/ Leveraging cloud unused resources for (Big) data application placement while achieving SLA /

Jean-Emile DARTOIS, **Jalil BOUKHOBZA**, Olivier BARAIS

Outline

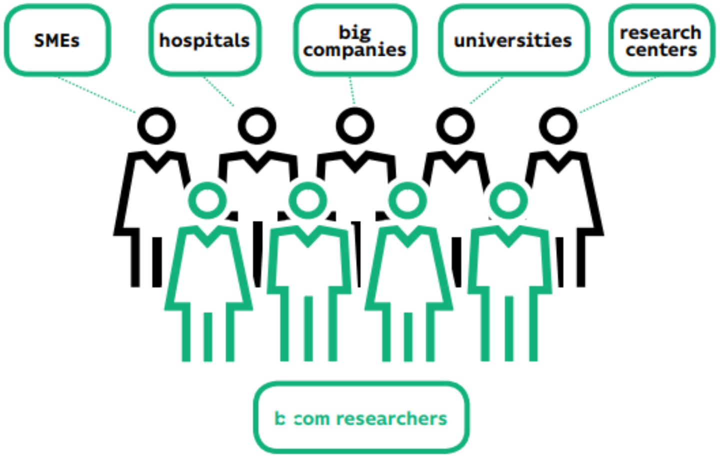
- b<>com
- Context and Motivation
- Problem Statement
- Objectives / Challenges
  - Challenge 1: Modelling I/O interference on SSD for Containers in a Cloud
  - Challenge 2: Using Quantile Regression for resource estimation
  - Challenge 3: MapReduce data placement on ephemeral Cloud resources

- b<>com is a private French innovation center designed to boost innovation in digital technologies.

- A unique co-investment model that provides knowledge, know-how and technology.

b<>com

/sharing expertise/



/3 fields of activity/

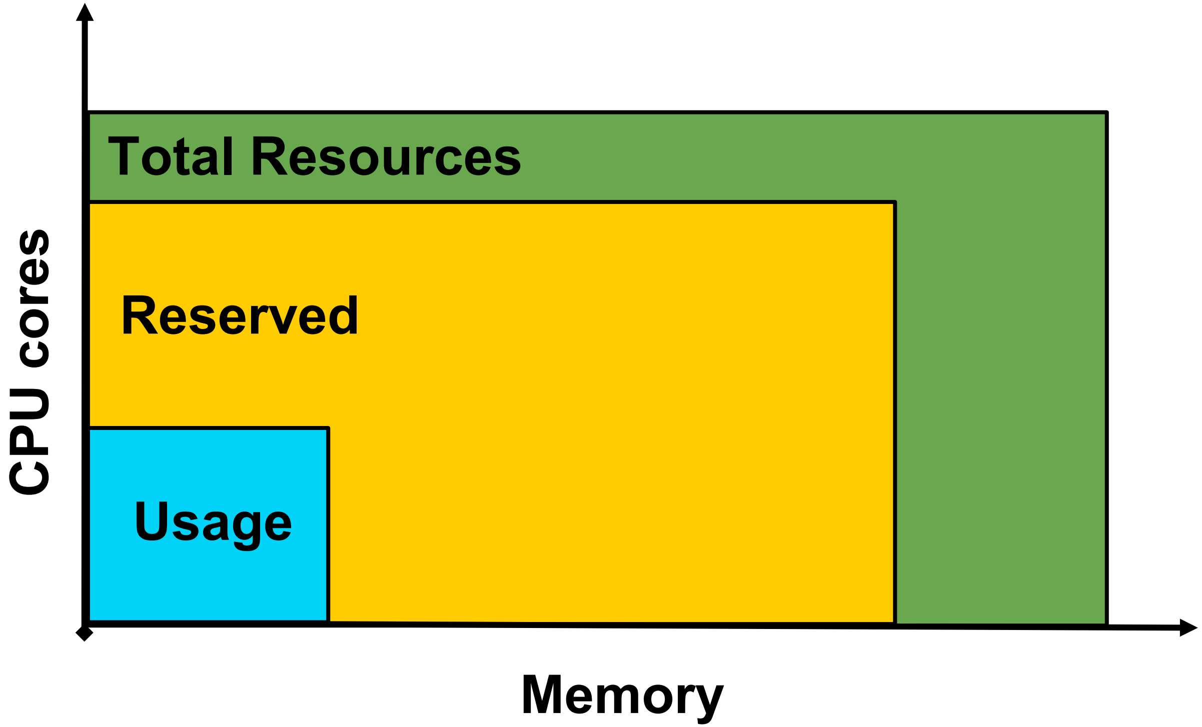
[key figures]

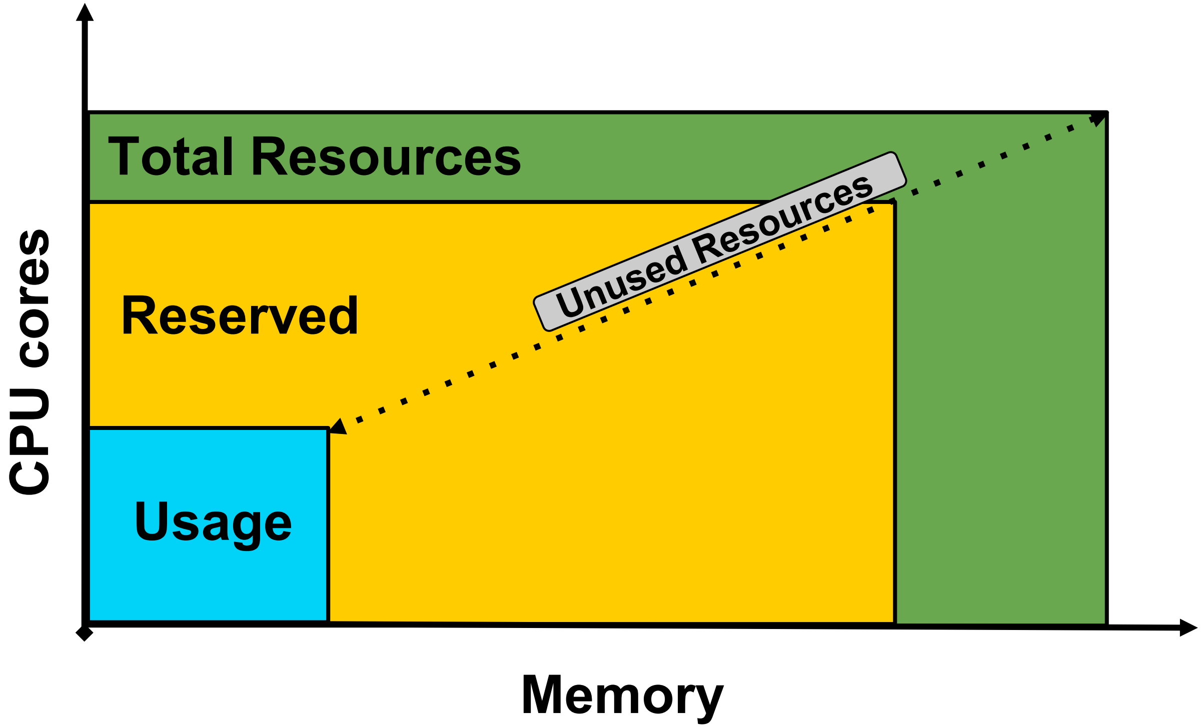
2012	year of creation	**15**	technological solutions
7000	m^2 scientific campus	**200**	papers & reports
250	people (25 PhD students)	**10**	european projects
14	nationalities	**66**	transferred technologies
35	shareholders	**4**	sites (3 in Western France, 1 in Paris)

