Adaptive Computing-plus-Communication Optimization Framework for Multimedia Processing in Cloud Systems

Mohammad Shojafar, Claudia Canali, Riccardo Lancellotti, *Members, IEEE,* and Jemal Abawajy, *Fellow, IEEE,*

Abstract—A clear trend in the evolution of network-based services is the ever-increasing amount of multimedia data involved. This trend towards big-data multimedia processing finds its natural placement together with the adoption of the cloud computing paradigm, that seems the best solution to cope with the demands of a highly fluctuating workload that characterizes this type of services. However, as cloud data centers become more and more powerful, energy consumption becomes a major challenge both for environmental concerns and for economic reasons. An effective approach to improve energy efficiency in cloud data centers is to rely on traffic engineering techniques to dynamically adapt the number of active servers to the current workload. Towards this aim, we propose a joint computing-plus-communication optimization framework exploiting virtualization technologies, called *MMGreen*. Our proposal specifically addresses the typical scenario of multimedia data processing with computationally intensive tasks and exchange of a big volume of data. The proposed framework not only ensures users the Quality of Service (through Service Level Agreements), but also achieves maximum energy saving and attains green cloud computing goals in a fully distributed fashion by utilizing the DVFS-based CPU frequencies. To evaluate the actual effectiveness of the proposed framework, we conduct experiments with *MMGreen* under real-world and synthetic workload traces. The results of the experiments show that *MMGreen* may significantly reduce the energy cost for computing, communication and reconfiguration with respect to the previous resource provisioning strategies, respecting the SLA constraints.

Index Terms—Energy efficiency, Multimedia data processing, Cloud resource management, Load balancing, Dynamic voltage and frequency scaling (DVFS), Traffic engineering.

1 Introduction

Internet-based services are evolving towards an ever-increasing amount of multimedia content, both in terms of number of resources (e.g., more videos and photos are uploaded every day in Web-based multimedia sharing applications such as Flickr or Youtube) and of their size (e.g., higher resolution videos) [1]. For example, the number of videos uploaded on YouTube recently reached a value in the order of 300 hours per minute, with videos that can be viewed both from small mobile devices or from full-HD screens[1], while Instagram is close to 60 millions of photo uploads per day, again with resolutions reaching tens of Mpixel[2]. This trend determines an evolution towards workloads characterized by significantly increased computational and communication requirements and higher variability, with major fluctuations throughout the day. To cope with such demands, cloud computing seems a very promising approach, because it provides an elastic, pay-as-you-go pricing model that can be used to address workload fluctuations, while the large data centers, that are typical of cloud infrastructures, can provide the computational power required to manage the huge amount of multimedia data of modern applications. However, such powerful cloud computing data centers must meet two separate goals. On one hand, they should reduce as much as possible the required energy, both for environmental reasons and to be competitive from an economic point of view even when computationally intensive tasks are carried out by the infrastructure. The impact of power consumption of such infrastructures is well described by EPA and NDRC reports [2], [3] that place the power consumption of data centers in the last years to 1.5% of the global demands (roughly comparable to the power consumption of countries such as Italy or Spain) and these numbers are expected to grow steadily as cloud systems become more and more popular. On the other hand, the cloud infrastructure must guarantee adequate performance (in order to meet QoS requirements), especially when resource-hungry applications with time-varying workloads are deployed on the cloud infrastructure. We consider the point of view of a service provider that uses a private cloud infrastructure for the delivery of multimedia data processing applications. Hence, the cloud provider has not only access to the underlying cloud infrastructure for management purposes, but also to the knowledge about application characteristics and QoS requirements [4]. In our specific problem, we consider a class of cloud-based

- *M. Shojafar, C. Canali and R. Lancellotti are with the Computer Engineering Department "Enzo Ferrari", University of Modena and Reggio Emilia, Via Vivarelli 10/1, 41125 Modena, Italy Italy*
  E-mail: {mohammad.shojafar, claudia.canali,riccardo.lancellotti}@unimore.it
- *J. Abawajy is with the Faculty of Science, Engineering and Built Environment, Deakin University, Geelong, Australia*
  Email: jemal@deakin.edu.au

Manuscript received May 26, 2016.

1. http://www.statisticbrain.com/youtube-statistics/
2. http://www.statisticbrain.com/instagram-company-statistics/

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.2016.2617367, IEEE Transactions on Cloud Computing

multimedia processing applications where clients upload contents that need to be annotated, for example to extract gestures [5] faces [6] or emotions [7] from multimedia data. SLA constraint is in the form of a maximum processing time for the requests. Due to the CPU-bound nature of these applications, energy management is critical, so we also need to minimize the overall computational-and-communication energy consumption [8].

Recently, researchers have focused on controlling the energy consumption of computing resources and communication links in data centers. Specifically, an optimized placement of computationally-intensive jobs on virtual machines (VMs) helps to increase the efficiency in CPU utilization, while optimized routing over multiple paths helps to increase efficiency in link utilization [4], [8]. This motivates our choice of proposing an energy-aware framework that considers both computational and network resource allocation in the cloud data centers [9]. In this context, virtualization enables the consolidation of heterogeneous applications onto fewer physical servers, while ensuring a fair resource allocation among competing applications. This approach achieves higher resource utilization and reduces energy costs by turning off under-utilized servers [10]. Furthermore, we take into account the presence of Dynamic Voltage and Frequency Scaling (DVFS) technology for dynamically tuning the processing frequency of the CPU according to the incoming workload. Finally, we include in our model a description of the data center network, to model also the transmission-related power consumption, which is likely to be non-negligible when large amount of data are transferred, as typically occurs with multimedia contents.

In this paper, we propose a new optimization framework, called *MMGreen*, to reduce the energy consumption of computing, communication and infrastructure reconfiguration in a cloud data center. Our approach operates at the granularity of a *data chunk*, that may be either an image or part of a video stream, reconfiguring the cloud infrastructure as needed. We model SLA as a constraint on the computational and communication time to process a data chunk. In a nutshell, the main goal of this work is to introduce a novel framework that minimizes joint computing-plus-communication energy in cloud data centers. Our solution, which is especially designed to work with multimedia applications characterized by variable workload, takes into account the allowed discrete processing frequencies for VMs hosted by DVFS-enabled CPU cores. It is important to note that this feature has an internal effect on the energy consumption of each CPU facing the incoming workload. The CPU dynamically changes its current (i.e., called *old*) operating frequency to a new one according to its incoming workload and to the related SLA constraints. The new frequency is called *optimum* frequency; the difference between old and optimum frequency is called *reconfiguration frequency*; the reconfiguration frequency is related to an energy cost, which is called *reconfiguration cost*.

