OPERA-P

OPportunistic Elastic Resource Allocation & Provisioner

A Big Data Plumbing with HTCondor and Hadoop Yarn

Feras Mahmoud Awaysheh &
Pablo Vázquez Caderno, et. al

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

CAMPUS VIDA CAMPUS DE EXCELENCIA INTERNACIONAL

CiTIUS Centro Singular de Investigación en Tecnoloxías da Información

HTCondor High Throughput Computing

Outlines

- Background
  - Scheduling Large-Scale Clusters (LSC)
  - State of Resource Management in Big Data
- Challenges
  - Proposed solutions
- Introducing OPERA-P
  - What is it and What's not
- OPERA-P architecture
- Use case - MapReduce
- Opportunities
- Future work

Scheduling Large-scale Clusters

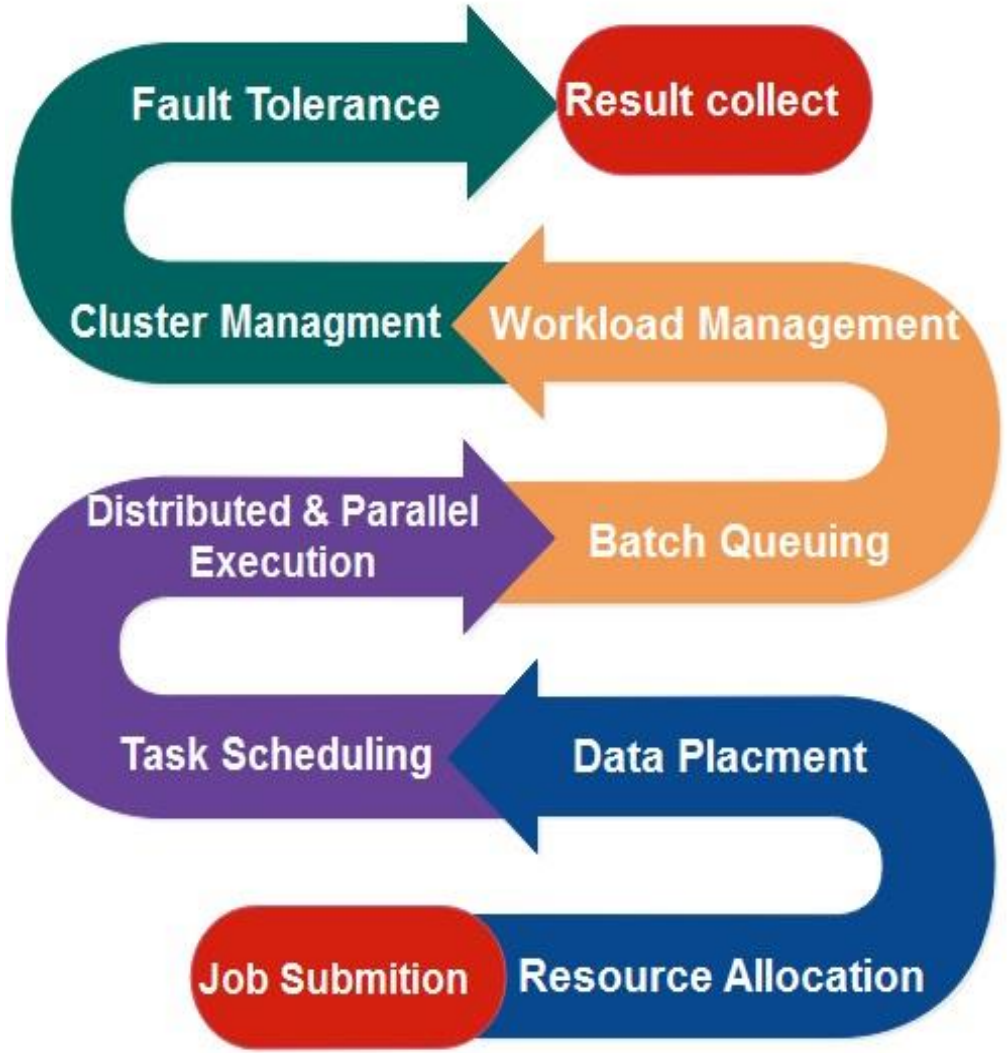
- Goals:
  - Highest Utilization
  - Maintain ultimate efficiency
  - Scalable (whenever, however we want)
  - High fault tolerance

- Issues:
  - Un-predictable load
  - Increasing workload, clients and cluster size
  - Common delusion
    - Network reliable and homogeneous
    - Transport cost is zero

Resource Management Terminology

- Different cluster scheduler architectures.
  - Monolithic schedulers
  - Two-level schedulers
  - Shared-state schedulers
  - Hybrid solutions

- Hide the details so that the user focus on application development
- Maintain in high availability, reliability and support frameworks to do so
- Open source resource management solution:
  - Hortworks, Cloudera, MapR - YARN
  - Apache Mesos and Myriad
  - Container-based Clusters - Docker Swarm, etc,.



LSDS Challenges

- The utilization problem…
- Multi-tenancy problem…
  - Virtualization impacts performance
  - Difficult to do short term borrowing of capacity
- Installing new infrastructure
  - Cost, cost and cost
- Fault tolerance, Failure management and security

Schedulers wish list!

