Amazon Elastic Compute Cloud (EC2) vs. in-House HPC Platform: a Cost Analysis

Joseph Emeras*, Sébastien Varrette†, Valentin Plugaru† and Pascal Bouvry†
*Interdisciplinary Centre for Security Reliability and Trust
†Computer Science and Communications (CSC) Research Unit
6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg, Luxembourg
Firstname.Name@uni.lu

***Abstract*—While High Performance Computing (HPC) centers continuously evolve to provide more computing power to their users, we observe a wish for the convergence between Cloud Computing (CC) and HPC platforms, with the commercial hope to see CC infrastructures to eventually replace in-house facilities. If we exclude the performance point of view where many previous studies highlight a non-negligible overhead induced by the virtualization layer at the heart of every Cloud middleware when running a HPC workload, the question of the real cost-effectiveness is often left aside with the intuition that, most probably, the instances offered by the Cloud providers are competitive from a cost point of view.**

In this article, we wanted to assert (or infirm) this intuition by analyzing what composes the Total Cost of Ownership (TCO) of an in-house HPC facility operated internally since 2007. This TCO model is then used to compare with the induced cost that would have been required to run the same platform (and the same workload) over a competitive Cloud IaaS offer. Our approach to address this price comparison is three-fold. First we propose a theoretical price - performance model based on the study of the actual Cloud instances proposed by one of the major Cloud IaaS actors: Amazon Elastic Compute Cloud (EC2). Then, based on the HPC facility TCO analysis we propose a hourly price comparison between our in-house cluster and the equivalent EC2 instances. Finally, based on the experimental benchmarking on the local cluster and on the Cloud instances we propose an update of the former theoretical price model to reflect the real system performance. The results obtained advocate in general for the acquisition of an in-house HPC facility, which balances the common intuition in favor of Cloud Computing (CC) platforms, would they be provided by the reference Cloud provider worldwide.

***Index terms*—Computers and information processing, Cloud Computing / High performance Computing.**
Engineering management, Economics, Costs.

I. Introduction

MANY public or private organizations have departments and workgroups that could benefit from High Performance Computing (HPC) resources to analyze, model, and visualize the growing volumes of data they need to conduct business. From a general perspective, HPC is essential for increased competitiveness and stronger innovation. There are two realistic scenarios today to access HPC capacities. One option consists in acquiring and operating an HPC system. However, for many companies and especially SMEs, this is seen as a non-viable solution since the Total Cost of Ownership (TCO) is perceived as too high and additional skills and man-power are needed to operate and maintain such a system. With the rapidly growing enthusiasm around the Cloud Computing (CC) paradigm, and more particularly of the Infrastructure-as-a-Service (IaaS) model which is best suited for HPC workload, a second viable option is foreseen and attracts more and more attention due to the massive advertisement toward the cost-effectiveness of this approach. In this model, users rent from providers a *resource* that include computing, storage and network or higher level services. At present, many large actors of the Cloud Computing market propose an IaaS service to their users, i.e. the possibility for users to rent virtual machines and other services such as network and storage to the provider. The cost model of IaaS is based on the actual resource usage of the user, who is thus billed depending on his activity. The computing resources are operated upon virtual machines and ran on a multi-tenant mode on the physical hardware. Characterized by their scalability and high-availability, IaaS platforms tend to be more and more efficient and are now sliding towards the territory of the traditional HPC facilities.

Because of their interesting characteristics and in despite of an overall lower performance, many IaaS implementations have been largely studied and benchmarked in the context of an HPC usage. While it is now widely established that running an HPC workload on top of IaaS resources induces a non-negligible performance overhead due to the hypervisor at the heart of every Cloud middleware, many people assume that this performance impact is counter-balanced by the massive cost savings brought by the Cloud approach. *But is it true?*
We propose in this paper a TCO analysis of an in-house HPC facility, instantiated by our own HPC facility [1] operated at University of Luxembourg (UL) since 2007, to enable the comparison of running a workload on the cloud or in-house premises. Moreover, although the comparative performance of Cloud vs. HPC systems received a wide audience in the recent literature, the analogous cost-coverage analysis remains at an

Manuscript submitted on February, 18th 2016; revise on July 16th, 2016. This work is an extended, improved version of the paper "Amazon Elastic Compute Cloud (EC2) vs. in-House HPC Platform: a Cost Analysis" presented at IEEE Cloud 2016 and published in: 9th IEEE International Conference on Cloud Computing, Cloud 2016, San Francisco, US, June 27 - July 2, 2016.

early stage, with very few contribution from the academic community. This is also due to the constant and rapid evolution of the Cloud instances offered by the different providers and the frequent price changes, making it hard to establish a fair comparison of the Cloud usage price with regards the equivalent HPC infrastructure operational costs. Furthermore the direct comparison of a given Cloud instance price with an HPC operating cost is biased and should be taken with precautions as it omits the performance aspect where the Cloud instance performance does not match with the one of an HPC node. It is to feed this gap that we propose in this article a new cost analysis model. Our approach to address this price comparison is threefold. First we propose a theoretical price - performance model based on the study of the actual Cloud instances proposed by one of the major Cloud IaaS actors according to Gartner's 2015 Quadrant[1]: Amazon Elastic Compute Cloud (EC2). Then, based on the TCO analysis of an HPC facility we propose a hourly price comparison between our in-house cluster and the EC2 instances. Finally, based on the experimental benchmarking on the local cluster and on the Cloud instances we propose to update the theoretical price model to reflect the real system performance.

This article is organized as follows: Section II reviews the related work as regards the cost analysis of Cloud platforms when compared to HPC facilities. Then, Section III provides a TCO analysis of an in-house HPC facility, summarize the characteristics of our in-house HPC with a focus on the different capital and operational costs required for the TCO estimation. A first contribution *i.e.* the TCO analysis of an HPC center and its instantiation with our platform at UL (reported in an average hourly node cost) is reported in this section. The Amazon EC2 instances, especially the ones relevant for HPC workloads and thus suited for a comparison with in-house HPC facilities, are depicted in Section IV. Then we propose a novel insight over Amazon EC2 instance prices through a new cost model that matches almost perfectly the corresponding computing performance. This new approach and its theoretical validation is depicted in Section V. Furthermore, we apply this novel cost model to propose a fair comparison with the TCO analysis performed over our in-house HPC facility in Section VI. This permits to conclude on the opportunity to privilege one or the other solution when considering the acquisition (or the renting) of an HPC platform. Sections VII and VIII respectively propose a performance analysis and comparison of the computing and storage aspects on EC2 and our in-house cluster. This performance study is then used to update the cost model to reflect the systems' actual performance. Finally, Section IX concludes the paper and provides some future directions and perspectives opened by this study.

II. Related Work

A. Cloud Performance Evaluation for HPC Applications

Evaluating the adequateness and the performance of cloud platforms for HPC applications has been largely studied in various ways. Many works addressed this problem using either synthetic benchmarks, real applications or even both to quantify the performance impact when running over an hypervisor (such as Xen, KVM, VirtualBox or VMWare), a Cloud middleware (such as OpenStack) or a commercial Cloud IaaS [2], [3], [4], [5], [6], [6], [7], [5], [8], [9], [10], [11], [12]. Depending on the experimental procedures, benchmarks and input data chosen, the loss due to the cloud varies among these experiments, but all agree that there exist a performance drop at scale.

This performance drop has been explained in several works. In [10], a real world climate prediction from the NASA was used to compare three instances from different public Cloud providers with NCSA local clusters. It was discovered that virtualization technology induced minimal overhead and that the main performance drop was due to poor networking capabilities of Clouds.

The same test has been driven in [11] on EC2 *cc1.4xlarge* instances and the conclusion was similar. For single node, EC2 might outperform HPC clusters but at scale and for application requiring intensive communications, the lack of Infiniband on EC2[2] resulted in significant performance loss. In [13], A. Gupta et al. quantified the virtualization overhead in terms of CPU on EC2 for *cc2.8xlarge* instances, as only 1-5% but showed that the network performance is a dominant factor. They also observed that for these instances, the network emulator was residing on core 0 on the Virtual Machines (VMs). According to [14], core 0 is also where all other interrupts are pinned to. This is problematic as it disrupts the processes running on this core and thus make it unsuitable for running HPC applications on it.

In [15], the performance evaluation of EC2 *cc1.4xlarge* instances based on real world climate and cardiac simulation applications along with the NPB benchmark suite showed that not only the interconnect is determinant but also the underlying file system, even on application that are classified as not strongly I/O intensive. In this work it was also observed that jitter due to virtualization has a minor effect on the applications performance.

In all these works the instances for the experiment could not benefit from the most recent Enhanced Networking in EC2. This feature, based on Single Root I/O Virtualization (SR-IOV) is an enhanced virtualization technology that promises to make high performance IO in virtualization by providing an uninterrupted path between the physical device and the VM. In [16], G. Lockwood et al. benchmarked the performance gain by activating or not this feature on a 10 Gb and Infiniband network. The conclusion of this work is that even though the performance gain in terms of latency and bandwidth is dramatic for Infiniband, it is almost negligible for 10 Gb.

