CPU & I/O-intensive distributed Computing w/ Docker on AWS

Keywords: Amazon Web Service (AWS), EC2, Apache Spark, Apache Mesos, Docker containers, Rancher, Cattle, MapReduce, Shuffle, Monte Carlo method.

by Toni Pohl[§] and Cedric Bhihe[§§]
Universitat Politècnica de Catalunya
Facultat d'Informàtica de Barcelona
([§]) toni.pohl@est.fib.upc.edu
([§§]) cedric.bhihe@gmail.com

OBJECTIVE OF THIS WORK:

To benchmark computing overheads for SPARK jobs on DOCKER (AWS-EC2), with and without APACHE MESOS and/or RANCHER abstraction layers, for CPU and data I/O-intensive use cases.

INTRODUCTION

- Dockerized environments, i.e. **Docker** (*D*) swarms, on Lx hosts are lean, quick to boot-strap, containerized contexts in which to carry out well isolated computations - see Fig.1. Some use cases are:
  - **(a)** multiple environments on 1 cluster (e.g. dev-ops, prod.)
  - **(b)** multi-applications clusters (e.g. web-services, etc.)
  - **(c)** multi-cluster batch jobs

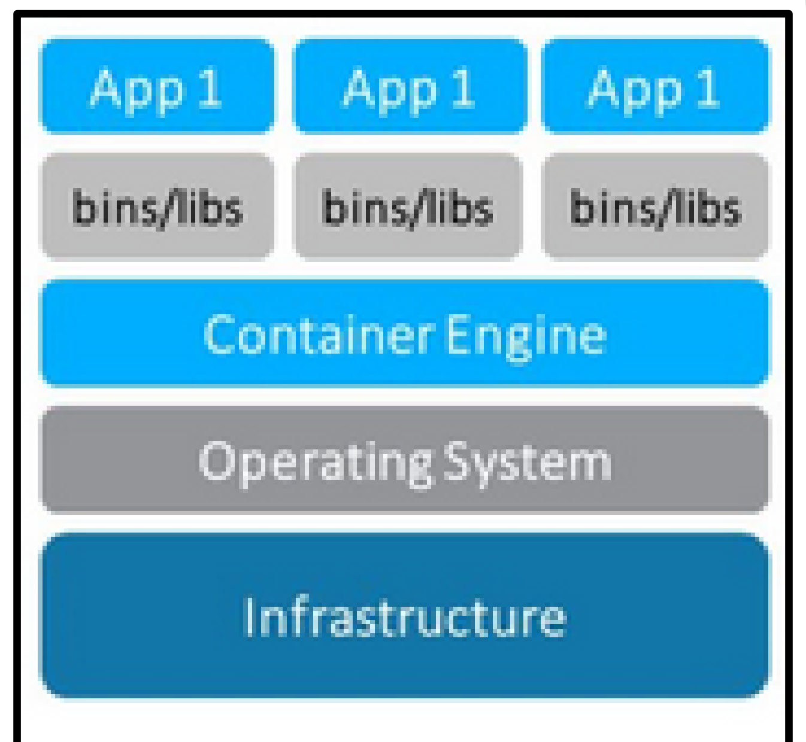


Fig.1: Containers

- For instance **MapReduce** batch-jobs with **Spark** may not only seek to optimize CPU usage, but also need to manage I/O and storage across thousands of cluster nodes, tolerating node churn and relying on agile networking between containers.
- The Spark *computing framework* on Docker may require ***help*** in:
  - **(i)** orchestrating infrastructure and scheduling D-containers,
  - **(ii)** managing users (RBAC, AD, LDAP)
  - **(iii)** leveraging network services (CNI: DNS, x-host comm.),
  - **(iv)** balancing load.

- Rancher (v1.5) on Docker (v17.03) provide the entire software stack needed to manage D-containers in production. See Fig. 2.