Context

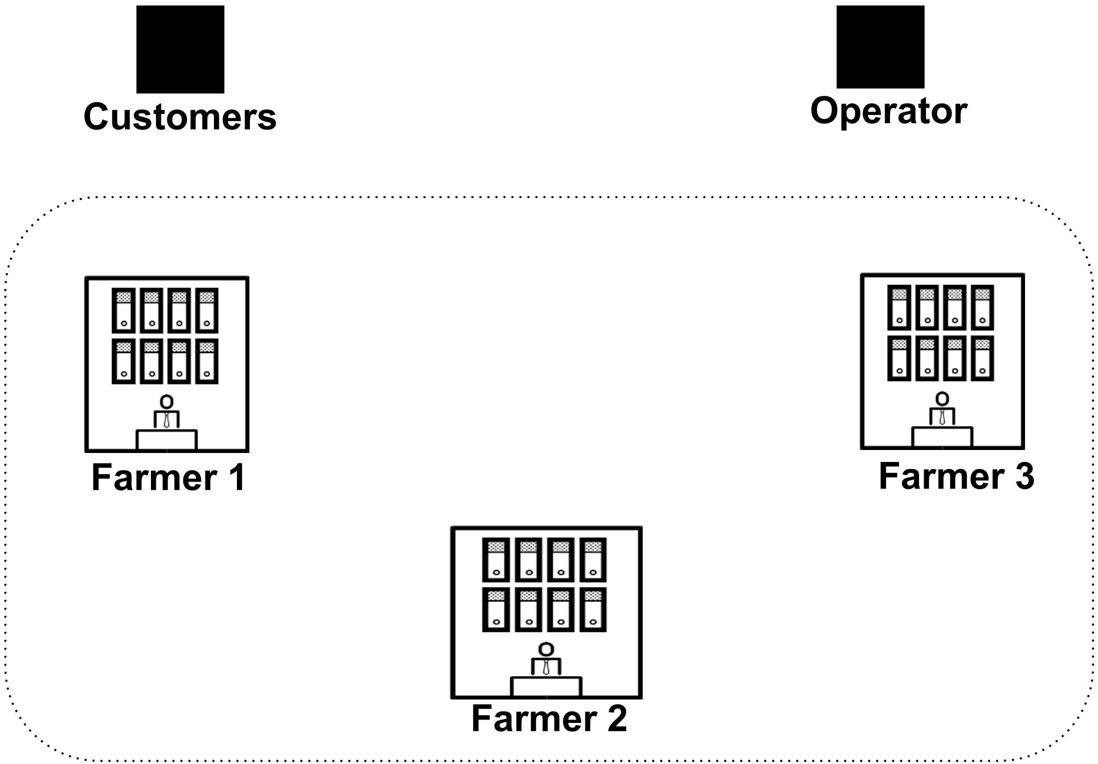
- **Past project: Watcher** https://wiki.openstack.org/wiki/Watcher
  - Flexible and scalable **resource optimization framework**
  - Provide a pluggable architecture for **optimization algorithms**

- **Current project: Falcon**
  - Most data-centers have a **low resource usage** (20% in case of CPU [M. Carvalho and al, 2014])
  - Allow organizations to **resell their computing infrastructure unused resources**

