Noname manuscript No.
(will be inserted by the editor)

Multi-Elastic Datacenters: Auto-Scaled Virtual Clusters on Energy-Aware Physical Infrastructures

Carlos de Alfonso · Miguel Caballer · Amanda Calatrava · Germán Moltó · Ignacio Blanquer

Received: date / Accepted: date

Abstract Computer clusters are widely used platforms to execute different computational workloads. Indeed, the advent of virtualization and Cloud computing has paved the way to deploy virtual elastic clusters on top of Cloud infrastructures, which are typically backed by physical computing clusters. In turn, the advances in Green computing have fostered the ability to dynamically power on the nodes of physical clusters as required. Therefore, this paper introduces an open-source framework to deploy elastic virtual clusters running on elastic physical clusters where the computing capabilities of the virtual clusters are dynamically changed to satisfy both the user application's computing requirements and to minimise the amount of energy consumed by the underlying physical cluster that supports an on-premises Cloud. For that, we integrate: i) an elasticity manager both at the infrastructure level (power management) and at the virtual infrastructure level (horizontal elasticity); ii) an automatic Virtual Machine (VM) consolidation agent that reduces the amount of powered on physical nodes using live migration and iii) a vertical elasticity manager to dynamically and transparently change the memory allocated to VMs, thus fostering enhanced consolidation. A case study based on real datasets executed on a production infrastructure is used to validate the proposed solution. The results show that a multi-elastic virtualized datacenter provides users with the ability to deploy customized scalable computing clusters while reducing its energy footprint.

Keywords Cloud Computing · Green Computing · Elasticity · Virtualization · Infrastructure Management

1 Introduction

Computer clusters are a very common computing facility used both for scientific institutions and enterprises. A cluster consists of a set of computing nodes connected using at least one high-speed low-latency network and it is usually managed by a Local Resource Management System (LRMS) used to manage the whole lifecycle of the jobs [1]. These jobs typically represent different workloads such as High Throughput Computing (HTC) or High Performance Computing (HPC).

However, physical clusters face several drawbacks. Firstly, they require a significant capital investment together with the costs required for housing and periodic hardware maintenance. Secondly, maintaining a physical cluster up and running on a 24/7 basis is very expensive, mainly due to the cost of energy [2]. Indeed, the energy is consumed both by the cluster itself and the cooling system required to maintain the environmental conditions. These physical clusters are typically overdimensioned to cope with increased workloads and, specially, peaks of demand. However, these peaks are rarely reached while underutilization is a very common scenario. In fact, as Williams et al. [3] described, in well-

The authors would like to thank the Spanish "Ministerio de Economía, Industria y Competitividad" for the project "BigCLOE' under grant reference TIN2016-79951-R.

The results of this work have been partially funded by EUBra-BIGSEA (690116), a Research and Innovation Action (RIA) funded by the European Commission under the Cooperation Programme, Horizon 2020 and the Ministério de Ciência, Tecnologia e Inovação (MCTI), RNP/Brazil (grant GA0000000650/04).

Instituto de Instrumentación para Imagen Molecular (I3M)
Centro mixto CSIC - Universitat Politècnica de València
Camino de Vera s/n, 46022, Valencia. E-mail: caralla@upv.es, micafer1@upv.es, amcaar@upv.es, gmolto@dsic.upv.es, iblanque@dsic.upv.es

provisioned datacenters, overload is unpredictable, relatively rare, uncorrelated, and transient.

Therefore, one of the challenges for computing clusters is to reduce their energy consumption. The energy saving techniques applied for clusters are basically, Static Power Management (SPM) techniques, which consist in using more efficient components, and Dynamic Power Management (DPM) techniques, that consist in adapting the infrastructure to the actual workload [4]. One common DPM approach for computing clusters is to power off those physical nodes that are idle and power them back on again as they are needed. This energy saving technique has been proven to provide substantial cost reduction in cluster infrastructures in our previous publication [5].

On the other hand, user applications typically have special requirements (libraries, compilers, Operating System (OS) versions, etc.) what leads to potential software conflicts on multi-tenant scenarios where multiple users share the same cluster. Also, virtualization and Cloud Computing have changed the way of managing a datacenter. Many datacenters are creating their on-premises Clouds to manage their servers using a Cloud Management Platform (CMP) such as OpenNebula [6], OpenStack [7], VMWare vCenter [8], etc. Then, the system administrators create the Virtual Machines (VMs) needed by the users instead of granting access to physical machines. Actually, virtual clusters use VMs as the computing nodes, which can support the same functionality as their physical counterparts. This way, virtual clusters can be specifically tailored to the hardware, software and configuration requirements of applications to be run on them. The ability to provision customized virtual clusters is beneficial both for the user, who can access computing resources on-demand and for the system administrator, since these virtual clusters are decoupled from the underlying execution infrastructure and no adaptation of applications to the computing environment is required.

Users are provided with customized virtual clusters running an specific version of an OS and a set of libraries, managed by the preferred LRMS of the user. Providing virtual clusters with a precise hardware and software configuration that matches the requirements of an application better guarantees its successful execution. Virtual clusters are provisioned on-demand and terminated when no longer required so that other virtual clusters can be deployed, thus providing the means of multiplexing access to the underlying physical computing resources. Virtual clusters can also benefit from the elasticity of Cloud infrastructure by terminating the idle VMs and provisioning new ones when they are needed. These elastic virtual clusters behave as physical clusters and, thus, DPM techniques can be applied to the virtual cluster to power on or off the nodes. In the case of the elastic virtual clusters, the working nodes are VMs that are deployed or terminated depending on the workload.

However, such dynamism in the creation and destruction of VMs in an on-premises Cloud typically leads to a fragmented distribution of the VMs in the physical servers [9]. In this situation, the request to deploy a VM can be denied because no single host has enough physical resources, even though the aggregation of physical resources from different nodes would allow the VM to be deployed. Increasing the consolidation ratio, where VMs are hosted in a fewer number of hosts, would allow the deployment of the VM. Moreover, the physical Cloud platform can also benefit from the aforementioned DPM technique to power off idle servers in order to reduce the overall energy consumption of the on-premises Cloud.

This paper describes the work towards a multi-elastic datacenter in which the users are delivered elastic virtual clusters that run on top of on-premises Cloud infrastructures supported by elastic energy-aware physical clusters. The physical nodes that are not hosting any VMs are automatically powered off and powered on again when needed, to introduce elasticity at a physical level. The VMs of the elastic virtual cluster can be vertically scaled, in terms of the allocated memory, according to the dynamic memory consumption patterns of the application (or applications) being executed. These VMs are automatically live-migrated between hosts to increase server consolidation and use a reduced number of physical hosts, thus, facilitating power management and stimulating horizontal scalability. Therefore, this creates a multi-elastic datacenter where automated horizontal elasticity is applied for physical and virtual computing resources together with automated vertical elasticity for the elastic virtual clusters. A set of open-source developments have been created and released in order to support this vision and deployed in production within our research group.