It is recognized that Cloud platforms performance does not fully match the requirements of HPC applications for the moment. This is due in particular to the Cloud interconnect capacity having a poor performance compared to HPC. However as proposed in [13], some applications may still benefit from the cloud if their workload does not involve

[1] https://www.gartner.com/technology/reprints.do?id=1-2G2O5FC&ct=150519&st=sb

[2] best network interconnect available at this time is 10 Gb.

latency-bound or communication-bound patterns. But another important concern is the variability of performance coming from the intrinsic property of the cloud being location independent and multi-tenant [17]. This aspect could be tackled by software techniques such as the one described in [18], where the authors proposed a dynamic load balancer to reduce this variability for tightly-coupled iterative HPC applications based upon Charm++ and to be ran on the Cloud.

B. HPC vs. Cloud Cost Analysis and Modeling

If it is currently established that Cloud platforms performance does not fully match the requirements of HPC applications as would do a physical HPC facility, the acclaimed cost effectiveness of the CC paradigm might counter-balance this performance overhead. Actually, several works already attempted to describe or model the cost of using a public cloud infrastructure instead of an HPC facility. In [14] the EC2 *cc2.8xlarge* instance is used as reference with *on-demand* pricing to compare the price and performance of the Cloud versus five in-house clusters. To be able to do this comparison, they propose a cost model for HPC cluster nodes. As the authors could not get or distribute data describing the cost of ownership, they propose an alternative model. Based upon the execution times of four NPB Benchmarks on their EC2 reference instance and local clusters, the alternative model proposes by the means of linear regression to attribute a node hourly price to each of their clusters. This approach has the advantage to be able to price cluster nodes, however it is now outdated as EC2 offers several instances that would fit an HPC workload and not only the sole *cc2.8xlarge* instance that was available at this time. Thus this model should be updated taking into consideration the instances evolution. Another lack of this study is that only *on-demand* pricing has been study although it may not be the most adequate pricing type when the workload is considered on a long period (months). In that case, the Cloud price considered is artificially too high due to the pricing type, only adapted for short workloads. Thus the reserved pricing types should also be taken into account in the study.

In [13], the authors propose a pricing model based on the study of cost and prices collected from several supercomputers installations and cloud offerings. The results of this comparison was that the cost ratio between an on-premise supercomputing and a cloud deployment is between 2 and 3 (meaning that one cluster core-hour cost is 2 to 3 times more expensive than a cloud's core-hour). Finally to better fit what may happen in reality due to price fluctuations they select a ratio between 1 to 5. Then, based on this model they compared the cost of running three applications on the cloud or on a cluster. According to this comparison, there is little sensitivity in the choice of the ratio and it is straightforward to determine the price break-even point. However, because of how the price model is computed the authors warned that this economic comparison should be taken conservatively. This motivates our approach of using real cost data for operating a local cluster.

In [19] the authors analyzed the cost of the Amazon EC2 cloud from different point of view: Memory size or CPU performance. They propose rankings of the instances based on their price regarding a given metric (i.e. memory size and bandwidth or cpu performance). What was observable was that instances were designed to be cost effective for one of the metrics only. This observation motivates our approach to include several instance types in the price model instead of a single one.

The work proposed in [20] approximates the cost of the local cluster nodes based on the EC2 *cc1* hourly price. To compute this node hourly price, they scaled the *cc1* price in order to reflect the performance differences between their cluster and the cloud instance. The authors warned that the price determined in this way should not be taken as a literal per node-hour cost. Despite the lack of representativeness of this approach it enabled the authors to determine a node hourly price for both *on-demand* and the different *reserved* pricing modes of EC2. In particular this showed that the users with the largest utilization rates could benefit from the reservation pricing mode on the *cc1* instance. Compared to the cluster operating cost it could lead to savings up to 3.75% for a 1-year contract.

Finally in [9], the comparison between the local cluster and the EC2 *cc1* instance was more straightforward as both had the same performance. The data taken into account for cluster pricing estimation was acquisition and operational expenses. However the authors approach diverge from ours and propose a price model of their in-house cluster based on the actual utilization of the cluster by the users. The claim here is that increased cluster utilization lowers the cluster cost. This is true if the point of view is job centric, however on a pure owner expense and operational vision it is wrong. The actual OPerating EXpense (OPEX) increases with the utilization as more energy is consumed by the nodes and thus the TCO is higher than for lower utilization rates. We propose first to compute the actual node hourly price based on the TCO with full utilization of the cluster to get the maximum cost. As the cost of renting an instance on the cloud is currently not dependent on the energy used (whether the instance is used at full load or not the price remains the same), it is thus fairer to compare the cloud instance price with the cost of operating the cluster at full load. With this information we can directly compare the cluster and cloud hourly costs. Then, based on the observation of the workload of the cluster we can easily determine the price per job and thus take into account the utilization in the final cost evaluation.

In [21], the authors compare the cost of running the same workload both on a cloud and on an existing campus cluster of non-dedicated resources. They take as reference instance the EC2 *c1.medium* as its performance is close to their local nodes and use the *on-demand* pricing mode. In the local cluster cost model, the costs taken into account are manpower, hardware costs (including switches, servers and computing nodes) and energy costs. The conclusion was that operating their workload on the cloud would be more expensive but would give a better throughput as there are no queue wait time in the cloud. The other interesting conclusion was that the cost of the cloud could be easily dominated by data management costs; about a third of the cost incurred by their workload, if ran on the cloud, would be data transfer only. This, added to data storage prices

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2628371, IEEE Transactions on Cloud Computing

should be taken into account to ensure that costs on the cloud are kept under control.

III. TCO Analysis of an HPC facility

While operating an HPC center, the induced cost does not come only from the buying of the machines that will support the workload. These machines have to be hosted within a dedicated and adequately equipped facility, they need to be interconnected and provided with a set of software suite enabling the users to properly use their computing power. Human and energy costs are also part of this global cost of operating an HPC center. In order to compute the actual TCO of operating such an HPC center we propose here an analysis of all the costs that are accountable in this operation.
The TCO of an HPC facility is decomposed into the different assets bought to run the infrastructure and which are accountable in the CApital EXpense (CAPEX) along with the different expenses necessary to provide the service and which are composing the OPEX. More precisely, CAPEX and OPEX for such an HPC facility are composed of the following items:

CAPEX

- machines
- servers
- storage
- interconnect
- room equipment
- building estimation

OPEX

- manpower
- energy (power and cooling)
- support
- software licenses

The different TCO items also have different depreciation periods. On the CAPEX side, computing nodes are considered amortized on a 4 year basis, servers on 3 years, storage on 5 years and network equipment on 8 years. Server rooms and their equipment on their side are supposed to be amortized in 15 years whereas the building in itself on 30 years. On the OPEX side, most costs (such as energy, manpower and cooling) are amortized on a 1 year basis and some software licenses and support are on a 3 years basis.

This TCO analysis will serve to compute our in-house HPC TCO and then, reported with the correct amortization period for each item, enable to compute a node hourly cost.

Presentation of the UL HPC Platform

The University of Luxembourg (UL) operates since 2007 an HPC facility [1] and the related storage by a small team of system administrators. The aspect of bridging computing and storage is a requirement of UL service – the reasons are both legal (certain data may not move) and performance related. Nowadays, people from the three faculties and/or the two Interdisciplinary centers within the UL, are users of this facility. At the time of writing, it consists of 4 clusters spread across 2 geographic sites featuring a total of 400 computing nodes (for 4284 cores) and a cumulative shared storage capacity of 4270 TB. In this study, we restrict the data and metrics collected to the two main clusters (namely *chaos* and *gaia*) that compose UL HPC facility – they contribute up to 80% (resp. 98%) of the total platform computing (resp. storage)

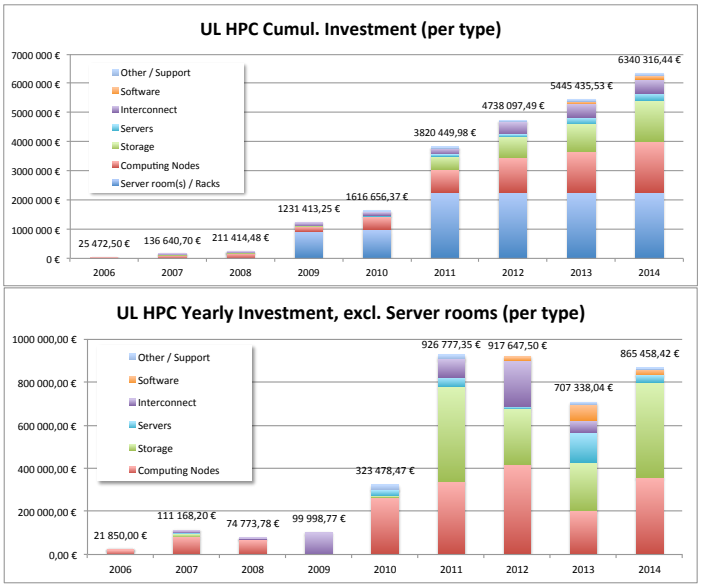


Figure 1. Investment in the UL HPC Facility for the time period 2007–2014. top: cumulative, incl. server rooms; bottom: yearly, excl. server rooms.

capacity. Within these clusters, the internal interconnect is based on an Infiniband QDR (40Gb/s) network built on top of and adapted topology ("fat tree" and/or "star" depending on the situation) to ensure the best performances for the process communications. It is worth to note that such a technology out-performs by far all Ethernet-based network in use among major Cloud providers (including Amazon). In all cases, the cumulative Hardware investment is now reaching 6.34 M€ and its evolution is depicted in the figure 1. The manpower is also a piece of information often missing in many cost analysis. In the table I, these costs are highlighted in the case of our facility. They reflect a reality where despite all our efforts, we were (and are still) outnumbered. While the TCO analysis proposed in this article relies on these numbers (more precisely, on the normalized prices for the year 2014), we estimate to 7.35 FTEs (thus $\simeq$ 88 Person Months (PMs)) the ideal manpower that would be required to maintain the platform at its current sizing.