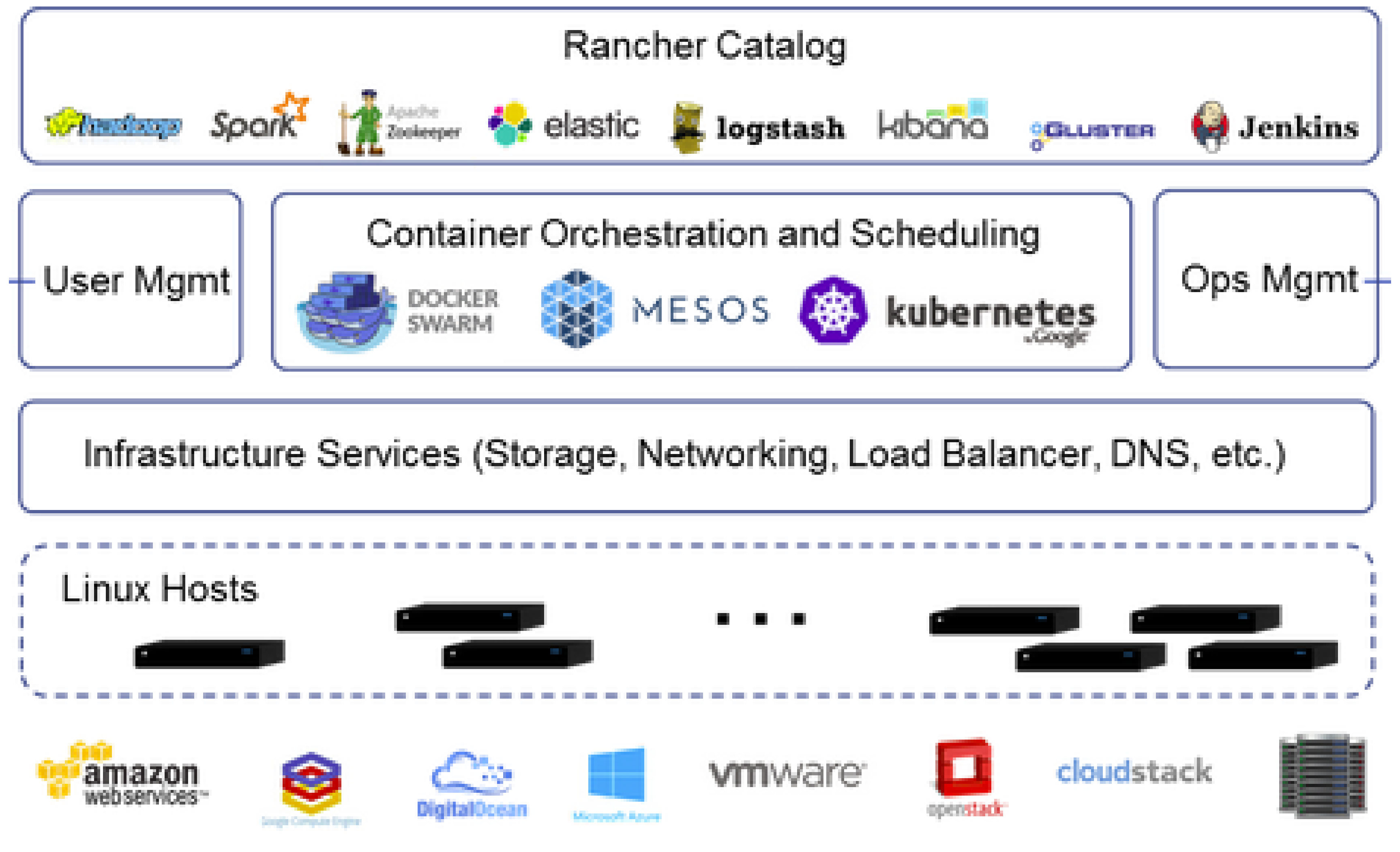


Fig.2: Rancher explained

PROPOSAL

- To measure the performance impact of Docker containers on Ubuntu v14.04 hosts, when running stress-tests via the computing framework, Apache Spark v1.6.2 on JVM7
- To submit model batch-jobs via Apache Spark [S], as follows:

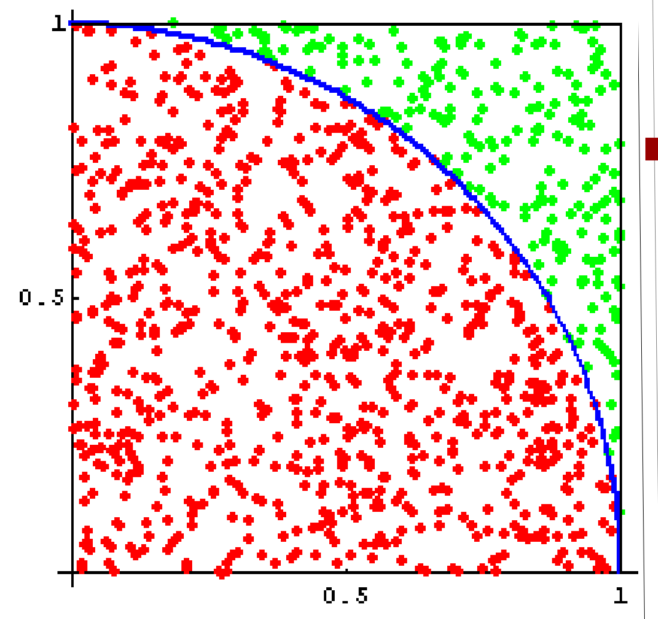
- directly on Ubuntu [U]	– ***SU***	: *baseline*
- on Rancher [R] on [U]	– ***SRU***	
- on Apache Mesos [M] on [U]	– ***SMU***	
- on [M], as scheduler for [R] on [U]	– ***SMRU***	

TESTING

- All experiments ran on AWS-EC2 "t2.large" with 1 master and 3 slaves using off-the-shelf AMI. The master ran on 1 of the slaves' host, but not in the same container when Rancher with Docker were used.
- All hosts were located in the same region and external IPs were used for cross-host communication.
- When used with Apache Mesos, the Spark client was executed on one of the 2 Mesos slaves, not located on the Mesos Master's host.
- All 4 setups were tested under CPU intensive (SparkPi) and Data intensive (JoinBench) conditions. Each test was replicated 5 times.