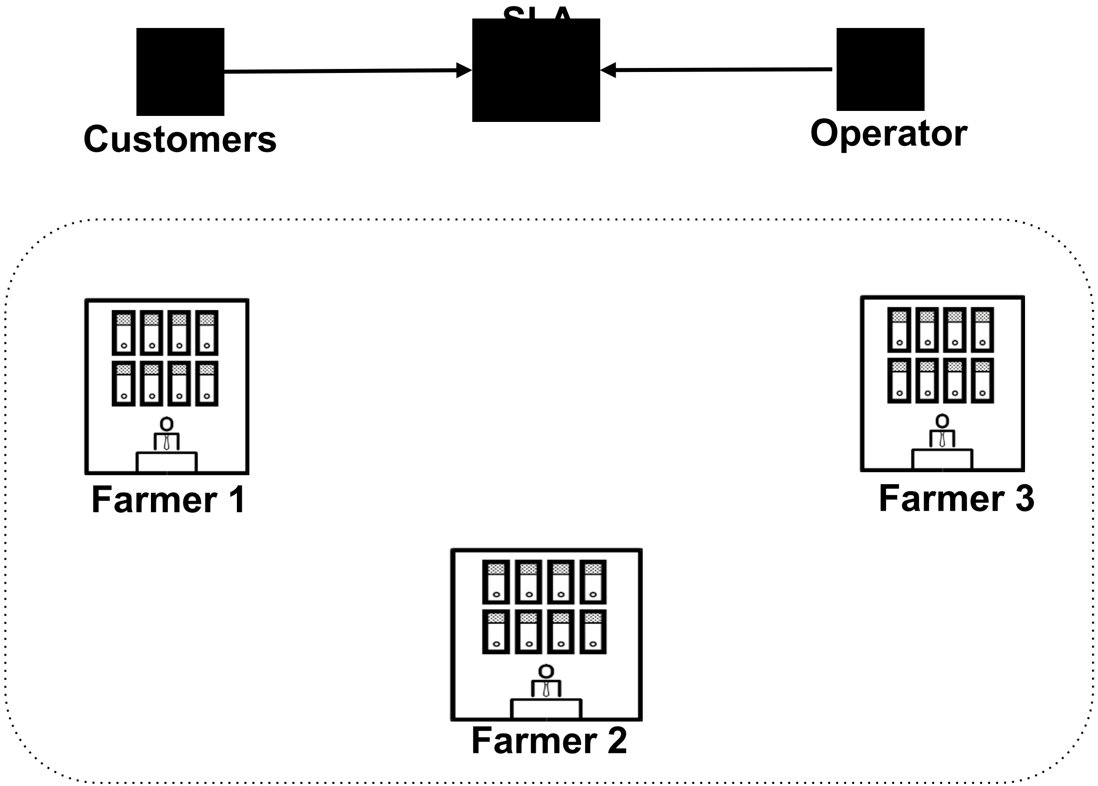

Total Resources
Reserved
Usage
CPU cores
Memory


Total Resources
Reserved
Usage
Unused Resources
CPU cores
Memory

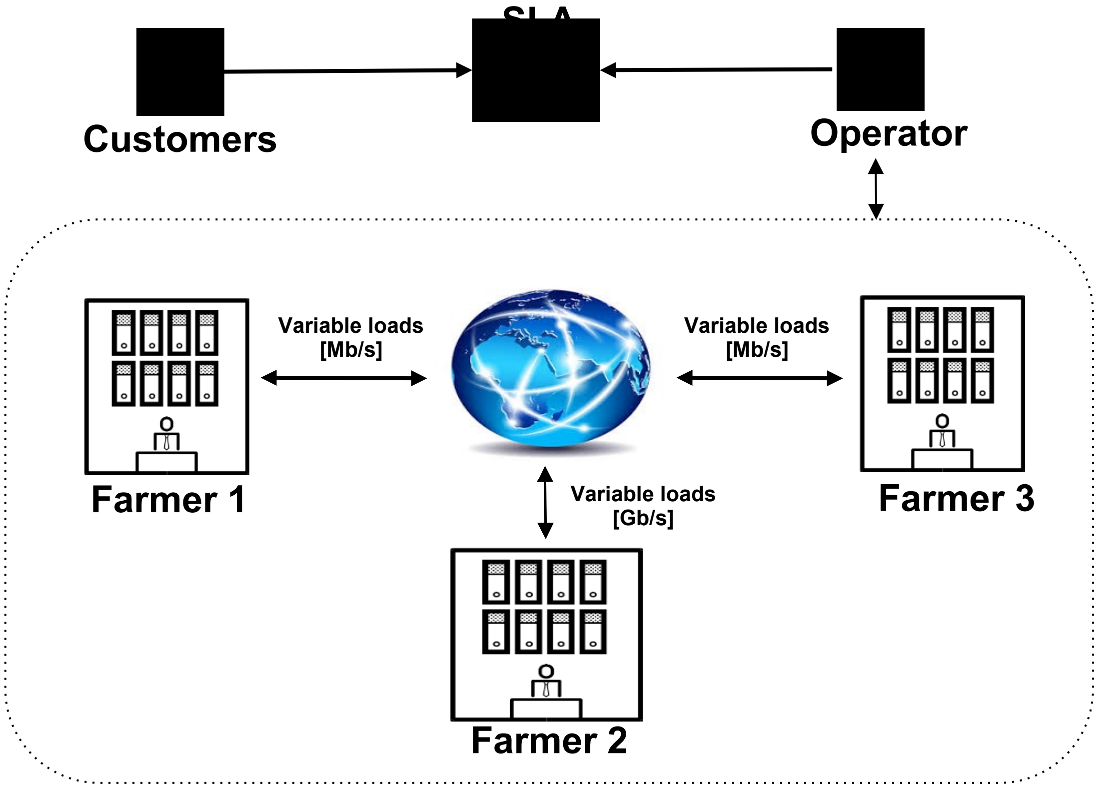
Problem Statement



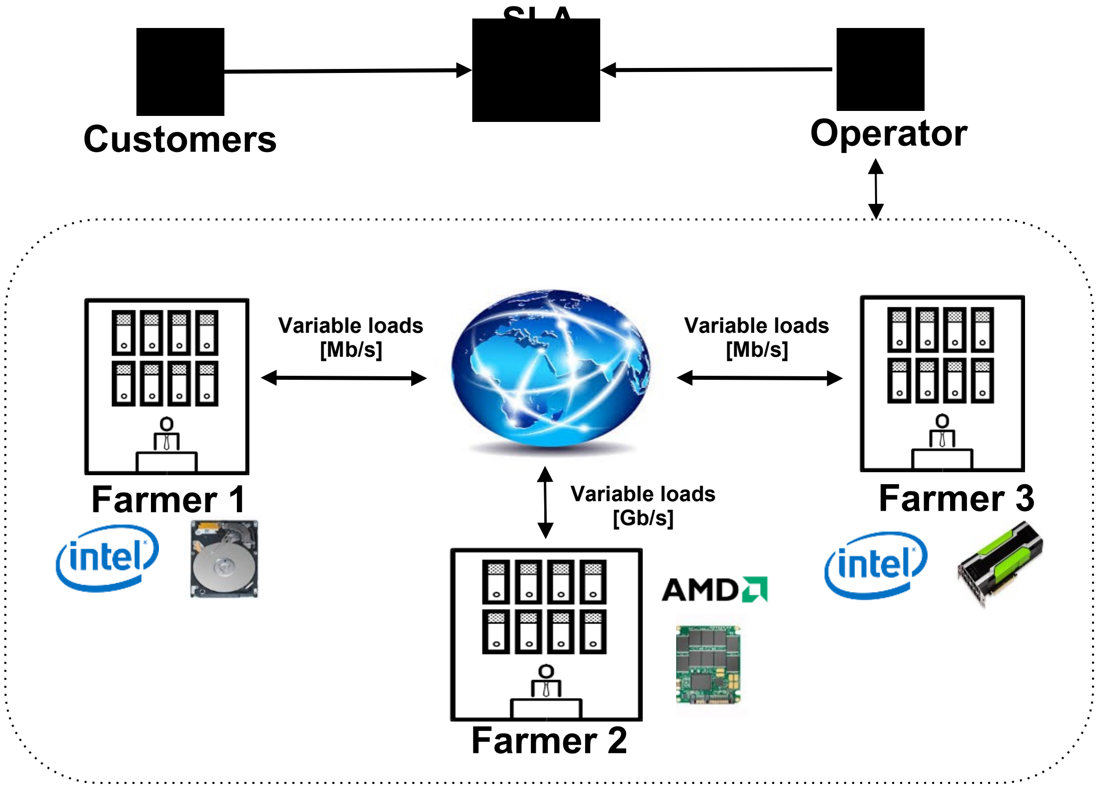