Table I. UL HPC Manpower

Year	PM effort	Total Annual Cost
2006	1	11 553,63 €
2007	4	32 118,40 €
2008	19	64 200,24 €
2009	13	60 450,24 €
2010	9	57 864,00 €
2011	13	87 379,20 €
2012	25	136 947,68 €
2013	49	194 280,10 €
2014	40	173 026,90 €

Another crucial question when performing a TCO analysis is the hypothesis linked to the usage of the platform. In our case, we have the chance to ensure a consistent and precise monitoring of the HPC resource usage (CPU, memory etc.) with regards the submitted jobs. For instance, a typical usage of the *gaia* cluster resources is illustrated in the figure 2. It

Table II. CHAOS AND GAIA COMPUTING NODES CHARACTERISTICS AND ESTIMATED TCO EVALUATION PER COMPUTING NODE.

	Node	CPUs	Memory GB	GPUs	Nb. Nodes	CPU Family	CPU Type	CPU Clock	Disk GB	GFLOPS	**Hourly Cost ($)**
CHAOS	d-cluster1	12	24	0	16	westmere	L5640	2.26	250	108.48	0.439
	e-cluster1	16	32	0	16	sandybridge	E5-2660	2.20	250	281.60	0.433
	h-cluster1	12	24	0	32	westmere	L5640	2.26	250	108.48	0.428
	r-cluster1	32	1024	0	1	nehalem	X7560	2.26	250	289.28	1.814
	s-cluster1	16	32	0	16	sandybridge	E5-2660	2.20	250	281.60	0.433
GAIA	gaia-[1-60]	12	48	0	60	westmere	L5640	2.26	256	108.48	0.453
	gaia-[123-154]	12	48	0	32	westmere	X5675	3.07	256	147.36	0.344
	gaia-[61-62]	12	24	1792	2	westmere	L5640	2.26	256	108.48	0.641
	gaia-[63-72]	12	24	10240	10	westmere	L5640	2.26	256	108.48	0.599
	gaia-[75-79]	16	64	12480	5	sandybridge	E5-2660	2.20	256	281.60	0.577
	gaia-[83-122]	12	48	0	40	westmere	X5670	2.93	256	140.64	0.344
	gaia-73	160	1024	0	1	sandybridge	E7-4850	2.00	256	2560.00	2.649
	gaia-74	32	1024	0	1	sandybridge	E5-4640	2.40	256	614.40	1.516

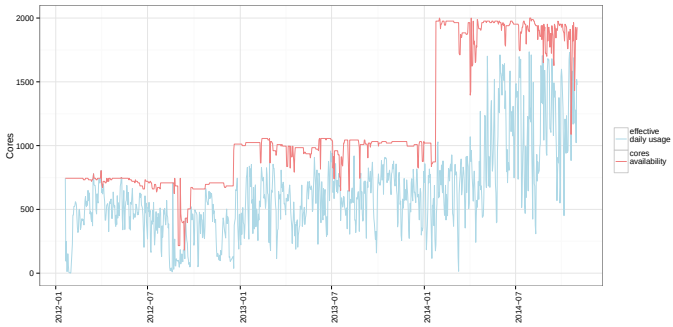


Figure 2. Example of used vs. available computing resources within the `gaia` cluster.

permits to derive an accurate ratio in terms of *used resources* with regards to the available computing components.

We use the HPC TCO analysis detailed in the beginning of this section to compute our platform TCO. The only item were we could not obtain a real price to integrate into the TCO is the building cost itself. The HPC center is hosted within the UL which, as a research institutions, benefits from governmental funding for its buildings and it is difficult to fairly estimate the price of the building attributed to our HPC center. To tackle this lack of information we use the data center building estimation tool provided by Intel[3] to estimate the building cost according to our infrastructure size. For 14 cabinets, 250 KW of power needed and 100 square meters necessary to host the cabinets we obtained an estimation for the building costs of 3 million US Dollar. According to Intel these include an estimation of all the engineering, construction and building equipment necessary to host a data center.

From this information, we are able to estimate the yearly TCO of our local HPC facility, amortized according to the different depreciation periods of the TCO items detailed in the beginning of this section. As `chaos` and `gaia` clusters are heterogeneous and composed of different node classes, as reported in Table II, we compute the hourly cost of each node depending on their class. For obvious privacy reasons, we cannot disclose the detailed CAPEX costs, yet the final amortized node hourly costs are reported in the last column of the Table II. To facilitate the comparison with Amazon EC2 instances prices, we reported this hourly cost in US Dollar rather than in Euro and costs are given as VAT exclusive.

[3]http://www.thecloudcalculator.com/calculators/build-vs-buy.html

IV. AMAZON EC2 INSTANCES FOR HPC

Since its launch in 2006 with the *m1.small* instance, Amazon has largely expanded the variety of instances and features it provides through its Elastic Compute Cloud (EC2) offer. They are now proposed in several families, each corresponding to a different computing need. Along the years and at new instance type releases, Amazon decreased the prices of the already existing ones. To ensure backward compatibility, most older instances are still distributed, however they do not provide a good price/performance ratio anymore regarding the most recent ones. Table III presents all the available instance types and their release dates as of early 2015 (except for the cc1.4xlarge which is not available anymore). An instance type e.g. *m1* or *c1* belongs to an instance family, e.g. *General Purpose* or *Compute Optimized*. For each instance type, there exist one or several models available that are not presented in this table. Instance models are described by an extension of the instance type (e.g. *m1.small*), possible values are: micro, small, medium, large, xlarge, 2xlarge, 4xlarge, 8xlarge. A given instance model corresponds to a particular performance

Table III. EC2 INSTANCE TYPES – GROUPED BY FAMILY.

Instance Family	**Instance Type**	**Processor Microarchitecture**	**Introduction Date**
General Purpose	m1	Xeon Family	2006-08-23
	m3	Ivy Bridge-EP	2012-10-31
	t2	Xeon Family	2014-07-01
	m4	Haswell-EP	2015-06-11
Memory Optimized	m2	Xeon Family	2010-02-22
	cr1	Sandy Bridge-EP	2013-01-21
	r3	Ivy Bridge-EP	2014-04-10
Compute Optimized	c1	Xeon Family	2008-08-08
	cc1	Nehalem-EP	2010-07-13
	cc2	Sandy Bridge-EP	2011-11-14
	c3	Ivy Bridge-EP	2013-11-14
	c4	Haswell-EP	2014-11-13
Storage Optimized	hi1	Xeon Family	2012-07-18
	hs1	Sandy Bridge-EP	2012-12-21
	i2	Ivy Bridge-EP	2013-12-20
Dense Storage	d2	Haswell-EP	2015-03-30
GPU	cg1	Nehalem-EP	2010-11-14
	g2	Sandy Bridge-EP	2013-11-05
Micro	t1	Xeon Family	2009-10-26

Table IV. EC2 INSTANCES SUITABLE FOR HPC WORKLOAD AND THEIR KEY CHARACTERISTICS.