Therefore, the main scientific contribution of this paper is to describe and assess the combination of open-source software components developed by the authors to provide users with efficient scalable cluster-based computing for on-premises Clouds. Efficiency is considered in terms of memory allocation and number of physical resources, thus linked to energy consumption. The proposed approach achieves to transparently run workloads on virtual clusters, customised for the application requirements, that can simultaneous scale both horizontally (number of nodes) and vertically (memory allocated to the VMs) on top a Cloud platform where

the physical nodes are powered on and off according to the computing requirements. This functionality is, to the authors' knowledge, unparalleled by the existing software in the Cloud computing space and, therefore, a significant contribution to the state of the art.

After the introduction, the paper is organised as follows. First, section 2 discusses the related work in the different areas covered by this paper. Next, section 3 describes the proposed approach together with the building blocks required to support a multi-elastic datacenter. Later, section 4 describes a case study in order to assess the advantages of the proposed approach. Finally, section 5 summarises the paper and points to future work.

2 Related work

Apart from the CMP, which manages the lifecycle of VMs running on the aforementioned physical infrastructure, several key components are needed for the multi-elastic datacenter: i) a tool to deploy customized elastic virtual clusters through VMs managed by the CMP; ii) an automated power management system for the physical infrastructure; iii) a mechanism to consolidate the VMs in the CMP in order to increase the VM-per-host ratio; iv) a system to automatically change the memory allocation of the VMs to the dynamic requirements of the applications being executed on them. The following subsections describe the main related works in these areas.

2.1 Elastic Virtual Clusters

There are different examples of tools to deploy virtual clusters in the literature, such as [10], [11] or [12]. These works mainly deal with the provision of the VMs and configuration of the cluster topology (e.g. connectivity, shared filesystem, ssh-ability, etc.). Some of them include configuration capabilities (e.g. installing applications, creating users, etc.). However, their approach lacks elasticity. In this sense, once a cluster has been delivered to the user, all the VMs will continue running even if they are idle. Such static behaviour may prevent from creating new clusters because of the lack of resources.

Focusing on elastic clusters, there are several works that address the problem. As an example, [13] and [14] evaluate the possibility of using Amazon EC2 to extend a physical cluster, depending on the workload. In [15], the authors explore the dynamic provision of working nodes in the cloud, depending on the size of the jobs in the queues, introducing several policies to limit the amount of working nodes to be powered on. The main limitation of these works is that they seem to be ad-hoc private implementations that have not been released as open-source components.

Focusing on ready-to-use open-source tools, the standard distribution of Hadoop [16] includes an easy-to-use mechanism to create a virtual cluster in Amazon Web Services (AWS). The main limitation is that it is exclusively designed for Hadoop and, in addition, the number of nodes for the cluster is not dynamically managed (although it is possible to manually add or destroy working nodes). StarCluster [17] is an open source cluster-computing toolkit built for a very limited number of platforms such as AWS. It uses pre-built Virtual Machine Images (VMI) with specific software installed. It is based on the Open Grid Scheduler LRMS (formerly known as SGE) and includes common libraries such as OpenMPI, OpenBLAS, Lapack, etc. It also features a module called Elastic Load Balancer that supports shrinking or expanding the cluster based on the statistics of the queues of the LRMS. However, the cluster is not self-managed since an external system (typically the user's computer running the StarCluster application) has to monitor the cluster to decide whether elasticity should be performed.

In this way, [18] is a development to create entire virtual clusters running a batch system such as HTCondor that grow and shrink automatically based on the usage, although this requires external continuous monitoring of the cluster. The caveat in this work is that it can only run their Virtual Machine Images, and requires a Cloud platform compliant with the Amazon EC2 interface.

2.2 Automated Power Management

The main CMPs do not offer automated power management out of the box, specially in the open-source versions of the products. For example, OpenQRM[1] introduces power saving features exclusively for the enterprise version, which is distributed under a commercial license. In the commercial cloud platforms, there are several solutions that offer automated power management features. This is the case of VMWare vSphere 5.5[2] which is capable of powering on and off physical hosts. However, it is restricted to the VMWare's hypervisor in addition to being very costly (beginning at USD 2,875.00 for the version able to manage the power of the nodes). Huawei's FusionSphere[3] also offers auto-

[1] http://www.openqrm-enterprise.com
[2] http://www.vmware.com/products/vsphere
[3] http://e.huawei.com/en/products/cloud-computing-dc/cloud-computing/fusionsphere/fusionsphere

mated power management. It builds up on OpenStack, but it is also distributed under a commercial license. However, commercial solutions are out of the scope of this paper due to the vendor lock-in effect that they produce.

2.3 Facilitating Power Management

There are works that try to reduce the number of physical servers needed to host the VMs deployed on a Cloud, specially at the scheduler level. In fact, most of the schedulers shipped in the default distributions of the CMPs include features for reducing the number of used servers. However, during the lifecycle of the platform (i.e. sequences of creation and destruction of VMs) the distribution of the VMs may prevent from achieving idle servers even when the running VMs could be hosted in a fewer number of servers.

There are several works that try to profit from live-migration features to consolidate the VMs in a platform into a few number of physical hosts. Some common approaches consist in applying reinforcement learning [19][20][21], fuzzy logic [22] or nature-based solutions such as the works in [23] (which is inspired on the movements of the ants in a colony), [24] and [25] (that are inspired on the behaviour of the swarms during migratory flights), or [26] (which is based on the movements of a bee colony). Other approach consist in modelling the problem as a *multidimensional bin packing* (mBP) problem where the physical nodes are modelled as multi-dimensional containers, and each dimension corresponds to a resource (typically CPU, memory or hard disk). In this field there are several proposal of works such as [27] that statically reduces the number of physical hosts, but it is not intended to be used in a continuously working platform. The work by Verma et al. [28] tries to combine VM placement with Dynamic Voltage and Frequency Scaling (DVFS). Finally, it is noticeable the work by Beloglazov et al. [29] that solves the bin packing problem and includes a scheduler to take into account energy saving criteria to re-place the VMs, or the works [30] and [31] that also try to reduce the number of used physical hosts.

