Computational Costs of Cloud Computing: Challenges and Solutions in High Performance Computing

Lúcia Drummond
Universidade Federal Fluminense

Outline

1. Cloud Computing and High Performance Computing
2. Approaches for Reducing Financial Costs in Clouds
   - Provider side
   - User Side
3. Challenges when using Spot VMs to reduce costs
4. Proposed Scheduling on Spots Strategies
5. A Case study: A Sequence Alignment Problem
6. Analysis of Storage Services for recording checkpoints

Conclusions and Future Works

Cloud Computing

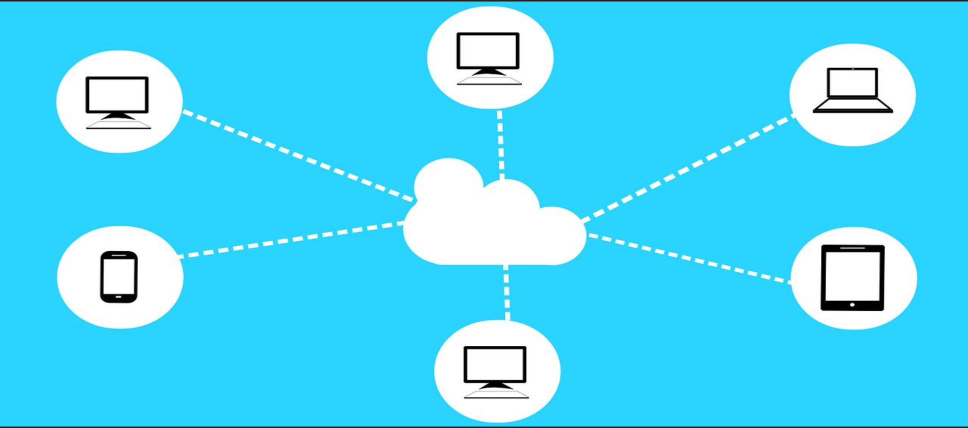
National Institute of Standards and Technology (NIST) 2011: Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud Computing Essential Characteristics

On-demand self-service: A consumer can unilaterally demand computing resources

Cloud Computing Essential Characteristics

Broad network access: Resources are available over the network and accessed through standard mechanisms - mobile phones, tablets, laptops, and workstations



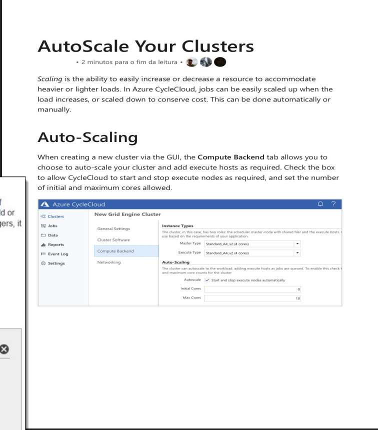
Cloud Computing Essential Characteristics

Resource pooling: The provider’s computing resources are pooled to serve multiple consumers - no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction - country, state, or datacenter.

Geographical Regions:
Azure 54
AWS 25
Google 21

Cloud Computing Essential Characteristics

Rapid elasticity: Resources can be elastically provisioned and released, in some cases automatically.

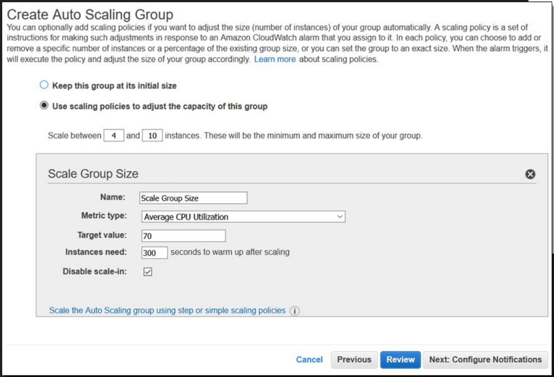

AutoScale Your Clusters

• 2 minutos para o fim da leitura •

Scaling is the ability to easily increase or decrease a resource to accommodate heavier or lighter loads. In Azure CycleCloud, jobs can be easily scaled up when the load increases, or scaled down to conserve cost. This can be done automatically or manually.

Auto-Scaling

When creating a new cluster via the GUI, the **Compute Backend** tab allows you to choose to auto-scale your cluster and add execute hosts as required. Check the box to allow CycleCloud to start and stop execute nodes as required, and set the number of initial and maximum cores allowed.



Cloud Computing Essential Characteristics

Measured service: Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

Spend by Service Category

AWS Service Category	Service Name	% Share		M/M%		R Q/Q%		YTD Y/Y%	Usage
⊞ Compute		60.66%	↑	12.36%	→	3.79%	↑	61.9%	$21.39M
⊞ Database		11.48%	→	8.14%	→	2.35%	↑	78.95%	$4.05M
⊟ SavingsPlansforAWSComputeusage	SavingsPlansforAWS...	6.01%	↑	10.71%	→	6.18%	↑	115.23%	$2.12M
	Subtotal	6.01%	↑	10.71%	→	6.18%	↑	115.23%	$2.12M
⊞ AWSPremiumSupport		4.92%	→	6.84%	→	4.13%	↑	23.22%	$1.74M
⊞ Storage		3.67%	→	6.44%	↑	21.28%	↑	191.37%	$1.30M
⊞ EC2Usage		2.45%			↓	-100.00%			$866.71K
⊞ Analytics		2.16%	↑	14.36%	↓	-1.45%	↑	104.34%	$764.07K
⊞ Networking & Content Delivery		2.03%	↑	15.71%	↑	23.93%	↑	314.37%	$718.09K

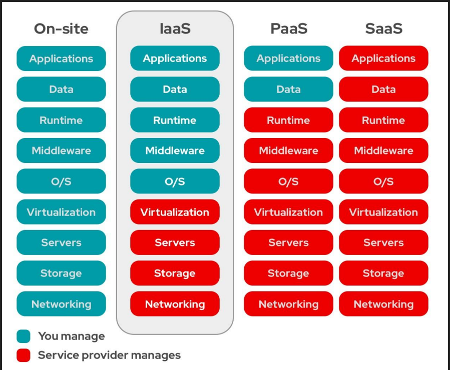
Cloud Computing

Technical and economic advantages:

- Immediate access to computational resources
- No upfront capital investments
- Pay-per-use model

Cloud Computing

Infrastructure as a Service

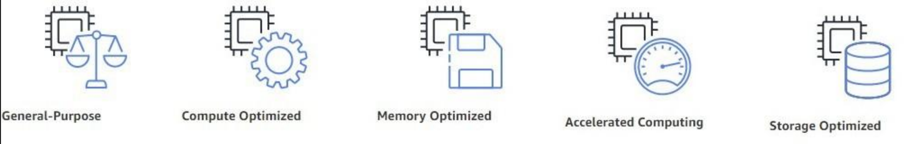


- Many types of Virtual Machines(VMs).
- Each type with different characteristics: network bandwidth, CPU, memory and cost.
- VMs can be used to create virtual *clusters* to execute parallel applications.

Cloud Computing

Infrastructure as a service - VM Families

- General-purpose (balance between CPU, memory, network)
- Compute-optimized (high performance cpu resources)
- Memory-optimized (data intensive applications)
- Accelerated computing (GPU, FPGA)
- Storage-optimized (local storage with high disk throughput)



Heterogeneity in Cloud Computing

Variety of instances per family in Public Clouds. Total:

- ~400 on Microsoft Azure
- ~250 on Amazon EC2
- ~110 on the Google Cloud

Instances with different number of cores, sets of memory and disk

Cloud Computing

Markets

Markets with different guarantees in terms of availability and prices

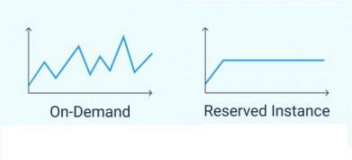
- On-demand VMs: High availability, cannot be interrupted by the provider

Cloud Computing
Markets

Markets with different guarantees in terms of availability and prices

-On-demand VMs

-Reserved VMs: Reduction in the pricing fees compared to pay-as-you-go pricing - up to 75% in Amazon EC2, time period usually considered as 1-year or 3-year, lower availability



Cloud Computing

Markets

Markets with different guarantees in terms of availability and prices

- On-demand VMs
- Reserved VMs
- Spot VMs: Offer up to 90% discount compared with on-demand prices in Amazon EC2, lower availability, interrupted by the provider when it needs the resources back

High Performance Computing

Supercomputers: computers composed of several processing units used to solve problems which can not be solved by regular computers in realistic times.

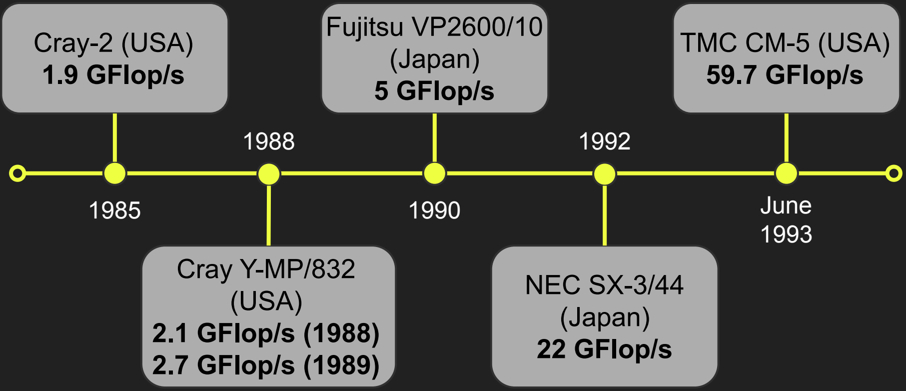
Metric: FLOPS (Floating Point operations per second)

High Performance Computing

- The **TOP500** project ranks and details the 500 most powerful computer systems in the world.
- Started in 1993 and publishes an updated list of the supercomputers twice a year.
- Bases rankings on HPL (High Performance Linpack)