Family	Type	ECUs	GFLOPS	vCPUs	Processor	Clock	Memory	Storage	GPUs	EBS-Opt	Network	SR-IOV
Compute optimized	c3.2xlarge	28.0	179.2	8	Xeon E5-2680v2	2.8	15.00	2 x 80 (SSD)	0	TRUE	1 Gib	TRUE
Compute optimized	c3.4xlarge	55.0	358.4	16	Xeon E5-2680v2	2.8	30.00	2 x 160 (SSD)	0	TRUE	1 Gib	TRUE
Compute optimized	c3.8xlarge	108.0	716.8	32	Xeon E5-2680v2	2.8	60.00	2 x 320 (SSD)	0	FALSE	10 Gib	TRUE
Compute optimized	c3.large	7.0	44.8	2	Xeon E5-2680v2	2.8	3.75	2 x 16 (SSD)	0	FALSE	100 Mib	TRUE
Compute optimized	c3.xlarge	14.0	89.6	4	Xeon E5-2680v2	2.8	7.50	2 x 40 (SSD)	0	TRUE	100 Mib	TRUE
Compute optimized	c4.2xlarge	31.0	371.2	8	Xeon E5-2666v3	2.9	15.00	EBS only	0	TRUE	1 Gib	TRUE
Compute optimized	c4.4xlarge	62.0	742.4	16	Xeon E5-2666v3	2.9	30.00	EBS only	0	TRUE	1 Gib	TRUE
Compute optimized	c4.8xlarge	132.0	1670.4	36	Xeon E5-2666v3	2.9	60.00	EBS only	0	TRUE	10 Gib	TRUE
Compute optimized	c4.large	8.0	92.8	2	Xeon E5-2666v3	2.9	3.75	EBS only	0	TRUE	100 Mib	TRUE
Compute optimized	c4.xlarge	16.0	185.6	4	Xeon E5-2666v3	2.9	7.50	EBS only	0	TRUE	100 Mib	TRUE
Compute optimized	cc2.8xlarge	88.0	665.6	32	Xeon E5-2670	2.6	60.50	4 x 840	0	FALSE	10 Gib	FALSE
Dense storage	d2.2xlarge	28.0	307.2	8	Xeon E5-2676v3	2.4	61.00	6 x 2000	0	TRUE	1 Gib	TRUE
Dense storage	d2.4xlarge	56.0	614.4	16	Xeon E5-2676v3	2.4	122.00	12 x 2000	0	TRUE	1 Gib	TRUE
Dense storage	d2.8xlarge	116.0	1382.4	36	Xeon E5-2676v3	2.4	244.00	24 x 2000	0	TRUE	10 Gib	TRUE
Dense storage	d2.xlarge	14.0	153.6	4	Xeon E5-2676v3	2.4	30.50	3 x 2000	0	TRUE	100 Mib	TRUE
GPU instances	cg1.4xlarge	33.5	185.6	16	Xeon x5570	2.9	22.00	2 x 840	896	FALSE	10 Gib	FALSE
GPU instances	g2.2xlarge	26.0	166.4	8	Xeon E5-2670	2.6	15.00	1 x 60 (SSD)	1536	TRUE	1 Gib	FALSE
GPU instances	g2.8xlarge	104.0	665.6	32	Xeon E5-2670	2.6	60.00	2 x 120 (SSD)	6144	FALSE	10 Gib	FALSE
Memory optimized	cr1.8xlarge	88.0	665.6	32	Xeon E5-2670	2.6	244.00	2 x 120 (SSD)	0	FALSE	10 Gib	FALSE
Memory optimized	r3.2xlarge	26.0	160.0	8	Xeon E5-2670v2	2.5	61.00	1 x 160 (SSD)	0	TRUE	1 Gib	TRUE
Memory optimized	r3.4xlarge	52.0	320.0	16	Xeon E5-2670v2	2.5	122.00	1 x 320 (SSD)	0	TRUE	1 Gib	TRUE
Memory optimized	r3.8xlarge	104.0	640.0	32	Xeon E5-2670v2	2.5	244.00	2 x 320 (SSD)	0	FALSE	10 Gib	TRUE
Memory optimized	r3.large	6.5	40.0	2	Xeon E5-2670v2	2.5	15.00	1 x 32 (SSD)	0	FALSE	100 Mib	TRUE
Memory optimized	r3.xlarge	13.0	80.0	4	Xeon E5-2670v2	2.5	30.50	1 x 80 (SSD)	0	TRUE	100 Mib	TRUE
Storage optimized	hi1.4xlarge	35.0	153.6	16	Xeon Family	2.4	60.50	2 x 1024 (SSD)	0	FALSE	10 Gib	FALSE
Storage optimized	hs1.8xlarge	35.0	256.0	16	Xeon E5-2650	2.0	117.00	24 x 2048	0	FALSE	10 Gib	FALSE
Storage optimized	i2.2xlarge	27.0	160.0	8	Xeon E5-2670v2	2.5	61.00	2 x 800 (SSD)	0	TRUE	1 Gib	TRUE
Storage optimized	i2.4xlarge	53.0	320.0	16	Xeon E5-2670v2	2.5	122.00	4 x 800 (SSD)	0	TRUE	1 Gib	TRUE
Storage optimized	i2.8xlarge	104.0	640.0	32	Xeon E5-2670v2	2.5	244.00	8 x 800 (SSD)	0	FALSE	10 Gib	TRUE
Storage optimized	i2.xlarge	14.0	80.0	4	Xeon E5-2670v2	2.5	30.50	1 x 800 (SSD)	0	TRUE	100 Mib	TRUE

of the instance in terms of vCPU, Memory, Storage, Network or such other performance characteristics.

A. Cloud Drawbacks for HPC Applications

To summarize, the cloud characteristics (which are also valid for EC2) that may impact HPC applications are as follows. (1) Clouds are run in *virtualized environments*. Although the virtualization overhead is not so important for CPU bound applications, the virtualized networking is still an important drawback for applications that need communications (i.e a large part of HPC applications). Even though the networking uses virtualization improvements such as SR-IOV, there exists a performance drop for inter-node communications. (2) There is generally no high performance network such as Infiniband yet available. This is still a problem for many applications whose performance is highly linked with the network performance. (3) Cloud by default uses multi-tenancy. This is a real problem as it does not ensure a reliable and stable behavior for applications. In particular, I/O operations in multi-tenancy platforms can face high I/O jitter. However EC2 also provides a charged *Dedicated Instances* service[4] to ensure the user will have the exclusive access to the nodes reserved. (4) Finally, spatial and link proximity of the reserved VMs is not guaranteed by default. Because of this intrinsic characteristic, inter-node bandwidth and latency can vary a lot. As for multi-tenancy, EC2 also provides for some instances the *placement group* feature, a logical grouping of instances within the same Availability Zone (isolated location within the same region) ensuring a low-latency, 10 Gb network between the nodes.

[4]https://aws.amazon.com/ec2/purchasing-options/dedicated-instances/

B. EC2 Instances for HPC

If we now aim at the execution of an HPC workload, one of the key constraint to address is the need of a reliable and efficient underlying network. Not all EC2 instances offer such a guarantee, however there exists a mechanism called *placement group* available for some of them and that allows the user to group a set of instances in the same cluster. This is an important requirement for HPC applications that use several nodes. Here we consider that most *classical* HPC applications fall into that category and thus need to be placed within the same cluster. In the Table IV, we detail the EC2 instances that allow placement groups and thus may be suitable for such an HPC workload. It is also important to say on an HPC point of view that the nodes that host the EC2 instances have HyperThreading activated, thus each vCPU on an instance is actually a HyperThread. In all cases, Amazon's EC2 provides several mechanisms to target High Performance.

- Enhanced Networking with SR-IOV. This hardware feature, proposed on the most efficient instances, provides higher network performance for the VMs: higher bandwidth and lower network latency and jitter.
  - Compute Optimized: *c3*, *c4*.
  - Memory Optimized: *r3*.
  - Storage Optimized: *i2*.
  - Dense Storage: *d2*.

Unfortunately GPU instances do not provide the Enhanced Networking feature but we have to consider them anyway for an HPC environment.
- Placement Groups (or cluster networking). This feature provides high bandwidth performance with a full bisection Ethernet network.
- Dedicated Instances. By default on EC2, cloud instances are multi-tenant VMs hosted on the same physical node. By opting for the dedicated instance option (paying) at launch time, the instance will have exclusive access to the underlying physical machine.
- EBS-Optimized. Elastic Bloc Store (EBS) is Amazon's persistent storage for EC2 instances (as opposed to instances local storage which is destroyed after the instance termination). It can be purchased with a provisioned IO option to increase and guarantee data transfer rates. EBS volumes have a maximum throughput of 128 MiB/s that can be attained only with instances proposing the EBS-optimized option.

One of the last instance type released by Amazon at this time, the *c4* instance (end of 2014), proposes EBS storage with no additional costs but does not provide local storage.

V. A Novel Price Model binding EC2 Instances with their HPC Performance

A. EC2 Instance Computing Performance Metric

EC2 instance performances are provided in EC2 Compute Units (ECUs). This metric represents the relative measure of the integer processing power of a given instance. Amazon does not give precisely how they measure ECUs, the only information available is that one ECU is equivalent to one Intel Xeon core running at 1.0-1.2 GHz. It is also clearly stated by Amazon that this metric might be revised in the future by adding or substituting measures that go into the definition of an ECU. According to Amazon, the goal of providing instance performance as ECU is to make it easier to compare CPU capacity between different instance types and thus to "*provide a consistent amount of CPU capacity no matter what the actual underlying hardware*".

Although the claim of using several benchmarks and tests to manage the consistency and predictability of the performance from an EC2 Compute Unit, it seems that this information might be sometimes misleading. In [22], the authors showed that for several instance types the performance measured experimentally was unreliable, unpredictable and not reflecting the ECU provided by EC2. This was due to the fact that the instances that were tested in this study were running indifferently on several types of CPU and it seems that for instances whose CPU model is described as being a *Xeon Family*, there is no guarantee of the actual CPU model on which the instance will be ran. In another study driven by Ostermann et al. in [8], the authors experimentally compared the actual GFLOPS regarding the announced ECUs for *m1* and *c1* instances and observed a real performance comprised between 39 to 44% of the advertised performance for *m1* instances and most of *c1* instances with an exception of 58.6% for the *c1.xlarge* instance.

Firstly, ECU is a non standard way to measure a processor performance and it actually makes it difficult to fairly compare the computing performance of an EC2 instance with an HPC node. Moreover, as this metric is not provided with a strict definition and as it was observed performance mismatch between the announced ECU and actual performance we need another metric such as FLOPS which is the traditionally used theoretical processor computing performance metric. As according to its definition, an ECU is equal to one Xeon core at 1.0-1.2 GHz, the ECU should reflect the processor FLOPS value. In order to verify the validity of this definition of ECU, we performed a linear regression of the ECU in function of processor FLOPS and obtained a linear model whose adjusted R^2 value is of 0.9 and a p-value of $2.2e-16$, meaning a good linear fitting and thus showing the collinearity between ECUs and GFLOPS. Because of this lack of information on how is really calculated an ECU and the fact that there exists indeed a linear relationship between ECU and GFLOPS, we propose to use GFLOPS instead of ECU in the sequel. This will enable us to compare the processing performance of EC2 instances and HPC computing nodes with a single metric that is well known and that can actually be measured.

