A Comparative Study of Containers & Virtual Machines in Big Data Environment

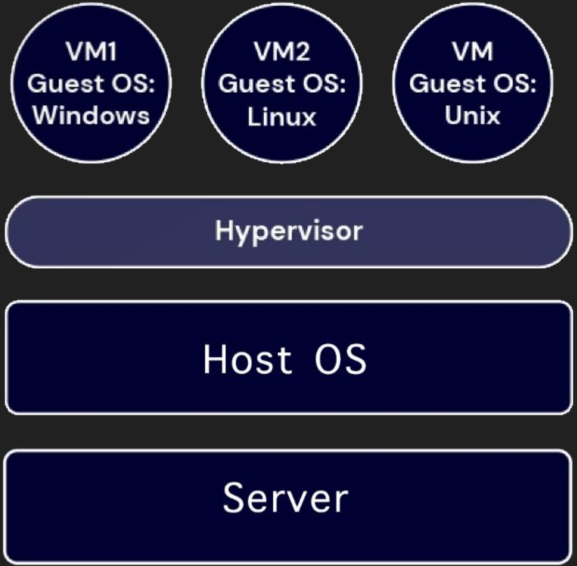
Ishita Jaisia
ij2056

Shiv Sinha
srs9969

Introduction

- ❏ **I**nvestigate how much conveniency containers and VMs can bring to the system administrators. Specifically, focus on how fast a big data computing environment can be setup from the scratch.
- ❏ **M**easure the impact of using containers and VMs on the performance and scalability of different big data workloads.
- ❏ **A**nalyze the reasons why containers and VMs can have different impacts on the big data workloads.

Virtual Machines



VM1
Guest OS:
Windows

VM2
Guest OS:
Linux

VM
Guest OS:
Unix

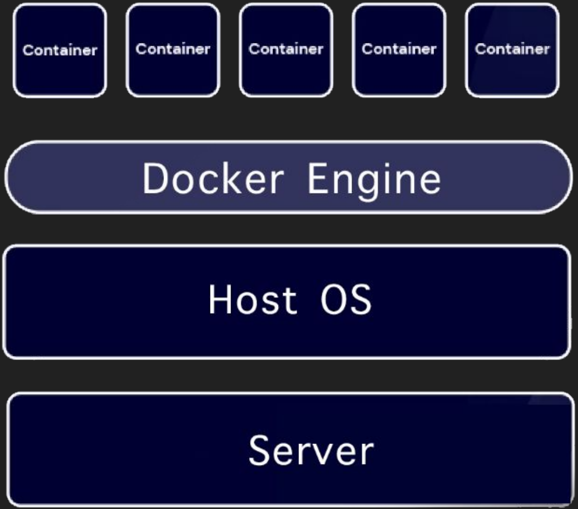
Hypervisor

Host OS

Server

What's
the
DIFF?

Docker container



Container

Container

Container

Container

Container

Docker Engine

Host OS

Server

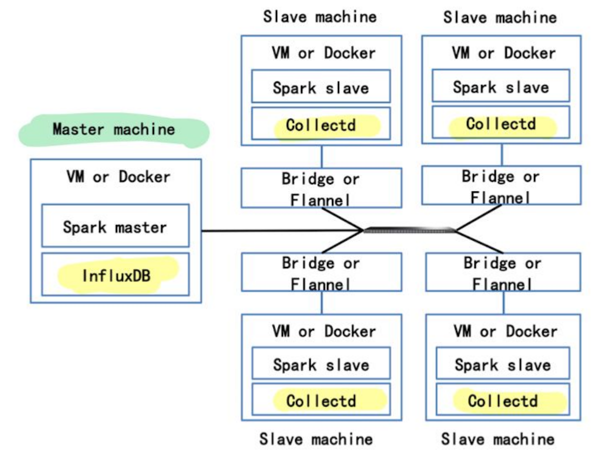
Experimental Setup

- ❑ 1 primary and 4 secondary machines
- ❑ 8 core each 3.40GHz
- ❑ 16GB RAM
- ❑ 1TB disk
- ❑ Ubuntu 16.04-64 bit running on 5 machines and all the VMs.
- ❑ Docker 1.12.6
- ❑ VM - Network Bridge
- ❑ Docker - Flannel

Spark Workloads

An open-source, distributed processing system used for **big data workloads**.