Problem Statement



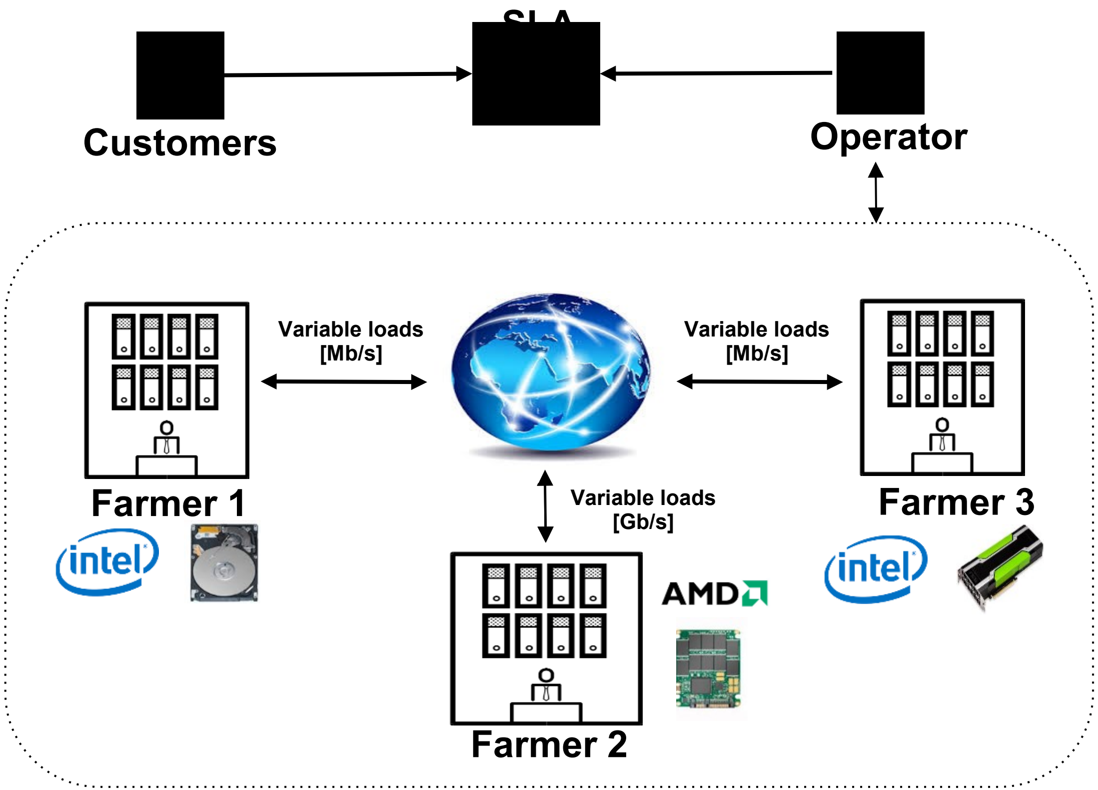
Problem Statement



Problem Statement

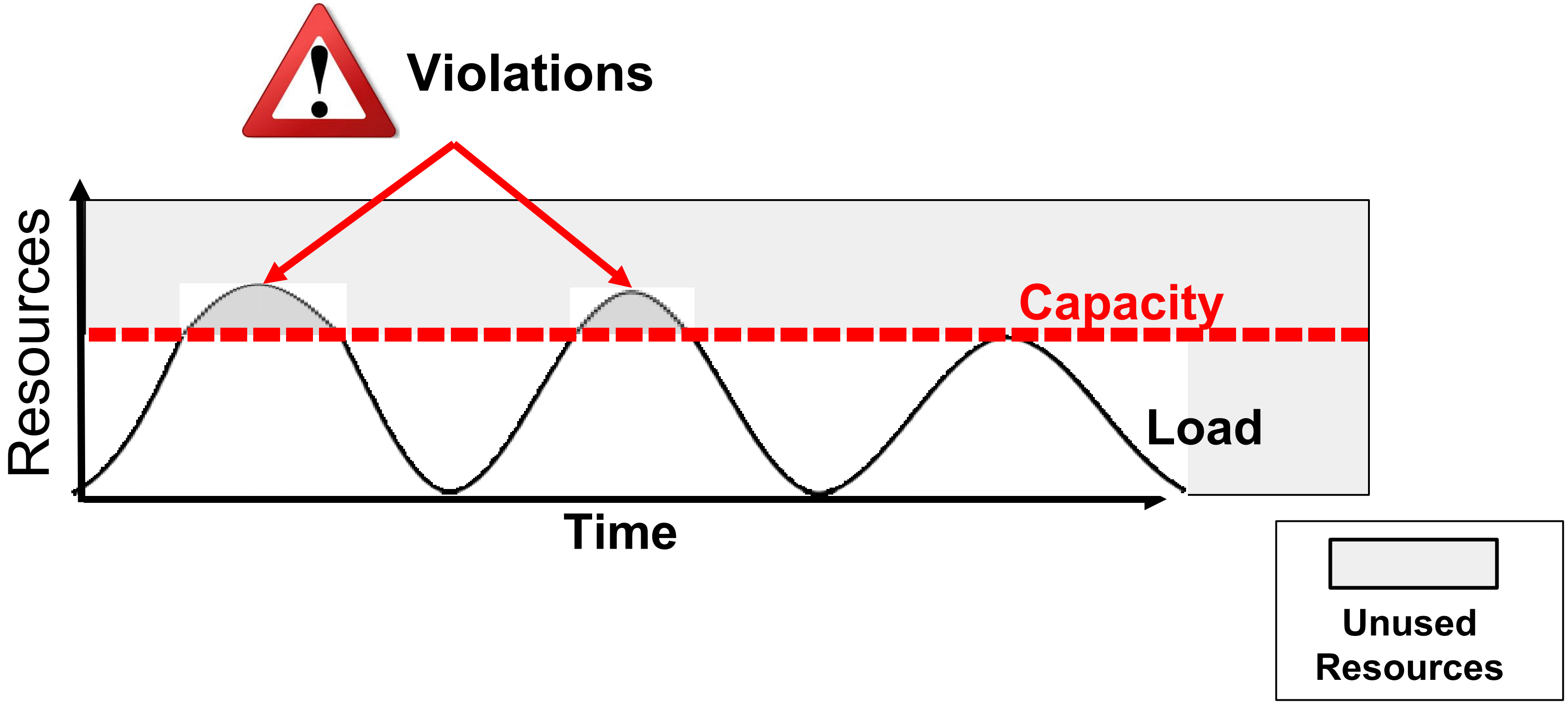


Problem Statement



How to decide/determine how much resource to sell?
How to guarantee SLA?

