*Docker Container-Based Big Data Processing using Message Passing Interface*

Subash Sunku

Dept. of Computer Science

University of Texas - San Antonio

San Antonio, TX, USA

Email: [Subash.sunku@my.utsa.edu](mailto:Subash.sunku@my.utsa.edu)

Alex Garcia

Dept. of Computer Science

University of Texas - San Antonio

San Antonio, TX, USA

Email: [alex.garcia@my.utsa.edu](mailto:alex.garcia@my.utsa.edu)

Shawn Saltsman

Dept. of Computer Science

University of Texas - San Antonio

San Antonio, TX, USA

Email: [Shawn.Saltsman@utsa.edu](mailto:Shawn.Saltsman@utsa.edu)

*Abstract*—Big data processing is progressively becoming essential for everyone to extract the meaningful information from their large volume of data irrespective of types of users and their application areas. Big data processing is a broad term and includes several operations such as the storage, cleaning, organization, modelling, analysis and presentation of data at a scale and efficiency. For ordinary users, the significant challenges are the requirement of the powerful data processing system and its provisioning, installation of complex big data analytics and difficulty in their usage.

Docker is a container-based virtualization technology and it has recently introduced Docker Swarm for the development of various types of multi-cloud distributed systems, which can be helpful in solving all above problems for ordinary users. This Docker container-based system is an inexpensive and user-friendly framework for everyone who has the knowledge of basic IT skills. Additionally, it can be easily developed on a single machine, multiple machines or multiple clouds.

MPI includes point-to point and collective communication routines, as well as support for process groups, communication contexts, and application topologies.

This paper demonstrates the architectural design and simulated development of the proposed Docker container-based big data processing system in multiple clouds and overview of MPI, a proposed standard message passing interface. Here is the division of work among my groupmates: I(Subash Sunku) worked on maintaining git repository, versioning of files, python script to download the big data and process and communication between containers using MPI, the other two worked on setting up the server, building the docker containers and analysis of big data like displaying graphs and plots using matplotlib.

*Keywords – Docker Container, Cloud, Docker, Hypervisor, Virtualization, MPI, Big Data, SSH, Git*

# Introduction

Hypervisor based virtualization uses a thin kernel called base layer which route instructions coming from virtual machines to the underlying hardware layer. The Windows Hyper-V uses a Microsoft kernel and Linux kernel is used by VMware ESX Server. The virtual machines (VMs) are created on top of the hypervisor in the hypervisor based virtualization. Here virtual machines (VMs) are completely installed Operating Systems.

The virtualization layer offers a file system and kernel service abstraction layer which isolates resources among all virtual machines called "Containers" and it ensures that each container appears as a standalone server.

Cloud-based big data processing systems are the most efficient and established infrastructure to fulfil the big data analysis requirement using MPI. The main advantages of establishing a message passing interface for such machines are portability and ease-of-use, and a standard message passing interface is a key component in building a concurrent computing environment in which applications, software libraries, and tools can be transparently ported between different machines[2].

This paper proposes the Docker container-based big data processing system in multiple clouds for everyone, which explores another potential dimension of Docker for big data analysis by passing info using MPI. This paper demonstrates the architectural design and simulated development of the proposed Docker-based big data processing system in multiple clouds and communication using MPI. This simulation of the big data processing system is based on a single machine consisting of VirtualBox, Docker, Containers and Windows.

# THEORETICAL BACKGROUND

## Container-based VIRTUALIZATION

Docker is a container technology which makes it easy to package and distribute software along with its other dependencies. It makes shipping of software code easy to stage or production or any other environment. Docker is written in Go, an open source programming language created in 2007 at Google by Robert Griesemer, Rob Pike, and Ken Thompson. Developer community is working aggressively on Docker API which have 15 revisions made so far in the past 1.5 years[1].

The container-based virtualization does not use complete virtual machines and hence no overhead of running a completely installed operating system. The advantage of this approach is that there is no need to duplicate functionality like hardware calls since there is just one operating system to take care of all hardware access[2].