- ❑ K-means
- ❑ PageRank
- ❑ Logistic Regression
- ❑ SQL Join



Deployment Conveniency

Time spent on building a 3-node spark cluster using containers vs VMs

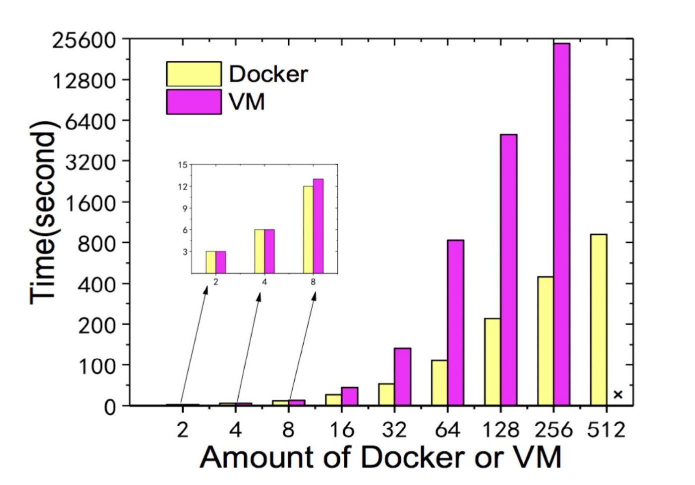
	Build Image	Setup Spark	Start Cluster	Total Time	Image Size
VM	28 min	13 min	6 min	46 min	4.1 GB
Container	6 min	12 min	5 min	23 min	1.1 GB

Why does VM take more time and space than Container?

- Containers share the same Host OS. VM requires the OS to be installed.
- 1 Image file can be shared among different containers whereas, each VM requires its own image to start with.

Bootup Efficiency

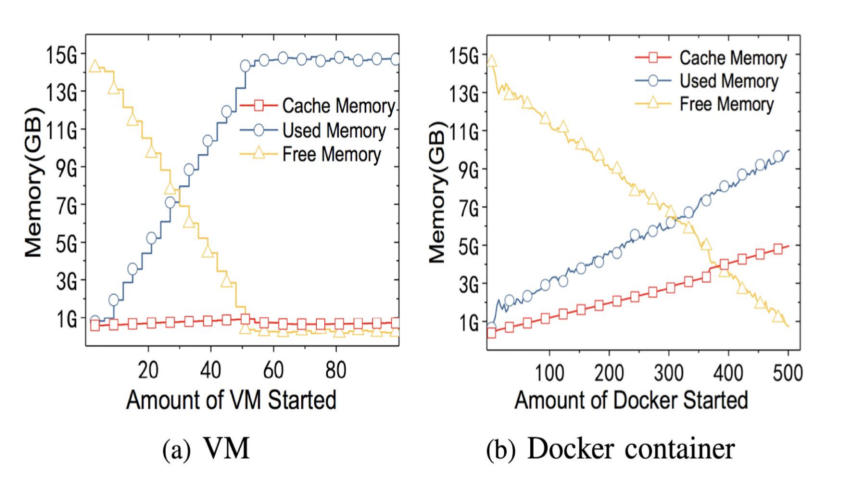
Time



When there are around 250 idle VMs on the host it takes more than 1000 seconds to boot up one VM.

Takes only 987 seconds to start 512 Docker container.