2.4 Adapting the VMs to the actual workload

Users tend to overestimate the amount of memory required by their applications resulting in unused memory that could be dedicated to additional VMs running on the same physical machine [32]. Moreover, CMPs typically offer different instance types (also known as *flavors* in the case of OpenStack), out of which the VMs are instantiated. These instance types define the amount of memory, cores and storage that will be allocated to the VM. The users select the instance type according to its constraints (e.g. the number of cores) even if they do not need the corresponding amount of memory. Apart from the waste of resources, over-dimensioning a VM also hinders VM consolidation into a reduced number of servers.

Some works have tried to adjust the resources of the VMs to the actual workload. As an example, the work shown in [33] tries to adapt the allocation of the CPU in the VMs running on the Xen hypervisor, but it does not consider changing the memory. There are also other works such as [34] and [35] that try to adapt the virtual memory to the actual needs of the applications running in the VMs using various methods. However, they consider it only at the host level instead of the whole physical infrastructure managed by the CMP. Therefore, the CMP is not able to oversubscribe the hosts considering the memory that is actually being used rather than the memory that is currently allocated. There are also works such as [36] that tackle the problem at the CMP level, but their approach does not provide countermeasures in case the host memory is overcommitted and a VM claims back the memory that it had originally requested.

As opposed to previous works, we propose a system that builds on open-source developments created in our research group in order to automatically manage elasticity both at the physical and the virtual levels, featuring vertical elasticity for virtual clusters. These features, combined with dynamic power management of physical clusters and automated consolidation of VMs via live migration, provides the foundation of a multi-elastic datacenter. As far the authors are aware there is no such approach currently available in the literature that addresses simultaneous multi-level elasticity of virtual infrastructures on physical infrastructures.

3 The Multi-Elastic Datacenter

The building blocks of the multi-elastic datacenter are summarized in Figure 1. Two layers are clearly distinguished: the Cloud Infrastructure (Physical Layer) and the Cloud Services & Applications (Cloud Layer). On the one hand, the Cloud infrastructure is managed by the system administrators who are responsible for installing and managing both the physical servers and the CMP. On the other hand, the Cloud Services & Applications layer provides the user with the elastic virtual clusters on which applications/jobs are executed.

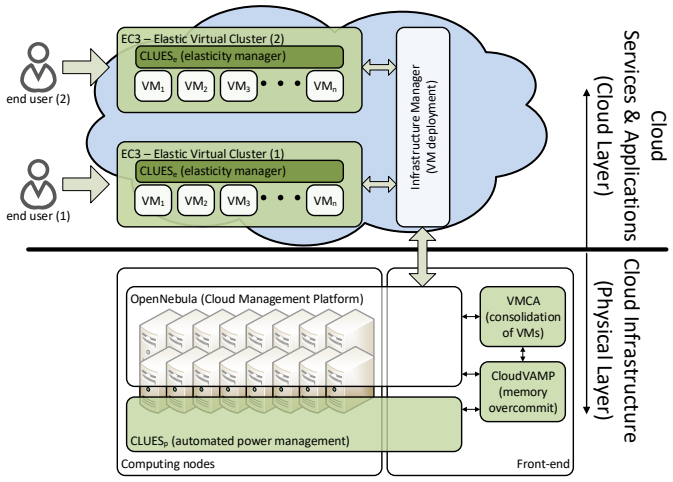


Fig. 1 The multi-elastic datacenter building blocks.

3.1 The Cloud Infrastructure Layer

The Cloud infrastructure comprises a set of physical servers arranged as an on-premises Cloud platform and managed by a CMP such as OpenNebula or OpenStack. The following components have to be installed by the system administrator to introduce the ability of dynamic power management, i.e., elasticity at the physical level:

- *CLUster Energy Saving (CLUES)*[4] [37] which is an automated power manager for computer clusters. It also supports plugins to integrate with CMPs such as OpenNebula, in order to power on and off the physical nodes depending on the requirements. For physical computer clusters, CLUES monitors the LRMS to decide when additional worker nodes have to be powered on according to different reactive policies. For virtual clusters, the semantics of powering on and off the nodes have been changed into deployment and termination of VMs. This way, CLUES can dynamically provision and relinquish VMs from a Cloud provider. For CMPs, CLUES intercepts the VM deployment requests to decide if physical nodes should be powered on. Using the same framework for elasticity at those three levels enable to reuse policies and maintain a consistent behaviour across the different layers.
- *Virtual Machine Consolidation Agent (VMCA)*[5] [9] starts from an existing VM distribution within an on-premises Cloud and produces a migration plan in order to achieve a set of idle nodes for CLUES to power them off. Live migrations are performed with the support provided by the KVM hypervisor and it has been integrated with the OpenNebula CMP.
- *Cloud Virtual Machine Automatic Memory Procurement (CloudVAMP)*[6] [38] is an automatic system that enables and manages memory over-subscription in an on-premises OpenNebula Cloud platforms. Using active monitoring of the VMs and considering the actual memory used by the VM, regardless of the initially allocated memory, it dynamically resizes the memory of the running VMs without downtime by leveraging memory ballooning techniques provided by the KVM hypervisor. Fully integrated with the CMP, it lets OpenNebula deploy additional VMs per server thus increasing the VM-per-host ratio and, as a side product, letting VMCA to obtain more idle servers. Live migration is employed to prevent memory overload of the physical hosts in order to restore the level of service.

The interface between the Cloud Infrastructure layer and the Cloud Services & Applications layer is the CMP. Apart from the common administrative tasks in the platform (e.g. creating OS disks, managing users and permissions, creating subnetworks, etc.), the interaction from the Cloud consists of creating or destroying VMs. The events that may happen in the physical side as a result from the creation or destruction of VMs are described below:

- When new VMs are created and there are not enough resources (typically in terms of memory or number of virtual CPUs) for them, $CLUES_p$ will power

[4] CLUES - `http://www.grycap.upv.es/clues`
[5] VMCA - `http://www.grycap.upv.es/vmca/`
[6] CloudVAMP - `http://www.grycap.upv.es/cloudvamp`

on one or more physical nodes to provide the required resources to run the new VMs. The VMs deployment request are held by the CMP until the physical servers are powered on. When the physical nodes are finally available, the VMs are deployed, and they will be started by the CMP on the appropriate host(s).
- When physical servers become idle, either because the VMs terminate or VMCA has migrated them to other physical nodes, $CLUES_p$ will power them off to save energy.
- If a VM is not using the memory requested during its creation, CloudVAMP will dynamically reduce its allocated memory without any downtime for the VM, according to the vertical elasticity rules described in [38].
- If a VM, whose memory was reduced, requires it (the applications running on the VM demand more memory than the one allocated to it after the reduction), CloudVAMP will check if the server in which it is hosted has enough free physical memory. If not, CloudVAMP will request $CLUES_p$ to power on a new physical server, and once the server is available, the VM will be live migrated to it. Once the VM is in a host that has enough free physical memory, CloudVAMP will increase the memory of the VM, according to the vertical elasticity rules. In the meanwhile, the VM will continue running, but in a state of memory pagination.
- If some VMs are in a physical host but they can be hosted among other servers without compromising the quality of service, VMCA will prepare and execute a migration plan to get that host free. Once the VMs are migrated, $CLUES_p$ will power off the physical node, since the host will become idle.