Problem Statement



b<>com

Challenges

(1) Estimating **the maximum performance reachable by the system** to determine the real system capacity

(2) Estimating **future unused resources**

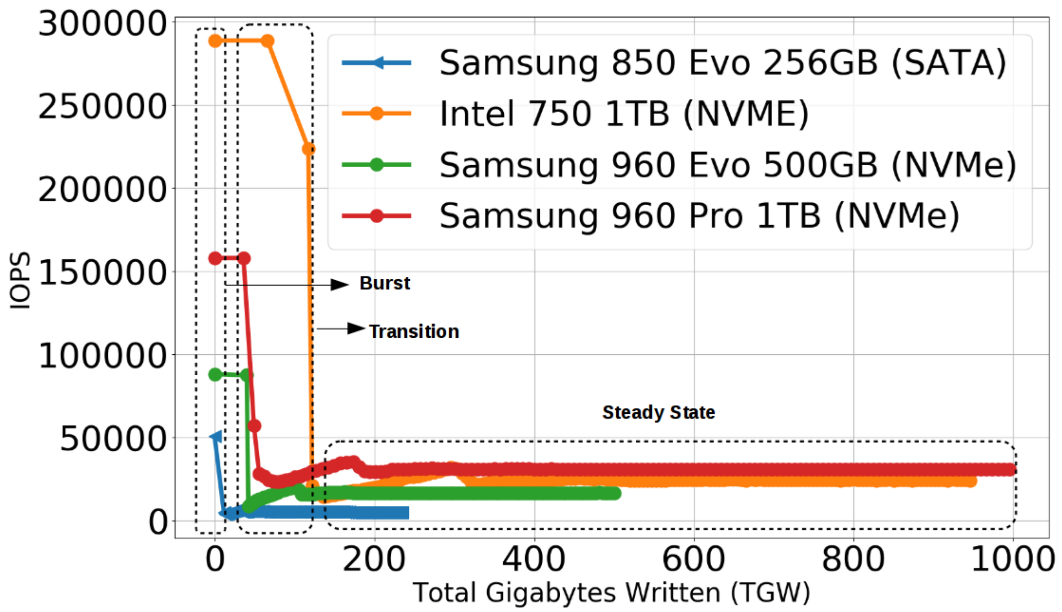
(3) Designing **strategies to deploy applications** on top of unused resources

Challenge #1 :

Investigating Machine Learning Algorithms for Modeling I/O Interference on SSD for Container-based Virtualization

b com

Background - SSD I/O interference



- **5x to 11x performance drop** (far below values reported on datasheets)
- **Performance is oscillating among three states** according to the sustained I/O traffic

Problem Statement

- We define **three types of I/O interference** on a given application I/O workload:
  - Interference due to SSD **internal mechanisms** (e.g. GC,wear leveling)
  - Interference due to **kernel I/O software stack** (e.g. page cache read-ahead and I/O scheduling)
  - Interference due to **co-hosted applications workloads**

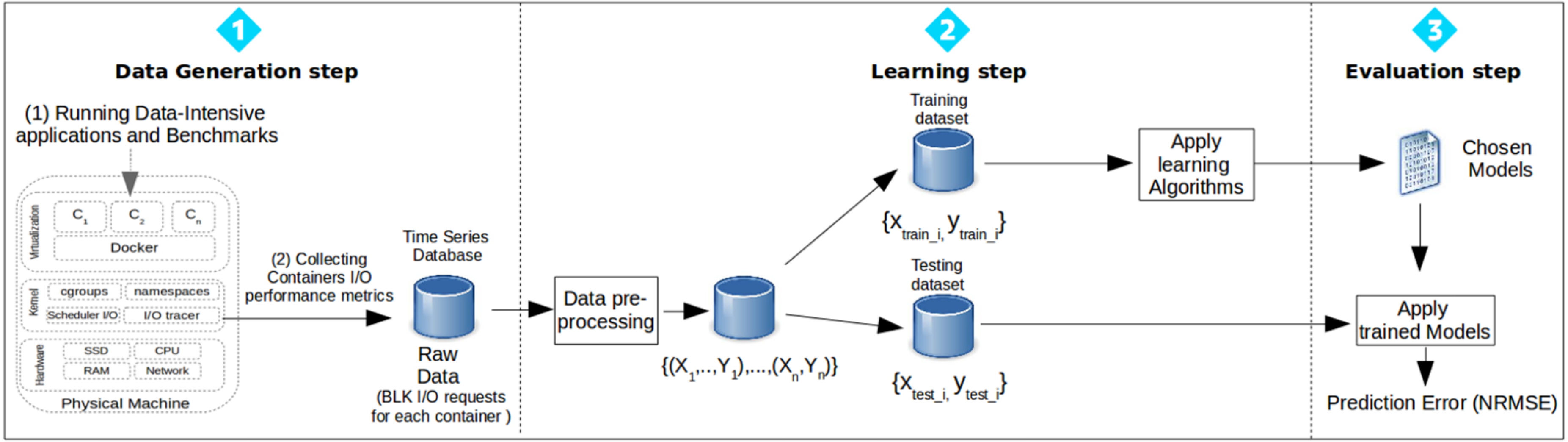
- **How to manage, prevent and solve I/O interference to guarantee SLA?**

Our Contribution

- A **methodology to build predictive models** for SSD I/O performance to solve I/O interference issues in container based clouds.
- To **explore** different **machine learning algorithms** for modeling I/O interference
- Often the hardest part of solving a machine learning problem is to **find the right algorithm** and the right **features /hyparameters.**

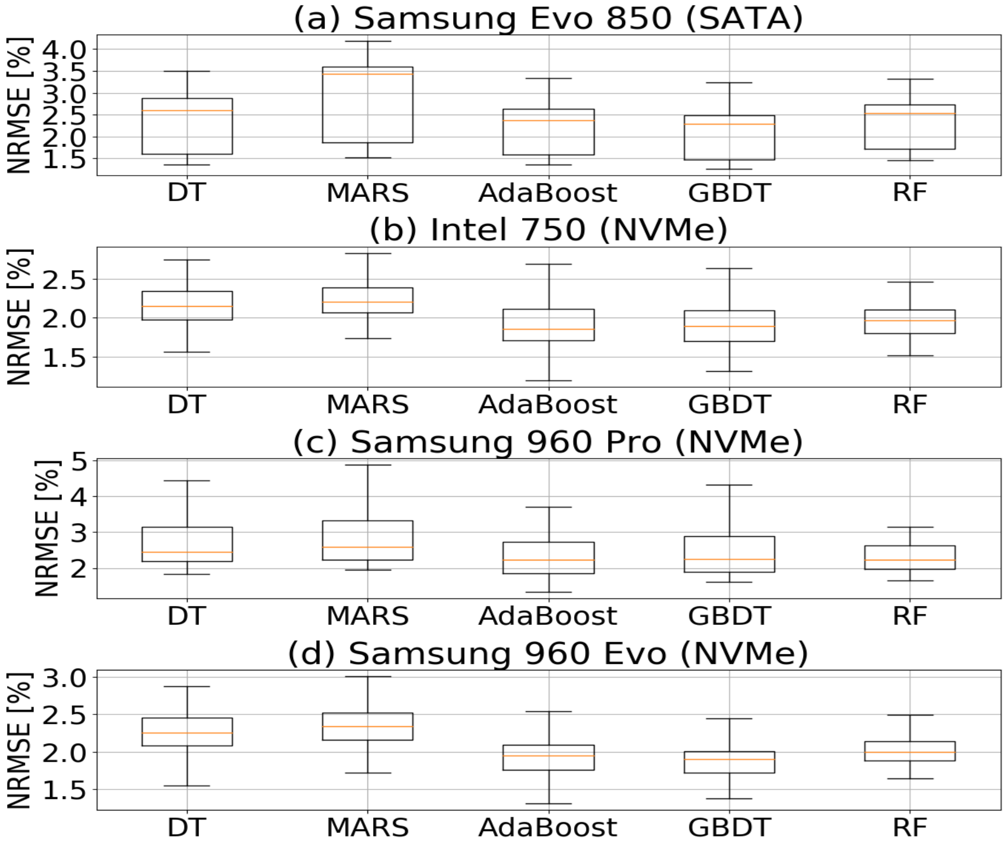
b<>com

Overall Approach



Name	Category	Description
web	Server application	N-tiers web application
email	Server application	Email server
fileserver	Server application	File server
video	Multimedia processing	H.264 video transcoding
freqmine	Data mining	Frequent itemset mining
compile	Software development	Linux kernel compilation
micro-benchmark	Synthetic Benchmark	I/O workload generator