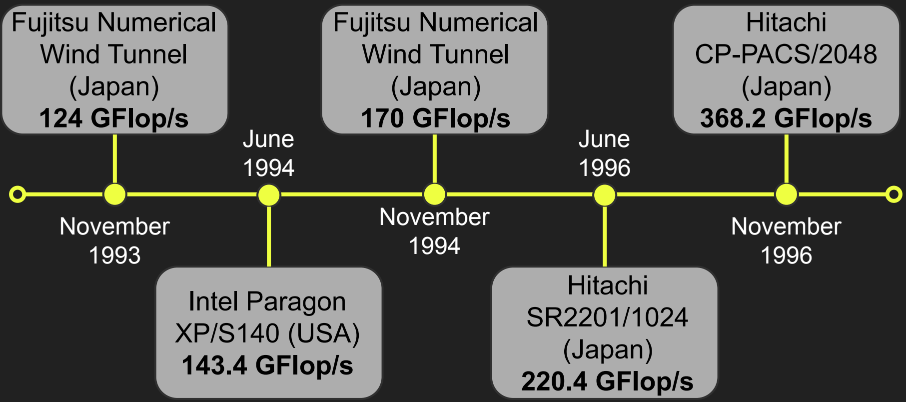
High Performance Computing - timeline

1985-1993 (GFlops)



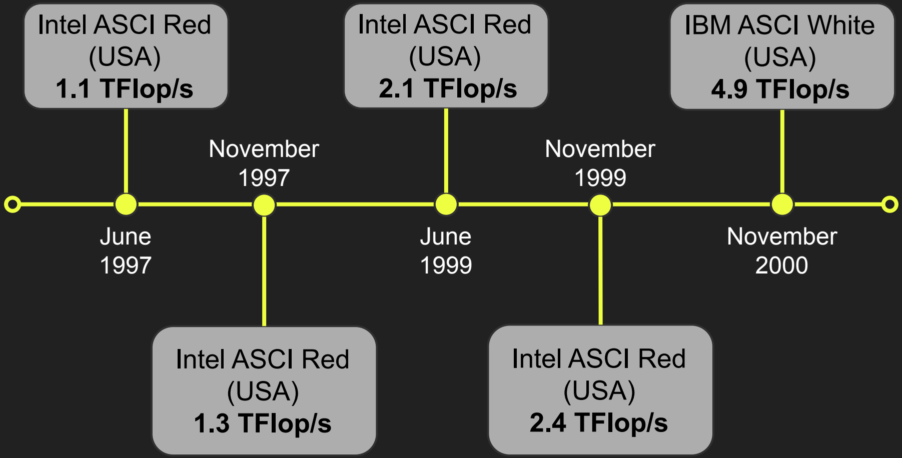
High Performance Computing and TOP 500

1993-1996(GFlops)



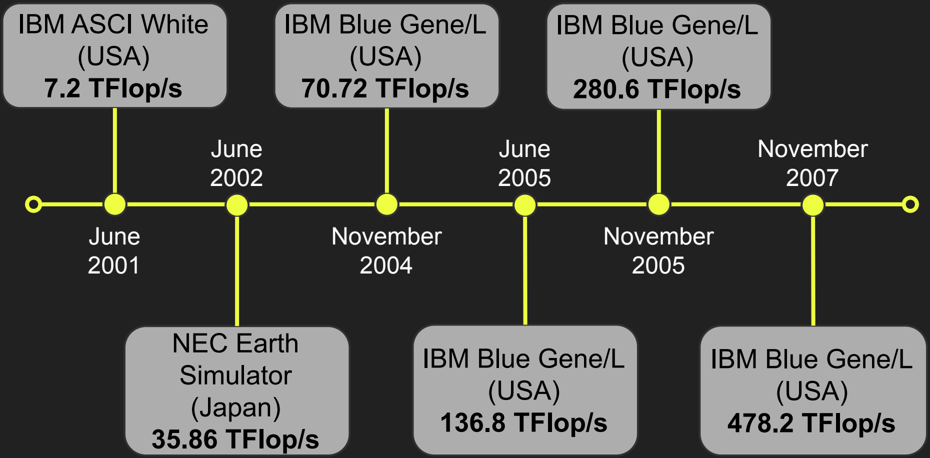
High Performance Computing and TOP 500

1997-2000 (TFlops)



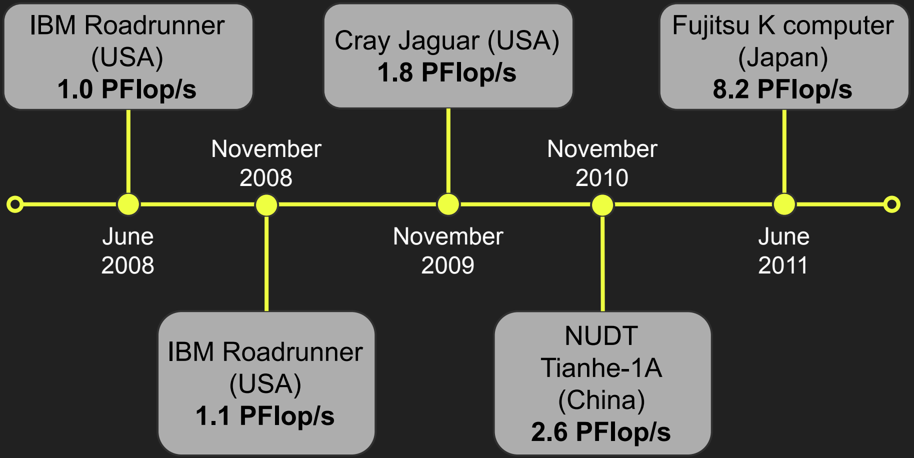
High Performance Computing and TOP 500

2001-2007 (TFlops)



High Performance Computing

2008-2011 (PFlops)



High Performance Computing

2011-2016 (PFlops)

November 2011

Fujitsu K computer (Japan)
10.5 PFlop/s

June 2012

IBM Sequoia Blue Gene/Q (USA)
16.32 PFlop/s

November 2012

Cray Titan (USA)
17.59 PFlop/s

June 2013

NUDT Tianhe-2A (China)
33.86 PFlop/s

June 2016

NRCPC Sunway TaihuLight (China)
93.01 PFlop/s

High Performance Computing Computing

2018-present (PFlops)

June 2018
IBM Summit (USA)
122.3 PFlop/s

November 2018
IBM Summit (USA)
143.5 PFlop/s

June 2019
IBM Summit (USA)
148.6 PFlop/s

June 2020
Supercomputer Fugaku (Japan)
415.53 PFlop/s

November 2020
Supercomputer Fugaku (Japan)
442.01 PFlop/s

June/November 2021
Supercomputer Fugaku (Japan)
442.01 PFlop/s

Cloud Computing and TOP500 November 2021

Position 10:

Voyager-EUS2 Microsoft Azure

~30 PetaFlops

Cloud Computing and Supercomputing

Reinventing High Performance Computing: Challenges and Opportunities, Daniel Reed, Dennis Gannon,Jack Dongarra
March 8, 2022

"Change is now in the wind… The major cloud vendors have invested in global networks of massive scale systems that dwarf today's HPC systems. Driven by the computing demands of AI, these cloud systems are increasingly built using custom semiconductors, reducing the financial leverage of traditional computing vendors. These cloud systems are now breaking barriers in game playing and computer vision, reshaping how we think about the nature of scientific computation. Building the next generation of leading edge HPC systems will require rethinking many fundamentals and historical approaches …"

Cloud Computing and High Performance Computing

Several scenarios:

- HPC in the Cloud: application executed in the cloud
- HPC plus Cloud: the Cloud is used together with other HPC resources
- HPC as a Service: HPC treated as a service by the Cloud provider

Cloud Computing and High Performance Computing

HPC as a Service, some examples to create and manage clusters:

- AWS ParallelCluster
- Azure CycleCloud

Cloud Computing - C+HPC Laboratory

Reducing Financial Costs in Clouds

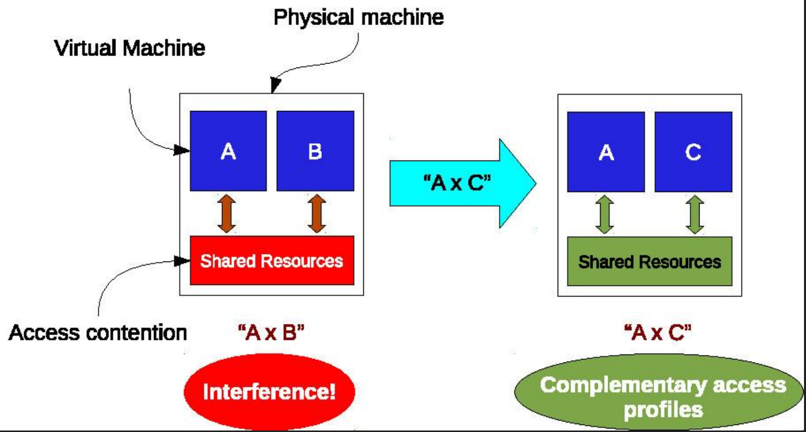
Resource management

Provider Side:

- Big computing infrastructures in different geographical areas: complex to manage and very costly to operate
- Power supply dominates the operational costs: Placing VMs to Datacenters, renewable energy generators co-located with the data centers.

Minimizing Physical Machines X Interference Problem

Applications can experience **cross-interference**.



Minimizing Physical Machines

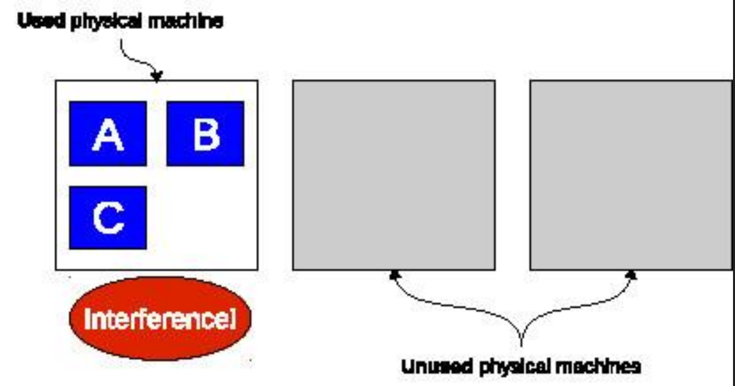
Minimizing the number of used physical machines.