- Applications request resources when they need them
  - Automated without user intervention
- Scale-out on demand to a free resources
  - Elastically provisioning
- Multi-tenancy with strong isolation
  - Sandbox with the required libraries etc,
- Minimal configuration
  - Updated/restarted without affecting current running tasks

The problem!

- Creating infrastructure silos to accommodate BD apps is inefficient

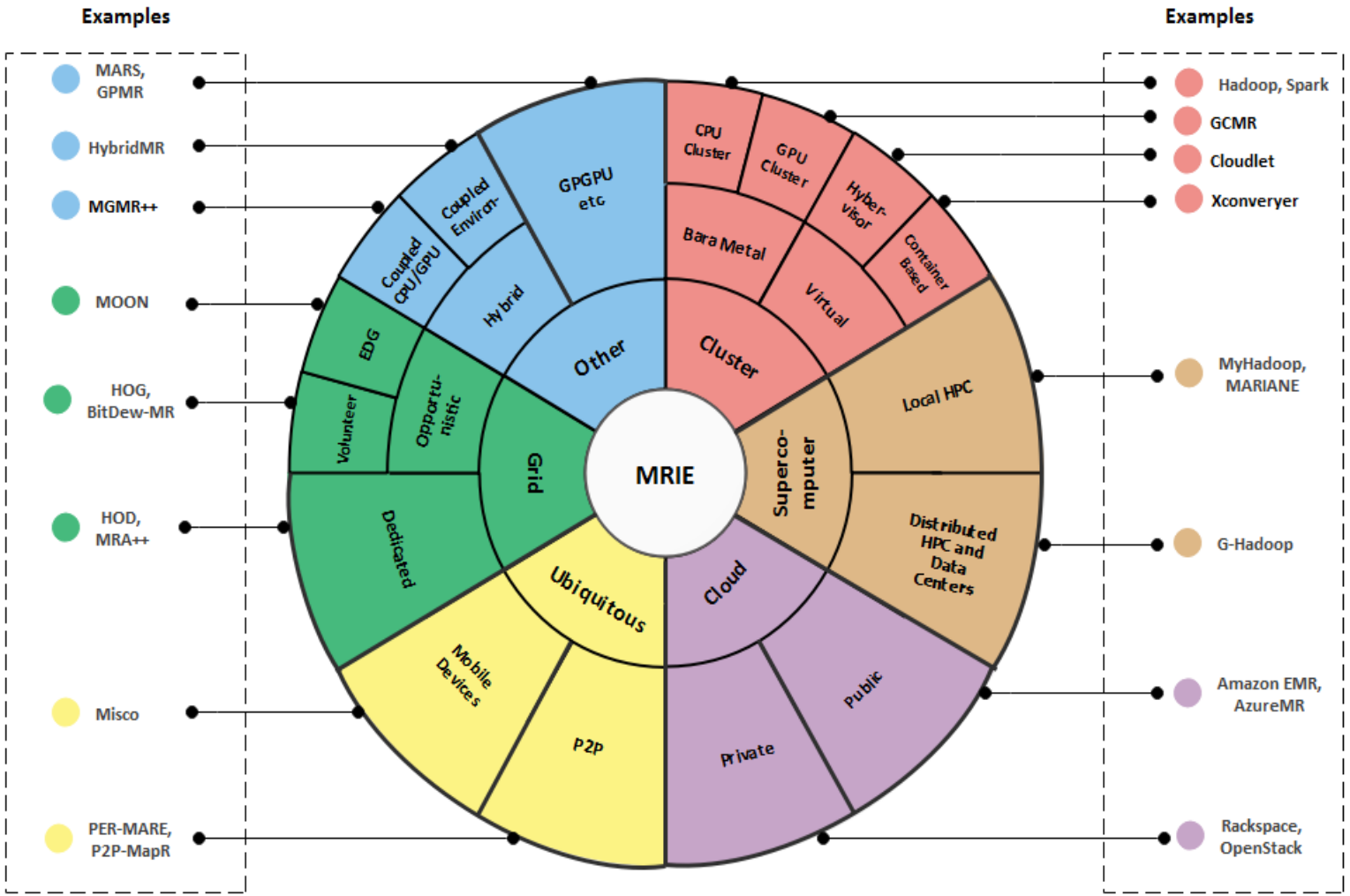
Low Utilization
= Higher cost

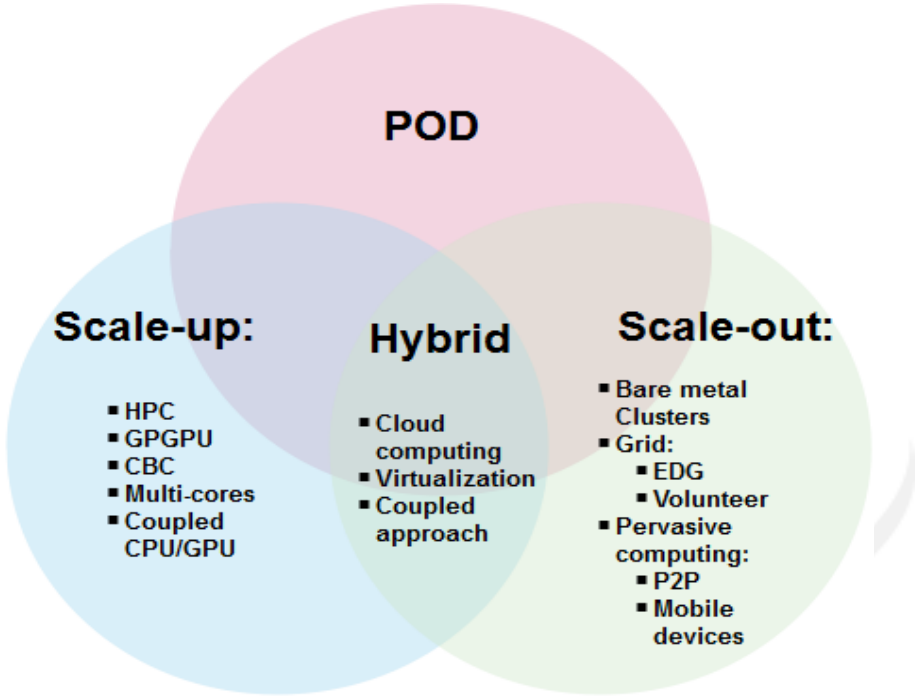
- Workloads tend to be bursty
  - Having periods of heavy utilization (e.g., Quarte reports) followed by periods of light utilization.
    - No one size fit all
- Deploying parallel infrastructures is not an easy task, it can be costly as well.
  - Wasted Computation within an enterprise

Current solutions

- Statically cluster sizing based on peak utilization
- Installing new infrastructure on-demand
  - Easy with Hadoop (scale-out)
  - Though, it's not elastic or auto-scale technique