In container-based virtualization all virtual machines uses the same kernel and doesn't need a hypervisor. The container based virtualization are used in Windows and Linux. The container based virtualization is suitable choice if optimal efficiency is priority and operating systems preference is less important.

## Overview of MPI

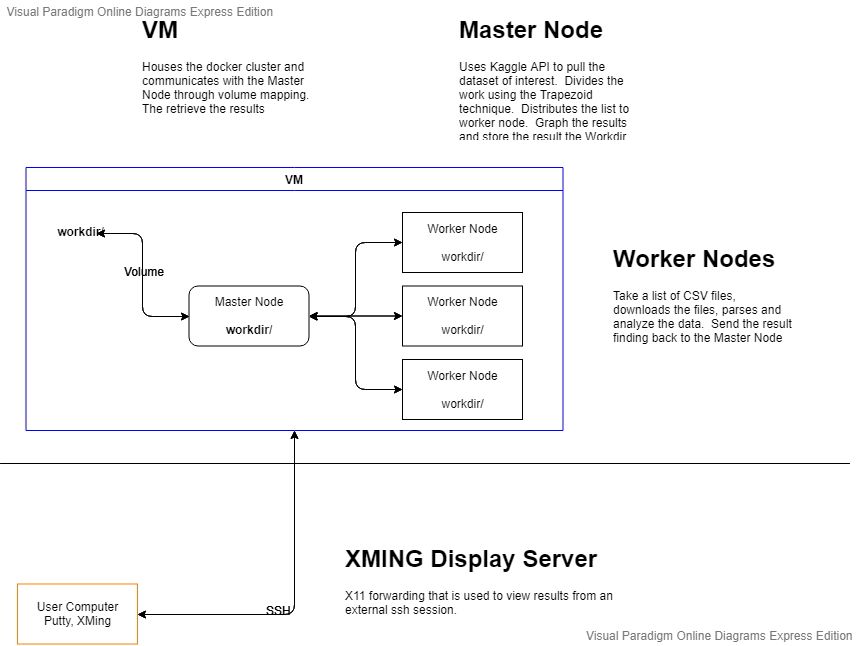
MPI is intended to be a standard message passing interface for applications running on MIMD distributed memory concurrent computers and workstation networks. We expect MPI also to be useful in building libraries of mathematical software for such machines. MPI is not specifically designed for use by parallelizing compilers. MPI does not contain any support for fault tolerance and provides reliable communications (or fails the program). MPI is a message passing interface, not a complete parallel computing programming environment. Thus, issues such as parallel I/O, parallel program composition, and debugging are not addressed by MPI[3].

## Big Data Analysis

Big data analysis is the process of mining and extracting meaningful patterns from massive input data for decision making, prediction, and other inferencing. Traditional data analysis is the process of applying standard statistical methods such as factor analysis, cluster analysis, correlation analysis, and regression analysis to explore the cleaned firsthand data of limited amount. This analysis is usually limited to testing a small number of hypotheses that we define well before the data collection unavoidable.

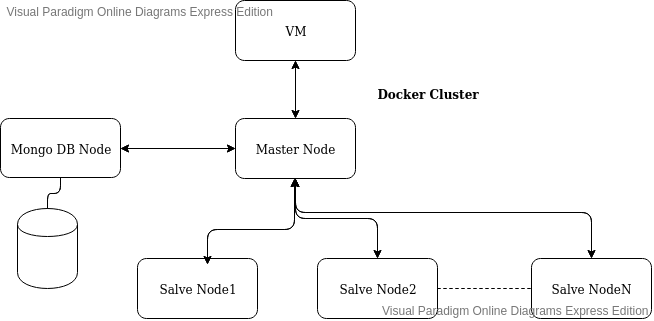
# ARCHITECTURAL design of a CONTAINER-BASED BIG DATA PROCESSING using MESSAGE PASSING INTERFACE

Fig. 1 shows the architectural design of docker container on host OS. Docker provides feasible and cheaper alternative to hypervisor based virtual machines. It has two major components - the open source containerization platform called Docker and Docker Hub which is a Software-as-a- Service (SaaS) platform to share and manage Docker containers. Docker uses a client server architecture model. The Docker client can talk to the Docker daemon which creates, run and distribute Docker containers. The Docker client and daemon can run on the same system or a Docker client can communicate through sockets or RESTful API to a remote Docker daemon.