Evaluation



- GDBT, AdaBoost and RF gave the best with an NRMSE of 2.5%

- The ranking of the tested algorithms was the same regardless of the SSD used.

- Adaboost, GDBT and RF provided the smallest dispersion proving there robustness to a changing I/O.

- We used fixed hyperparameters to tune RF and DT. This makes them simpler to use

Challenge 1- Conclusion

- **Predicting I/O performance** in container-based virtualization is necessary to **guarantee SLO**
- **Machine learning is a relevant approach to predict SSD I/O performance in a container-based virtualization**
- **We advise to use Random Forest.**
- We will use our approach to develop a strategy to **improve container placement** in cloud infrastructure in order to avoid performance issues before users are impacted.

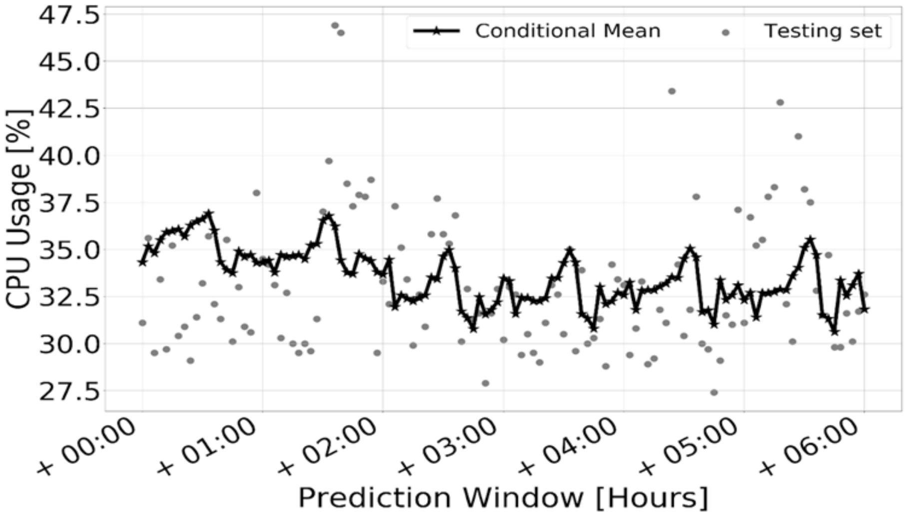
Challenge #2:

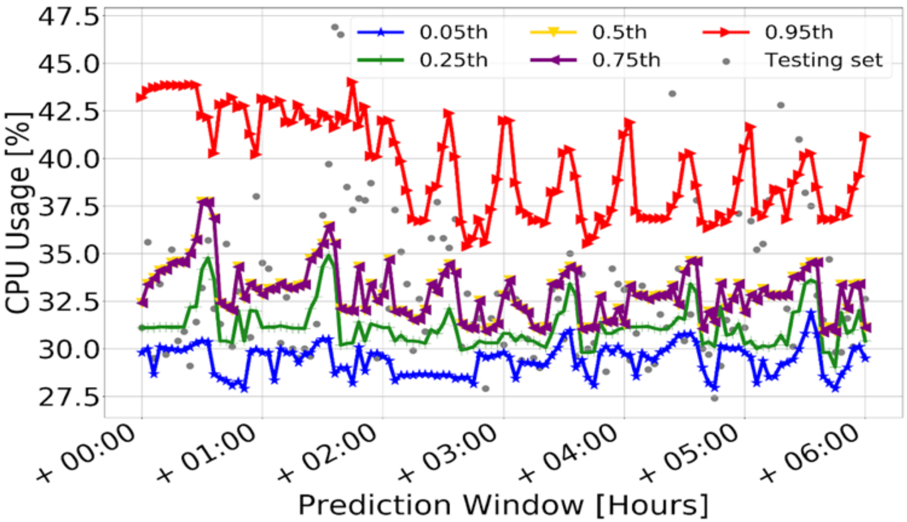
Using Quantile Regression for Reclaiming Unused Cloud Resources while achieving SLA

[IEEE CloudCom,2018]

- Estimate **Future Demand** to provide **SLA guarantees**
- Machine learning could be used to **estimate future unused resources**
- Vision: Quantile regression could provide a **flexible trade-off between the potential amount of resources to reclaim and the risk of SLA violations**

Background - Quantile Regression





The **direct** approach consists in minimizing a sum of asymmetrically weighted absolute residuals based on:

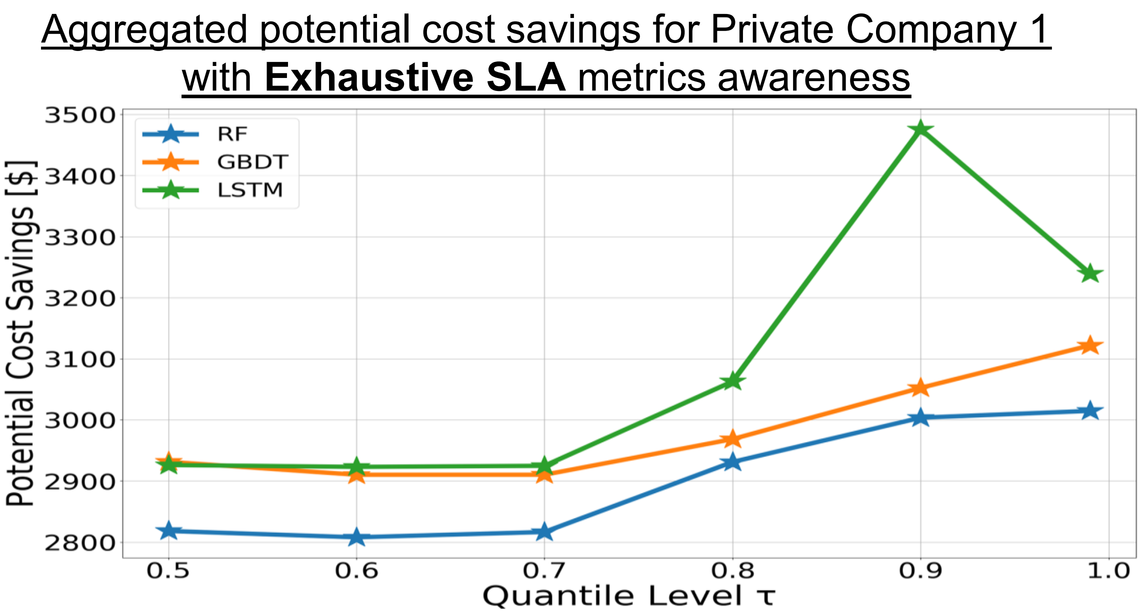
$$q_\tau(x) = \arg\min_{\mu(x)} \mathbb{E}\left(\rho_\tau(Y - \mu(x)) | X = x\right)$$