3.2 The Cloud Services & Applications Layer

Users deploy their elastic virtual clusters by means of EC3 (Elastic Cloud Computing Cluster)[7] [39]. This is a tool to deploy self-managed cost-efficient elastic virtual clusters on top of Cloud platforms. It supports different LRMS such as SLURM, Torque, Mesos and SGE. EC3 relies on the Infrastructure Manager[8] [44] to provision the VMs on multiple back-ends, including, but not limited to, public Clouds such as Amazon Web Services and on-premises Clouds such as OpenNebula and OpenStack. EC3 deploys a front-end node with CLUES specifically configured to be able to deploy and terminate VMs, instead of dealing with physical hosts. This is called $CLUES_e$ in Figure 1. This way, when additional worker nodes are required, new VMs are automatically deployed up to a user-specified maximum on the Cloud platform. It also includes support for hybrid deployments across multiple Clouds (either on-premises and/or public), migration capabilities and automatic checkpointing together with cost-efficient mechanisms such as the usage of spot instances, a potentially cost-reducing instance type available in Amazon EC2.

Once the user has deployed an elastic virtual cluster i, some events will happen on this layer:

- When the user submits new jobs to the cluster, $CLUES_e^i$ will check if there are enough free working nodes to execute the job. If this is not the case and the maximum cluster size has not been reached, it will ask the IM to deploy additional VMs to be integrated as new working nodes in the cluster in the LRMS. Notice that this procedure is transparent to the user, who only notices a delay since the time the job is submitted to the LRMS and the time the job starts executing.
- When the working nodes become idle for a while, $CLUES_e^i$ will terminate the corresponding VMs in order to free the used computing resources.

CLUES support multiple customizable elasticity rules, described deeply in [37]. As an example, CLUES can be configured to power on single nodes or groups of nodes based on a sensor system, and power them off when they are idle a configuring period of time.

3.3 Complex actions in the Multi-Elastic Datacenter

During the lifecycle of the multi-elastic datacenter, some complex actions may be triggered as a result of the interaction of the user with the clusters in the Cloud. These complex actions are the result of the interaction of the different building blocks of the multi-elastic data center. Examples of such actions are summarized below:

- A user submits a job to cluster i. $CLUES_e^i$ detects that there are not free working nodes and requests a VM to the IM. The IM deploys a new VM through the CMP. $CLUES_p$ detects that there are not free physical servers and powers on a new one. When the physical host is on, the VM is started and the IM can integrate the VM in the cluster i. The job can finally be started. Again, this process is completely transparent for the user.
- A VM which is part of the cluster i is not using the memory requested. CloudVAMP reduces the size of the memory of the VM. The VM is alone in one of the physical servers, but VMCA detects that it can

[7] EC3 - `http://www.grycap.upv.es/ec3`
[8] IM - `http://www.grycap.upv.es/im`

fit into other server. Therefore, VMCA live-migrates the VM, and now the physical host is idle. If the server is still idle after for a certain amount of time, $CLUES_p$ powers off the physical server.
- A VM j in cluster i (VM_j^i) whose memory was downsized, starts using the memory again. CloudVAMP detects that there is not enough allocated physical memory in the server in which the VM is hosted and requests $CLUES_p$ to power on a new physical server. When the physical server is available, CloudVAMP live-migrates VM_j^i to the new server and resizes the granted memory.

All the aforementioned components are distributed as open-source and made available in GitHub[9]. Additionally, some of the components have been adopted in large-scale research infrastructures. In particular, the IM has been integrated in the VMOps Dashboard of EGI (European Grid Infrastructure), see [40] for details, while EC3 has been integrated in the EGI Access service to provide Virtual Elastic Clusters as a Service for the Long Tail of Science (LToS), see [41] for details.

3.4 Integrating the Components

The aforementioned components address different individual problems to enhance the usability of a datacenter dedicated to provide virtual infrastructures:

- IM as cloud infrastructure provisioner: Delivering customized VMs.
- $CLUES_e$ as a horizontal elasticity manager: Adapting the virtual infrastructure (i.e. number of VMs) to the actual workload to rationalize virtual resources.
- VMCA as a facilitator for idling physical servers: Compacting the virtual resources to get servers idle.
- $CLUES_p$ as an automated power manager for physical servers, in order to save the energy of idle resources.
- CloudVAMP as a vertical elasticity manager, in order to enhance the efficiency of the usage of the virtual resources by adapting them (i.e. the virtual memory) to the actual workload.

While it is possible to use each of the components individually, their combination provides unique features for multi-elastic datacenter. Some examples of combined actions are described as follows:

- The physical servers are powered off because they are not hosting any VM. Then, $CLUES_e$ may request new VMs to the IM, which requests them to the CMP, which in turn requests $CLUES_p$ to power on physical servers.
- When the VMs are not used anymore, $CLUES_e$ requests the IM to terminate them, causing to delete them from OpenNebula, thus leaving some physical servers idle, that will be powered off by $CLUES_p$.
- CloudVAMP can shrink the memory of some VMs so that VMCA can move them to a single physical node and $CLUES_p$ powers off the previously used nodes.

Notice that the integration of these software is key to build the multi-elastic datacenter. This involves the integration of with the existing components (i.e. the CMP, the physical servers, hypervisors, etc.), and the integration of the software to deliver this functionality. The summary of integrations are described as follows:

- A plug-in for CLUES was developed to create $CLUES_e$. It consists of a power manager that whenever CLUES is requested to power on a new host, it queries the IM for a new customized VM. The main challenges in this case are selecting the appropriate type of VM considering the node requested by $CLUES_e$, managing the interaction with the IM to correctly create the VM and monitoring the status of the VMs to assure that only the VMs correctly integrated with the LRMS are maintained. Finally, all the created virtual resources are terminated when a VM is no longer needed (as requested by $CLUES_e$). The plugin also considers the possibility of having heterogeneous nodes and a cloud bursting mode, thus supporting virtual clusters in multi-cloud hybrid environments.
- A plug-in was developed to create $CLUES_p$. It consists of a power manager that uses the Intelligent Platform Management Interface (IPMI) to remotely power on and power off the physical servers. Also, a connector with OpenNebula was developed in order to monitor the physical infrastructure to obtain information about the usage of the physical virtualization servers. The main challenges in this case are to deal with the periods of the OpenNebula monitoring system and how to activate or deactivate the servers to prevent OpenNebula from scheduling VMs to physical nodes that are being powered off.
- The implementation of VMCA contains a connector with OpenNebula to monitor the VMs that are running in the platform. It also implements the actions carried out in the platform (e.g. migrating a VM).
- The distribution of CloudVAMP is integrated with the monitorization system of OpenNebula both to publish information about the actual usage of memory in the VMs, and to update the information about the reservable virtual memory to enable OpenNebula to overcommit the memory.