1. Overall architectural design

Fig. 2 shows the architectural design of Docker container communicating through mpi to and fro between master node(including 1 master, n Workers and 1 database node).



# SIMULATED DEVELOPMENT/METHODOLOGY

This section demonstrates the simulated development of a Docker container-based Kaggle stock market data with the help of MPI and git based repository into three steps: building a Docker cluster for the processing of data and building data volume containers for the management of data and finally git repository for versioning of scripts and files created.

We divided the work among ourselves and decided to progress in the following way: Mr. Alex Garcia worked on the docker and he created the yml files and dockerfiles to build the docker containers, He created the base structure for the project. Mr. Shawn Saltsman did created our profiles on cc server and updates user groups and installed basic libraries in the VM to work on, he then mainly worked on analysis part to display the plots of stocks rise and fall during pandemics. I am responsible for the most of coding part – python – downloading data from Kaggle, processing it, dividing csv files among master and worker nodes and communication between them, creating scripts – bash file to execute the whole project in a single go, it contains commands to build the containers, get the ip address of containers, creating hostfile and scp it to containers and testing the actual python script to run on stock data and display graphs, maintaining git repository to maintain versioning of files and utilizing 3rd part tools like putty and Xming to display graphs and plots generated during execution.

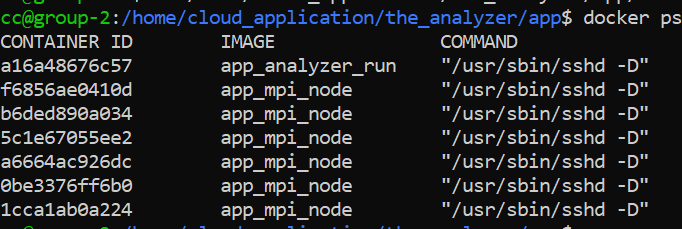


Fig.3 : Building of Docker containers on VM cc server

## Building a Docker Cluster for the Processing of Data

This experimental simulation of the big data processing system is based on Virtual machine, Docker containers, Git Lab repository and Kaggle dataset. Here the big data system is developed as a cluster of 1 master node and 6 worker node containers as shown in fig. 3.

The Docker Cluster contains the self-contained solution, Docker file having commands to install Ubuntu - operating system, SSH – a network protocol to connect to worker node containers from master node, Kaggle api to download datasets directly from Kaggle website and python3 libraries like numpy, pandas, matplotlib etc., to perform data analysis, and lastly using Yml file we can having VM accessing volume of master container, distributed works and display results.

We created a docker image that includes python3, Kaggle, mpi script and mongoDB. The image can be built using the command as in fig. 3:

docker-compose up -d --scale mpi\_node=4 –build

The above command once run in VM will create a docker container with 4 worker nodes. We tested our python script which contains code to pull data from Kaggle and display results as shown in fig. 4 below.

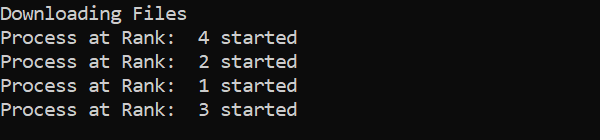
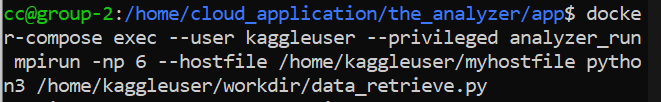


Fig 4 Testing our script using docker composed mpi nodes

## Git Based repository

Throughout the project we maintained a git repository to have versioning of files. The folder structure of Git repository is as below.