→ Reduction of operating costs

Minimizing Physical Machines X Interference Problem

Minimizing the number of used physical machines.
→ Reduction of operating costs

Conflicting objectives...

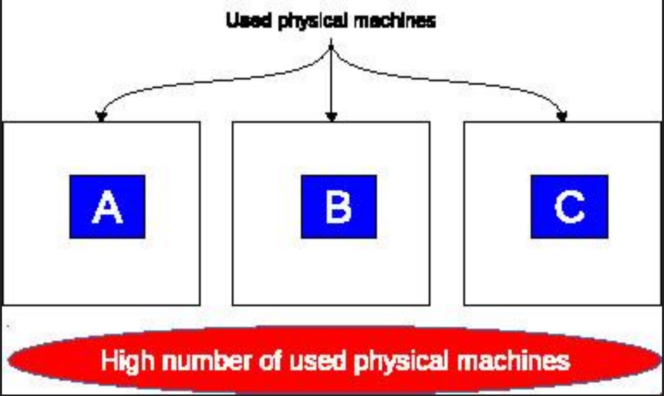


Minimizing Physical Machines x Interference Problem

Minimizing the number of used physical machines.

→ Reduction of operating costs

Conflicting objectives...



Interference Prediction: multivariate problem

Interference is influenced by:

- Concurrent access to **Cache**.
- Concurrent access to **DRAM**.
- Concurrent access to **Virtual Network**.
- **Similarity** between applications access profiles.

Multivariate problem.

Interference Prediction: multivariate problem

Interference is influenced by:

- Concurrent access to **Cache**.
- Concurrent access to **DRAM**.
- Concurrent access to **Virtual Network**.
- **Similarity** between applications access profiles.

Multivariate problem.

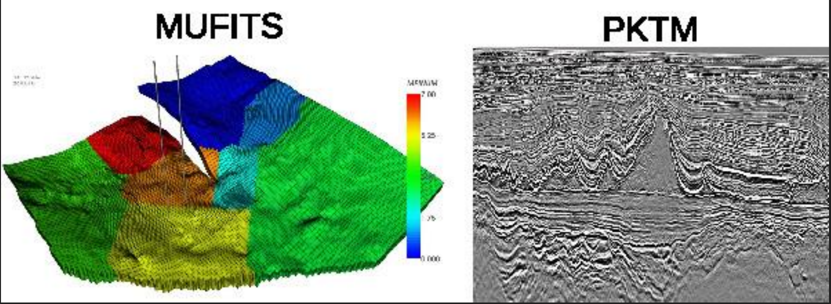
Multiple Regression Analysis

Some Results with several different co-locations

HPCC benchmark:

- FFT: Fast Fourrier Transform.
- DGEMM: matrix-matrix multiplication.
- PTRANS: matrix transpose.
- HPL: linear equation solver.

Applications from **petroleum industry**:



Some Results

For each co-location:

1. **Predicted** interference level.
2. **Real** interference level.
3. Prediction error: **difference** between interference levels.

- ► Median: 3.59%
- ► Maximum: 13.88%


The Journal of Systems and Software 128 (2017) 150–163

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

A multivariate and quantitative model for predicting cross-application interference in virtual environments

Maicon Melo Alves*, Lúcia Maria de Assumpção Drummond

Instituto de Computação, Universidade Federal Fluminense, Niterói, Brazil

ARTICLE INFO

Article history:
Received 23 October 2016
Revised 13 March 2017
Accepted 2 April 2017
Available online 3 April 2017

ABSTRACT

Cross-application interference can drastically affect performance of HPC applications executed in clouds. The problem is caused by concurrent access of co-located applications to shared resources such as cache and main memory. Several works of the related literature have considered general characteristics of HPC applications or the total amount of SLLC accesses to determine the cross-application interference. However, our experiments showed that the cross-application interference problem is related to the amount

An Interference-Aware Strategy for Co-locating High Performance Computing Applications in Clouds

Maicon Melo Alves[1,2(✉)], Luan Teylo[1], Yuri Frota[1], and Lúcia Maria de A. Drummond[1]

[1] Fluminense Federal University, Niterói, Brazil
{mmelo,luanteylo,yuri,lucia}@ic.uff.br
[2] Petroleo Brasileiro S. A. - Petrobras, Rio de Janeiro, Brazil

Abstract. Cross-interference may happen when applications share a common physical machine, affecting negatively their performances. This problem occurs frequently when high performance applications are executed in clouds. Some papers of the related literature have considered this problem when proposing strategies for Virtual Machine Placement. However, they neither have employed a suitable method for predicting interference nor have considered the minimization of the number of used physical machines and interference at the same time. In this paper, we present a solution based on the Iterated Local Search framework to solve the Interference-aware Virtual Machine Placement Problem for HPC applications in Clouds (IVMP). This problem aims to minimize, at the same time, the interference suffered by HPC applications which share common physical machines and the number of physical machines used to allocate them. Experiments were conducted in a real scenario by using applications from oil and gas industry and applications from the HPCC benchmark. They showed that our method reduced interference in more than 40%, using the same number of physical machines of the most widely employed heuristics to solve the problem.

1 Introduction

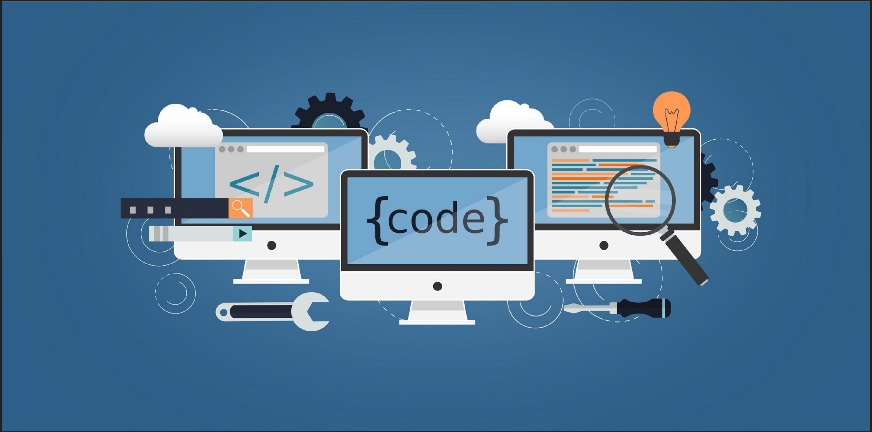
High Performance Computing (HPC) applications are typically executed in dedicated datacenters. However, in the past few years, cloud computing has emerged as an attractive option to run these applications due to several advantages that it brings when compared with a dedicated infrastructure. Clouds provide a rapid provisioning of resources and a significant reduction in operating costs related to energy, software licence and hardware obsolescence [6,15]. Moreover, with the recent increase of the number of cores in modern physical machines, HPC applications can scale up to many cores even when fully allocated to a single physical machine. Consequently, clouds can offer a reasonable service performance for small-scale HPC applications, that can be entirely placed into a single physical machine, thus avoiding the performance penalty usually imposed by the physical network of clouds [8].

C. Bianchini et al. (Eds.): WSCAD 2018, CCIS 1171, pp. 3–20, 2020.
https://doi.org/10.1007/978-3-030-41050-6_1

Reducing Financial Costs in Clouds

User side - Dimensioning and Scheduling algorithms:

- Instance families and markets



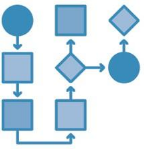
Cloud Dimensioning

How many VMs of each type and how long does an application require?

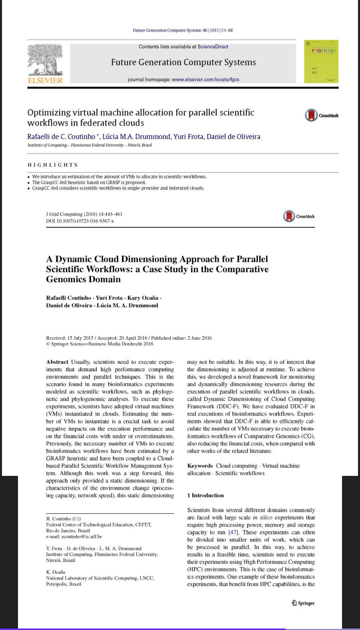
Optimization Problem!

Cloud Dimensioning of Scientific Workflows

Input: memory, storage and processing demand in GFlops for executing a scientific workflow



Output: a virtual cluster


Future Generation Computer Systems 46 (2015) 51–68

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Optimizing virtual machine allocation for parallel scientific workflows in federated clouds

Rafaelli de C. Coutinho *, Lúcia M.A. Drummond, Yuri Frota, Daniel de Oliveira

Institute of Computing – Fluminense Federal University – Niterói, Brazil

HIGHLIGHTS

- We introduce an estimation of the amount of VMs to allocate in scientific workflows.
- The GraspCC-fed heuristic based on GRASP is proposed.
- GraspCC-fed considers scientific workflows in single-provider and federated clouds.

J Grid Computing (2016) 14:443–461
DOI 10.1007/s10723-016-9367-x

A Dynamic Cloud Dimensioning Approach for Parallel Scientific Workflows: a Case Study in the Comparative Genomics Domain

Rafaelli Coutinho · Yuri Frota · Kary Ocaña · Daniel de Oliveira · Lúcia M. A. Drummond

Received: 15 July 2015 / Accepted: 20 April 2016 / Published online: 2 June 2016

Abstract Usually, scientists need to execute experiments that demand high performance computing environments and parallel techniques. This is the scenario found in many bioinformatics experiments modeled as scientific workflows, such as phylogenetic and phylogenomic analyses. To execute these experiments, scientists have adopted virtual machines (VMs) instantiated in clouds. Estimating the number of VMs to instantiate is a crucial task to avoid negative impacts on the execution performance and on the financial costs with under or overestimations. Previously, the necessary number of VMs to execute bioinformatics workflows have been estimated by a GRASP heuristic and have been coupled to a Cloud-based Parallel Scientific Workflow Management System. Although this work was a step forward, this approach only provided a static dimensioning. If the characteristics of the environment change (processing capacity, network speed), this static dimensioning may not be suitable. In this way, it is of interest that the dimensioning is adjusted at runtime. To achieve this, we developed a novel framework for monitoring and dynamically dimensioning resources during the execution of parallel scientific workflows in clouds, called Dynamic Dimensioning of Cloud Computing Framework (DDC-F). We have evaluated DDC-F in real executions of bioinformatics workflows. Experiments showed that DDC-F is able to efficiently calculate the number of VMs necessary to execute bioinformatics workflows of Comparative Genomics (CG), also reducing the financial costs, when compared with other works of the related literature.