$$\rho_\tau(u) = \begin{cases} \tau u & u \geq 0 \\ (\tau - 1)u & u < 0 \end{cases}$$

The **indirect** approach is performed in two steps, the first one estimates the conditional CDF. Then, the τ th conditional quantile of Y given X = x is obtained via inversion of the estimated conditional CDF [23] based on:

$$q_\tau(x) = F^{-1}(\tau | x)$$

Potential cost savings



- **All learning algorithms:**
  - **Increase potential cost savings** with the increase of τ
  - **Increase up to 20%** cost savings compared to **median-estimation** based approach (τ=0.5)
- When **τ > 0.9** the **reduction of unused resources** is **higher** than **the decrease of SLA violations**

b<>com

Challenge 2 - Conclusion

- **Flexibility: Regression Quantile** is useful to **maximize cost savings** (**up to 20%** compared to traditional approaches) for the **4 datasets**

- **Exhaustivity**: Only CPU leads to no savings

- **Robustness**: RF is the most robust algorithm but LSTM performs better with potential cost savings (**Underestimation** is not **penalized)**

- **Applicability**: GBDT smallest overhead and LSTM had the highest overheads.

Challenge #3 :

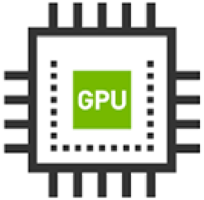
Cuckoo: Opportunistic MapReduce on Ephemeral and Heterogeneous Cloud Resources

(On-going)

Introduction

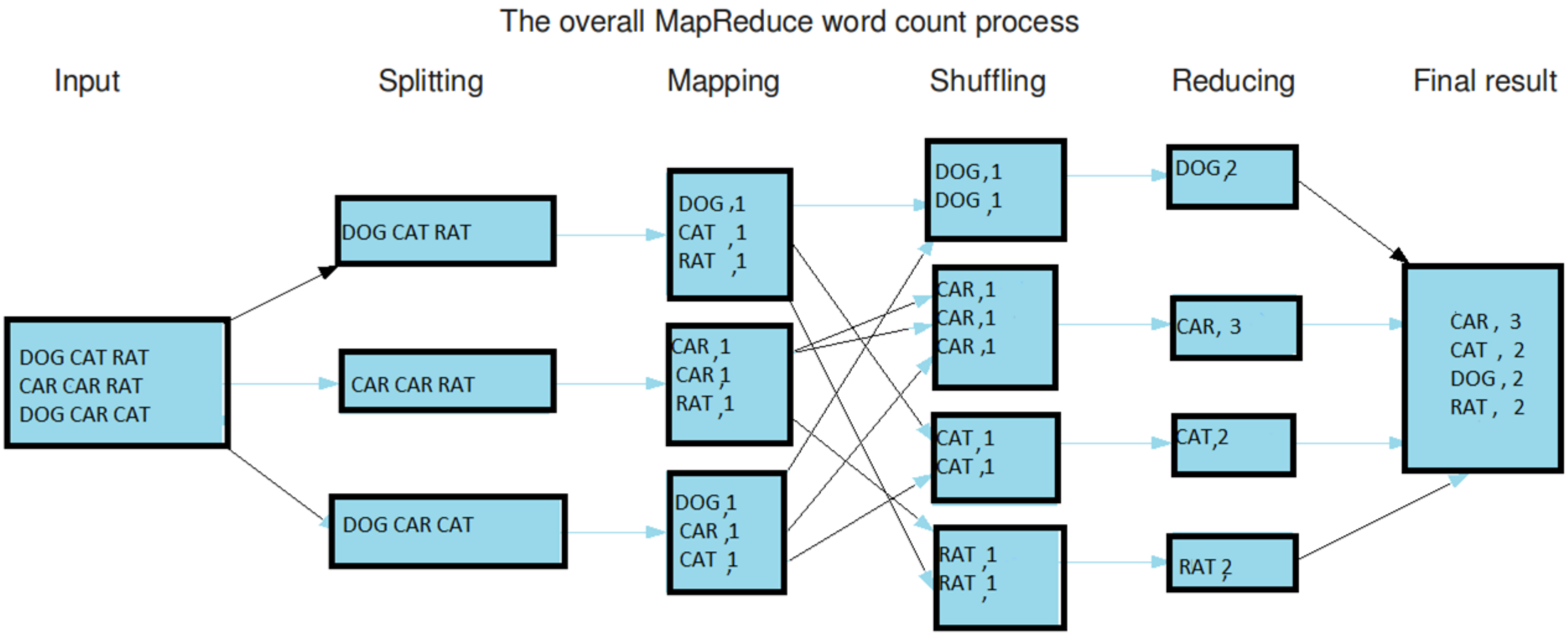
- **Big data** in cloud computing is a growing trend [El-seoud, 2017]

- *Big data processing* demand a considerable amounts of cloud resources and are **costly** [Montero C, 2014]



- Opportunity to **reduce costs** by processing **big data on top of unused resources**