[9] GitHub Organization: `https://github.com/grycap`

4 Case Study

This section describes a case study in order to validate the proposed solution to produce multi-elastic datacenters on realistic settings. For this, we adopted real workloads obtained form the Grid Workloads Archive [42] and considered a scenario where two virtual elastic clusters were deployed by means of EC3 on the same Cloud infrastructure configured with the aforementioned tools. Both clusters executed the same job submission pattern. The second virtual cluster (C2) was created three hours after the first one (C1). The maximum size of each virtual cluster was fixed to 12 nodes, considering the workload used. The following configuration was specified for $CLUES_e$, i.e., the elasticity manager of the virtual clusters: i) no limit to the number of nodes concurrently provisioned and ii) idle nodes where powered off after 600 seconds.

Concerning memory ballooning, CloudVAMP was configured to keep a minimum memory size per VM of 384 MB, required for the Operating System to properly function and allowing each node to maintain a 30% of free memory, which corresponds to a Memory Overprovisioning Percentage (MOP) slightly increased compared to the 20% value used in our earlier work in vertical elasticity [38]. For this case study, the amount of cluster reconfigurations caused by adding additional nodes to the clusters introduced additional memory usage peaks which were better accommodated by these increased free memory safety margin.

The real workload, used in both cluster executions, is a fragment extracted from the *GWA-T-3 NorduGrid* dataset offered by the Grid Workloads Archive (from line 4 to line 16 of the *.gwf* file), and represented in Figure 2 in terms of number of jobs. For the sake of reproducibility of results, the duration of each job has been reduced down to a 200%. The jobs executed in that dataset are 13 sequential tasks [43], with an average duration of 186 min (in the reduced time). In our case, we used a synthetic memory-consuming application[10] that is able to reproduce a pattern of memory usage that consists of three periods, (i) increasing from 0 MB to 500 MB, (ii) maintaining the consumption in those 500 MB of memory, and (iii) reducing the usage of memory from 500Mb to 0 Mb. This behaviour can be appreciated in Figure 3, where the granted memory for a node of the cluster C1 is represented. The rationale behind the pattern is to use a dynamic memory consumption pattern in order to trigger the activation of CloudVAMP for the adjustment of the allocated memory to the VMs.

[10] https://github.com/grycap/synthalloc

The underlying physical infrastructure used is composed by an heterogeneous blade-based system that has four kind of nodes: 2 x (2 quad-core L5430@2.6 Ghz, 16 GB), 2 x (2 quad-core multithreaded E5520@2.26 GHz, 16 GB), 6 x (2 quad-core multithreaded E5620@2.4 GHz, 16 GB) and 3 x (4 quad-core multithreaded E7520 @1.86 GHz, 64 GB), with a total amount of 128 cores and 352 GB of RAM. The blade system is backed by a 16 TB SAN connected via a private gigabit ethernet network. This system is managed by OpenNebula 5.2.1, using KVM as the underlying hypervisor.

For the deployment of the virtual cluster nodes we have relied on pre-configured VMIs, since this approach will reduce in 70% the contextualization phase for each virtual node [39]. Therefore, the VMI selected is based on Ubuntu 14.04 with SLURM 14.11 and NFS preinstalled. Each VM, that corresponds with a node of the cluster, has been deployed with one CPU and requesting 1024 Mb of RAM memory.

The migration of VMs is a resource-consuming task that may interfere with the performance of the VMs. Therefore, the effects of VMCA could not be analysed under this case study, since its use is only advised when the infrastructure is stable. During the execution of a case study, VMs are frequently created and terminated. Instead, the focus has ben set on the multi-level elasticity achieved by the integration of a vertical elasticity memory oversubscription tool (CloudVAMP) with a horizontal elasticity manager (CLUES) for the execution of applications with dynamic memory-consuming patterns on virtual elastic clusters. In a production environment, VMCA is suggested to be triggered under special needs (e.g. when the VMs claim the borrowed memory to CloudVAMP and they will incur in memory overcommiting), or in periods of reduced activity. A detailed analysis of VMCA is published in [9].

4.1 Results

In this subsection, we first analyse the results obtained from the execution, and then discuss the main contributions of the proposed solution. Figures 3 and 4 cover the main results of the case study.

It is important to point out that the underlying physical infrastructure employed is used in production and during the execution of the case study, $CLUES_p$, i.e., the energy-aware elasticity manager, did not have to power on physical nodes on which the VMs of the virtual clusters would be running. The impact of $CLUES_p$ in the workload of a physical cluster have been analysed in paper [37]. The results presented in that paper show that only "2.9% of the jobs had to wait for a node to be switched on, with an average waiting time of 1 minute

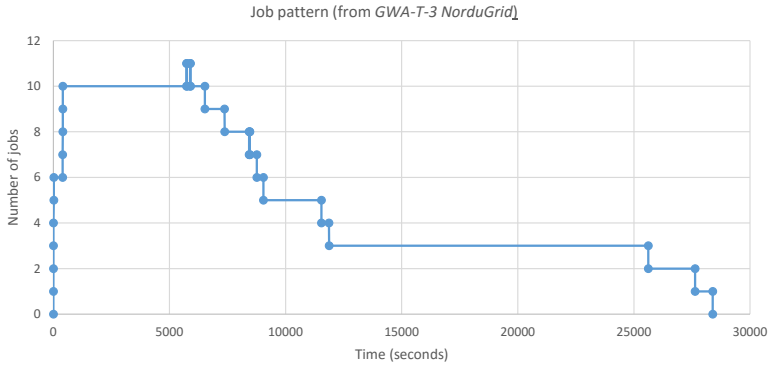


Fig. 2 Workload of the case study, extracted from the *GWA-T-3 NorduGrid* dataset. The blue line represents the evolution of the workload in terms of number of jobs.