Keywords Cloud computing · Virtual machine allocation · Scientific workflows

1 Introduction

Scientists from several different domains commonly are faced with large scale *in silico* experiments that require high processing power, memory and storage capacity to run [47]. These experiments can often be divided into smaller units of work, which can be processed in parallel. In this way, to achieve results in a feasible time, scientists need to execute their experiments using High Performance Computing (HPC) environments. This is the case of bioinformatics experiments. One example of these bioinformatics experiments, that benefit from HPC capabilities, is the

R. Coutinho (✉)
Federal Center of Technological Education, CEFET, Rio de Janeiro, Brazil
e-mail: rcoutinho@ic.uff.br

Y. Frota · D. de Oliveira · L. M. A. Drummond
Institute of Computing, Fluminense Federal University, Niterói, Brazil

K. Ocaña
National Laboratory of Scientific Computing, LNCC, Petrópolis, Brazil

Springer

Dimensioning - Some Results in AWS

Workflow SciPhylomics

- Dimension: GraspCC
- Redimension: Initialized with a metaheuristic (GraspCC) and with 1 VM m3.large
- Literature: SciDim
- Scheduling and load balancing by SciCumulus

Approaches	*GRASPCC*	*SciDIM*	*Redimension*	
			GraspCC	*Min VMs*
Set of VMs	13 m3.xlarge, 1 m3.large	12 m3.large, 3 m3.medium	t_0: 13 m3.xlarge, 1 m3.large t_1: 13 m3.large	t_0: 1 m3.large t_1: 11 m3.large, 2 m1.small t_2: 6 m3.xlarge, 1 m3.large
Execution Time	137 minutes	179 minutes	156 minutes	151 minutes
Financial Cost	U$10, 56	U$5, 67	U$7, 16	U$5, 12

Scheduling on Spots - Challenges

Scheduler: define where and when the tasks should be executed. Usually, bounded by restrictions such as deadline or memory limits.

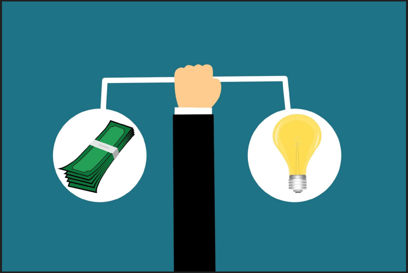
- Fault-tolerance and reliability: these instances can be revoked so using for sensitive applications without special fault-tolerance is not advised.

Scheduling to Spots - Challenges

- Quality of service and service level agreement: no guarantee on QoS and SLAs. In some cases, some policies can be considered to improve the overall availability and performance

Scheduling to Spots - Challenges

- Price prediction: dynamic pricing approach that depends on the supply and demand. Users offer a bid to the service provider. Submitting an appropriate bid decreases the possibility of an unpredictable termination.

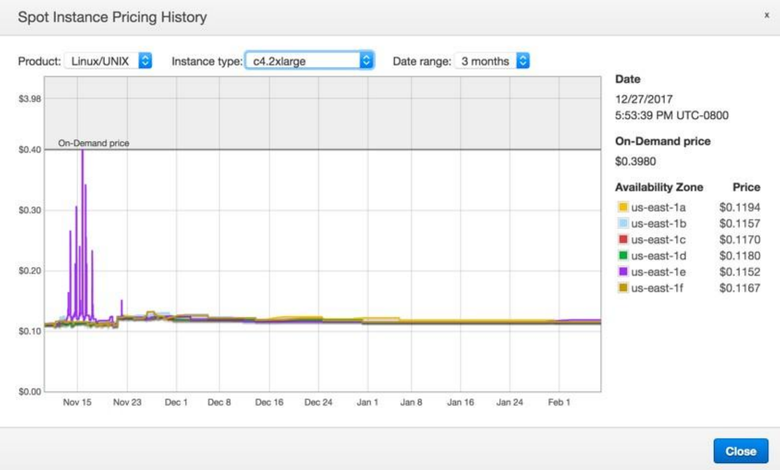


Spot VMs in Amazon Web Services

In December 2017 Amazon EC2 changed the spot VMs pricing model. To contract a spot VM the user had to give a bid and the VM was allocated only if its current price is smaller than that bid

Spot VMs in Amazon Web Services

Before the new price model, the spot market prices faced constant variations because it was driven by the users' bids



Spot VMs

Amazon also announced:

- Reduction in the number of interruptions
- The **Hibernation** feature

Hibernation-prone Spot VMs

When a spot VM is hibernated:

- Its memory and context are saved
- If VM resumes, all the tasks restart from the hibernate point
- During the hibernation the user is not charged by the VM allocation

Spot VMs

Challenges:

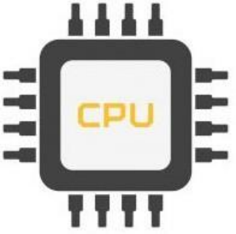
Fault-tolerance and reliability

Quality of service and service level agreement

Some Works

Scheduling on spot and on-demand CPU instances

- Subramanya et al. (2015) present the SpotOn, uses fault-tolerance mechanisms to mitigate the impact of spot revocations
- Varsheney et al. (2019) propose AutoBot, a framework that uses checkpoints alongside spot and on-demand VMs for executing BoT applications

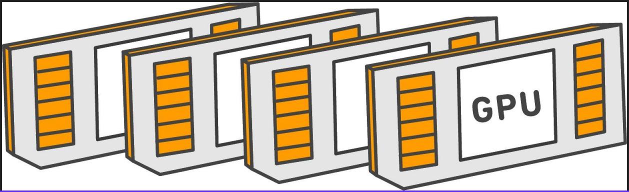


Scheduling with spot and on-demand GPU instances

- Lee and Son (2017) propose DeepSpotCloud, a framework that uses AWS Spot GPU instances for Deep Learning tasks.

- Zhou et al. (2019) implemented a fault-tolerant stencil computation on the AWS Spot GPU instances, using an application-specific checkpointing mechanism.

- Wagenländer et al. (2020) proposed Spotnik, a system to Machine Learning using Spot GPU VMs and synchronizes the communication phase to deal with revocations.



Cloud Computing Companies

Companies which monitor and optimize resources use from all markets

Hibernation Aware Dynamic Scheduler

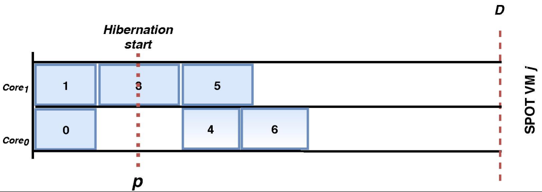
HADS uses hibernation-prone spot instances to minimize the monetary cost of Bag of Tasks applications (BoTs) execution, respecting their deadline constraints.

Problem Definition

Problem Statement

In an environment with hibernated-prone VMs, a hibernated VM can resume either in time or not to meet the deadline

If the VM does not resume in time, the deadline can be violated. To avoid a **temporal failure**, the affected tasks have to be migrated to other VMs



$$Q_{jp} = \{3, 4, 5, 6\}$$

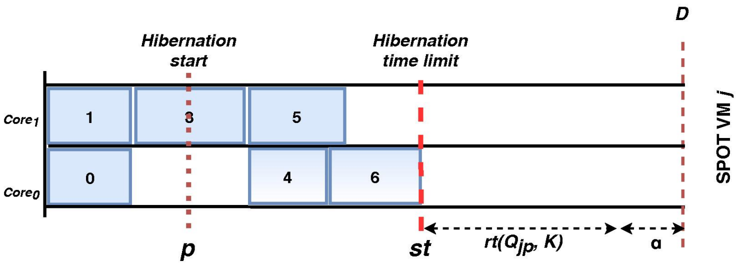
Set of affected tasks

Problem Definition

To compute the maximum period that a VM can remain hibernated without violating the deadline D, we consider that:

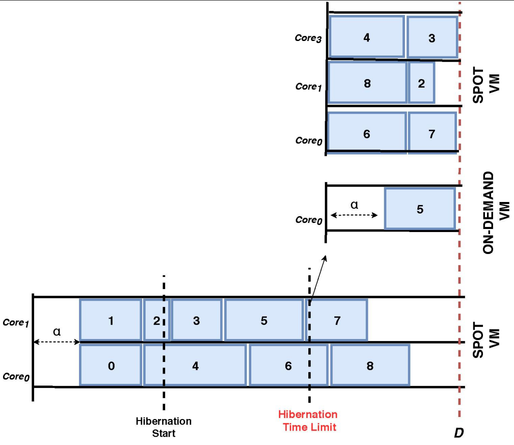
- A VM takes α periods of time to be ready to receive any task
- The number of periods required to execute all affected tasks in Q_{jp} on a set of VMs, K, is given by $rt(Q_{jp}, K)$.

$$st = (D - \alpha) - rt(Q_{jp}, K)$$

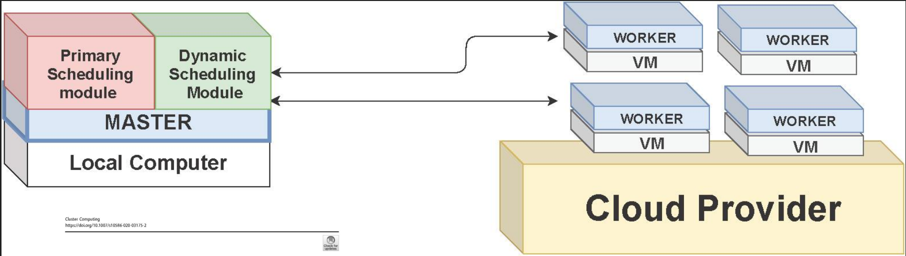


Problem Definition

If the VM does not resume in time, we have to migrate tasks to a set K of VMs



Architecture of the Framework



Cluster Computing
https://doi.org/10.1007/s10586-020-03175-2

Check for updates

A dynamic task scheduler tolerant to multiple hibernations in cloud environments

Luan Teylo[1] · Luciana Arantes[2] · Pierre Sens[2] · Lucia M. A. Drummond[1]

Received: 6 January 2020 / Revised: 18 August 2020 / Accepted: 21 August 2020

Abstract
Cloud platforms usually offer several types of Virtual Machines (VMs) with different guarantees in terms of availability and volatility, provisioning the same resource through multiple pricing models. For instance, in the Amazon EC2 cloud, the user pays per use for on-demand VMs while spot VMs are instances available at lower prices. However, a spot VM can be terminated or hibernated by EC2 at any moment. In this work, we propose the Hibernation-Aware Dynamic Scheduler (HADS) that schedules Bag-of-Tasks (BoT) applications with deadline constraints in both hibernation prone spots VMs and on-demand VMs. HADS aims at minimizing the monetary costs of executing BoT applications on Clouds ensuring that their deadlines are respected even in the presence of multiple hibernations. Results collected from experiments on Amazon EC2 VMs using synthetic applications and a NAS benchmark application show the effectiveness of HADS in terms of monetary costs when compared to on-demand VM only solutions.