Background - Hadoop



source:http://a4academics.com/tutorials/83-hadoop/840-map-reduce-architecture

Problem Statement

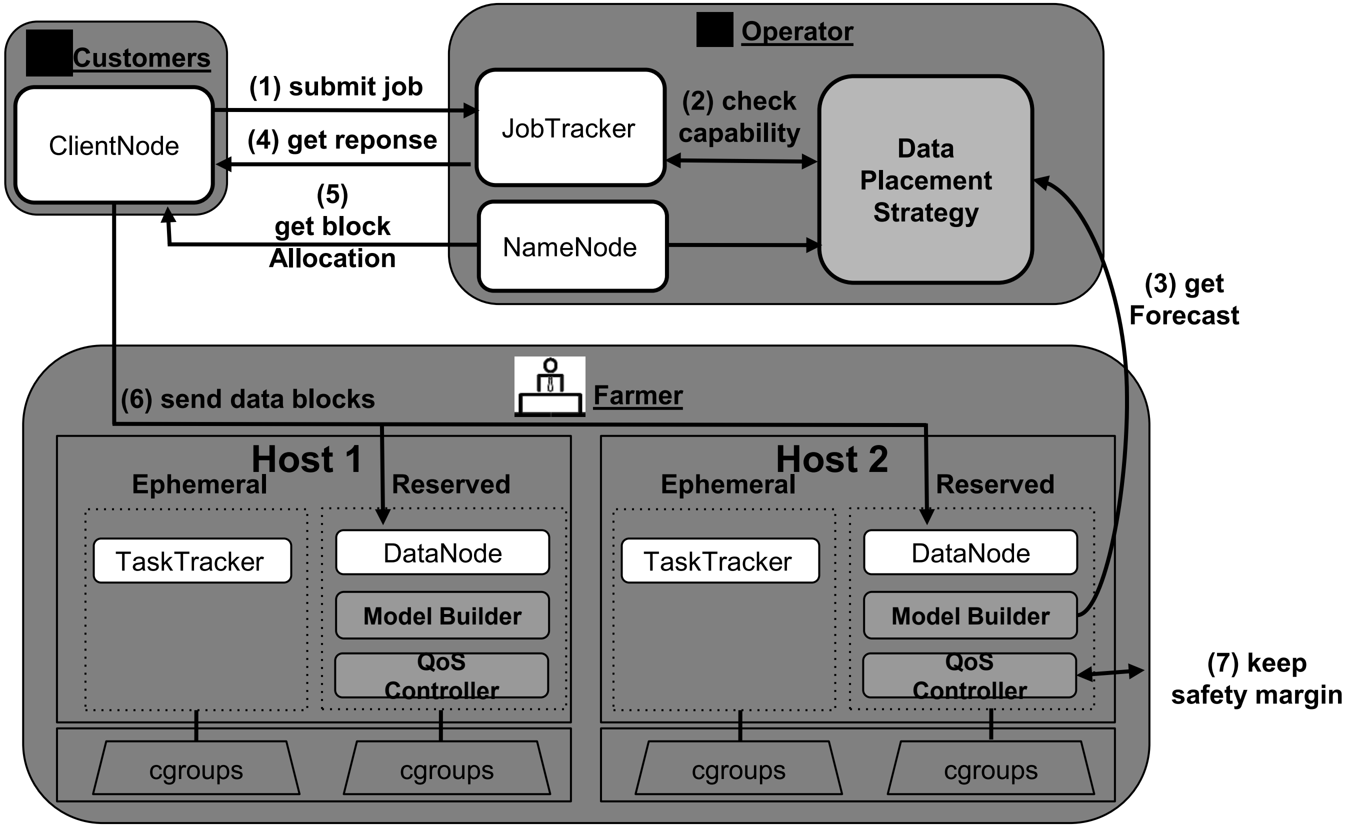
- **Cloud Heterogeneity**
  - processing capabilites
- **Resources Volatility**
  - overestimation or underestimate of the future unused resources may lead to performance degration (*e.g.*, remote or speculative tasks)
- **Resources Isolation**
  - reclaimed resources **must be released or evicted in case of starvation** whenever farmers require them again

➢ **How to minimize the number of recomputations? How to avoid any interference on farmer workloads?**

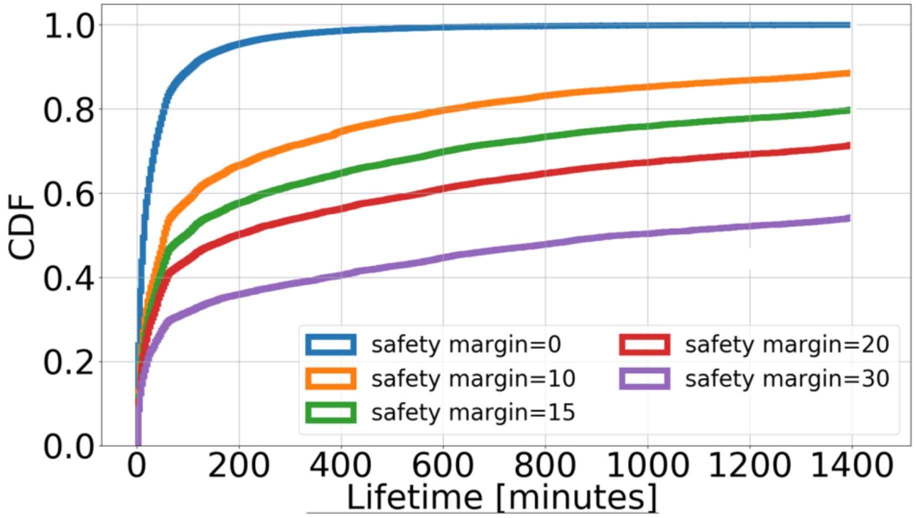
Our Contribution

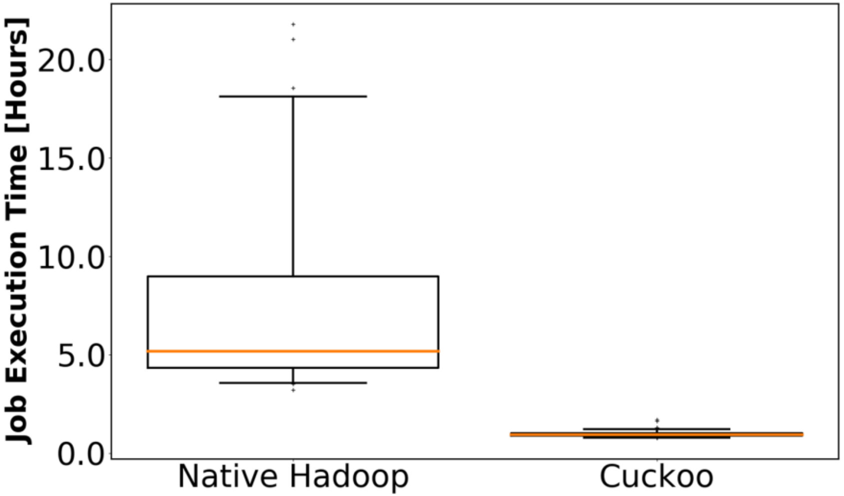
- **Data Placement Strategy** based on weighted-Round-Robin algorithm:
  - Processing capacities → heterogeneity
  - Estimating Future unused resources → volatility
- A **QoS Controller** to deal with underestimation or overestimation of the available unused resources → isolation
  - Keeping a portion of CPU and memory unused to prevent interference
  - Adjusting dynamically containers resource limits

Overall approach



Evaluation (preliminary)





- Increase the ephemeral container lifetimes up to median ~900 minutes with a safety margin of 30% but at a cost of less resources
- **Cuckoo outperforms Hadoop by up to 4.4x**

THANKS/MERCI

Any questions ?