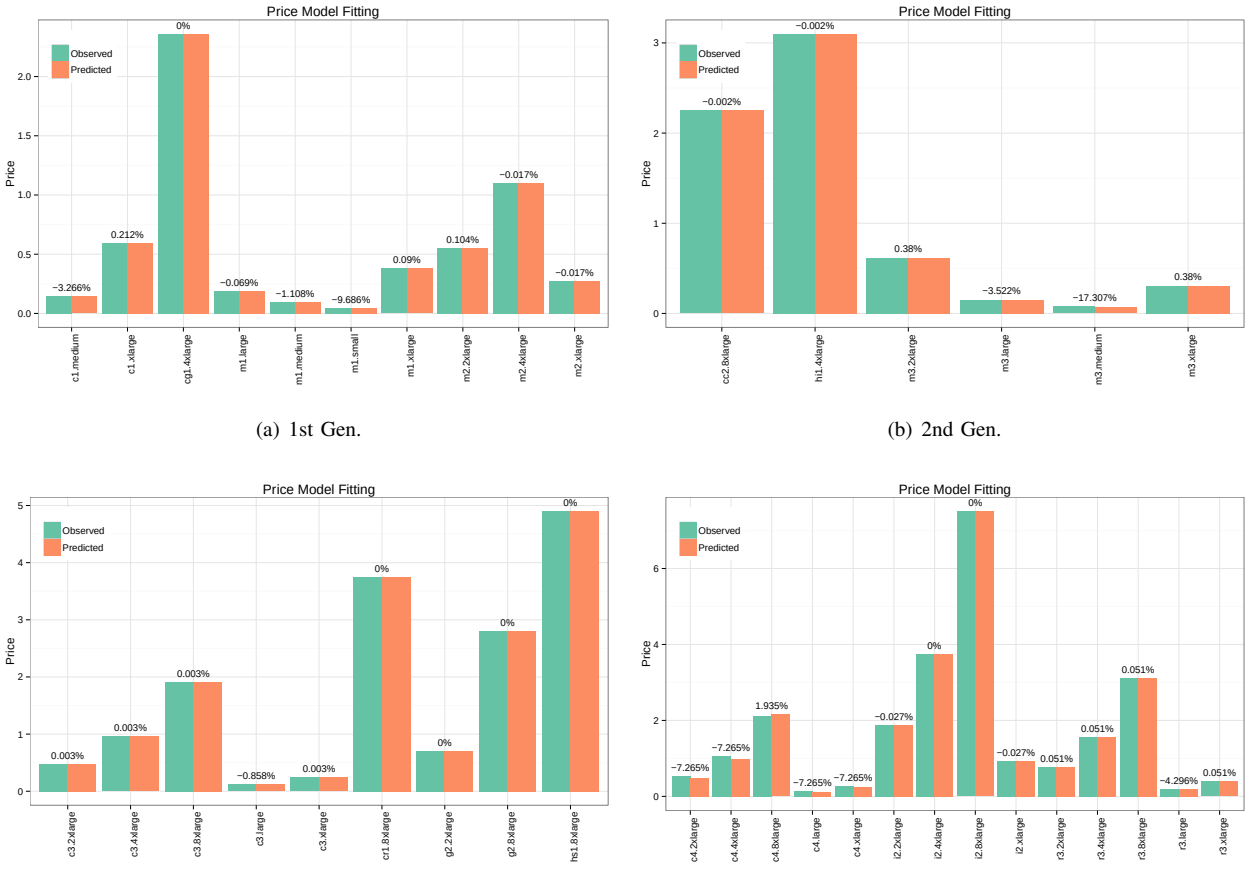
B. Toward a new Amazon EC2 Price Model linked to their respective Computing Performance

Very often in the context of HPC workloads, input and output data should be transferred to/from the computing facility and this often involves a large amount of data. Thus we decided to focus in EC2 instances in Europe. Only two regions are available in Europe at this time, one in Ireland and one in Frankfurt (Germany), we choose to focus on the Ireland region as its instance prices are cheaper and Frankfurt region was launched in the end of 2014 and does not offer all the instance types yet. In order to have a coherent instance price that enables to compare with the cluster costs and taking into account the hardware performance, we propose to use a multiple linear regression approach in order to compute an Amazon EC2 price model. Based on this model we will be able to determine for our local machines a fair EC2 equivalent price that will reflect their hardware performance, as proposed later in the section VI. Note that Amazon offers different purchasing modes for its instances. The "*On-Demand*" mode is the most common and allows to pay a fixed rate by the hour with no commitment. In the rest of this section, we present our computed results for our price model using the proposed rate for this mode. So to establish an accurate price model, we have to take into consideration additional characteristics than the pure peak performance of the considered resource (in GFLOPS). Indeed, we have observed that instances that belong to the same type have prices that tend to be linearly dependent on their performance. *i.e.* instance *c3.4xlarge* is two times more expensive than *c3.2xlarge* but its number of cores, available memory and disk size are two times larger. Thus for each instance type we compute a pricing linear model based on linear regression analysis. We select the model parameters in two steps. (1) First we perform an automated stepwise selection, then (2) from the meaningful parameters detected

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2628371, IEEE Transactions on Cloud Computing

Table V. COEFFICIENTS OF OUR NEW EC2 PRICE MODEL FOR ON DEMAND INSTANCES, WITH FITTING PERFORMANCE EVALUATION.

Model	Types	GFLOPS (α)	MemGiB (β)	DiskGiB (γ)	GPUs (δ)	Adj. R^2	P-Value
1st Gen.	m1, c1, m2, cg1	0.0039522	0.0061130	0.0000670	0.0015395	0.9999909	0e+00
2nd Gen.	cc2, m3, hi1	-0.0035266	0.0355353	0.0007284	0.0000000	0.9999785	1e-07
3rd Gen.	hs1, cr1, g2, c3	0.0017209	0.0106101	0.0000655	0.0001644	1.0000000	0e+00
4th Gen.	i2, r3, c4	0.0009952	0.0081883	0.0007605	0.0000000	0.9998832	0e+00
5th Gen.	m4, d2	0.0000000	0.0173750	0.0000342	0.0000000	1.0000000	0e+00



(a) 1st Gen. (b) 2nd Gen.

(c) 3rd Gen. (d) 4th Gen.

Figure 3. Evaluation of Proposed EC2 Price Model Fittings (with error rates) against actual On Demand instance prices.

we manually assess the ones that are the most representative in the model via R^2 shrinkage.

Among many parameters evaluated, we established that the significant ones are: Processor speed (GFLOPS), Memory size (GB), Disk Size (GB) and number of GPU cores. It follows that the new price model for EC2 instances prices can be described through the Equation 1:

$$\begin{aligned} Instance\_Price = \alpha * GFLOPS \\ + \beta * Memory\_GB \\ + \gamma * Disk\_GB \\ + \delta * Nb\_GPUs \end{aligned} \quad (1)$$

Furthermore, we have detected that the instances that were released in the same time period could be grouped as a same **factorial predictor** in the linear model. We thus obtain a pricing model with 5 factorial predictors, that depend on the instance *generation*, *i.e.* release period of the instance. We labeled these model's factorial predictor: "1st Gen", "2nd Gen", "3rd Gen", "4th Gen" and "5th Gen" to reflect this matching instance generation. In fact, we can see the linear model with factorial predictors as the sum of 5 more simple linear models without factorial predictor. To simplify we will take this approach from now on, but the result is exactly the same. Refining this process several times gives us five models, whose parameters for the On Demand pricing model are described in Table V. We have evaluated the

five models individually and we provide these results in Figure 3, the error rate in percentage is also given for each instance. Due to space limitation and because it present a perfect linear fitting, the 5th model evaluation figure is not presented here. We can see that the fitting is very good for all the models and **the error rate is reasonably low in each case**.

VI. Application of the new EC2 Price Model against the in-house UL HPC Facility

Hourly Price Model

With the new price model computed in V-B, we are now able to give an EC2 equivalent price for each class of node of our local clusters. This means that even though not all the cluster nodes have a perfect cloud instance match, we are still able to determine what would be its equivalent price on EC2 if that matching instance was available. This information is later used to assess the interest of operating a given node class regarding renting an On Demand instance with the same performance on the cloud. Such an estimation is proposed in Figure 4 which presents the ratio, for each node class in the cluster, of the *EC2 Equivalent Price* based on the closest possible existing EC2 instance, versus the operating cost of the considered node as evaluated in the section III (see Table II). We insist on reminding that the operating cost provided in Table II represents the maximum cost scenario where the HPC infrastructure is used at 100%. For a lower utilization of the cluster, this operating cost will be decreased due to lower energy usage.

What we observe is that for some node classes, the ratio is close to 1, meaning that operating in-house these HPC nodes costs about the same price as renting the corresponding instance on Amazon. Thus it may be interesting to rent these kind of resources on EC2 instead of operating them locally. However for **all the other ones** the results are more mitigated.