- Virtual Machines
  - High virtualization costs
  - VM licensing
  - Data movement issues
- New development environments
  - Adaptive and untraditional analytical environments

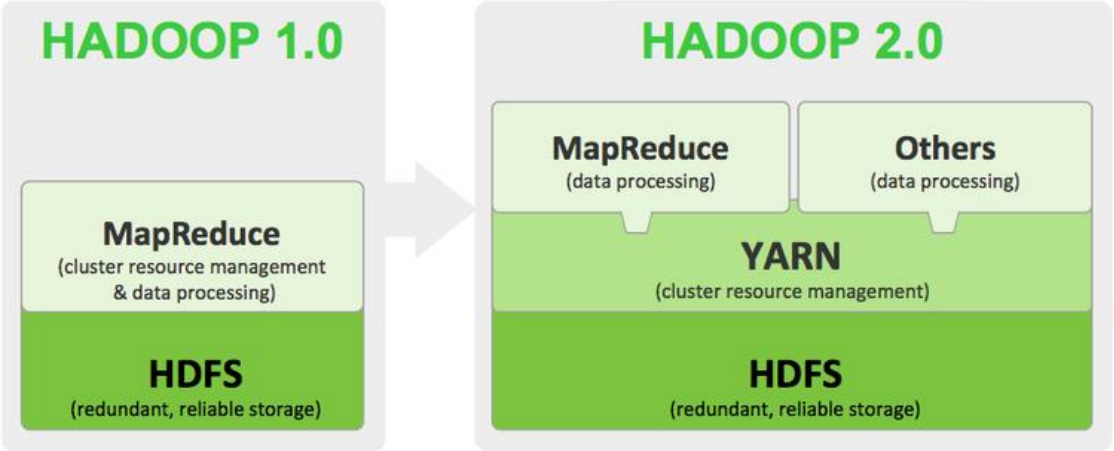
Big Data job execution Environments





Big Data era - The Hadoop Stack

- Hadoop moves to become a BD operating system

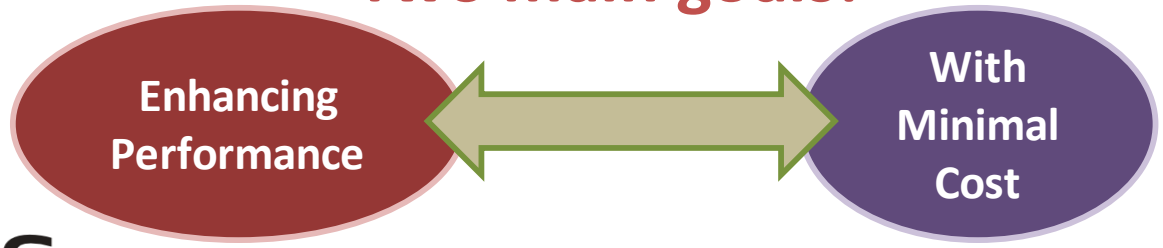


  - Store all your data in one place (HDFS)
  - Interact with that data in multiple ways (YARN Platform + Apps (frameworks))
  - Scale as you go, shared, multi-tenant, secure and multi-workloads type

What problem are we trying to solve?

- Many new complex solutions ⇨ Leads to costly, siloed, under-utilized infrastructure
- Running the fat elephant in a high-throughput computing – Toward efficient, elastic and free auto-scale of Hadoop clusters on-demand
- Creating a BD platform without infrastructure partitioning to accommodate Data-intensive apps efficiently

Two main goals:



Effective usage of resources!!