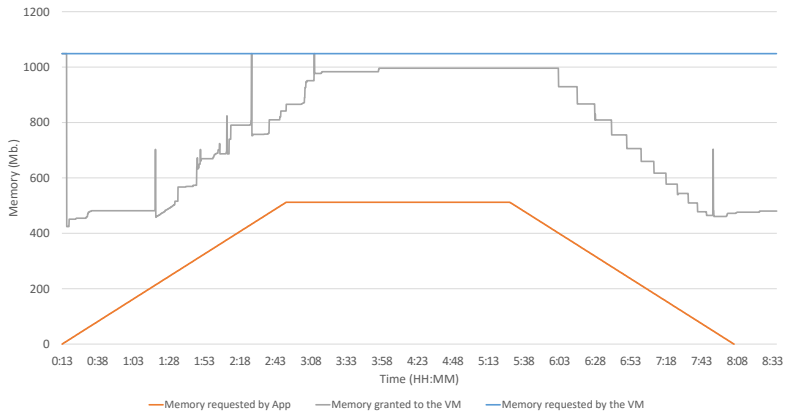


Fig. 3 Granted memory of a working node from cluster C1 in contrast with the job requested memory. The orange line represents the memory requested by the job. The grey line represents the memory granted to the VM (the peaks are related to memory consumption by other applications running inside the VM). The blue line depicts the initial memory requested by the VM.

and 54 seconds". Such time is short enough for this use-case since the mean of the job duration is 186 minutes. Notice that the only effect in the case study would be an increased time since a VM is deployed until the VM is up and running to include the time required to power on the physical node. The delay of the jobs (in case they are affected by powering on a physical server) will be less than 1%.

Figure 3 shows the evolution of the granted memory for a VM of cluster C1. The orange line represents the three different memory consuming phases solicited by the application, as described above, during its execution. Every job has a different duration execution time. The grey line of the graph represents the data reported by the hypervisor about the granted memory to the VM while the blue line represents the initially requested memory of the VM (1024 MB). In fact, the memory requested when deploying the VM represents an upper bound to the memory allocated at any given time to the VM.

Notice in the figure that a newly deployed VM receives all the requested memory (1024 MB), but as soon as CloudVAMP detects that the VM has free memory beyond the thresholds set by the 30% MOP, it steals the unused memory in order to make room in the physical node for other VMs to be deployed on that node by the OpenNebula scheduler. The amount of memory borrowed never leaves the VM with less than 384 MB of RAM and less that 30% of free memory.

The spiky peaks in the granted memory to the VM (gray line) appear to be related to memory consumption by other applications running inside the VM, specially concerning the horizontal elasticity of the virtual cluster, i.e. when the whole cluster is reconfigured by Ansible when a new virtual node is deployed or a virtual node is terminated because no jobs are available for execution.

Finally, the figure shows the grace period (600 s.) that CLUES gives to the idle virtual node, before terminating it, just in case further jobs were submitted to be deployed in that cluster. This strategy enables to

gently accommodate an incoming job without requiring an additional deployment of a virtual node, at the expense of an increased energy consumption in the underlying physical infrastructure. Notice that this grace period can be configured by the user.

Figure 4 describes the evolution of the total requested memory and the memory granted to the different virtual nodes of the two clusters (Figure 4(a)) and the evolution of the size of the clusters in terms of number of nodes, i.e. VMs (Figure 4(b)).

Figure 4(a) differentiates the results for cluster C1, where each blue line represents the granted memory to each node of the cluster, and for cluster C2, where red lines are used. The light grey area in the background represents the total memory assigned for both clusters during the execution, managed by CloudVAMP. This tool was able to dynamically and transparently change the memory allocated to VMs depending on the current workload. Its effect can be noticed when comparing the grey area, which represents the total memory initially requested by each VM (1024 MB per node), with the blue and red lines, which represent the actual granted memory to each VM. Specifically, CloudVAMP introduced a 29.13% memory saving, thus allowing increased server consolidation ratio and, thus, better usage of resources.

Figure 4(b) shows the elasticity evolution of both clusters, C1 and C2, in terms of number of nodes. The size of both clusters was dynamically adapted to their current workload by CLUES, a fact that can be appreciated comparing the grey lines (the aggregated job submission pattern for both clusters) and orange (aggregated number of nodes for clusters C1 and C2). An appreciable delay between both lines in the graph depicts the time needed for the VMs to be deployed and configured to be integrated as a new a node of each virtual cluster, i.e., the contextualization process. This introduces a delay in the execution of jobs unless an idle node is available in the cluster, an scenario that occurred in both clusters for the last three jobs of the workload.

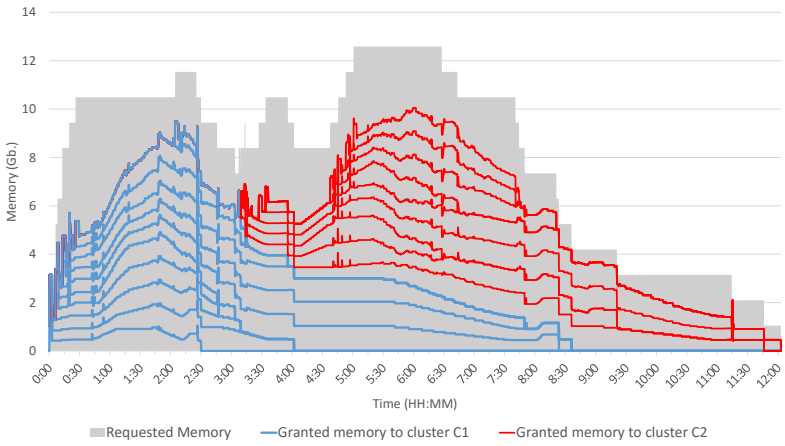
5 Conclusions and further work

This paper has introduced open-source components to manage multi-elastic datacenters, where elastic virtual clusters run on top of elastic energy-aware physical clusters. This way, the computing capabilities of the virtual clusters are dynamically changed to satisfy both the user application's computing requirements and to reduce the amount of energy consumed by the underlying physical cluster that supports an on-premises Cloud. For these, both horizontal elasticity, to add or remove nodes of the virtual cluster to adjust to the workload, and vertical elasticity techniques, to dynamically change the memory allocation of the VMs, have been combined. These developments can be adopted as an integrated approach to achieve better resource usage without requiring any additional effort by the users, which use the virtual computing clusters as if they were physical ones.