* The Analyzer
  + App
    - Docker files, yml files, requirement files, script to run whole project.
  + Work directory
    - Contains python scripts.
  + SSH
    - Contains rsa public and private keys to ssh into a network
  + Kaggle
    - Kaggle.json to access Kaggle website using Kaggle api

We used the basic git commands to keep the project files updated and versioned, commands are as follows:

git init – to initialize git repository

git add . – to add all the modified files

git clone remote\_server – to clone from remote server

git commit – to commit local changes

git push – to push changed to remote location.

Link to our git repository:

<http://129.114.25.79/cloud_2k_project_cs5573>

Our project primary goal is to do statistical analysis on stock market data during pandemics: H1N1 that occurred around 2009-2010 and current ongoing COVID-19. We picked our stock market data from Kaggle website which is created by jacksoncrow and it contains about 3GB of data with almost 4000+ individual csv files (stocks exchange listings). Each stock file has the following parameters: Date, open price, close price, High price, low price, volume of stocks that changed hands.

We concentrated mainly on closing prices of stocks during period January 2009 to August 2010 when first pandemic occurred and latest stock data from Feb 1st2020.

Fig 5 shows the sample data of some of the stocks available.

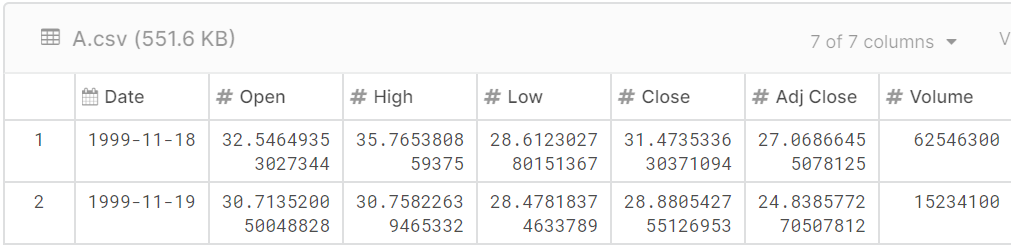


Fig 5 Sample stocks data from Kaggle website

## Message passing and Analysis:

The main concept of MPI comes into picture when downloading the dataset from Kaggle website which is of 3GB data. The master node will download the data and divide it among the worker nodes. Each worked node will process the available csv files into dataframes using pandas and generate plots using matplotlib. Some basic MPI4py commands used are:

**Comm.Get\_size() –** calculate no. of processes

**Comm.Get\_rank() –** get process rank

**Comm.send(data, dest) –** send data to dest

**Comm.recv(data, source) –** receive data

Coming to the analysis part, we took the data before, during and after pandemic and plotted line/bar graphs of percentage change of stock prices and found that pandemics adversely affect stock market. Some of fascinating results can be found in next section.

The plots generated by our python script using matplotlib are later displayed on windows system (as linux system does not have a GUI) using putty and Xming Display server - 3rd party tools.