Memory



(a) VM

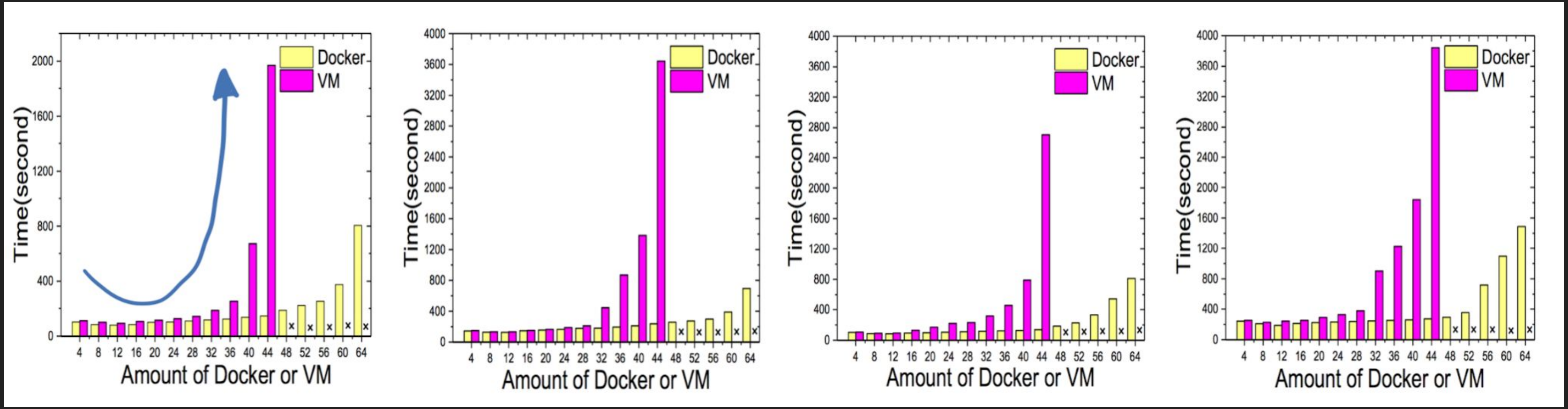
(b) Docker container

Docker container takes less memory than a VM after it boot up.

While a VM takes 0.23GB, whereas a Docker container only takes 0.03GB memory.

Application Performance

*Avg of 5 runs



K-means

Logistic Regression

Pagerank

SQL Join

Cluster size 4

Containers : 145 sec

VMs : 149 sec

Cluster size 44

Containers : 238 sec

VMs : 3643 sec

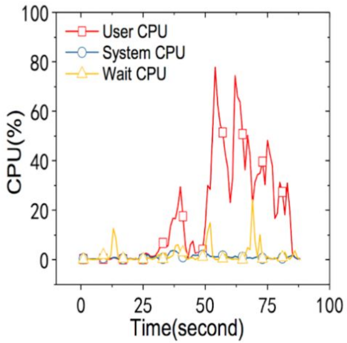
Cluster size 32

Containers : 118 sec

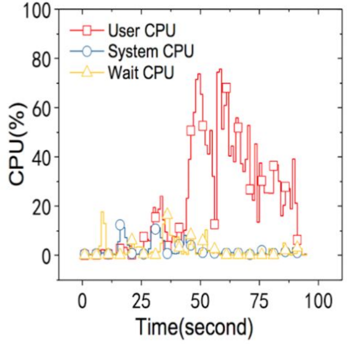
VMs : 316 sec

Scalability diff?...

CPU Utilization - PageRank



(a) 2 containers/machine



(e) 2 VMs/machine

CPU Utilization Categories:

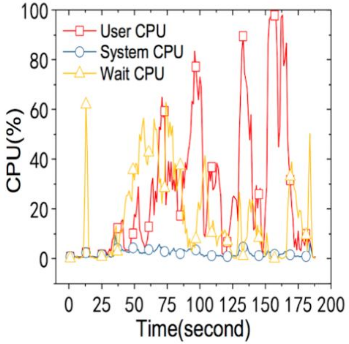
❑ User Level CPU

❑ Wait Time

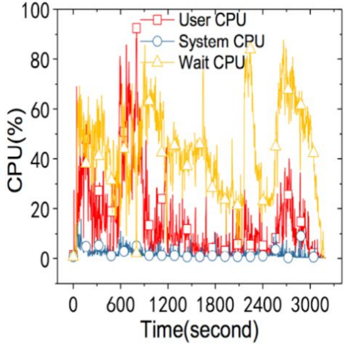
❑ System Level CPU

In both sides, we see High Wait time and User Level CPU as Pagerank workload is a user level application.