Keywords Cloud computing · Dynamic BoT scheduling · Temporal failures · Spot VM hibernation · Monetary cost minimization

1 Introduction

In the past few years, cloud computing has emerged as an attractive option to run different classes of applications due to several advantages over other platforms, such as: (i) immediate access to computational resources, (ii) no upfront capital investments, and (iii) pay-per-use model. Some cloud providers offer several classes of Virtual Machines (VMs) with different guarantees in terms of availability and volatility, provisioning the same resource through multiple pricing models. Amazon EC2, for example, offers VMs in two main markets: on-demand and spot.

On-demand VMs can be deployed at any time, offering high availability since they cannot be interrupted by Amazon provider while allocated by a user. On the other hand, spot VMs are unused EC2 resources with a huge discount (according to Amazon the discount can be up to 90% when compared to on-demand prices) but can be revoked and terminated by Amazon whenever it requires the resources back.

Since December 2017, Amazon EC2 has defined a VM allocation policy where spot prices are more stable and with little differences over the days, i.e., they do not vary according to users' resource requests demand [23]. Furthermore, Amazon EC2 has introduced the spot VM hibernation feature that hibernates a spot VM instead of terminating it definitively [4]. When a VM hibernates, its memory and context are saved. In the case of on-demand VMs, only the user can decide to hibernate/resume a VM while, in the case of spot VMs, Amazon hibernates a given VM when it needs its resources and resumes it when cloud resource demands decrease. When a VM is resumed, all tasks that were executing at the moment of the hibernation

✉ Luan Teylo
luanteylo@ic.uff.br

Luciana Arantes
luciana.arantes@lip6.fr

Pierre Sens
pierre.sens@lip6.fr

Lucia M. A. Drummond
lucia@ic.uff.br

1 Instituto de Computação, Universidade Federal Fluminense, Niterói, Brazil

2 Sorbonne Université, CNRS, INRIA, LIP6, Paris, France

Published online: 03 September 2020

Springer

2019 31st International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)

A Bag-of-Tasks Scheduler Tolerant to Temporal Failures in Clouds

Luan Teylo
Fluminense Federal University
Institute of Computing
Niteroi, Brazil
luanteylo@ic.uff.br

Luciana Arantes
Sorbonne Université
CNRS, INRIA, LIP6
Paris, France
luciana.arantes@lip6.fr

Pierre Sens
Sorbonne Université
CNRS, INRIA, LIP6
Paris, France
pierre.sens@lip6.fr

Lúcia Maria de A. Drummond
Fluminense Federal University
Institute of Computing
Niteroi, Brazil
lucia@ic.uff.br

***Abstract*—Cloud platforms offer different types of virtual machines which ensure different guarantees in terms of availability and volatility, provisioning the same resource through multiple pricing models. For instance, in Amazon EC2 cloud, the user pays per hour for *on-demand* instances while *spot* instances are unused resources available for a lower price. Despite the monetary advantages, a spot instance can be terminated or hibernated by EC2 at any moment. Using both hibernation-prone spot instances (for cost sake) and on-demand instances, we propose in this paper a static scheduling for applications which are composed of independent tasks (bag-of-task) with deadline constraints. However, if a spot instance hibernates and it does not resume within a time which guarantees the application's deadline, a temporal failure takes place. Our scheduling, thus, aims at minimizing monetary costs of bag-of-tasks applications in EC2 cloud, respecting its deadline and avoiding temporal failures. Performance results with task execution traces, configuration of Amazon EC2 virtual machines, and EC2 market history confirms the effectiveness of our scheduling and that it tolerates temporal failures.**

***Index Terms*—Clouds, Temporal failures, Scheduling**

I. Introduction

In the past few years, cloud computing has emerged as an attractive option to run different classes of applications due to several advantages that it brings when compared with a dedicated infrastructure. Clouds provide a significant reduction in operational costs, besides offering a rapid elastic provisioning of computing resources like virtual machines (VMs) and storage. However, in cloud environments, besides the usual goal of minimizing the execution time of the application, it is also important to minimize the monetary cost of using cloud resources, i.e., there exists a trade-off between performance and monetary cost.

Infrastructure-as-a-Service (IaaS) existing cloud platforms (e.g., Amazon EC2, Microsoft Azure, Google Cloud, etc.) enable users to dynamically acquire resources, usually as virtual machines (VMs), according to their application requirements (CPU, memory, I/O, etc.) in a pay-as-you-use price model. They usually offer different classes of VMs which ensure different guarantees in terms of availability and volatility, provisioning the same resource through multiple pricing models. For instance, in Amazon EC2, there are basically three classes: (i) *reserved* VM instances, where the user pays an upfront price, guaranteeing long-term availability; (ii) *on-demand* VM instances which are allocated for specific time periods and incur a fixed cost per unit time of use, ensuring availability of the instance during this period; (iii) *spot* VM instances which are unused instances available for lower price than on-demand price.

The availability of spot VMs instances fluctuates based on the spot market's current demand. If there are not enough instances to meet clients demands, the VM can be interrupted by the cloud provider (temporarily or definitively). Despite the risk of unavailability, the main advantage of spot VMs is that their cost is much lower than on-demand VMs since the user requests unused instances at steep discounts, reducing the costs significantly. With Amazon's more recent announcement, an interrupted spot can either terminate or hibernate[1]. Whenever a spot instance is hibernated by EC2, its memory and context are saved to the root of EC2 Block Storage (EBS) volume and, during the VM's pause, the user is only charged for EBS storage. EC2 resumes the hibernated instance, reloading the saved memory and context, only when there is enough availability for that type of instance with a spot price which is lower than the user's maximum price. Contrarily to terminated instances whose user is warned two minutes before the interruption of them, hibernated instances are paused immediately after noticing the user.

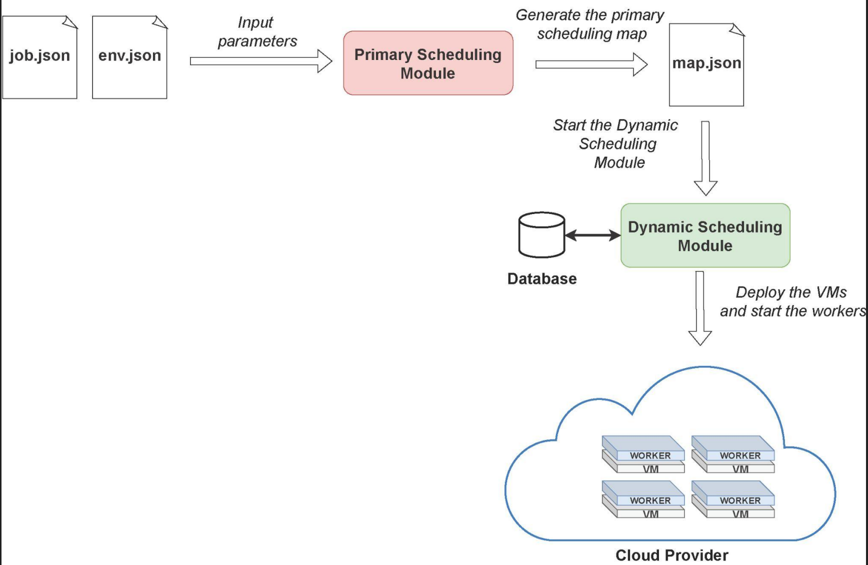
Our proposal in this work is to provide a static cloud scheduler for Bag-of-Tasks applications using, for cost sake, hibernate-prone spot instances as much as possible, respecting the application deadline constraints while also minimizing the monetary costs of bag-of-tasks applications. However, if a spot instance hibernates, it might happen that it will not resume within a time which guarantees the deadline constraints of the application. In this case, a *temporal failure* would take place, i.e., correct computation is performed but too late to be useful (inability to meet deadlines). Thus, in order to avoid temporal failure in case of spot instance hibernation, our scheduler statically computes the time interval that an

[1] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-interruptions.html

978-1-7281-4194-7/19/$31.00 ©2019 IEEE
DOI 10.1109/SBAC-PAD.2019.00033

144

Architecture of the Framework



Hibernation-Aware Dynamic Scheduler

System Architecture

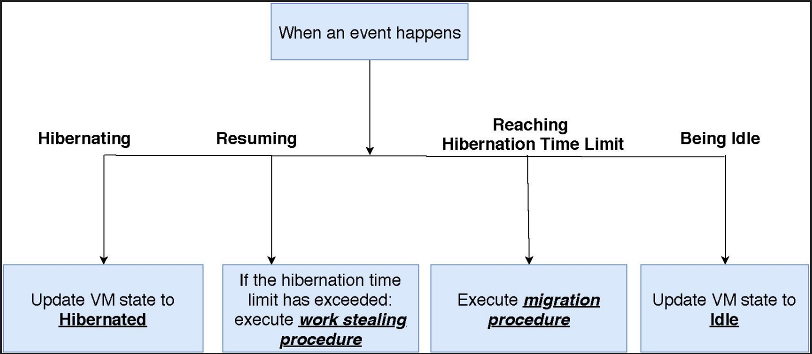
Each VM can be in one of the following states during the execution:

- *Busy*
- *Idle*
- *Hibernated*
- *Terminated*

Hibernation-Aware Dynamic Scheduler

Dynamic Module

The Dynamic Scheduler Module is an event-driven algorithm that performs some actions in response to events that can increase the monetary cost or cause temporal failures



Hibernation-Aware Dynamic Scheduler

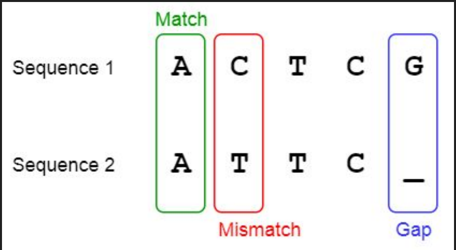
Burstables Virtual Machines

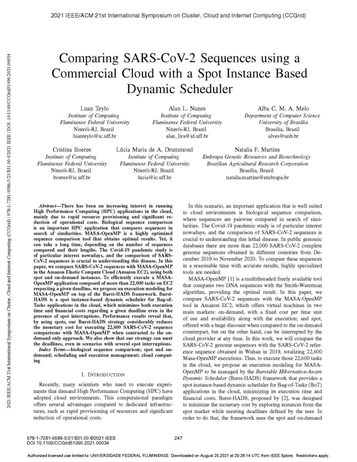
- Scheduling Bag-of-Tasks in Clouds using Spot and Burstable Virtual Machines. IEEE Transactions on Cloud Computing, online, to be published.

The Dynamic Scheduler Module is an event-driven algorithm that

Case study: A Sequence Alignment Problem in CPU

Comparison of 22,600 Sars-Cov-2 sequences with the Wuhan sequence





2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)

2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid) | 978-1-7281-9586-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/CCGrid51090.2021.00034

Comparing SARS-CoV-2 Sequences using a Commercial Cloud with a Spot Instance Based Dynamic Scheduler

Luan Teylo
Institute of Computing
Fluminense Federal University
Niterói-RJ, Brazil
luanteylo@ic.uff.br

Alan L. Nunes
Institute of Computing
Fluminense Federal University
Niterói-RJ, Brazil
alan_lira@id.uff.br

Alba C. M. A. Melo
Department of Computer Science
University of Brasilia
Brasilia, Brazil
alves@unb.br

Cristina Boeres
Institute of Computing
Fluminense Federal University
Niterói-RJ, Brazil
boeres@ic.uff.br

Lúcia Maria de A. Drummond
Institute of Computing
Fluminense Federal University
Niterói-RJ, Brazil
lucia@ic.uff.br

Natalia F. Martins
Embrapa Genetic Resources and Biotechnology
Brazilian Agricultural Research Corporation
Brasilia, Brazil
natalia.martins@embrapa.br

***Abstract*—There has been an increasing interest in running High Performance Computing (HPC) applications in the cloud, mainly due to rapid resource provisioning and significant reduction of operational costs. Biological sequence comparison is an important HPC application that compares sequences in search of similarities. MASA-OpenMP is a highly optimized sequence comparison tool that obtains optimal results. Yet, it can take a long time, depending on the number of sequences compared and their lengths. The Covid-19 pandemic study is of particular interest nowadays, and the comparison of SARS-CoV-2 sequences is crucial to understanding this disease. In this paper, we compare SARS-CoV-2 sequences with MASA-OpenMP in the Amazon Elastic Compute Cloud (Amazon EC2), using both spot and on-demand instances. To efficiently execute a MASA-OpenMP application composed of more than 22,000 tasks on EC2 respecting a given deadline, we propose an execution modeling for MASA-OpenMP on top of the Burst-HADS framework. Burst-HADS is a spot instance-based dynamic scheduler for Bag-of-Tasks applications in the cloud, which minimizes both execution time and financial costs regarding a given deadline even in the presence of spot interruptions. Performance results reveal that, by using spots, our Burst-HADS strategy considerably reduces the monetary cost for executing 22,600 SARS-CoV-2 sequence comparisons with MASA-OpenMP when contrasted to the on-demand only approach. We also show that our strategy can meet the deadlines, even in scenarios with several spot interruptions.**

***Index Terms*—biological sequence comparison; spot and on-demand; scheduling and execution management; cloud computing**

I. Introduction

Recently, many scientists who need to execute experiments that demand High Performance Computing (HPC) have adopted cloud environments. This computational paradigm offers several advantages compared to dedicated infrastructures, such as rapid provisioning of resources and significant reduction of operational costs.

In this scenario, an important application that is well suited to cloud environments is biological sequence comparison, where sequences are pairwise compared in search of similarities. The Covid-19 pandemic study is of particular interest nowadays, and the comparison of SARS-CoV-2 sequences is crucial to understanding this lethal disease. In public genomic databases there are more than 22,000 SARS-CoV-2 complete genome sequences obtained in different countries from December 2019 to November 2020. To compare these sequences in a reasonable time with accurate results, highly specialized tools are needed.

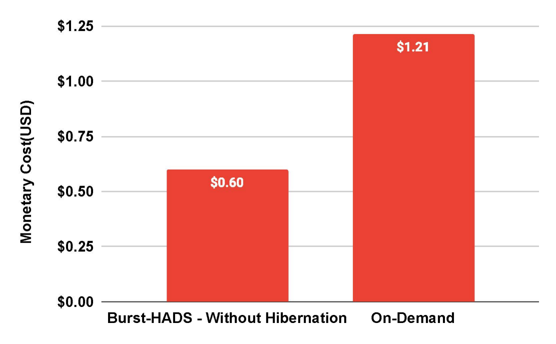
MASA-OpenMP [1] is a multithreaded freely available tool that compares two DNA sequences with the Smith-Waterman algorithm, providing the optimal result. In this paper, we compare SARS-CoV-2 sequences with the MASA-OpenMP tool in Amazon EC2, which offers virtual machines in two main markets: on-demand, with a fixed cost per time unit of use and availability along with the execution; and spot, offered with a huge discount when compared to the on-demand counterpart, but on the other hand, can be interrupted by the cloud provider at any time. In this work, we will compare the SARS-CoV-2 genome sequences with the SARS-CoV-2 reference sequence obtained in Wuhan in 2019, totalizing 22,600 Masa-OpenMP executions. Thus, to execute those 22,600 tasks in the cloud, we propose an execution modeling for MASA-OpenMP to be managed by the *Burstable Hibernation-Aware Dynamic Scheduler* (Burst-HADS) framework that provides a spot instance-based dynamic scheduler for Bag-of-Tasks (BoT) applications in the cloud, minimizing its execution time and financial costs. Burst-HADS, proposed by [2], was designed to minimize the monetary cost by exploring instances from the spot market while meeting deadlines defined by the user. In order to do that, the framework uses the spot and on-demand

978-1-7281-9586-5/21/$31.00 ©2021 IEEE
DOI 10.1109/CCGrid51090.2021.00034

Experimental Results

Case study: A Sequence Alignment Problem

- 24 spot instances (136 vCPUs)
- 21 minutes and 23 seconds
- Monetary cost reduction of 50.4%



Experimental Results

Case study: A Sequence Alignment Problem

Emulation of the hibernation process, using Poisson distribution to model the hibernation and resuming events

Evaluation in three different scenarios of hibernation and resuming events

ID	*hibernating*	*resuming*	λ_h	λ_r
c_1	$k_h = 5$	$k_r = 0$	5/3600	0/3600
c_2	$k_h = 5$	$k_r = 2.5$	5/3600	2.5/3600
c_3	$k_h = 5$	$k_r = 5$	5/3600	5/3600

Experimental Results

Case study: A Sequence Alignment Problem

c_2 shows how resuming events impact the monetary cost. Although the average number of hibernations was greater than c_1, the cost reduction was 29.75%.

scenario	HADS					Diff (%)	
	# hibernation	# resuming	# on-demand	cost($)	makespan	cost($)	makespan
c_1	6.67	0.00	8.33	1.185	2978.66	2.06%	-133.25%
c_2	9.67	6.33	3.33	0.85	2785.66	29.75%	-118.141%
c_3	8.33	4.66	4.00	0.82	2350.33	32.41%	-84.0512%

Sequence Alignment Application in GPU

- MASA-CUDAlign achieves an outstanding performance using hundreds of GPUs
  - Such a platform is not available for most scientists

- Multiple GPUs available in EC2
  - Nvidia:Kepler, Maxwell, Turing, Volta and Ampere arch

A Fault Tolerant and Deadline Constrained Sequence Alignment Application on Cloud-Based Spot GPU Instances

Rafaela C. Brum[1(✉)], Walisson P. Sousa[2], Alba C. M. A. Melo[3], Cristiana Bentes[2], Maria Clicia S. de Castro[2], and Lúcia Maria de A. Drummond[1]

[1] Fluminense Federal University, Niterói, Brazil
rafaelabrum@id.uff.br, lucia@ic.uff.br
[2] State University of Rio de Janeiro, Rio de Janeiro, Brazil
walisson.sousa@pos.ime.uerj.br, cris@eng.uerj.br, clicia@ime.uerj.br
[3] University of Brasilia, Brasilia, Brazil
alves@unb.br

Abstract. Pairwise sequence alignment is an important application to identify regions of similarity that may indicate the relationship between two biological sequences. This is a computationally intensive task that usually requires parallel processing to provide realistic execution times. This work introduces a new framework for a deadline constrained application of sequence alignment, called MASA-CUDAlign, that exploits cloud computing with Spot GPU instances. Although much cheaper than On-Demand instances, Spot GPUs can be revoked at any time, so the framework is also able to restart MASA-CUDAlign from a checkpoint in a new instance when a revocation occurs. We evaluate the proposed framework considering five pairs of DNA sequences and different AWS instances. Our results show that the framework reduces financial costs when compared to On-Demand GPU instances while meeting the deadlines even in scenarios with several instances revocations.