SparkPi test

- Computes an approximation to π via Monte Carlo (MC); example-app that ships with the Spark installation (*scala code on GitHub*).
- Parallelize with 10,000 slices .
- A MC statistical simulation method evaluates the ratio of a unit radius circle's area (π), to the containing square's area (4) by randomly choosing point's real coordinates x and y, st. $(x,y) \in [0,1] \times [0,1]$. The result value for $\pi \cong 4\ m / n$, where m is the number of points that satisfy $x^2+y^2 \leq 1$ and n is the total number of points tried for a 10,000 slice run.



- This non-deterministic *CPU intensive* method converges slowly.

JoinBench test

- Apache Spark extended Hadoop's MapReduce processing, by introducing the Resilient Distributed Dataset (RDD) for in-memory cluster-computing.
- Expensive I/O operations consist in shuffling data accross a cluster of partitions. A Shuffle happens between multiple operations, e.g. when a table join-by-key is performed (Fig. 3).

ID	A	B
1	100	400
2	200	500
3	300	600

ID	X	Y
1	10	11
2	12	13
3	14	15

ID	L_id	R_id
1	1	1
2	2	3
3	1	2

Join

- During a shuffle data no longer lives only in memory. Data partition, compression and I/O may occur multiple times.
- Our JoinBench test performs a *TableJoin* between 3 tables with rows sized 1,000; 20,000 and 3,000,000.