Specifically, our work aims to:

- define an architectural framework and principles for energy-efficient cloud data centers;
- develop an energy-aware resource allocation and provisioning algorithm that improve the energy efficiency of a data center under SLA constraints;
- develop an adaptive version of the scheduling algorithm for energy-efficient mapping of multimedia data fractions over the available VMs.

Notable features of the resulting *MMGreen* framework are the ease of implementation and the ability to manage time-varying workloads at the minimal reconfiguration cost. To the best of our knowledge, this is the first paper addressing all these points with such level of details in modeling the three involved components: computation, communication and reconfiguration.

We tested *MMGreen* against multiple state-of-the-art solutions for data center resource allocation using both synthetic and realistic workloads related to a multimedia application deployed on a cloud infrastructure. Our results show that *MMGreen* clearly outperforms the alternatives in terms of energy savings. Moreover, a sensitivity analysis is carried out on the main parameters of the proposed framework.

The rest of the paper is organized as follows. Section 2 discusses the relevant related work. The system model and architecture are defined in Section 3. The proposed *MMGreen* framework is discussed in Section 4. Simulation results are presented in Section 5, and Section 6 concludes the paper with some final remarks.

2 Related work

In this section, we briefly present a review of the most relevant research efforts, explore the basic ideas behind these techniques and highlight how our work complements current research and advances in energy-efficient multimedia cloud systems.

In last few years multimedia applications have been recognized to represent a major challenge for cloud computing systems [11] because they place great overhead not only on CPU and storage requirements, but also on the communication infrastructure. Recently, most studies in the field have proposed solutions that allow mobile devices to access multimedia rich applications by offloading the computing intensive tasks on the cloud servers [12], [13]. These works focus on delivering high quality multimedia services that can guarantee the agreed QoS and save energy on the mobile devices to increase their lifetime. However, they do not consider the energy-related issues at the data center level, which are particularly challenging in the context of multimedia applications requiring high amounts of CPU and bandwidth resources. On the other hand, solutions for energy-efficient management of cloud resources do not specifically focus on multimedia applications, as we discuss in the rest of this section.

Most of the existing approaches for energy-saving in cloud data center focus on scheduling jobs between computing servers and providing energy efficiency by means of some hardware techniques, such as DVFS [8], [10], [14], [15]. Schedulers that exploit this feature have been categorized in [16] as static and sequential. STAtic Scheduler, namely STAS, has static power consumption which is independent of clock rates, device usage scenarios, and system status for the energy management; although STAS does not incur any reconfiguration cost arising from dynamic frequency scaling and consolidation, it induces overbooking of computing

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.2016.2617367, IEEE Transactions on Cloud Computing

resources [16]. On the other hand, SEquential Schedulers, namely SES, exploit perfect future workload information, in order to perform offline resource provisioning at the minimum reconfiguration cost. Specifically, their approach [14], [15] is to formulate the afforded minimum-cost problems as sequential optimization problems, and solve them by using limited look-ahead control. Hence, the effectiveness of these solutions relies on the ability to predict accurately future workload and the performance degrades when the workload exhibits unpredictable fluctuations. Furthermore, these approaches neglect the provisioning of communication resources, which are considered in our proposal.

Multiple studies apply these approaches to process large amounts of data in a cloud-based environment: for example [17], [18] focus on VM allocation under SLA constraints. However, such studies do not take into account the communication-related aspects of the problem. The joint analysis of the computing-plus-communication energy consumption and online job decompositions is the focus of *GreenDCN* [19] and *Hybrid NetDC* [20]. However, such approaches do not consider inter-switching costs, and assume a fixed end-to-end link cost, which is not realistic. Our proposed method fixes these problems and takes into account the communication costs under QoS constraints in the *MMGreen* framework.

Finally, Cordeschi et al. [21] propose a traffic engineering-based approach that takes into account the data center network and aims to reduce the number of active servers, while simultaneously balancing the resulting communication traffic flows. However, this work does not consider some important elements affecting the energy consumption of the data center, such as inter-costs for reconfiguration among various discrete ranges of processing frequencies, idle costs for the end-to-end links, and idle discrete frequencies for each VM, while the proposed *MMGreen* method takes them into account. As multimedia workloads in a cloud environment may exhibit a significant variability in their intensity, a detailed modeling of frequency variations and idle states makes our approach much more suitable for the considered application scenario.

3 System Model and Considered MMGreen Architecture

This section introduces the cloud data center model used in *MMGreen*. Specifically, we describe the components of the proposed architecture that is shown in Fig. 1. In subsection 3.1, we describe the general scheme of the proposed architecture that minimizes the joint computing-plus-communication energy for multimedia processing. Subsection 3.2 explains the fundamental components of the proposed framework. Finally, subsection 3.3 presents the energy-aware part of the architecture.

3.1 The MMGreen reference architecture

The *MMGreen* architecture is composed of two components. The first one, which is shown in the lower part of Fig. 1, consists of the front-end part of the data center that manages the incoming workload and is in charge of the configuration of the cloud data center infrastructure. The second one,