Keywords: Cloud computing · Spot GPU · Sequence alignment

1 Introduction

Biological sequence alignment is an important and extensively used computational procedure that arranges sequences of DNA, RNA, or proteins to identify regions of similarities and differences. However, this procedure presents quadratic complexity in terms of execution time. Many parallel versions of the sequence alignment algorithm were proposed in the literature exploiting different types of architectures focusing on providing realistic execution times [3,7,15,18]. Among these solutions, MASA-CUDAlign [18] stands out for the comparison of huge

L. Sousa et al. (Eds.): Euro-Par 2021, LNCS 12820, pp. 317–333, 2021.
https://doi.org/10.1007/978-3-030-85665-6_20

Overall Architecture

- Controller
  - Chooses the initial Spot VM
    - Based on financial cost
    - Deadline
  - Handles eventual revocations

- Worker
  - Monitors the application
  - Notifies the Controller of a future revocation

Architecture of proposed framework

Experimental Setup

- 5 pairs of DNA sequences of human and chimpanzee chromosomes
  - Chromosome 19 ⇒ 59MB and 61MB
  - Chromosome 20 ⇒ 64MB and 66MB
  - Chromosome 21 ⇒ 47MB and 33MB
  - Chromosome 22 ⇒ 51MB and 38MB
  - Chromosome Y ⇒ 57MB and 26MB

- All GPU instance types chosen costs less than 1 USD/hour[1]
  - *g2.2xlarge* ⇒ Kepler arch (Nvidia K520)
  - *g3s.xlarge* ⇒ Maxwell arch (Nvidia M60)
  - *g4dn.xlarge* ⇒ Turing arch (Nvidia T4)
  - *g4dn.2xlarge* ⇒ Turing arch (Nvidia T4)
  - *p2.xlarge* ⇒ Kepler arch (Nvidia K80)

[1] prices obtained in November 2020

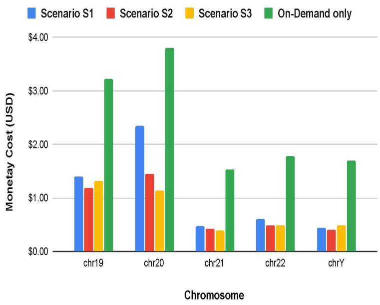
Results with simulated revocations

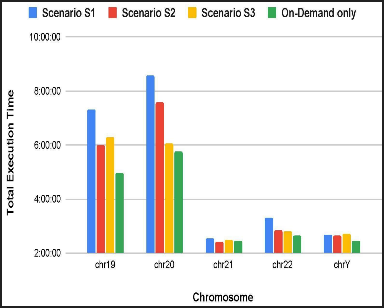
- Poisson model to simulate revocations ⇒ $\lambda = 1/k_r$
  - k_r ⇒ average time of a failure occurrence

- 3 scenarios
  - S1 with k_1 = 2 hours
  - S2 with k_1 = 4 hours
  - S3 with k_r = 6 hours

Results with simulated revocations

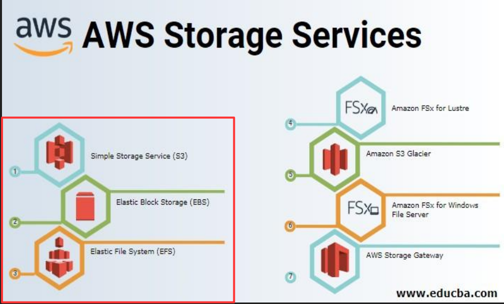
- Compared to on-demand only execution
  - 60% of average cost reduction
  - 13% of average execution time increase

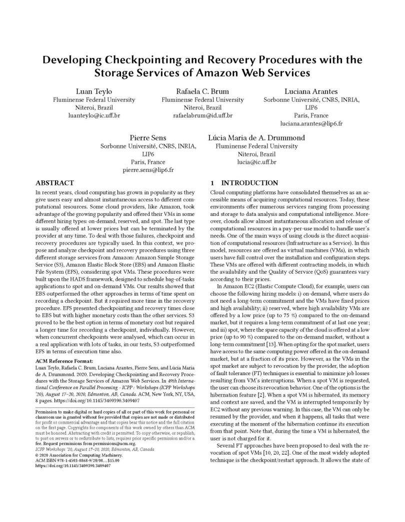




Storage Services

We are interested in general-purpose storage options that can be used to store and recover checkpoints files





Developing Checkpointing and Recovery Procedures with the Storage Services of Amazon Web Services

Luan Teylo, Fluminense Federal University, Niteroi, Brazil, luanteylo@ic.uff.br

Rafaela C. Brum, Fluminense Federal University, Niteroi, Brazil, rafaelabrum@id.uff.br

Luciana Arantes, Sorbonne Université, CNRS, INRIA, LIP6, Paris, France, luciana.arantes@lip6.fr

Pierre Sens, Sorbonne Université, CNRS, INRIA, LIP6, Paris, France, pierre.sens@lip6.fr

Lúcia Maria de A. Drummond, Fluminense Federal University, Niteroi, Brazil, lucia@ic.uff.br

ABSTRACT

In recent years, cloud computing has grown in popularity as they give users easy and almost instantaneous access to different computational resources. Some cloud providers, like Amazon, took advantage of the growing popularity and offered their VMs in some different hiring types: on-demand, reserved, and spot. The last type is usually offered at lower prices but can be terminated by the provider at any time. To deal with those failures, checkpoint and recovery procedures are typically used. In this context, we propose and analyze checkpoint and recovery procedures using three different storage services from Amazon: Amazon Simple Storage Service (S3), Amazon Elastic Block Store (EBS) and Amazon Elastic File System (EFS), considering spot VMs. These procedures were built upon the HADS framework, designed to schedule bag-of-tasks applications to spot and on-demand VMs. Our results showed that EBS outperformed the other approaches in terms of time spent on recording a checkpoint. But it required more time in the recovery procedure. EFS presented checkpointing and recovery times close to EBS but with higher monetary costs than the other services. S3 proved to be the best option in terms of monetary cost but required a longer time for recording a checkpoint, individually. However, when concurrent checkpoints were analysed, which can occur in a real application with lots of tasks, in our tests, S3 outperformed EFS in terms of execution time also.

ACM Reference Format:

Luan Teylo, Rafaela C. Brum, Luciana Arantes, Pierre Sens, and Lúcia Maria de A. Drummond. 2020. Developing Checkpointing and Recovery Procedures with the Storage Services of Amazon Web Services. In *49th International Conference on Parallel Processing - ICPP : Workshops (ICPP Workshops '20), August 17–20, 2020, Edmonton, AB, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3409390.3409407

ICPP Workshops '20, August 17–20, 2020, Edmonton, AB, Canada

ACM ISBN 978-1-4503-8868-9/20/08...$15.00
https://doi.org/10.1145/3409390.3409407

1 INTRODUCTION

Cloud computing platforms have consolidated themselves as an accessible means of acquiring computational resources. Today, these environments offer numerous services ranging from processing and storage to data analysis and computational intelligence. Moreover, clouds allow almost instantaneous allocation and release of computational resources in a pay-per-use model to handle user's needs. One of the main ways of using clouds is the direct acquisition of computational resources (Infrastructure as a Service). In this model, resources are offered as virtual machines (VMs), in which users have full control over the installation and configuration steps. These VMs are offered with different contracting models, in which the availability and the Quality of Service (QoS) guarantees vary according to their prices.

In Amazon EC2 (Elastic Compute Cloud), for example, users can choose the following hiring models: i) on-demand, where users do not need a long-term commitment and the VMs have fixed prices and high availability; ii) reserved, where high availability VMs are offered by a low price (up to 75 %) compared to the on-demand market, but it requires a long-term commitment of at last one year; and iii) spot, where the spare capacity of the cloud is offered at a low price (up to 90 %) compared to the on-demand market, without a long-term commitment [13]. When opting for the spot market, users have access to the same computing power offered in the on-demand market, but at a fraction of its price. However, as the VMs in the spot market are subject to revocation by the provider, the adoption of fault tolerance (FT) techniques is essential to minimize job losses resulting from VM's interruptions. When a spot VM is requested, the user can choose its revocation behavior. One of the options is the hibernation feature [2]. When a spot VM is hibernated, its memory and context are saved, and the VM is interrupted temporarily by EC2 without any previous warning. In this case, the VM can only be resumed by the provider, and when it happens, all tasks that were executing at the moment of the hibernation continue its execution from that point. Note that, during the time a VM is hibernated, the user is not charged for it.

Several FT approaches have been proposed to deal with the revocation of spot VMs [10, 20, 22]. One of the most widely adopted technique is the checkpoint/restart approach. It allows the state of

Amazon Simple Storage Service (S3)

- Can be used to store and recover any amount of data
- Provides storage to a wide range of objects sizes, up to 5TB
- The price of each stored GB per month is US$0.023 [a]

[a] price of standard class in region us-east-1 (April 2020)

Amazon Elastic Block Store (EBS)

- Offers local storage volumes with capacity varying from 1GB to 16TB
- EBS volumes are persistent and can be kept even without any VM associated with it
- Price of US$0.10 per GB per month[a]

[a] price in region us-east-1 (April 2020)

Amazon Elastic File System (EFS)

- Provides a scalable file system
- Compatible with the Network File System version 4 (NFSv4.0 or NFSv4.1).
- Charges $0.30 per GB stored per month[a]

[a] price of frequently access class in region us-east-1 (April 2020)

Evaluating AWS Storage Services

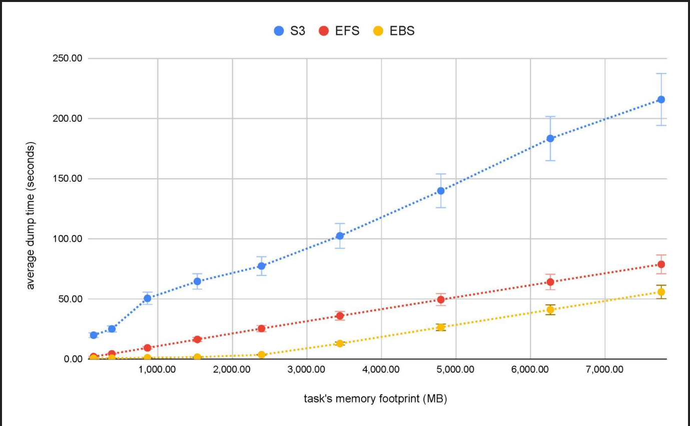
Dump Time

Dump time: time spent writing out the checkpoint files.
Set of synthetic tasks with memory footprint varying from 140 MB to 7,750 MB (one task by memory size).