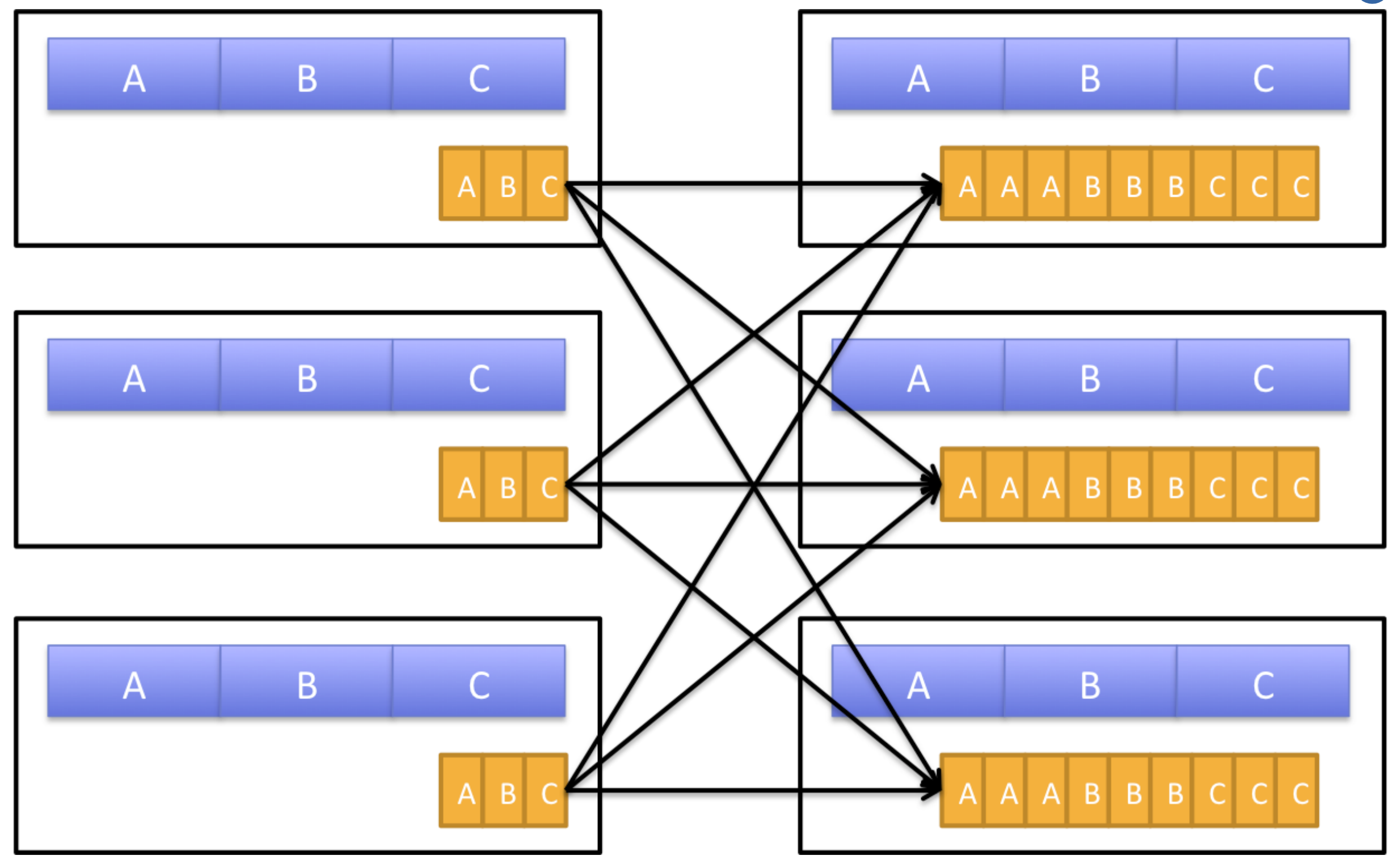


Fig. 3: Spark key-value pair shuffle (all-to-all)

RESULTS

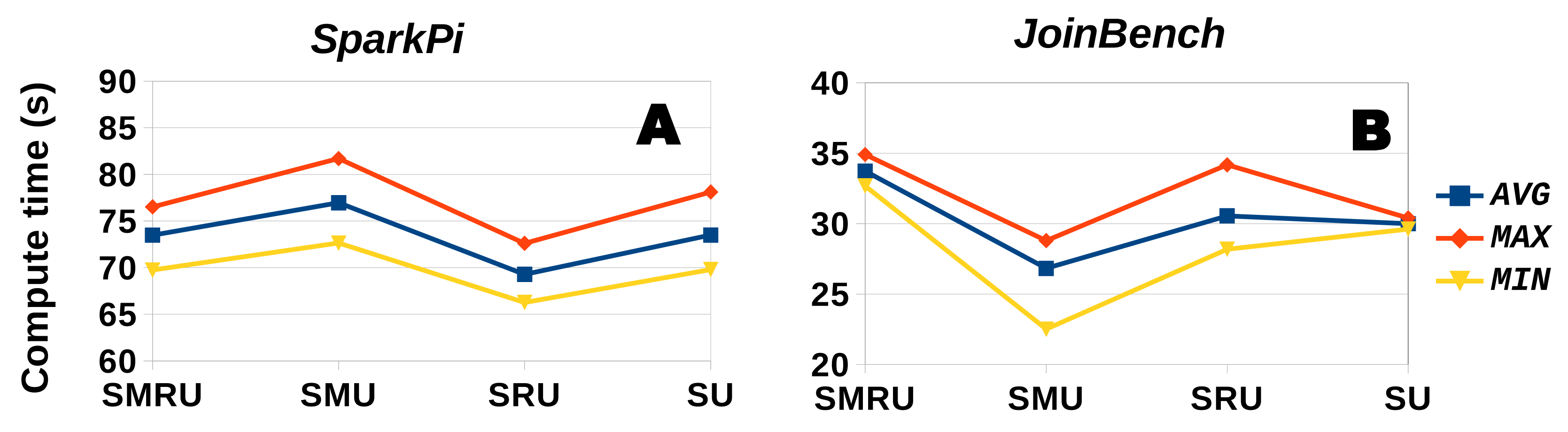


Fig. 4: Time to completion (in seconds) for 2 tests and 4 setups.

DISCUSSION / CONCLUSION

- Results (Fig.4) show maximum, average and minimum values for computational times in seconds for each test and all 4 setups.
- Compute time variability does not exceed ±5% for SparkPi tests and ± 15% for JoinBench tests. This may be due to a small number of nodes in the cluster used for testing.
- Differentiated performance impacts could be attributed to the use of *Rancher with Docker* (RwD) in both tests:
  - ☞ For SparkPI (graph A), RwD has a positive impact whether Apache Mesos is used as a scheduler or not.
  - ☞ For JoinBench (graph B), the opposite holds. RwD seems to have a detrimental effect on performance, whether Apache Mesos is used or not.

FURTHERING THIS WORK ...

- A greater number of nodes, although much costlier to test on AWS/EC2, might reveal an overall beneficial performance impact of Apache Mesos.
- The effect of distributing hosts and in different AWS regions may also contribute to the heterogeneity of the computing cluster and could similarly reveal the beneficial role of the Mesos scheduler.
- ....

Reference: *http://docs.rancher.com/rancher/v1.5/en/*
http://rancher.com/playing-catch-docker-containers/