Current Hadoop Yarn

- Don’t approve dynamically Sharing of cluster resources
- Emphasizes infrastructure silos and static partitioning
- Not elastic or automated scaling
- Don’t take advantage of on premise capabilities

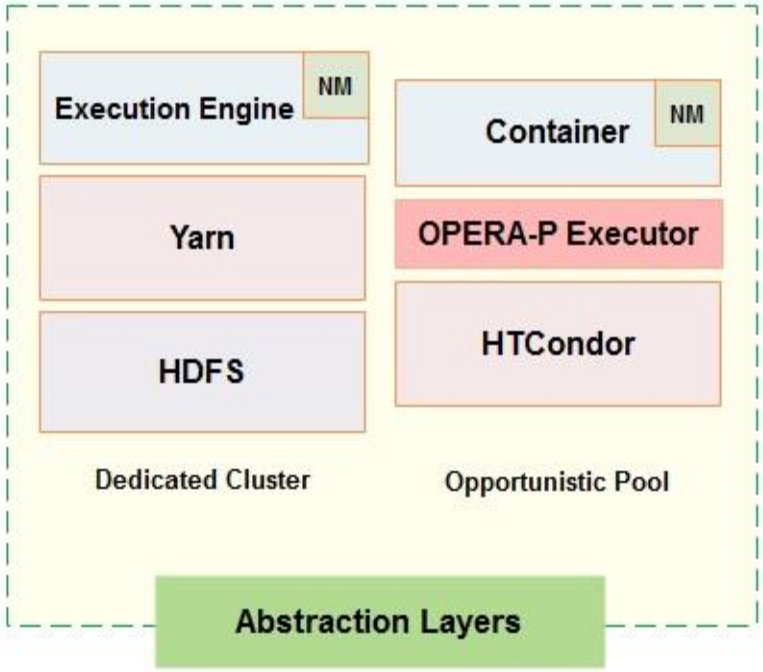
OPERA-P: a BD Platforms Provisioner

- A POD service that automatically and continuously spawns containers in a HTCondor pool
  - According to the available resources
- An opportunistically analytical environment
  - Runs as a standalone instance on each HTCondor machine
  - Represent a new CaaS service
  - Disposable pilot approach → one job - one container
- Extended BD platform
  - No additional cost
  - Better return on infrastructure investments

OPERA-P: a BD Platforms Provisioner

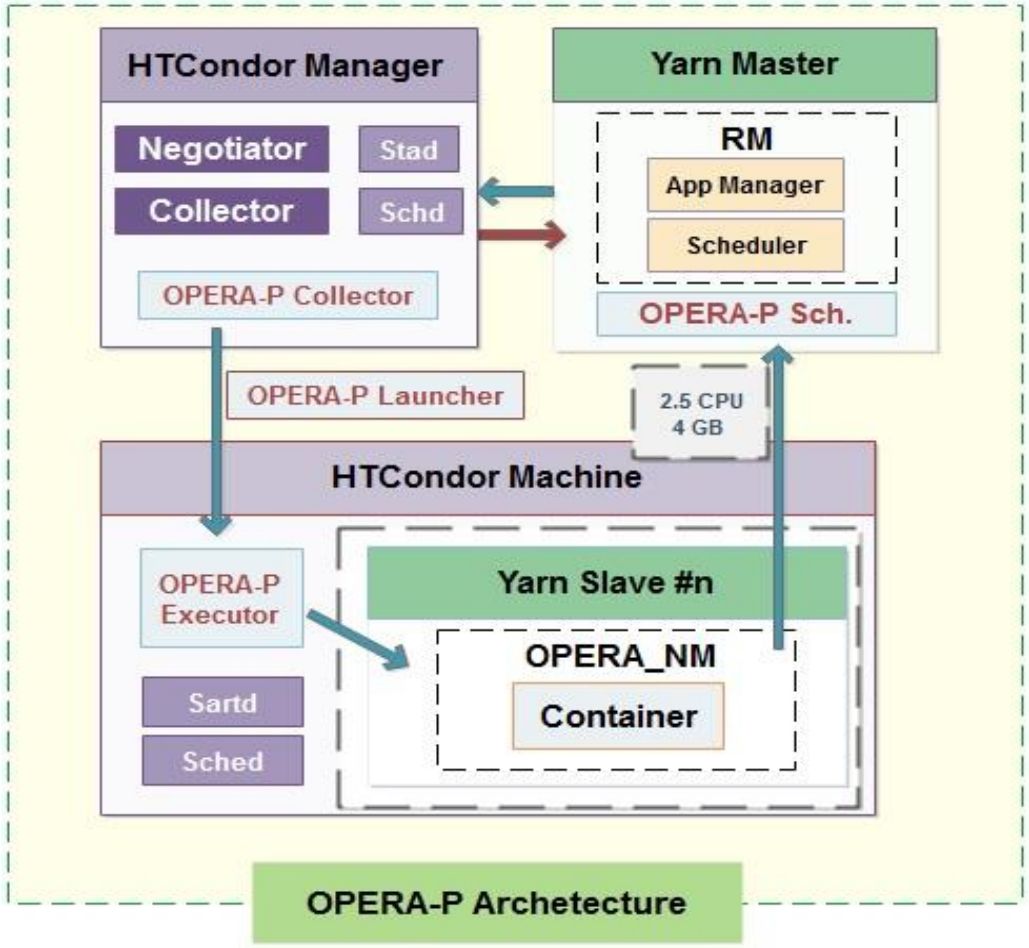
Cont…

- OPERA-P increase agility by provisioning a logical Hadoop clusters by multiple Condor Machines (or at POD)
- This model means that a shared pool of resources can be shared among many BD processing frameworks
  - Each capable of allocating additional resources elastically when needed and releasing them when not.