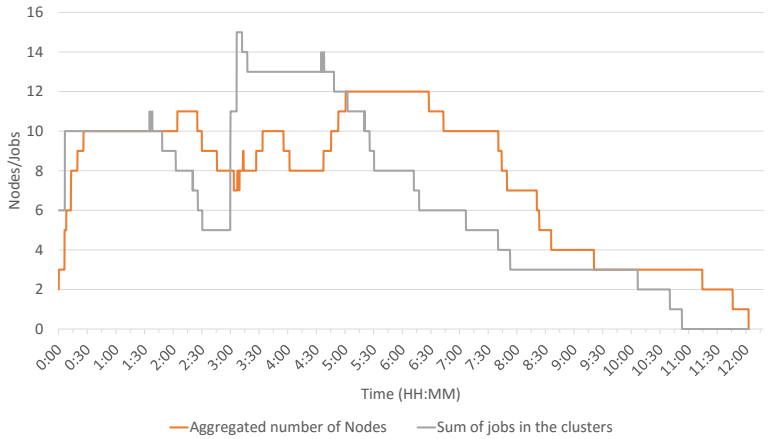
Future work involves addressing the dynamic allocation of CPUs for each VM, a feature that could not be initially developed due to the lack of support by the KVM hypervisor used by CloudVAMP. OpenNebula has added support for *cgroups* in cooperating with KVM, thus paving the way for further research in this area. Also, the components will be evolved to support Container Orchestration Platforms instead of Cloud Management Platforms, where challenges in the area of integrated vertical and horizontal elasticity require further research activity.

References

1. Buyya, R.: High Performance Cluster Computing: Architectures and Systems. Prentice Hall PTR, Upper Saddle River, NJ, USA (1999)
2. de Alfonso, C., Caballer, M., Alvarruiz, F., Moltó, G.: An Economic and Energy-Aware Analysis of the Viability of Outsourcing Cluster Computing to the Cloud. Future Generation Computer Systems (International Journal of Grid Computing and eScience) **29**, 704–712 (2013). DOI 10.1016/j.future.2012.08.014. URL `http://www.sciencedirect.com/science/article/pii/S0167739X12001720?v=s5`
3. Williams, D., Jamjoom, H., Liu, Y.H., Weatherspoon, H.: Overdriver: handling memory overload in an oversubscribed cloud. ACM SIGPLAN Notices **46**(7), 205 (2011). DOI 10.1145/2007477.1952709. URL `http://dl.acm.org/citation.cfm?id=2007477.1952709`
4. Valentini, G., Lassonde, W., Khan, S., Min-Allah, N., Madani, S., Li, J., Zhang, L., Wang, L., Ghani, N., Kolodziej, J., Li, H., Zomaya, A., Xu, C.Z., Balaji, P., Vishnu, A., Pinel, F., Pecero, J., Kliazovich, D., Bouvry, P.: An overview of energy efficiency techniques in cluster computing systems. Cluster Computing **16**(1), 3–15 (2013). DOI 10.1007/s10586-011-0171-x
5. De Alfonso, C., Caballer, M., Hernández, V.: Efficient Power Management in High Performance Computer Clusters. In: Proceedings of the 1st International Multi-Conference on Innovative Developments in ICT, Proceedings of the International Conference on Green Computing 2010 (ICGreen 2010), pp. 39–44 (2010)
6. OpenNebula: OpenNebula Cloud Software. `https://opennebula.org/`. [Online; accessed 12-June-2017]
7. OpenStack: OpenStack Cloud Software. `http://openstack.org`. [Online; accessed 12-June-2017]
8. VMWare: VMWare vCenter Server. `https://www.vmware.com/products/vcenter-server.html`. [Online; accessed 12-June-2017]
9. De Alfonso, C., I., B.: Automatic consolidation of virtual machines in on-premises cloud platforms. In: Cluster,



(a) Evolution of the total memory consumption.



(b) Evolution of size of the clusters considering the workload pattern.

Fig. 4 Performance evaluation of the proposed scenario for the complete execution.

Cloud and Grid Computing, IEEE/ACM International Symposium on, p. 10701079 (2017). DOI http://doi.org/10.1109/CCGRID.2017.128

10. Chase, J.S., Irwin, D.E., Grit, L.E., Moore, J.D., Sprenkle, S.E.: Dynamic virtual clusters in a grid site manager. In: Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing, HPDC '03, pp. 90–. IEEE Computer Society, Washington, DC, USA (2003). URL `http://dl.acm.org/citation.cfm?id=822087.823392`
11. Doelitzscher, F., Held, M., Reich, C., Sulistio, A.: Viteraas: Virtual cluster as a service. In: Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on, pp. 652–657 (2011). DOI 10.1109/CloudCom.2011.101
12. Wei, X., Wang, H., Li, H., Zou, L.: Dynamic deployment and management of elastic virtual clusters. In: Chinagrid Conference (ChinaGrid), 2011 Sixth Annual, pp. 35–41 (2011). DOI 10.1109/ChinaGrid.2011.31
13. de Assuncao, M.D., di Costanzo, A., Buyya, R.: Evaluating the cost-benefit of using cloud computing to extend the capacity of clusters. In: Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing, HPDC '09, pp. 141–150. ACM, New York, NY, USA (2009). DOI 10.1145/1551609.1551635. URL `http://doi.acm.org/10.1145/1551609.1551635`
14. Marshall, P., Keahey, K., Freeman, T.: Elastic site: Using clouds to elastically extend site resources. In: Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on, pp. 43–52 (2010). DOI 10.1109/CCGRID.2010.80
15. Niu, S., Zhai, J., Ma, X., Tang, X., Chen, W.: Cost-effective cloud hpc resource provisioning by building semi-elastic virtual clusters. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '13, pp. 56:1–56:12. ACM, New York, NY, USA (2013). DOI 10.1145/2503210.2503236. URL `http://doi.acm.org/10.1145/2503210.2503236`
16. Bialecki, A., Cafarella, M., D., C., O., O.: Hadoop: A framework for running applications on large clusters built of commodity hardware. Tech. rep., Apache Hadoop (2005). URL `http://hadoop.apache.org`
17. MIT: StarCluster Elastic Load Balancer. URL `http://web.mit.edu/stardev/cluster/docs/0.92rc2/manual/load_balancer.html`
18. appliance, C.C.S.: Creating elastic virtual clusters. http://cernvm.cern.ch/portal/elasticclusters (2015)
19. research project, T.G.: The games research project (2013). URL `http://www.green-datacenters.eu`