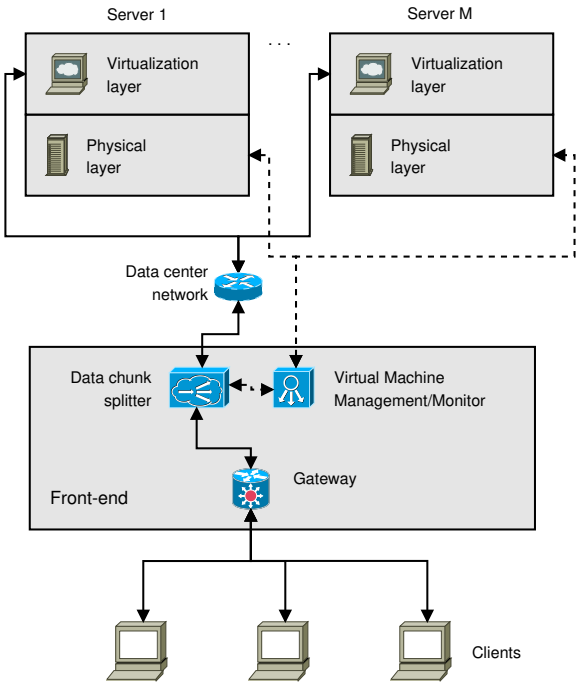


Fig. 1: The considered *MMGreen* architecture.

at the top of Fig. 1, consists of the computing resources, in the form of physical servers connected to the front-end part by the data center network. Each server consists of a virtualization layer (that is, the VMs running on the server) and a physical layer providing the actual computational resources for the processing of the incoming multimedia data. Each server connects to the front-end component via a communicating link through the data center network, as shown in the figure; the intra-cluster communication is supported through message passing.

In the front-end we identify a gateway receiving the incoming workload. Furthermore, we have a management layer with two components namely Data Chunk Splitter and Virtual Machine Management/Monitor (VMM). The first distributes the incoming multimedia data chunks among the M VMs through the end-to-end links (each bidirectional arrowed line drawn from the Data Chunk Splitter to the VMs in Fig. 1). The VMM dynamically manages the virtualization layer to map the available resources onto multiple (possibly heterogeneous) VMs.

3.2 Computing Resources

The computing resources in the *MMGreen* architecture are the data center servers hosting the VMs. The computing cost is calculated based on the energy consumed during the processing of chunks of the dispatched workload for each VM. In this paper, we assume that VMs deployed over a server can change their share of server resources according to the model described in [22], that is the typical approach for the management of private clouds. This model tends to face conditions of high computational demand by means of few large VMs instead of many small VMs. This motivates our simplifying choice to consider just one VM on each physical server. A server hosting a VM has three general modes: OFF, active and inactive. In OFF mode, the server

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.2016.2617367, IEEE Transactions on Cloud Computing

power is turned OFF. Turning servers ON or OFF is the task of the long-term management system of the datacenter that is out of the scope of the current paper. In *active mode*, the server is ON and executes tasks. Finally, in *inactive mode* the server is ON but does not execute any task. Since the time for turning a server ON or OFF is relatively high, we introduce an *idle state* (inactive mode) for each VM, which indicates that the VM is not processing data and is in a low power state, consuming less energy. In our system we assume that powering ON or OFF a server is a decision taken by a long-term server consolidation strategy, that must ensure the ability of the infrastructure to have at every time enough servers to process the expected maximum incoming workload for that time period. Hence, the long-term consolidation strategy must handle the daily patterns typical of the workload of most Internet-based services. Multiple solutions for long-term consolidation strategy are already available in literature, such as [23], [24]. The focus of our proposal, instead, is on the fast changes of the workload intensity, that occur with a time scale of seconds and that cannot be addressed by traditional server consolidation solutions.

3.2.1 Workload Model

The workload is modeled as a series of multimedia data chunks sent from the clients to the cloud platform for processing. The considered scenario is the case of applications for annotating multimedia contents, for example to perform face recognition and movement tracking from images and video resources.

We define the length of a data chunk as L_{tot} $[bit]$. The chunk is then split into M quotas (fractions) that must be assigned to the active VMs and passed over contention-free parallel end-to-end links to reach each VM for processing.

Data processing of multimedia resources is characterized by a QoS requirement defined through a SLA in the form of a maximum time allowed to process the data. Hence, chunk processing (execution plus communication delay) must be carried out within a time T_t $[s]$.

3.2.2 VMs and Servers Characterization

In the context of energy-aware resource allocation, the attributes of the generic server i can be defined as:

$$\{f_i^{idle}, f_i^{max}, P_i^{Idle}, A(i), C_{eff}(i)\}, i = 1, 2, \ldots, M, \quad (1)$$

where f_i^{idle} and f_i^{max} are the idle and maximum CPU frequencies, P_i^{Idle} is the idle CPU power consumption and the parameters $A(i)$ and $C_{eff}(i)$ represent the active percentage of gates and effective capacitance load [25], [26]. As we consider that each server hosts only one VM, we can refer the energy-oriented attributes of the i-th physical server directly to the i-th VM hosted on that server. Furthermore, for the sake of simplicity we consider a homogeneous data center, so in the following we remove the reference to the VM and server i from the notation. We consider that the i-th VM can process multimedia data according to the application deployed on the cloud data center. We define F^{max} as the maximum processing rate (in bit/s). Furthermore, we assume to consider a CPU-bound application so that the CPU frequency of the server is linearly correlated to the processing rate. Hence, the processing rate F^{max} corresponds to the CPU frequency f^{max}.

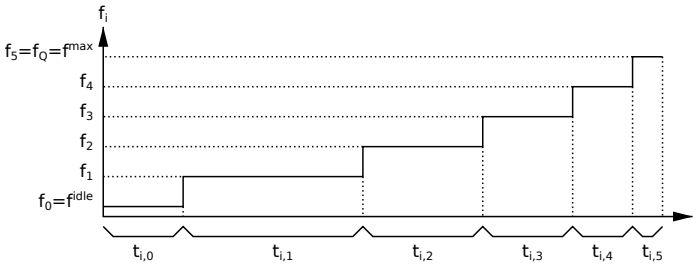


Fig. 2: The discrete range of frequencies considered for $VM(i)$

3.3 Energy Consumptions in MMGreen

In this subsection, we detail the energy model, which is categorized into three types of energies: ComPutational cost (denoted *CPc*), REconfiguration cost (denoted *REc*), and CoMmunication cost (denoted *CMc*).

3.3.1 ComPutational Cost (CPc) in MMGreen

DVFS technique is applied in VMs processors to reduce the energy consumption by decreasing the VMs frequencies. It is assumed that each VM can operate at multiple processing frequencies and each (discrete) frequency is active for a specific time [27].