Pagerank failed in a 48 VMs cluster while successfully finished in 183 seconds in a 48 containers cluster.

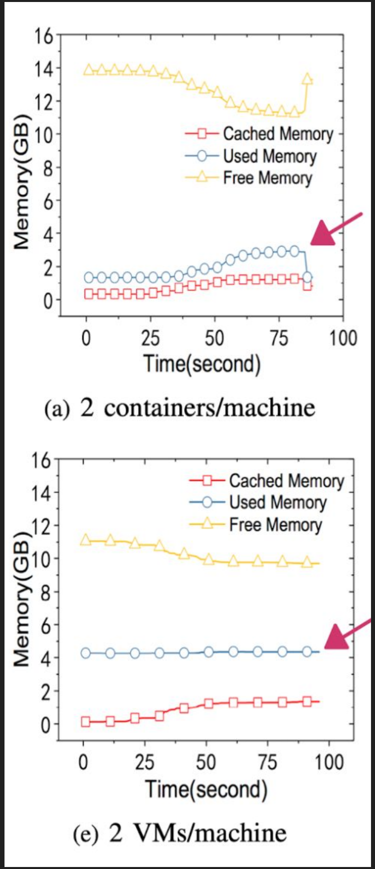


(d) 12 containers/machine



(h) 12 VMs/machine

Memory Utilization - PageRank



(a) 2 containers/machine

(e) 2 VMs/machine

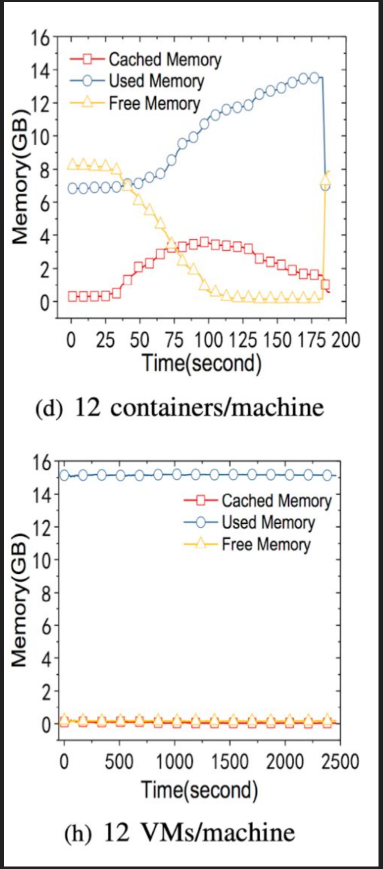
Memory Utilization Categories :

- ❑ Cached memory
- ❑ Used memory
- ❑ Free memory

The amount of memory allocated to a container is very small at the beginning and then increases based on the demands of the application in the container.

VM occupies more memory at the beginning.

A container releases its memory after it finishes its workload while a VM still holds the memory even after it becomes idle. (*left graph red arrows*)



(d) 12 containers/machine

(h) 12 VMs/machine

Conclusion

The extensive measurement study shows:

❏ Dockers container is more convenient than VM for system administrators both in **deployment** and **bootup** stages.

❏ With different big data workloads, Dockers container shows much better **scalability** than virtual machines.

❏ With the same workload, Docker containers achieves **higher CPU and memory utilization**.

What we would have done differently?

- ❏ Compared Spark with Singularity containers inplace of Docker containers in the Big data environment.
- ❏ Secondly, Singularity natively supports MPI which would have been interesting to compare with Spark in terms of the speedup related to different distributed workloads and CPU, memory consumption and network latency.