Evaluating AWS Storage Services

Dump Time Without Concurrence

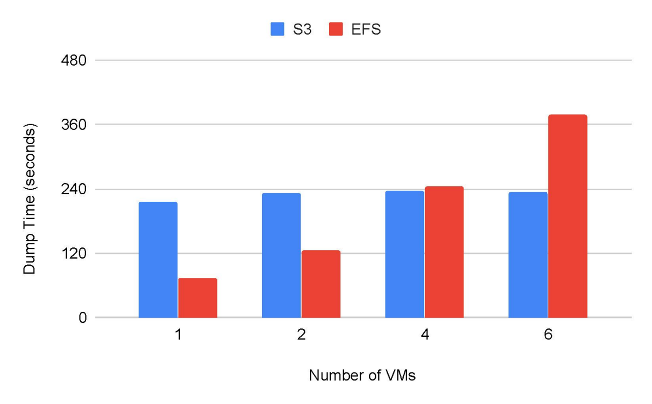
The dump time with S3 presented an increment of **72.57%** and **89.37%** on average when compared to EFS and EBS, respectively



Evaluating AWS Storage Services

Dump Time With Concurrence

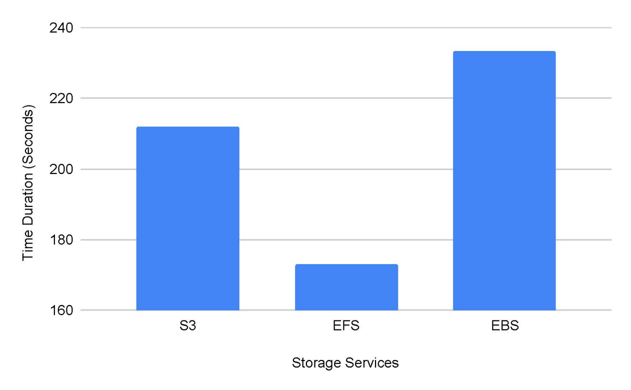
In the scenario with six VMs, the dump time with concurrent checkpoint recording increased **37.89%** with EFS in comparison to S3



Evaluating AWS Storage Services

Recovery Procedure Evaluation

The time of EBS is **9.14%** higher than S3 and **25.86%** higher than EFS



Evaluating AWS Storage Services

Monetary Cost for Long-Running Tasks

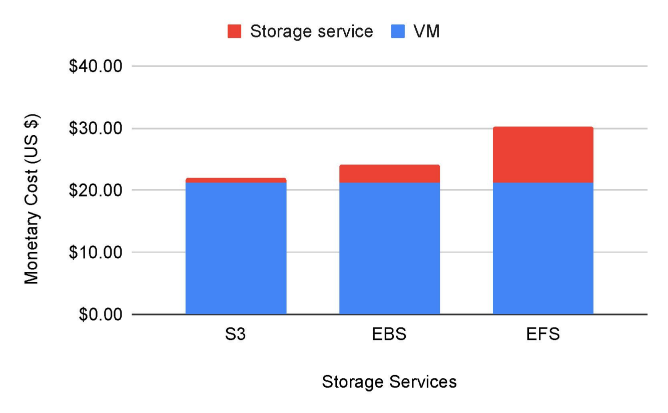
An application with only one task executing for 30 days without any interruption or revocation and storing 30GBs of data

Checkpoint Interval (h)	# of Checkpoints	Total Execution Time (h)			Total Monetary Cost (US$)		
		S3	EBS	EFS	S3	EBS	EFS
1	720	763.14	731.16	735.75	$23.13	$24.50	$30.64
5	144	728.63	722.23	723.15	$22.11	$24.24	$30.27
10	72	724.31	721.12	721.57	$21.99	$24.20	$30.22
15	48	722.88	720.74	721.05	$21.94	$24.19	$30.20
20	36	722.16	720.56	720.79	$21.92	$24.19	$30.20
25	28	721.68	720.43	720.61	$21.91	$24.18	$30.19

Evaluating AWS Storage Services

Monetary Cost for Long-Running Tasks

While the user pays **US$0.69** for the 30 GBs stored for 30 days in S3, in EBS and EFS those costs are **US$3.0** and **US$9.01**



Some Conclusions

Optimization algorithms play a fundamental role to solve scheduling and dimension problems when we consider HPC applications (many tasks) on clouds (several types of VMs and markets).
Dynamic approaches are also necessary!

Spot Vm is an interesting alternative to reduce financial costs

- but fault tolerance mechanisms inside the application or offered by a framework are necessary
- the simple storage service seems to be more attractive when we consider financial costs and spent time for recording and recovering concurrent checkpoints

Future work

- Multi-cloud
- Other models of Applications
    - Federated Learning
    - Map-Reduce (Spark)
- Analyses of HPC as a Service
    - Financial costs
    - Usability
    - Execution time

R. Brum, L. Drummond, M. C. Castro and G. Teodoro, “Towards Optimizing Computational Costs of Federated Learning in Clouds,” in 1st Workshop in Cloud Computing (WCC 2021)

A. L. Nunes, A. C. M. A. Melo, C. Boeres, D. de Oliveira and L. M. A. Drummond, “Towards Analyzing Computational Costs of Spark for SARS-CoV-2 Sequences Comparisons on a Commercial Cloud,” in XXII Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD 2021)

TEAM

Rafaela Brum

Luan Teylo

Maicon Melo Alves

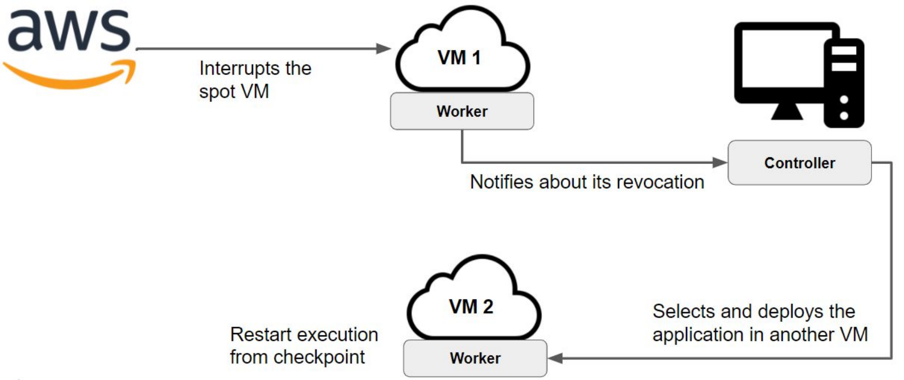
Rafaelli Coutinho

Alan Lira

Thank You

lucia@ic.uff.br

Migrating to a new VM



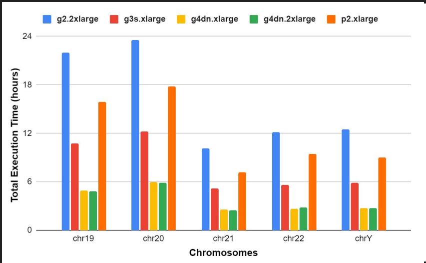
Framework Overhead

- Comparing to Spot *g4dn.xlarge* execution
- Less than 3% of overhead

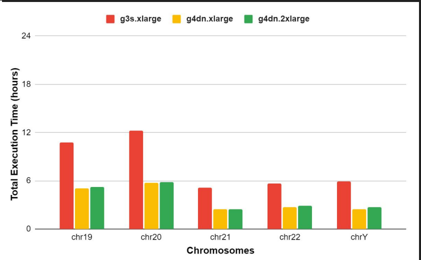
Seq.	**With framework**			**Without framework**	
	M^D	**Exec. time**	**Cost**	**Exec. time**	**Cost**
chr19	22:00:00 (79,200 s)	05:10:05	\$0.81	05:01:55	\$0.79
chr20	24:00:00 (86,400 s)	05:45:07	\$0.90	05:44:30	\$0.91
chr21	10:00:00 (36,000 s)	02:28:12	\$0.39	02:26:04	\$0.38
chr22	12:00:00 (43,200 s)	02:45:58	\$0.43	02:40:54	\$0.42
chrY	10:00:00 (36,000 s)	02:33:11	\$0.40	02:26:37	\$0.39

Results without the framework

- Similar execution times on on-demand and Spot VMs



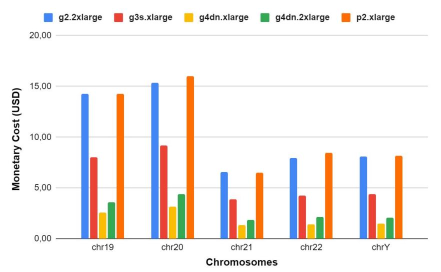
On-demand VMs



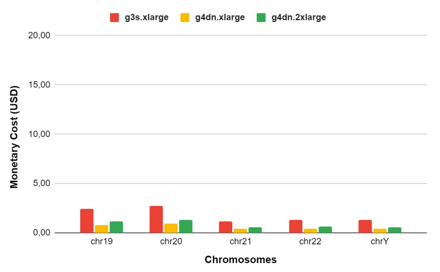
Spot VMs

Results without the framework

- Costs reduction of up 72% with spot instances
  - *g2.2xlarge* and *p2.xlarge* suffered recurrent revocations



On-demand VMs



Spot VMs

Alumni and Current Students

Alan Lira
Luan Teylo
Maicon Melo Alves
Rafaela Brum
Rafaelli Coutinho

Challenges in Application Execution on Clouds

Portability

- Lack of application and code documentation
- Scientific applications have very specific problem domains
- Migration requires a high level knowledge of application
- Applications developed without version control or comments creates a bottleneck in migration (legacy code)

Resources

- Network
  - Scaling, resilience
- Storage
  - Local and distributed file systems
- Processing
  - Number and types of instances

Storage Services

AWS offers several storage services with different features and purposes.