In general, DVFS technology allows to work at various frequencies: Q is the number of frequency segmentations between the (real) minimum and maximum frequencies for each VM processor that is able to work with DVFS technology. For instance, AMD Turion MT-34 can operate at six frequencies ranging which are from 800 to 1800 MHz [28], while Crusoe RTM-5800 makes available 5 discrete frequencies falling into the interval 300 to 933 MHz [29]. Including also the idle state as the lowest frequency, we can write:

$$f_0 \triangleq f^{idle} < f_1 < f_2 < \ldots < f_Q \triangleq f^{max}. \quad (2)$$

The time required to change frequency is limited to few tens of μs in state-of-the-art DVFS-enabled multi-core computing platforms [16], [27]. So the current technology supports change of frequency at run-time to tune the data processing rate to the data center needs [30]. From a more practical perspective, in DVFS-based VMs CPU, VM i is able to work with frequency f_j for the duration t_{ij}. Hence, $F_j t_{ij}$ is the resulting amount of processed data in *bit*s. As shown in Fig. 2, each VM can scan its available frequency range when processing an incoming set of data.

Other components of a system, such as memory, bus, etc, operate at a single frequency and consume the same power in both active and idle states [28]. For this reason, we focus only on the CPU power. According to [25], [26], the dynamic power consumption P_{dyn} of each CPU server (and, consequently, VM) working at frequency f is given by:

$$P_{dyn} = AC_{eff} f v^2, \quad (3)$$

where A, C_{eff} have already been introduced in the previous section, while f and v are the CPU frequency and supply

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.2016.2617367, IEEE Transactions on Cloud Computing

voltage, respectively. The frequency and the voltage are correlated to each-other according to Eq. (4) [25], [26].

$$f = \alpha \frac{(v - v_{th})^2}{v}, \tag{4}$$

where, v_{th} is threshold voltage, which is much lower than v [25], [26] and α is a constant. Joining Eq. (3) and (4) we can express instantaneous power consumption as a cubic function of the frequency f. Hence, if we define $P^{idle} \geq 0$ as the power consumption when the VM is in an inactive state, the total computational cost becomes:

$$\mathcal{E}_{CPc}(i) \triangleq \sum_{j=0}^{Q} A' C_{eff} f_j^3 t_{ij}, \;\; i = 1, \ldots, M, \tag{5}$$

where $A' = A/\alpha$, t_{ij} is the time period during which the CPU of the i-th VM is running at frequency f_j, $\forall i = 1, \ldots, M$, $j = 0, \ldots, Q$ (Q denoting the discrete frequencies for each VM in Q+1 discrete ranges).

3.3.2 REconfiguration cost (REc) in MMGreen

The basic task of the VMM is to manage suitable frequency-scaling mechanisms, to allow the server hosting the VMs to adjust in real-time their processing frequency f_i [31]. We note that switching from frequency f_1 to frequency f_2 incurs an energy cost $\mathcal{E}_{REc}(f_1; f_2)$ $[Joule]$ [27], [31]. Although the actual behavior of the switching energy-function $\mathcal{E}_{REc}(f_1; f_2)$ depends on the adopted DVFS technique and the underlying physical CPUs [32], any practical $\mathcal{E}_{REc}(f_1; f_2)$ function typically retains the following (mild) properties [27], [31]: *i)* the function $\mathcal{E}_{REc}(f_1; f_2)$ depends on the absolute frequency gap $|f_1 - f_2|$; *ii)* $\mathcal{E}_{REc}(.)$ vanishes at $f_1 = f_2$ and it is non decreasing in $|f_1 - f_2|$; and, *iii)* it is jointly convex in f_1, f_2. A common practical model that retains the aforementioned formal properties is the following:

$$\mathcal{E}_{REc}(f_1; f_2) = k_e \left(f_1 - f_2\right)^2 \; [Joule], \tag{6}$$

where k_e $Joule/(Hz)^2$ dictates the energy cost induced by an unit-size frequency switching. Typical values of k_e for current DVFS-based virtualized computing platforms are quite low and in the order of few hundreds of μJ per MHz^2 [31].

Hence, f_1 and f_2 are fixed and just related to the defined discrete ranges of available frequencies for each VM. We assume that the number of discrete available frequencies is the same for each VM. According to the fraction of workload allocated to it, each VM is able to work with some ranges of discrete frequencies that we called *active discrete frequencies*. The switching cost in *MMGreen* is split into *internal* and *external* costs. The *internal* cost takes into account the reconfiguration cost of changing the internal-switching among active discrete frequencies of $VM(i)$, while the *external* cost is related to the difference between the first active discrete frequency for the next incoming workload and the last used active discrete frequency for the previous workload. For example, if we consider a VM that has 5 discrete frequencies and this VM is able to work with the 3 lowest frequencies based on the assigned fraction of workload, we consider these three frequencies as a list of active discrete frequencies for this workload fraction, then calculate internal and external frequency differences, and compute the reconfiguration energy according to eq. (6).

3.3.3 CoMmunication cost (CMc) in MMGreen

In *MMGreen* model, we assume that each VM communicates to the scheduler via a *dedicated* (i.e., contention free) reliable link that works at the transmission rate of R_i (bit/s), $i = 1, \ldots, M$.

We assume that the link is bidirectional and symmetric [33]. Furthermore, we assume that the one-way transmission-plus-switching operation over the i-th link drains a (fixed) power of P_i^{CMc} $(Watt)$. P_i^{CMc} can be expressed as: $P_i^{CMc} \equiv P_T^{CMc}(i) + P_R^{CMc}(i)$, where $P_T^{CMc}(i)$ is the power required by the (one-way) transmission and switching, and $P_R^{CMc}(i)$ is the power demanded by the received circuit. The actual value of P_i^{CMc} depends on the switching unit, the noise affecting the i-th link, as well as the demanded reliability (e.g., the target big error rate or BER to be attained on the i-th link [34]). In the following, we assume that the set of link powers $\left\{P_i^{CMc},\; i = 1, \ldots, M\right\}$ is assigned.