20. Cioara, T., Anghel, I., Salomie, I., Copil, G., Moldovan, D., Kipp, A.: Energy aware dynamic resource consolidation algorithm for virtualized service centers based on reinforcement learning. In: Parallel and Distributed Computing (ISPDC), 2011 10th International Symposium on, pp. 163–169 (2011). DOI 10.1109/ISPDC.2011.32
21. Farahnakian, F., Liljeberg, P., Plosila, J.: Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning. In: Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on, pp. 500–507 (2014). DOI 10.1109/PDP.2014.109
22. Masoumzadeh, S., Hlavacs, H.: Integrating vm selection criteria in distributed dynamic vm consolidation using fuzzy q-learning. In: Network and Service Management (CNSM), 2013 9th International Conference on, pp. 332–338 (2013). DOI 10.1109/CNSM.2013.6727854
23. Feller, E., Rilling, L., Morin, C.: Energy-aware ant colony based workload placement in clouds. In: Grid Computing (GRID), 2011 12th IEEE/ACM International Conference on, pp. 26–33 (2011). DOI 10.1109/Grid.2011.13
24. Pop, C.B., Anghel, I., Cioara, T., Salomie, I., Vartic, I.: A swarm-inspired data center consolidation methodology. In: Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics, WIMS '12, pp. 41:1–41:7. ACM, New York, NY, USA (2012). DOI 10.1145/2254129.2254180
25. Marzolla, M., Babaoglu, O., Panzieri, F.: Server consolidation in clouds through gossiping. In: Proceedings of the 2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WOWMOM '11, pp. 1–6. IEEE Computer Society, Washington, DC, USA (2011). DOI 10.1109/WoWMoM.2011.5986483
26. Ghafari, S., Fazeli, M., Patooghy, A., Rikhtechi, L.: Beemmt: A load balancing method for power consumption management in cloud computing. In: Contemporary Computing (IC3), 2013 Sixth International Conference on, pp. 76–80 (2013). DOI 10.1109/IC3.2013.6612165
27. Ajiro, Y., Tanaka, A.: Improving packing algorithms for server consolidation. In: Int. CMG Conference, pp. 399–406. Computer Measurement Group (2007)
28. Verma, A., Ahuja, P., Neogi, A.: pmapper: power and migration cost aware application placement in virtualized systems. In: Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, Middleware '08, pp. 243–264. Springer-Verlag New York, Inc., New York, NY, USA (2008)
29. Beloglazov, A., Abawajy, J., Buyya, R.: Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Gener. Comput. Syst. **28**(5), 755–768 (2012). DOI 10.1016/j.future.2011.04.017
30. Guazzone, M., Anglano, C., Canonico, M.: Exploiting vm migration for the automated power and performance management of green cloud computing systems. In: Proceedings of the First International Conference on Energy Efficient Data Centers, E2DC'12, pp. 81–92. Springer-Verlag, Berlin, Heidelberg (2012). DOI 10.1007/978-3-642-33645-4_8. URL `http://dx.doi.org/10.1007/978-3-642-33645-4_8`
31. Shi, L., Furlong, J., Wang, R.: Empirical evaluation of vector bin packing algorithms for energy efficient data centers. In: Computers and Communications (ISCC), 2013 IEEE Symposium on, pp. 000,009–000,015 (2013). DOI 10.1109/ISCC.2013.6754915
32. Tomás, L., Tordsson, J.: Improving cloud infrastructure utilization through overbooking. In: Proceedings of the 2013 ACM Cloud and Autonomic Computing Conference on - CAC '13, p. 1. ACM Press, New York, New York, USA (2013). DOI 10.1145/2494621.2494627
33. Dawoud, W., Takouna, I., Meinel, C.: Elastic vm for cloud resources provisioning optimization. In: A. Abraham, J. Lloret Mauri, J. Buford, J. Suzuki, S. Thampi (eds.) Advances in Computing and Communications, *Communications in Computer and Information Science*, vol. 190, pp. 431–445. Springer Berlin Heidelberg (2011). DOI 10.1007/978-3-642-22709-7_43
34. Tasoulas, E., Haugerund, H.r., Begnum, K.: Bayllocator: a proactive system to predict server utilization and dynamically allocate memory resources using Bayesian networks and ballooning. In: Proceedings of the 26th international conference on Large Installation System Administration: strategies, tools, and techniques, pp. 111–122. USENIX Association (2012)
35. Hines, M.R., Gordon, A., Silva, M., Da Silva, D., Ryu, K., Ben-Yehuda, M.: Applications Know Best: Performance-Driven Memory Overcommit with Ginkgo. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science, pp. 130–137. IEEE (2011). DOI 10.1109/CloudCom.2011.27
36. Litke, A.: Manage resources on overcommitted KVM hosts. Tech. rep., IBM (2011). URL `http://www.ibm.com/developerworks/library/l-overcommit-kvm-resources/`
37. De Alfonso, C., Caballer, M., Alvarruiz, F., Hernández, V.: An energy management system for cluster infrastructures. Comput. Electr. Eng. **39**(8), 2579–2590 (2013). DOI 10.1016/j.compeleceng.2013.05.004. URL `http://dx.doi.org/10.1016/j.compeleceng.2013.05.004`
38. Moltó, G., Caballer, M., de Alfonso, C.: Automatic memory-based vertical elasticity and oversubscription on cloud platforms. Future Generation Computer Systems **56**, 1–10 (2016). DOI 10.1016/j.future.2015.10.002. URL `http://dx.doi.org/10.1016/j.future.2015.10.002`
39. Calatrava, A., Romero, E., Moltó, G., Caballer, M., Alonso, J.M.: Self-managed cost-efficient virtual elastic clusters on hybrid Cloud infrastructures. Future Generation Computer Systems **61**, 13–25 (2016). DOI 10.1016/j.future.2016.01.018. URL `http://authors.elsevier.com/sd/article/S0167739X16300024http://linkinghub.elsevier.com/retrieve/pii/S0167739X16300024`
40. Caballer, M., Chatziangelou, M., Calatrava, A., Moltó, G., Pérez, A.: IM integration in the EGI VMOps Dashboard. In: EGI Conference 2017 and INDIGO Summit 2017 (2017)
41. Calatrava, A., Caballer, M., Moltó, G., Pérez, A.: Virtual Elastic Clusters in the EGI LToS with EC3. In: EGI Conference 2017 and INDIGO Summit 2017 (2017)
42. Iosup, A., Li, H., Jan, M., Anoep, S., Dumitrescu, C., Wolters, L., Epema, D.H.: The grid workloads archive. Future Generation Computer Systems **24**(7), 672 – 686 (2008). DOI http://dx.doi.org/10.1016/j.future.2008.02.003. URL `http://www.sciencedirect.com/science/article/pii/S0167739X08000125`
43. Nordugrid dataset, the grid workloads archive (Online; accessed 27-March-2017). URL `http://gwa.ewi.tudelft.nl/datasets/gwa-t-3-nordugrid/report/`
44. Caballer, M., Blanquer, I., Moltó, de Alfonso, C: Dynamic Management of Virtual Infrastructures. Journal of Grid Computing **13**, 53–70 (2015). DOI 10.1007/s10723-014-9296-5. URL `http://link.springer.com/article/10.1007/s10723-014-9296-5`