OPERA-P Architecture

- Resource management framework:
  - second level
    - HTCondor offers resource
    - Framework schedulers accept or reject offered resource
- Lightly used resource allocation:
  - Thanks to HTCondor
  - Elastically provisioning needed frameworks on-demand
- Frameworks Integration
  - Hybrid analytical environment
  - Modify framework scheduler in the container to com- with Yarn master through its API



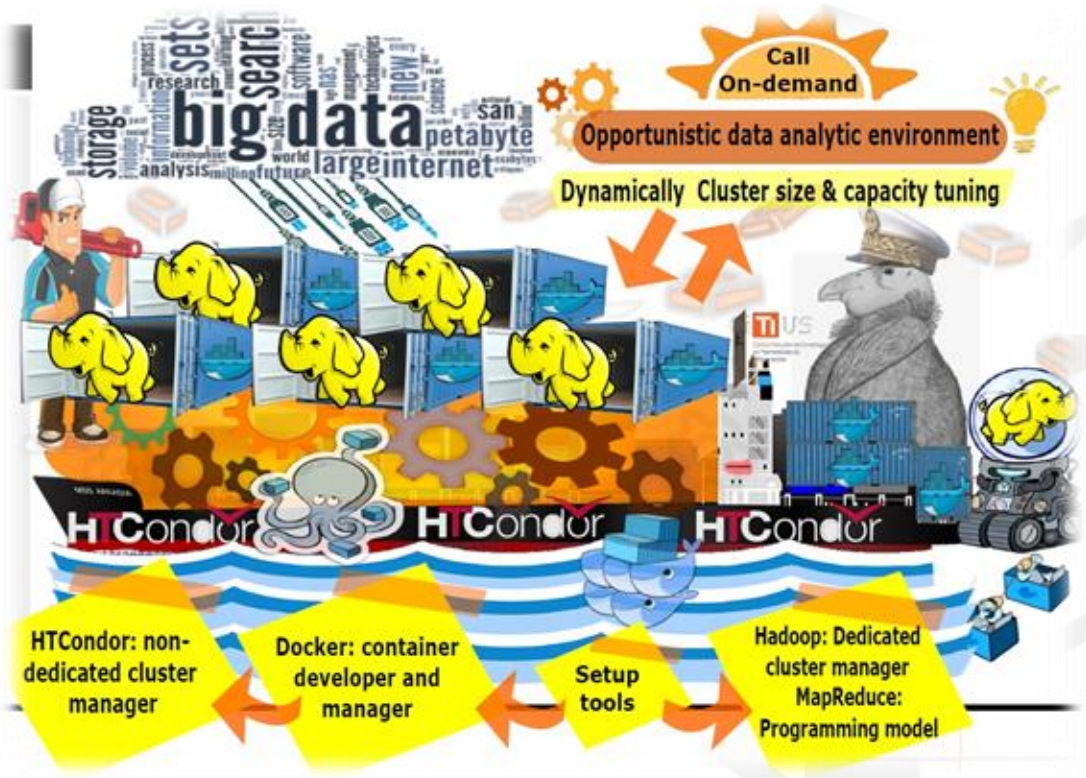
What and What's not OPERA

YES

- ✓ A provisioning service
- ✓ Workflow manager
- ✓ Opp- analytical platform

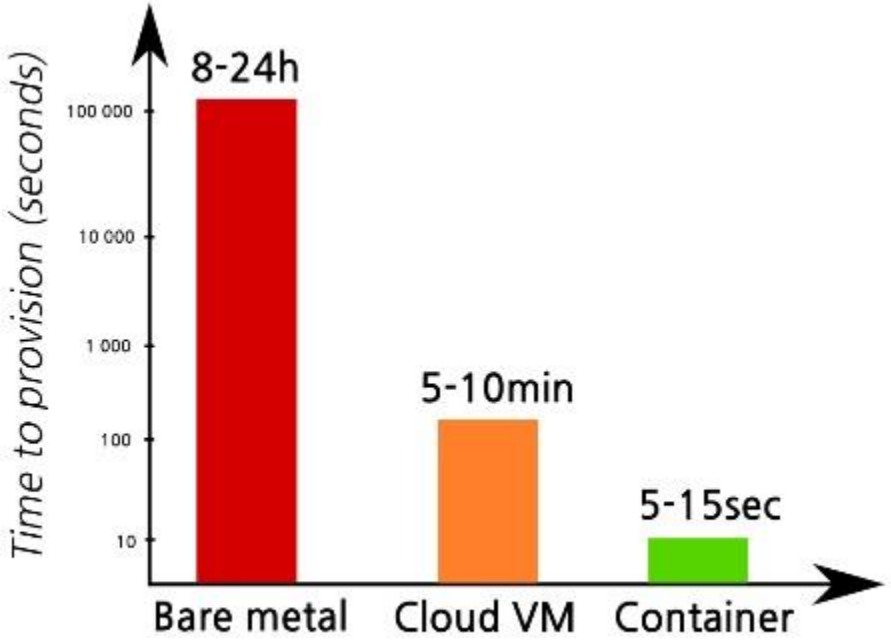
NOPE

- ✗ Container scheduler
- ✗ Hadoop distribution
- ✗ MR implementation



Design considerations