About the actual value of P_i^{CMc}, we note that in order to limit the implementation cost, current data centers utilize off-the-shelf rackmount physical servers which are interconnected by commodity Fast/Giga Ethernet switches. Furthermore, they implement TCP protocols to attain end-to-end reliable communication [35]. In this regard, we note that the data center-oriented versions of the TCP New Reno protocol proposed in [35], [36] allow the managed end-to-end transport connections to operate in the Congestion Avoidance state during 99.9% of the working time, while assuring the same end-to-end reliable throughput of the TCP New Reno protocol. Therefore, the communication power cost of the proposed model can be simplified as in [35], [36], [37]:

$$P_i^{CMc}(R_i) = \Omega_i \left(\overline{RTT_i} R_i\right)^2 + P_i^{idle}, \qquad i = 1, \ldots, M, \tag{7}$$

where $\Omega_i \triangleq \frac{1}{g_i} \left(\frac{1}{MSS}\sqrt{\frac{2v}{3}}\right)^2 \quad i = 1, \ldots, M$; MSS $[bit]$ is the maximum segment size; $v \in \{1, 2\}$ is the number of per-ACK acknowledged segments; g_i $[Watt^{-1}[$ is the coding gain-to-receive noise power ratio of the i-th end-to-end connection; $\overline{RTT_i}$ is the average round-trip-time of the i-th end-to-end connection (e.g., $\overline{RTT_i}$ less than $1ms$ in typical data centers [36]); and P_i^{Idle} is the idle power cost for the i-th end-to-end link.

Hence, the corresponding one-way transmission delay equates: $D(i) = \sum_{j=1}^{Q} F_j t_{ij}/R_i$, so that the corresponding one-way communication energy $\mathcal{E}^{CMc}(i)$ is:

$$\mathcal{E}^{CMc}(i) \triangleq P_i^{CMc}(R_i) \left(\sum_{j=1}^{Q} \frac{F_j t_{ij}}{R_i}\right) \; [Joule]. \tag{8}$$

Specifically, the energy consumption of end-to-end link does not effect the policy of computation, and is completely independent. Table 1 summarizes the notations used in the paper.

TABLE 1: Notation

Symbol	Meaning/Role
$f_i\ [MHz]$	CPU frequency of server hosting $VM(i)$
$f_i^{\max}\ [MHz]$	Max CPU frequency of server hosting $VM(i)$
$f_i^{idle}\ [bit/s]$	Idle fixed frequency for $VM(i)$
$F_j\ [bit/s]$	Computing rate for CPU at frequency f_j
$L_{tot}\ [Mbit]$	Data chunk size
$R_i\ [bit/s]$	Communication rate of the i-th link
$R_t\ [bit/s]$	Aggregate communication rate of the LAN
$T\ [s]$	Per-chunk maximum allowed computing time
$t_{ij}\ [s]$	Computing time for $VM(i)$ at F_j
$T_t\ [s]$	Per-chunk maximum allowed total time
$\mathcal{E}_{CPc}(i)[Joule]$	Computing energy consumed for $VM(i)$
$\mathcal{E}_{REc}(i)[Joule]$	Reconfiguration cost for $VM(i)$
$\mathcal{E}^{CMc}(i)[Joule]$	Network energy consumed for i-th link
$P_i^{idle}[Watt]$	Idle power for i-th link
$P^{Idle}[Watt]$	Idle power for each CPU

4 *MMGreen* Optimization Problem and Solution

In this section we introduce *MMGreen*, our adaptive joint computing-plus-communication framework for resource allocation, that takes into account DVFS-based active discrete frequencies and their time fractions for each VM. Specifically, this problem aims at properly tuning the workload fractions $\{F_j t_{ij}, i = 1,\ldots,M, j = 0,\ldots,Q\}$ and the end-to-end link data transferring rates $\{R_i, i = 1,\ldots,M\}$ to minimize the overall resulting computing-plus-communication energy, formally defined as:

$$\mathcal{E}_{tot} \triangleq \sum_{i=1}^{M} \mathcal{E}_{CPc}(i) + \sum_{i=1}^{M} \mathcal{E}_{REc}(i) + \sum_{i=1}^{M} \mathcal{E}^{CMc}(i) \ \ [Joule], \quad (9)$$

where $\mathcal{E}_{REc}(i)$ is the reconfiguration cost of $VM(i)$ under the SLA constraint T_t on the allowed per-chunk processing and communication time. The last term of (9) depends, in turn, on the (one-way) delays $\{D(i), i = 1 \ldots M\}$ introduced by the Virtual LAN (Fig. 1 end-to-end virtual links). We recall that our proposal takes into account the use of DVFS technologies, so we consider that the operating frequency of each VM lies in a limited ranges of discrete frequencies. So the effect of operating at an optimal frequency is achieved by switching the VMs CPU frequencies over the possible values for different time durations. However, the presence of a specific range of discrete frequencies introduces a non-convexity in the problem that we address as follows: each VM moves from one of its discrete frequencies to the next one for processing the assigned workload; hence, the time is divided into $Q+1$ discrete unknown time variables. Therefore, for each VM we have the known vector of frequencies and the unknown vector of the corresponding time periods, where each element of the vector (i.e., t_{ij} for j-th time period of $VM(i)$) represents the length of the period during which the VM i works at frequency f_j. The system keeps the list of the active servers to decide for the next incoming workload arriving from the gateway. This information is necessary to distribute the incoming multimedia data chunks across the available servers, to minimize the average consumed energy while respecting the constraints on the execution time. From a formal perspective, the *Objective Problem (OP)* assumes the following form:

$$\min_{\{R_i,\, t_{ij}\}} \sum_{i=1}^{M}\sum_{j=0}^{Q} \left(AC_{eff} f_j^3 t_{ij}\right) + \sum_{i=1}^{M} \mathcal{E}_{REc}(i) + \quad (10.1)$$
$$+ \sum_{i=1}^{M}\sum_{j=1}^{Q} 2P_i^{CMc}(R_i) \left(\frac{F_j t_{ij}}{R_i}\right),$$

subject to:

$$\sum_{i=1}^{M}\sum_{j=1}^{Q} F_j t_{ij} = L_{tot}, \quad (10.2)$$

$$\sum_{j=1}^{Q} t_{ij} \le T, \quad i = 1,\ldots,M, \quad (10.3)$$

$$\sum_{j=1}^{Q} \frac{2F_j t_{ij}}{R_i} + T \le T_t, \ \ i = 1,\ldots,M, \quad (10.4)$$

$$\sum_{i=1}^{M} R_i \le R_t. \quad (10.5)$$

The above optimization problem can be understood as follows. Eq. (10.1) represents the joint computing-plus-communication cost, which takes into account the VMs frequency switching cost for each incoming workload. The equality in (10.2) states that the summation of products of the processing rates by their duration for all VMs should be equal to the incoming workload L_{tot}. The inequalities (10.3) and (10.4) introduce a parameter T that is the maximum time for the computation. The overall computing-plus-communication time, that is the object of the SLA, is thus split in two contributions detailed in constraints (10.3) and (10.4). Specifically (10.3) introduces a constraint on the computational time, while (10.4) refers to the communication time.