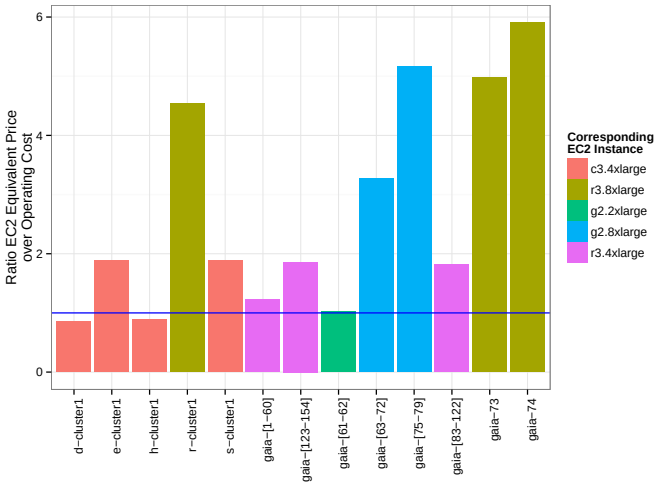


Figure 4. Ratio between the EC2 Equivalent Price of our in-house HPC Resources with regards their respective TCO estimation.

Some node classes also have low ratio (around 2) but some others have very high ratio (above 4). For these the renting of such kind of instance would be simply too costly.

The above comparison is based on "On Demand" instance prices. In this purchase model, the customer pays a fixed rate by the hour with no commitment and we just showed that in this scenario, the acquisition of an in-house HPC facility is probably a more cost-effective solution. Actually, Amazon proposes cheaper options for the instance prices:

- *Reserved* instances provide a capacity reservation, and offer a significant discount on the instance hourly charge.
- *Spot* instances enable to bid a price for a given instance, and thus may lead to even greater saving.

This last renting mode remains too hard to model accurately. However we extended our pricing model against the purchased rate offered in the "Reserved" mode. There are three payment options in this case:

1) *No Upfront*: this option provides access to a Reserved Instance without requiring an upfront payment.
2) *Partial Upfront*: a part of the Reserved Instance has to be paid upfront and the remaining hours in the term are billed at a discounted hourly rate, regardless of usage.
3) *All Upfront*: a full payment is made at the start of the term, with no other costs incurred for the remainder of the term regardless of the number of hours used.

For reserved purchase option the important thing to know is that the option is attached to one instance only and the reservation price has to be paid even though the instance is used or not. Thus the corresponding upfront price has to be paid for each EC2 instance needed. Moreover, for the reserved purchase it is possible to subscribe a contract based on 1 year or 3 years. However due to the frequent releases of new EC2 instances and the price drops of the older ones, it seems too risky to subscribe to a 3 years contract even though the savings are more important than for the 1 year basis. Because of this we will consider only the 1 year reserved option.

While we also achieved a very good fitting for all the models with a low error rate, it seemed more pertinent to us to propose from these results a general overview of the in-house HPC nodes costs when compared to all the considered EC2 purchasing mode (*i.e.* On Demand, All Upfront, Partial Upfront and No Upfront) and prices for the different renting modes. This complete cost analysis is depicted in the Figure 5. It can be seen that there exist a few cases (for the `d-cluster1-*`, `h-cluster1-*` and `gaia-[61,62]` nodes) where the EC2 Reserved instances are more cost-effective than the in-house resources. Again, for all the other cases, and thus even when a cheaper purchasing alternative is chosen for EC2 instances, it remains more efficient, from a pure cost point of view, to consider the acquisition of the corresponding HPC resources rather than renting them on the Cloud for an infrastructure of the size of the UL.

VII. Experimental Performance Impact of EC2 and Local Cluster over MPI Jobs

In Section V-B we proposed a cost model that enables to give an EC2 equivalent price to each of our local cluster

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2628371, IEEE Transactions on Cloud Computing

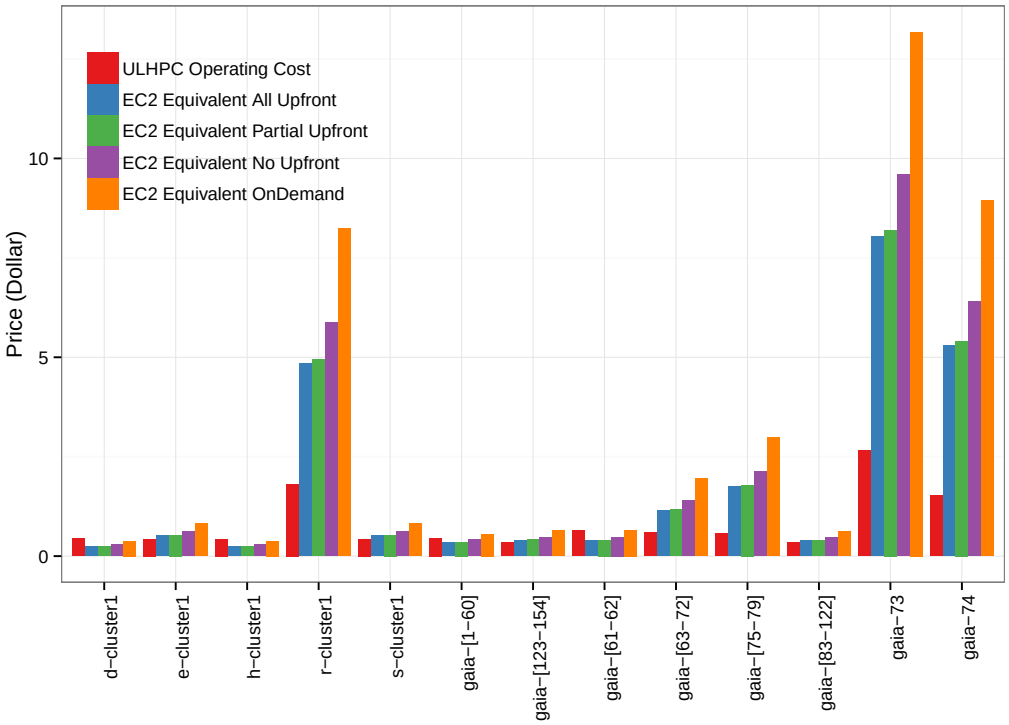


Figure 5. Comparison of the in-house HPC resources operating cost with regards an equivalent Amazon EC2 instance for the different purchasing options (On demand, All upfront, Partial upfront and No upfront).

node depending on its hardware characteristics. However, this cost model proposed is based on the theoretical computing performance of the processor and not on what can actually be obtained. In order to refine this cost model and to adapt it to the actual performance of the nodes, we propose to benchmark both EC2 and *gaia* cluster nodes. From this evaluation we will refine the cost model proposed by integrating the actual benchmark score in the cost model equation. For that purpose we use the HPCG[5] benchmark [23] proposed by Dongarra *et al.* According to the authors, HPCG is *"designed to exercise computational and data access patterns that more closely match a broad set of important applications"*. This benchmark has been proposed by its authors as an alternative of the HPL benchmark with a more relevant metric for ranking HPC systems. Its load is principally composed of matrix-vector multiplication on sparse matrices and uses it also intensively the memory.

A. Experiment Conditions

We use the HPCG Benchmark version 2.4 on several EC2 instances and *gaia* nodes with a local problem dimension of 112^3. We compile and run HPCG both on EC2 and *gaia* with Intel's toolchain version 15.0.3 and Intel MPI version 5.0-3 from Intel's *parallel studio XE 2015 update3*. We drove two sets of independent experiments: one at small scale with a number of cores being around 160 and a medium scale experiment with about 1024 cores.

1) EC2: All benchmarks are run on the *eu-west-1* region, corresponding to Ireland and we use the *Amazon Linux AMI 2015.03.1 (HVM)* RedHat based image (using VPS, with HVM virtualization). The instances are in the same placement group to ensure faster networking connections and use EC2 enhanced networking feature (SR-IOV). By default on EC2 the number of instances that can be deployed at a given time is quite limited and we requested a limit increase to Amazon. We were granted 64 instances of type *c3* and 10 instances of types *r3* and *g2*. Unfortunately we were not able to get a higher resource limit from Amazon. Due to cost considerations we run only 10 iterations of HPCG for each instance type in each experiment set.

2) gaia cluster: On *gaia* cluster we run HPCG on 13 nodes (156 cores) for the small scale experiment then on 85 nodes (1020 cores) for the medium scale one. It is noteworthy to remind that *gaia* benefits from an Infiniband QDR (40Gb/s) interconnect. We run 100 iterations of HPCG on *gaia* cluster for each experiment set.

B. Experiment Results

The results obtained from running the HPCG benchmark on EC2 and *gaia* nodes are provided in Table VI and Table VII.

[5]http://hpcg-benchmark.org/

Table VI. RESULTS FROM SMALL SCALE EXPERIMENT WITH HPCG

	EC2 *c3.8xlarge*	EC2 *g2.8xlarge*	EC2 *r3.8xlarge*	EC2 *c3.4xlarge*	EC2 *r3.4xlarge*	*gaia* cluster
Number of nodes	5	5	5	10	10	13
Number of cores	160	160	160	160	160	156
Theoretical GFLOPS	3584	3328	3200	3584	3200	1828.32
Reached avg. GFLOPS	38.14	27.59	37.48	38.37	35.54	48.8
Reached GFLOPS std. dev.	0.04	0.07	0.05	1.08	0.44	0.15
Reached vs. theoretical (%)	**1.06**	**0.83**	**1.17**	**1.07**	**1.11**	**2.67**

Table VII. RESULTS FROM MEDIUM SCALE EXPERIMENT WITH HPCG

	EC2 *c3.8xlarge*	EC2 *c3.4xlarge*	*gaia* cluster
Number of nodes	32	64	85
Number of cores	1024	1024	1020
Theoretical GFLOPS	22937.6	22937.6	10121.28
Reached avg. GFLOPS	202.51	215.72	234.21
Reached GFLOPS std. dev.	13.8	1.77	0.67
Reached vs. theoretical (%)	**0.88**	**0.94**	**2.31**

They present for the different EC2 instance types tested and for *gaia* the theoretical performance provided by the cores and the actual results obtained with HPCG. We also provide the standard deviation of the benchmark result and the ratio of performance reached versus theoretical performance. Apart from *c3.8xlarge* whose results show a relatively high standard deviation, the results are very stable with low standard deviations.