- On-premise prototype
  - Network is reliable
  - Minimal Data transport cost
- Light weight virtualization
  - Near Bare-metal
  - Thanks to Docker containers
- Resource capping and isolation
  - Workloads don't interfere with operational applications



Architecture [1]

1- Introduction to Apache Mesos: https://www.slideshare.net/tomasbart/introduction-to-apache-mesos

Fault tolerance

"design your system for failure"

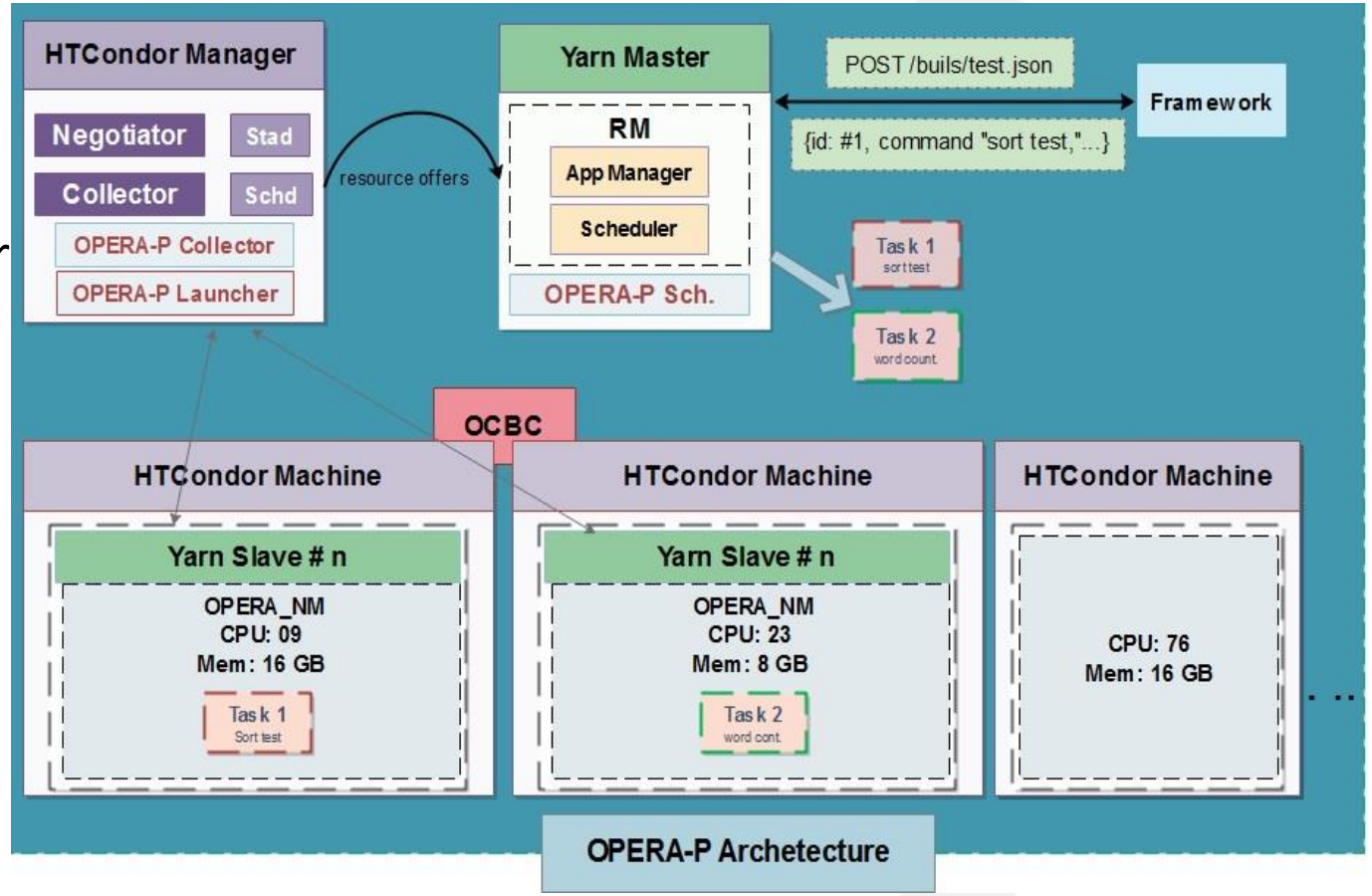
- Every component must have redundancy
  - No single point of failure!
- Tuning the heartbeat
- Majority voting (result checking):
  - Leaving work-done flag until collect two out of three results
  - Directly enhance fault tolerance as well