The inequality in (10.5) assures the amount of data transferred through the data center does not exceed the overall data center network capacity. This equation works like a water-filling problem in order to control aggregate end-to-end link bandwidth load balancing and adjust the bandwidth for each VM according to their assigned workload fractions.

The overall problem is non-convex because of the non-convexity of the communication terms of the objective function in (10.1). Note that the rest of the constraints are affine or can be easily written in convex form in their considered range. For this reason, we choose to split these three different activities (e.g., computation, reconfiguration of frequencies and communication) and schedule them separately for an efficient execution. Hence, we consider three tasks to be considered: *Computation-aware*, *Communication-aware*, and *Reconfiguration-aware* tasks.

From a *Computation-aware* point of view, we can simply write the *computation optimizing problem* as follows:

$$\min_{t_{ij}} \ AC_{eff} \sum_{i=1}^{M}\sum_{j=0}^{Q} f_i^3 t_{ij}, \quad (11)$$

$$\text{subject to (10.2), (10.3).} \quad (11.1)$$

On the basis of this observation, eq. (11) is linear in the control variable t_{ij} and can be easily solved based on two constraints (10.2), (10.3). We can solve this linear problem by the equation system reported in the **Appendix A**.

From a *Communication-aware* point of view, the third term in (10.1) is non-convex in the variables R_i and $F_j t_{ij}$. Formally speaking, for any assigned nonnegative vector $\overrightarrow{F_j t_{ij}}$ of the workload fractions (chunk sizes), CMc is generally non-convex in the communication rate variables $\{R_i, i = 1, \ldots, M\}$, and the resulting optimization problem is:

$$\min_{R_i} \sum_{i=1}^{M} \sum_{j=1}^{Q} 2P_i^{CMc}(R_i) \left(\frac{F_{ij} t_{ij}}{R_i} \right), \tag{12}$$

$$\text{subject to (10.4) and (10.5).} \tag{12.1}$$

It is proved in **Proposition 1** that this problem can be expressed in the convex form reported below.

Proposition 1. The expression of $\mathcal{E}^{CMc}$ can be put in the following form (see the **Appendix B** for the proof):

$$\begin{aligned} &\sum_{i=1}^{M} \sum_{j=1}^{Q} 2P_i^{CMc}(R_i) \left(\frac{F_j t_{ij}}{R_i} \right) = \\ &(T_t - T) \sum_{i=1}^{M} \sum_{j=1}^{Q} P_i^{CMc} \left(\frac{2F_j t_{ij}}{T_t - T} \right). \end{aligned} \tag{13}$$

The following *Proposition 2* describes the feasibility conditions for the optimization problem in (9).

Proposition 2. The following set of conditions is *necessary and sufficient for the feasibility* of the optimization problem in (10.1)-(10.5) (see the **Appendix C** for the proof):

$$L_{tot} \leq R_t \frac{(T_t - T)}{2}, \tag{14.1}$$

$$L_{tot} \leq \sum_{i=1}^{M} T F_Q. \tag{14.2}$$

From a *Reconfiguration-aware* point of view, the second term in (10.1) (i.e., $\mathcal{E}_{REc}(i)$) can be split into two reconfiguration costs. The first one is the cost of changing discrete frequencies of $VM(i)$ from f_j to f_{j+k} (i.e., k steps movement to reach to the next *active discrete frequency*) and span $t_{i(j+k)}$ seconds. The second cost is the reconfiguration cost for the switching from the current *active discrete frequency* of $VM(i)$ to the first *active discrete frequency* of $VM(i)$ in the next slot time. Note that *active discrete frequencies* are found based on their related times-quota variables. In other words, an *active discrete frequency* f_j for VM i is characterized by $t_{ij} > 0$. We use the FCFS (First Come, First Serve) technique for visiting each frequency in the active discrete frequency list of VM i: it means that we start from the first active discrete frequency, f_{ik}, and move to the second active discrete frequency in the list, $f_{i(k+1)}$. Therefore, we calculate the difference as follows: $\Delta f_{ik} \triangleq f_{i(k+1)} - f_{ik}$, and the resulting reconfiguration cost is $k_e \Delta f_{ik}^2$. We continue until the end of the active frequency list. If we consider homogeneous VMs, the total cost of internal-switching for all VMs is: $k_e \sum_{i=1}^{M} \sum_{k=0}^{K} (\Delta f_{ik}^2)$, where $k \in \{0, 1, \ldots, K\}$, $K \leq Q$ is the number of active discrete frequencies for VM i (the *internal* reconfiguration cost). On the other hand, the external-switching cost is calculated as multiplication of k_e with the quadratic differences between the last active discrete frequency of VM i for the current workload and the primary active discrete frequency of VM i in the next incoming workload, which is denoted as Ext_Cost (the *external* reconfiguration cost). In a nutshell, the total reconfiguration energy can be written as: $\sum_{i=1}^{M} \mathcal{E}_{REc}(i) = k_e \sum_{i=1}^{M} \sum_{k=0}^{K} (\Delta f_{ik})^2 + k_e \sum_{i=1}^{M} Ext\_Cost$. In the worst case, $K = Q$, we need to move Q steps to f_0. In this case, we need to visit all the possible active discrete frequencies of each VM i (internal-switching cost: $k_e\, M \sum_{k=0}^{Q} (\Delta f_k)^2$); external-switching cost: $k_e\, M (f_Q^t - f_0^{t-1})^2$, where t is the current time and $(t-1)$ is the previous time (refers to the incoming workload in the previous time slot), and f_Q^t and f_0^{t-1} express the maximum discrete frequency of each VM (i.e., we assume VMs are homogeneous) and idle discrete frequency (the first frequency range of each VM) while the VM received the t-th workload and the $(t-1)$-th workload, respectively.

