containers. In *Performance Analysis of Systems and Software (ISPASS), 2015 IEEE International Symposium On*. IEEE, 171–172.

[8] Lei Gu and Huan Li. 2013. Memory or time: Performance evaluation for iterative operation on hadoop and spark. In *High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC), 2013 IEEE 10th International Conference on*. IEEE, 721–727.

[9] Di Liu and Libin Zhao. 2014. The research and implementation of cloud computing platform based on docker. In *Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 2014 11th International Computer Conference on*. IEEE, 475–478.

[10] Stefan Prodan. 2016. dockprom. https://github.com/stefanprodan/dockprom. *GitHub repository*.