Use case

EME: An Enhanced Mapreduce Environment

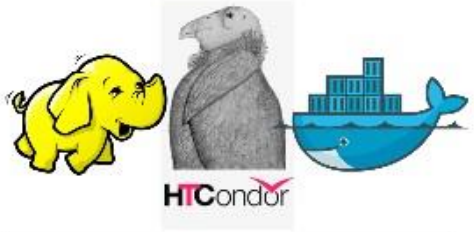
How it works

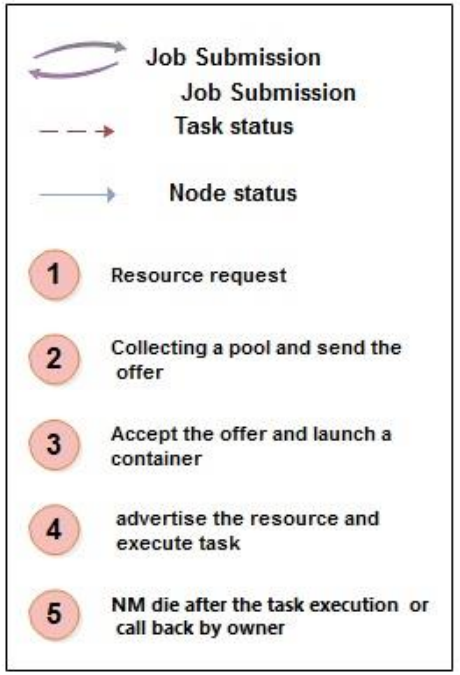
- Very similar to how multiple apps run concurrently on a laptop or smartphone
- New threads are spawned, and more resources are joining the Hadoop cluster as they are needed
- OPERA-P will match the request to incoming HTCondor resource offers and can then consume the resources as it sees fit
- HTCondor, in turn, will pass it on to its worker machines, and launches pilot containers among the underutilized nodes (idle workstations)