# results

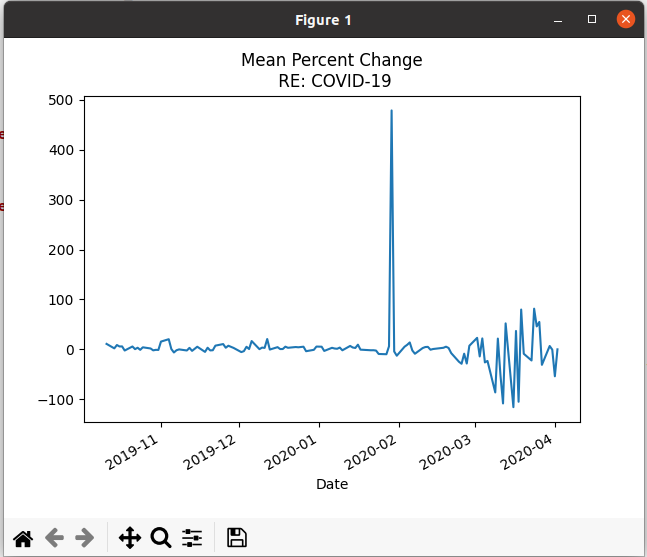
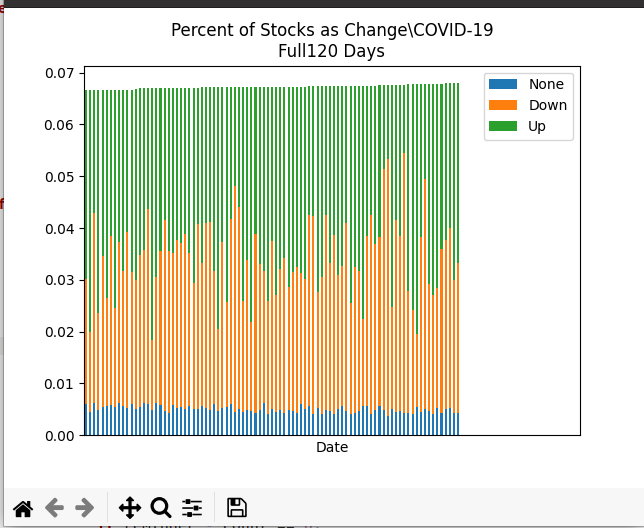
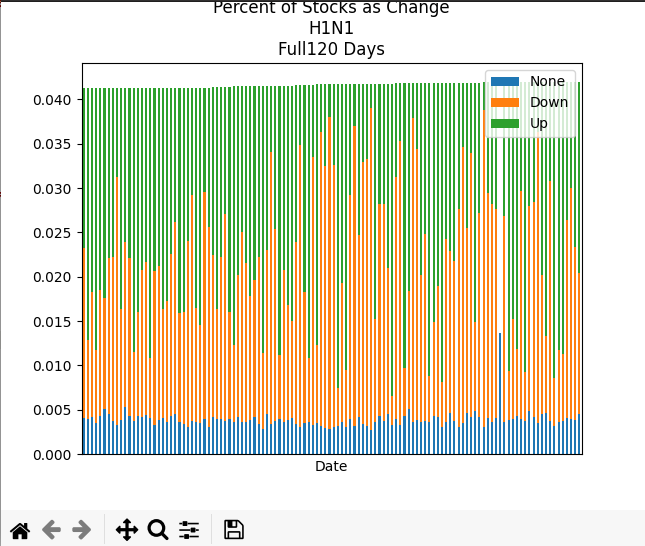
fig:5 stock market change during COVID -19 pandemic Fig: 6Stock market changes during COVID: 120 days report

Fig: 7 stock market changes during H1N1 : 120 days report

As we can see in the above graphs figure 5, due to pandemics, stock market having dings during period March 2020 to April 2020. Also, we can see the individual stocks changes during H1N1 and COVID-19. In figures 6 and 7, blue part shows the stocks did not have much difference and orange ones are stocks that had downfall and green ones are raised. The analysis is based on 1000 files.

# conclusion

This paper proposed the Docker container-based big data processing system in multiple clouds/containers in this case for everyone, which explored another potential dimension of Docker. It demonstrated the architectural design and simulated development of the proposed Docker container-based big data processing systemin multiple clouds.

To conclude on stock market, we can see that stocks are negatively impacted during pandemics.

# References

1. Nitin Naik – ‘Docker Container-Based Big Data Processing System in Multiple Clouds for Everyone’, IEEE 2017
2. Sachchidanand Singh, Nirmala Singh, ‘Containers & Docker: Emerging Roles & Future of Cloud Technology’, IEEE, 2016
3. The MPI Forum, ‘MPI: A Message Passing Interface’, ACM 1993.
4. N. Naik, “Building a virtual system of systems using Docker Swarm in multiple clouds,” in IEEE International Symposium on Systems Engineering (ISSE). IEEE, 2016.
5. container-based virtualization (operating system-level virtualization) - <http://searchservervirtualization.techtarget.com/definition/containerbasevirtualization-operating-system-level-virtualization>