5 Experimental results

This section evaluates the performance of the *MMGreen* computing-plus-communication optimization framework for a set of offered workloads, and compares the simulation results with alternative solutions based on the *Lyapunov* method [10], the Networked Data Centers (*NetDC*) [21] and the *Hybrid NetDC* [20] approaches. It is worth to note that the last two solutions take into account reconfiguration and communication costs; on the other hand, *Hybrid NetDC* compared to *NetDC* has higher energy consumption in terms of energy provisioning due to the fixed end-to-end link power for each VMs. We also tested *MMGreen* with one of the most commonly adopted solutions: the STAtic Scheduler (*STAS*) [16].

5.1 Experimental Setup

In order to evaluate the computing-plus-communication energy $\bar{\mathcal{E}}_{tot}$ consumed by the proposed *MMGreen* solution, we implemented a prototype of the adaptive framework as part of a simulated cloud environment. The prototype is based on a paravirtualized environment using Xen 3.3 as VMM and Linux 2.6.18 as guest OS kernel. The framework is implemented at the driver domain (i.e., Dom0) of the legacy Xen 3.3. Out of approximately 1100 lines code needed for implementing the framework, 45% is directly based on the existing Xen/Linux code. The reused code includes part of the Linux's TCP New Reno congestion control suite and Xen's I/O buffer management.

The simulator, namely TEST-DVFS, works by using CVX solver, the state-of-the-art Stanford optimizing solver over Matlab [38]. TEST-DVFS simulates the algorithm in DVFS-enabled data centers by enabling DVFS functionalities not only for the components performance model but also for the offered workloads and energy models.

5.1.1 TEST-DVFS implementation

The goal of the implemented testbed is to demonstrate the effectiveness of *MMGreen* framework in reducing

computing-plus-communication energy compared to the other available techniques, and to support a sensitivity analysis with respect to its parameters.

The TEST-DVFS testbed consists of the following modules:

1) *Workload module*: This module is developed to simulate various types of offered workloads that can be either parameter-based synthetic workloads or a real trace;
2) *Component module*: All the considered components of the system (e.g., VMs, channels, DVFS - see Fig. 1) are implemented in this module;
3) *Working module*: The working module focuses on the energy model, scheduling types and network topology.

5.1.2 Test Workload

In this performance evaluation we consider both a synthetic and a realistic workload. For both workloads, we assume that client requests for processing multimedia data arrive at the data center, and each request refers to the processing of a data chunk.

For the synthetic workload, Tables 2 and 3 summarize the test parameters for the TEST-DVFS simulator: Table 2 lists the parameters with fixed values, while Table 3 shows the ranges of the parameter values used for the sensitivity analysis, with their default values. Values of framework parameters come from studies in [39], [40], and a preliminary set of test validets the energy consumption results of our simulator against data from real servers[3]. Values related to workload application parameters ($\overline{L}_{tot}$ and Peak-to-Mean Ratio (PMR)) refer to a multimedia-oriented application that provides an annotation service for images. Specifically, the considered application receives as input a set of high resolution JPEG images (possibly part of an MJPEG video stream) and performs a task of face detection using the Viola-Jones algorithm [41]. The faces recognized by the application are marked in the image and an out data stream is sent back to the clients. Using a prototype implementation of the application we also measured the processing rates for different frequencies on an Intel Nehalem Quad-core Processor [27] system (parameter F in Table 2). In order to account for the effects of the reconfiguration costs and the time-fluctuations of the offered workload on the energy performance of the tested solutions, as in [10], we model the synthetic workload as an independent identically distributed (i.i.d.) random sequence $\{L_{tot}(m), m = 0, 1, \ldots\}$, (where m is the index of input chunk), whose sizes are uniformly distributed over the interval $[max(0, \overline{L}_{tot}(2 - PMR)), \overline{L}_{tot}PMR]$.

It is worth to note that a qualifying point of our experimental setup is to consider different values for the PMR parameter, in order to evaluate the case of highly variable workloads that are typical of modern data centers hosting multimedia-oriented applications. Testing the sensitivity of the performance of the proposed framework to the PMR of the offered workload is one of the goals of these simulations.

3. https://www.energystar.gov/ia/products/prod˙lists/enterprise_servers_prod_list.xls

TABLE 2: Fixed parameters test values

Parameter	
$R_t = 10$ $[Gb/s]$	$T_t = 5$ $[s]$
F={0.65, 8.12, 9.27, 11.01, 11.60} $[Mb/s]$	$A = 1$
f={0.15, 1.86, 2.13, 2.53, 2.66} $[GHz]$	$Q = 4$
$\overline{RTT}_i = 700$ $[\mu s]$	$C_{eff} = 1$
$P_i^{idle} = 5$ $[mW]$	$P^{Idle} = 5$ $[W]$
$\overline{L}_{tot} = 90$ $Gbit$	

TABLE 3: Variable parameters test values

Parameter	Range	Default	Unit
M	{5000 : 500 : 8000}	5000	-
T	{1.25 : 0.25 : 4}	3	s
k_e	{0.005, 0.05, 0.5}	0.005	$J/(Mbit/s)^2$
Ω_i	{0.5, 5, 50}	5	mW
PMR	{1.5 : 1 : 3.5}	1.5	-

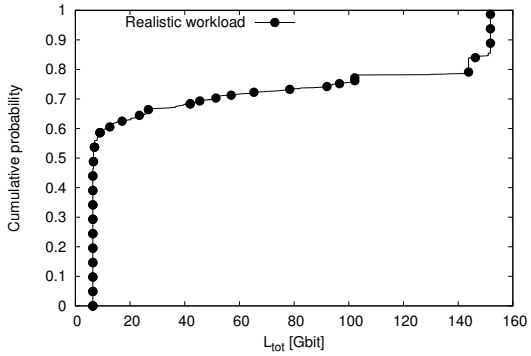


Fig. 3: Cumulative distribution of realistic workload

Our performance evaluation also considers a realistic workload based on the traces coming from [42]. In particular, we consider the traces related to EDU1 data center, that describe the activity of a real data center processing big amount of data, including multimedia applications. Fig. 3 shows the cumulative distribution of $\overline{L}_{tot}$ for the realistic workload, which is characterized by $\overline{L}_{tot} = 54.27$ $Gbit$ and $PMR = 3.79$. As for the framework parameters, we consider the values described in Tables 2 and 3.