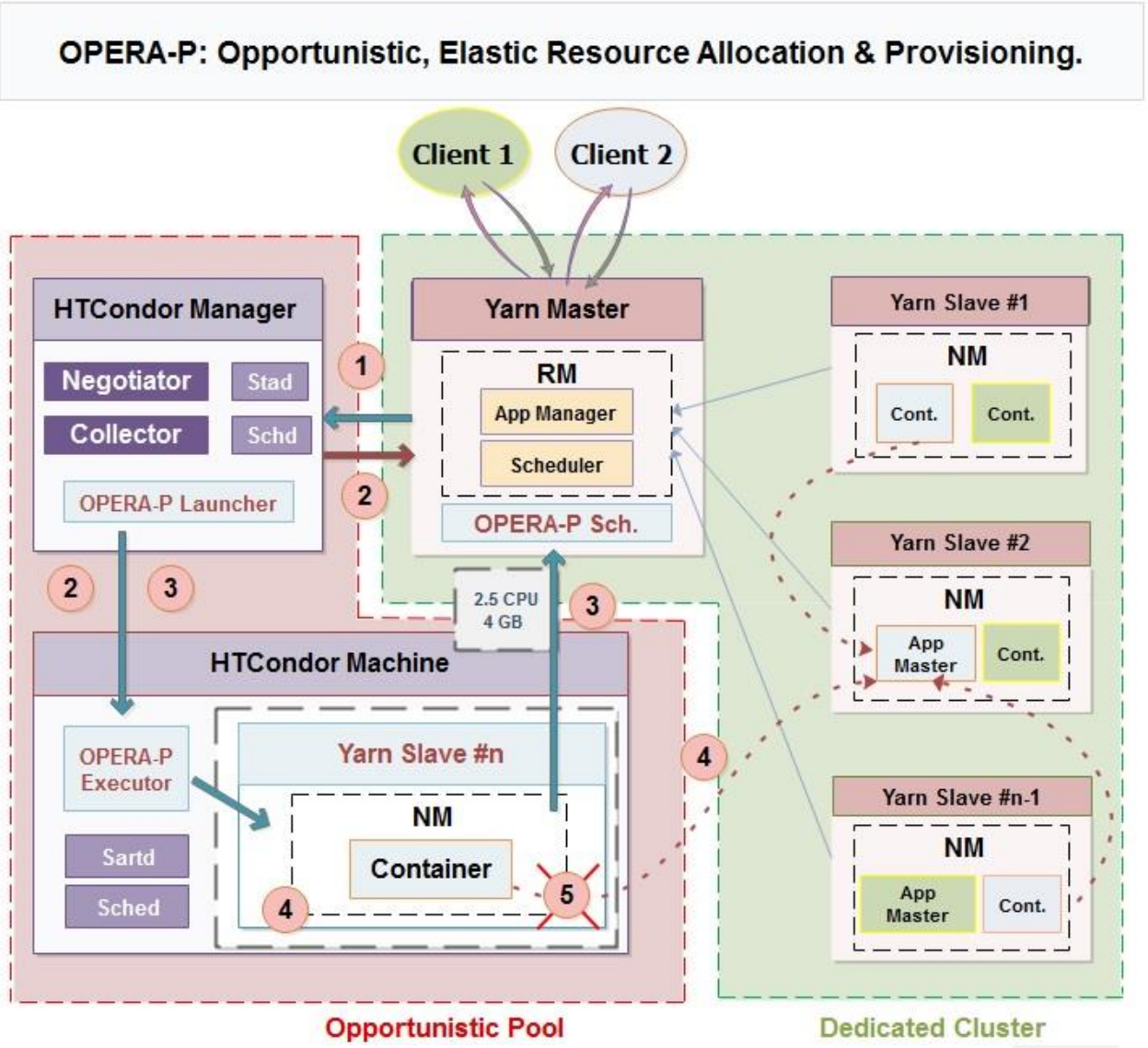


Use case example

CiTIUS
Centro Singular de Investigación en Tecnoloxías da Información



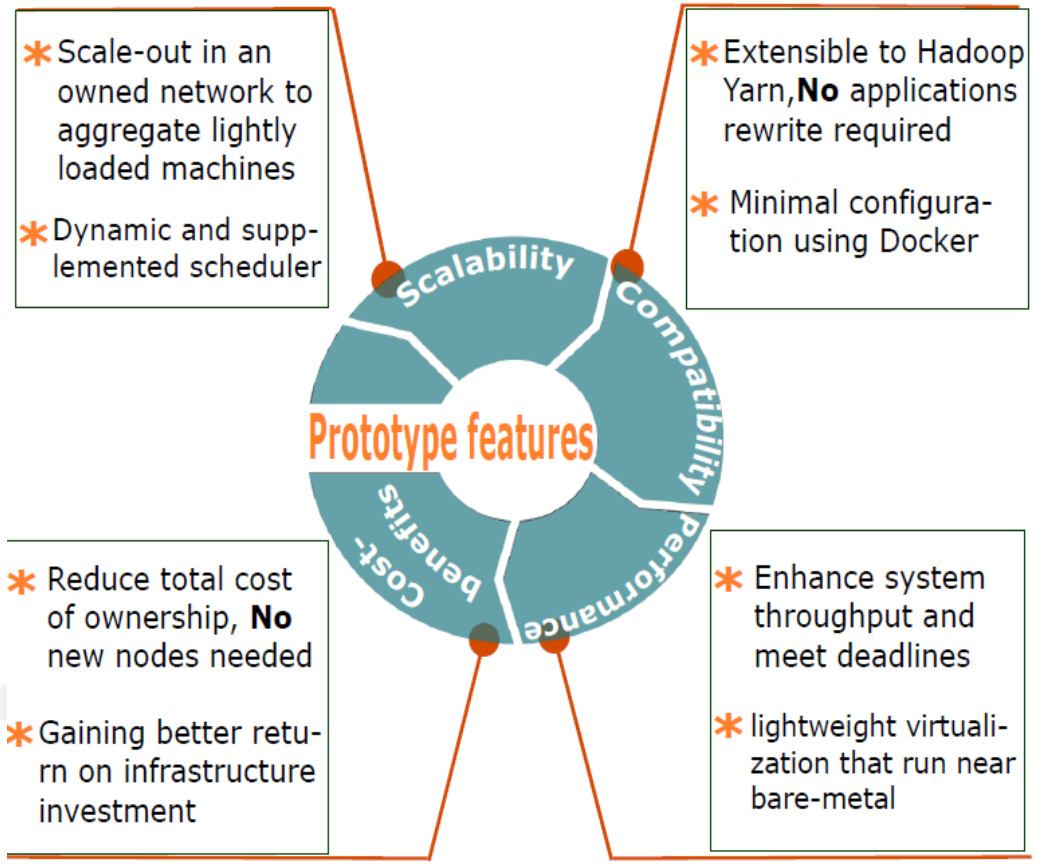




CiTIUS

Feature

- High utilization
  - High throughput , an enhanced performance
- Scalable (Scale-out/up)
  - Automated and elastic on-demand
- Isolation
  - Ensuring that workloads don't interfere with operational applications
- Load balancing
  - Various load patterns, with deferent workloads (job types)
- Minimal cost
  - Installed infrastructure (Capital expander)
  - Operational & Scaling cost



Opportunities

- Exploit more than 2TB of RAM & 65PB HDD available resources at the CiTIUS
- Opportunistic Container-based Cluster (OCBC)
  - A new CaaS service
- A 3D models
  - Running dedicated only
  - Running Opportunistic only
  - Provisioning BD platforms on-demand
- Organizations can deploy, manage, and monitor their BD system, on both dedicated Hadoop cluster and opportunistic HTCondor pool as a single machine

Conclusion & Future

- OPERA-P is an enabling technology to take advantage of leveraging all of available resources within an enterprise or cloud as a single pool of resources
- OPERA-P provides a seamless bridge from the pool of resources available in HTCondor to the YARN tasks that want those resources.
- OPERA prototype can easily be adapted to other resource managers, e.g., Apache Mesos and Docker Swarm
- OPERA-P is an ongoing project, we start prototyping in a virtualized cluster and, when proving its usefulness, test it in a bare-metal environment.

Thank you for your attention

Unidad de Innovación:

feras.awaysheh@usc.es & pablo.caderno@usc.es

citius.usc.es

galicia CiTIUS

UNIÓN EUROPEA

XUNTA DE GALICIA