In both experiment sets, we observe that for the same number of cores, the performance reached is generally the same but with a theoretical performance more than doubled for EC2 regarding *gaia*. This largely impacts the reached vs. theoretical ratio that is quite low for EC2.

It is both expected and well visible that the ratio of GFLOPS reached by HPCG versus the theoretical processor performance is higher for the *gaia* cluster than for EC2. Indeed the use of Infiniband network and the fact that the local cluster does not use HyperThreading (as opposed to EC2 cores that are hyperthreaded cores within the VM) largely improve the performance of the run.

C. Refined Cost Model Adapted to Actual Performance

Based upon the results we got from the benchmarks, we observe that for HPCG *gaia* has an efficiency improvement factor between 2 and 3 compared to EC2, depending on the instance type used and the size of the platform used. This performance improvement ratio is summarized in Table VIII. As we could not benchmark the EC2 *g2.2xlarge*, we assume its efficiency factor to be the same as for *g2.8xlarge*.

The results of the updated cost model show without a doubt that as for now, operating a local HPC platform is more cost effective on the hourly rate. Even though cloud platforms are improving from day to day there are still a lack of performance regarding the classical HPC infrastructures, this results in a higher cost *in fine* for the Amazon EC2 cloud versus local HPC regarding the same obtainable performance.

Table VIII. EFFICIENCY IMPROVEMENT OF *gaia* VERSUS EC2

	c3.4xlarge	*r3.4xlarge*	*r3.8xlarge*	*g2.8xlarge*
***gaia*'s efficiency improvement factor**	2.5	2.4	2.3	3.2

VIII. SHARED STORAGE ASPECTS

At this time EC2 in itself does not provide a large distributed storage File System service. However another Amazon service called Simple Storage Service (S3) proposes to serve as a storage service with a pay-per-GB stored and retrieved approach. S3 does not permit to work directly on the files stored.

In order to compare storage capabilities on EC2 and on the local clusters we take the following scenario. Data is stored on large storage (S3 for EC2 and GPFS or Lustre for cluster). This data is copied upon the local disk of the machine allocated to the user then processed and finally pushed back to the large storage. We measure the time to transfer a randomly generated 1GB file from and to the large storage and repeat the experiment 10 times. Results of transfer times are provided in Table IX. We observe that the time for retrieving a 1GB file from S3 to EC2 (on the same region) and on GPFS and Lustre to *Gaia* is approximately the same but with less variation for *Gaia*. However where difference is large is for the time necessary to send the file from the machine to the storage. Whereas for EC2 this time is comparable with data retrieving time, for *Gaia* it is much shorter and with very little variation.

In order to compare local storage performance on *EC2* and *Gaia* machines we benchmark their respective disk performance. For that purpose we test the *c3.8xlarge* instance and one of *Gaia*'s node with the IOzone Benchmark [24] in automatic mode on the previously used 1GB file. Both configurations have SSD disks and as they also have a large memory, we use IOzone with direct IO operations to ensure that the file will not be cached in memory. The results of IOzone are proposed in Figure 6. For all type of operations the disk performances on EC2 and *Gaia* are comparable with a tendency for *Gaia* to be slightly better in terms of speed but most importantly with a more normally distributed speeds values (it is remarkable that in EC2 *c3.8xlarge* all operation speeds have long-tailed distributions).

Table IX. DATA TRANSFER TIMES FOR A 1GB FILE ON *EC2* AND *Gaia*.

		GET data from storage			
		min	max	avg	sd
EC2	VM disk w/ **S3**	5.999	10.914	**7.014**	1.678
Gaia	node disk w/ **GPFS**	5.685	6.737	**6.173**	0.300
	node disk w/ **Lustre**	7.007	8.567	**7.5538**	0.423
		PUT data on storage			
		min	max	avg	sd
EC2	VM disk w/ **S3**	6.046	10.414	**7.924**	1.742
Gaia	node disk w/ **GPFS**	0.631	0.704	**0.6654**	0.020
	node disk w/ **Lustre**	1.917	2.044	**1.975**	0.043

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2016.2628371, IEEE Transactions on Cloud Computing

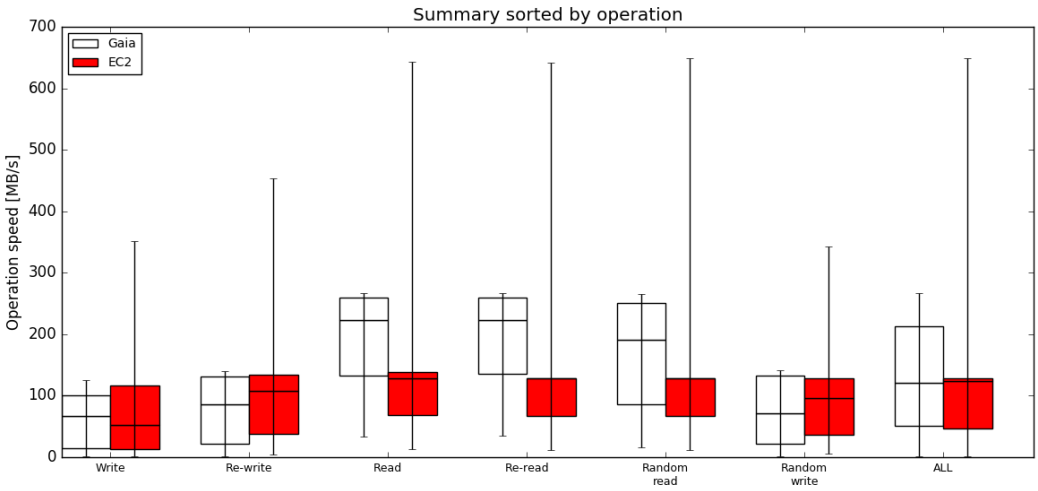


Figure 6. Local storage performance for EC2 and *Gaia* machines. Both disk types are SSD.

IX. Conclusion

Yearly Price Model Applied to Real Cluster Usage

In the previous sections we compared the costs of operating a local cluster node versus its equivalent price on EC2 based on our cost model. We saw that depending on what the user specifically asks to the system in terms of resources to run his job, the associated price largely varies. Now we investigate how this price difference impacts the costs on a yearly basis. For that purpose we analyzed all job requests on the reference period of year 2014. Using the logs form the batch scheduler we can extract which nodes were allocated to the jobs. Based on the prices computed in Figure 5, the actual performance obtained from Section VII-B and the knowledge of nodes associated to jobs we are capable of calculating the yearly cost for the year 2014 to operate the in-house cluster and the EC2 equivalent price according to their respective performances.

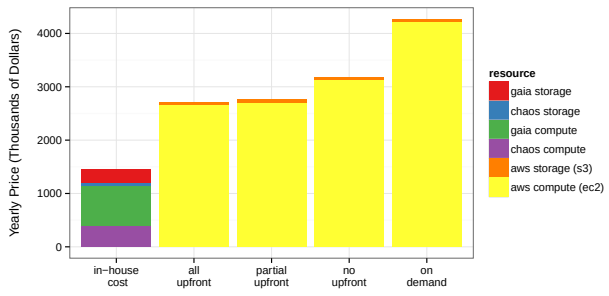


Figure 7. Annual cost for year 2014 of UL workload if operated locally or rented on EC2. Cost model adapted to real nodes performance and with storage prices included.

We present these results in Figure 7. This result particularly shows that the migration of HPC workloads to the cloud is not only a problem of adaptability of the cloud performance to HPC needs but also a problem of correctly determining which types of job are good candidates to be run on the cloud in order to prevent such cost overheads.

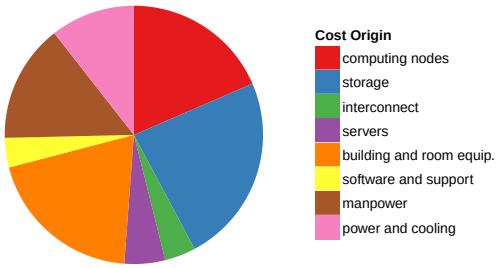


Figure 8. UL HPC yearly amortized TCO and its distribution.

The objective of the present study was to evaluate the real cost-effectiveness of Cloud Computing (CC) platform when compared to an in-house HPC facility. To answer this question, we propose in this article three main contributions:

1) a representative Total Cost of Ownership (TCO) analysis of a medium-size academic HPC facility, which is based on our own experience while managing this platform within our research institution since 2007
2) a novel price model for the instances offered by the main Cloud IaaS actors (Amazon EC2) able to compete with the performances of such an HPC platform. Our proposed cost model is flexible and rely on the inherent performance metrics (from an HPC perspective) of the considered instance resources.
3) the application of this new model against the resources

of our local clusters to offer, when confronted to the operating cost evaluated in the first part of this study, a fair comparison of the prices and thus of the investments linked to each of the solution (local own infrastructure vs. rented resource on the Cloud).

Our proposed cost analysis is not only accurate (the fitting of our novel cost model exhibits a low error rate), it also advocates in general in favor of the acquisition of an in-house HPC facility.

The perspective opened by the work are manifold. Firstly, we wish to extend our analysis over the Spot instance that permit to bid for the price of the resources. This offers a chance for better cost-savings on the instance rate prices, and thus might indicate other classes of HPC resources for which the renting option makes sense. Also, we hope to benefit from the planned move to a new campus, and thus the possibility to monitor the building cost when implementing the brand new HPC server rooms within this new site, to integrate the actual building cost in our TCO analysis. This will also give us the opportunity to update our model with regards the cost of cutting-edge HPC technologies (Direct Liquid Cooling equipment, Infiniband EDR interconnect etc.)

Acknowledgments: The experiments presented in this paper were carried out using the HPC facilities of the University of Luxembourg [1] – see http://hpc.uni.lu.

REFERENCES

[1] S. Varrette, P. Bouvry, H. Cartiaux, and F. Georgatos, "Management of an academic hpc cluster: The ul experience," in *Proc. of the 2014 Intl. Conf. on High Performance Computing & Simulation (HPCS 2014)*. Bologna, Italy: IEEE, July 2014, pp. 959–967.

[2] X. Besseron, V. Plugaru, A. H. Mahmoudi, S. Varrette, B. Peters, and P. Bouvry, "Performance Evaluation of the XDEM framework on the OpenStack Cloud Computing Middleware," in *Proc. of the 4th Intl. Conf. on Parallel, Distributed and Grid Computing for Engineering (PARENG 2015)*. Dubrovnik, Croatia: Civil-Comp Press, Mar. 24–28 2015.

[3] M. Guzek, S. Varrette, V. Plugaru, J. E. Pecero, and P. Bouvry, "A Holistic Model of the Performance and the Energy-Efficiency of Hypervisors in an HPC Environment," *Intl. J. on Concurrency and Computation: Practice and Experience (CCPE)*, vol. 26, no. 15, pp. 2569–2590, Oct. 2014.

[4] S. Varrette, V. Plugaru, M. Guzek, X. Besseron, and P. Bouvry, "HPC Performance and Energy-Efficiency of the OpenStack Cloud Middleware," in *Proc. of the 43rd Intl. Conf. on Parallel Processing (ICPP-2014), Heterogeneous and Unconventional Cluster Architectures and Applications Workshop (HUCAA'14)*. Minneapolis, MN, US: IEEE, Sept. 2014.

[5] K. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. Wright, "Performance analysis of high performance computing applications on the amazon web services cloud," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, Nov 2010, pp. 159–168.

[6] Z. Hill and M. Humphrey, "A quantitative analysis of high performance computing with amazon's ec2 infrastructure: The death of the local cluster?" in *Grid Computing, 2009 10th IEEE/ACM International Conference on*, Oct 2009, pp. 26–33.

[7] S. Akioka and Y. Muraoka, "Hpc benchmarks on amazon ec2," in *Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on*, April 2010, pp. 1029–1034.

[8] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "A performance analysis of ec2 cloud computing services for scientific computing," in *Cloud Computing*, ser. Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, D. Avresky, M. Diaz, A. Bode, B. Ciciani, and E. Dekel, Eds. Springer Berlin Heidelberg, 2010, vol. 34, pp. 115–131. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12636-9_9

[9] Y. Zhai, M. Liu, J. Zhai, X. Ma, and W. Chen, "Cloud versus in-house cluster: Evaluating amazon cluster compute instances for running mpi applications," in *State of the Practice Reports*, ser. SC '11. New York, NY, USA: ACM, 2011, pp. 11:1–11:10. [Online]. Available: http://doi.acm.org/10.1145/2063348.2063363

[10] Q. He, S. Zhou, B. Kobler, D. Duffy, and T. McGlynn, "Case study for running hpc applications in public clouds," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, ser. HPDC '10. New York, NY, USA: ACM, 2010, pp. 395–401.

[11] P. Mehrotra, J. Djomehri, S. Heistand, R. Hood, H. Jin, A. Lazanoff, S. Saini, and R. Biswas, "Performance evaluation of amazon ec2 for nasa hpc applications," in *Proceedings of the 3rd Workshop on Scientific Cloud Computing Date*, ser. ScienceCloud '12. New York, NY, USA: ACM, 2012, pp. 41–50. [Online]. Available: http://doi.acm.org.proxy.bnl.lu/10.1145/2287036.2287045

[12] R. R. Exposito, G. L. Taboada, S. Ramos, J. Tourino, and R. Doallo, "Performance analysis of hpc applications in the cloud," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 218 – 229, 2013, including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X12001458

[13] A. Gupta, P. Faraboschi, F. Gioachin, L. Kale, R. Kaufmann, B.-S. Lee, V. March, D. Milojicic, and C. Suen, "Evaluating and improving the performance and scheduling of hpc applications in cloud," *Cloud Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[14] A. Marathe, R. Harris, D. K. Lowenthal, B. R. de Supinski, B. Rountree, M. Schulz, and X. Yuan, "A comparative study of high-performance computing on the cloud," in *Proceedings of the 22nd international symposium on High-performance parallel and distributed computing*. ACM, 2013, pp. 239–250.

[15] P. Strazdins, J. Cai, M. Atif, and J. Antony, "Scientific application performance on hpc, private and public cloud resources: A case study using climate, cardiac model codes and the npb benchmark suite," in *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2012 IEEE 26th International*, May 2012, pp. 1416–1424.

[16] G. K. Lockwood, M. Tatineni, and R. Wagner, "Sr-iov: Performance benefits for virtualized interconnects," in *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment*, ser. XSEDE '14. New York, NY, USA: ACM, 2014, pp. 47:1–47:7. [Online]. Available: http://doi.acm.org.proxy.bnl.lu/10.1145/2616498.2616537

[17] C. Gong, J. Liu, Q. Zhang, H. Chen, and Z. Gong, "The characteristics of cloud computing," in *Parallel Processing Workshops (ICPPW), 2010 39th International Conference on*, Sept 2010, pp. 275–279.

[18] A. Gupta, O. Sarood, L. Kale, and D. Milojicic, "Improving hpc application performance in cloud through dynamic load balancing," in *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, May 2013, pp. 402–409.

[19] I. Sadooghi, J. Hernandez Martin, T. Li, K. Brandstatter, Y. Zhao, K. Maheshwari, T. Pais Pitta de Lacerda Ruivo, S. Timm, G. Garzoglio, and I. Raicu, "Understanding the performance and potential of cloud computing for scientific applications," *Cloud Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.

[20] A. Carlyle, S. Harrell, and P. Smith, "Cost-effective hpc: The community or the cloud?" in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, Nov 2010, pp. 169–176.

[21] A. S. McGough, M. Forshaw, C. Gerrard, S. Wheater, B. Allen,

and P. Robinson, "Comparison of a cost-effective virtual cloud cluster with an existing campus cluster," *Future Generation Computer Systems*, vol. 41, no. 0, pp. 65 – 78, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X14001344

[22] J. O'Loughlin and L. Gillam, "Towards performance prediction for public infrastructure clouds: An ec2 case study," in *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, vol. 1, Dec 2013, pp. 475–480.

[23] M. A. Heroux, J. Dongarra, and P. Luszczek, *HPCG Benchmark Technical Specification.*, Oct 2013. [Online]. Available: http://www.osti.gov/scitech/servlets/purl/1113870

[24] W. D. Norcott and D. Capps, "Iozone filesystem benchmark," 2003. [Online]. Available: http://www.iozone.org

Joseph Emeras, PhD is a Research Associate at the University of Luxembourg since 2014 within the PCOG team of Prof. Dr. Pascal Bouvry. His general interests are the performance evaluation of distributed platform through the analysis and modeling of workload traces. More recently his research focused on analyzing Cloud pricing offers and developing pricing models.

Sébastien Varrette, PhD is a Research Associate at the University of Luxembourg since 2007. Within the PCOG team of Prof. Dr. Pascal Bouvry, Dr. Varrette is managing the HPC facility. In parallel, his research work focuses on Distributed Computing Platforms (clusters, grids or clouds), with a particular interest on the security and performance evaluation of distributed or parallel executions. He's active in various scientific committees and technical workgroups. Up to now, Dr. Varrette co-authored four books and more than nine book chapters in computer science. He also wrote about 60 research or popularization articles in scientific journals, or international conference proceedings.

Valentin Plugaru is an HPC engineer part of the HPC team since 2014. Beginning with 2012 he has collaborated with Prof. Bouvry's team on research in Energy Efficiency and Performance Evaluation of HPC/Cloud environments. His general interests span R&D in High Performance Computing, Grid and Cloud Computing.

Pascal Bouvry, PhD is a full professor of the University of Luxembourg and the head of the ILIAS research unit and the DS-CSCE doctoral school. He is is also faculty member of the SnT (interdisciplinary center for security, reliability and trust), associate editor of IEEE Cloud Computing magazine, associate editor of Elsevier Swarm and Evolutionary Computation journal, founding member of IEEE TC on Cybernetics for Cyber-Physical Systems, and vice communication chair of IEEE STC on sustainable computing. His team (PCOG) is composed of 25 researchers working on Parallel computing and Optimization applied to Cloud Computing and HPC (scheduling, energy-efficiency, security), Ad-Hoc Networks (Vanets simulation and service optimization) and Biology (gene sequencing, regulatory networks, protein folding).