It is worth to note that another qualifying point of our experiments is to consider for both workloads a variable number of VMs. We recall that powering ON or OFF a server is a decision taken by a long-term server consolidation strategy: evaluating the performance of the framework for different numbers of VMs is important because the framework is not always guaranteed to work with an optimal number of active VMs for the incoming workload. In all experiments each tested point has been evaluated by averaging over 1000 independent runs.

5.2 Performance Comparison

In this section, we compare the performance of *MMGreen* with *NetDC* [21], *Lyapunov* [10] and *Hybrid NetDC* [20] alternatives in terms of energy-savings for both the synthetic and realistic workloads previously introduced. We highlight that the *NetDC* and *Lyapunov* frameworks are not solutions that

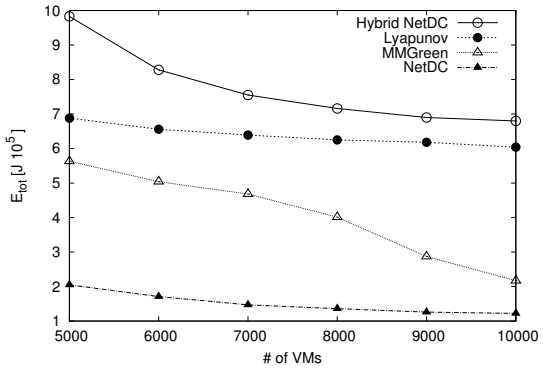


Fig. 4: Performance comparison: synthetic workload

can be directly implemented in a real system, because they consider that CPU frequency may assume any value within a given operating range and do not take into account the presence of a limited set of available operating frequencies for every CPU. For these reasons, we consider the energy consumptions achieved by these frameworks as purely theoretical results. On the other hand, the *Hybrid NetDC* implements the solution of OP, that is the constrained minimization problem in (10.1)-(10.5) over the variable $\{t_{ij}\}$ with $R_i = R_t$ and provides a solution that can be applied to real systems.

5.2.1 Synthetic Workload

In the first comparison, we consider the standard scenario described in Section 5.1.2 for the synthetic workload and we compare the energy consumption of the considered solutions.

The plot of Fig. 4 shows the per-VM total energy $\overline{\mathcal{E}}_{tot}$ for the considered frameworks as the number of VMs grows to 5000 to 10000. The curves in the graph clearly show that the energy consumption of the different solutions can span over nearly one order of magnitude, with the *Hybrid NetDC* framework providing worse results than the alternatives. This poor performance can be explained by considering that the *Hybrid NetDC* is not taking into account the communication costs, and this leads to an explosion of communication-related energy that is concentrated in short bursts where the link utilization is maximum. The other frameworks, which are aware of communication costs, achieve significant energy savings. The *Lyapunov* framework achieves an energy consumption that is 18% to 64% higher when compared to the *MMGreen* alternative, confirming the advantage of considering periodic reconfiguration of the operating CPU frequency. Finally, the *NetDC* framework achieves the best performance. However, this optimal result is based on the unrealistic assumption that CPU can operate at any frequency within an allowed range instead of considering just a discrete set of values. Hence, this result should be considered as a theoretical lower bound for the energy consumption.

A second analysis takes into account the static scheduler (*STAS*), that is commonly used in cloud data centers [16]. Indeed, current virtualized data centers usually rely on static resource provisioning, where, by design, a *fixed* number of VMs constantly run at the highest possible frequency f_i^{max} (corresponding to the maximum processing rate F_i^{max}) [33].

TABLE 4: Energy savings attained by *MMGreen*, *Lyapunov* and *HybridNetDC* over *STAS*

PMR	MMGreen	Lyapunov	HybridNetDC
1.5	61%	50%	37%
2.5	59%	47%	33%
3.5	57%	41%	28%

The goal is to constantly provide the computing capacity needed for satisfying the peak input workload, that is $\overline{L}_{tot} \cdot PMR\ (Mbit)$. It is worth to note that, although the *STAS* solution does not experience reconfiguration costs, the approach induces resource overbooking and wastes a significant amount of energy, because CPU runs at the maximum frequency even when the workload is far from the peak conditions. However, the capacity planning studies in [33] refer to static schedulers and this provides an additional motivation for considering the energy performance of the *STAS* as a benchmark. Table 4 reports the average energy savings provided by the *MMGreen*, *Lyapunov* and *HybridNetDC* frameworks over the static one (*STAS*) for the basic experimental scenario described Section 5.1.2. The energy savings of a frameworks S is defined as $\frac{\overline{\mathcal{E}}_{tot}^{STAS} - \overline{\mathcal{E}}_{tot}^{S}}{\overline{\mathcal{E}}_{tot}^{STAS}}$.

In order to guarantee a fair comparison of the different solutions, the numerical results of Table 4 have been evaluated by forcing the aforementioned frameworks to utilize the same number of VMs used by the *STAS* scheduler.

If we look at the results in Table 4 we observe three significant facts. First, the average energy saving of the *MMGreen* framework over the *STAS* alternative is in the order of 60% even considering the discrete processing frequencies and the related reconfiguration energy overhead. This result confirms that *MMGreen* is an effective solution to cope with the sudden time-variations exhibited by the workload. Second, if we compare the *MMGreen* solution with the other considered alternatives, we achieve a significantly higher energy saving (higher than 20%). Third, as the PMR increases, the energy saving is reduced. This last effect can be explained considering that, as PMR grows, every considered framework must configure the system to increase the CPU frequency (although this may occur for short periods of time). This behavior clearly reduces the difference with the *STAS* scheduler that always run at the highest CPU frequency, thus explaining the reduction in energy saving. However, we remark that even as PMR increases, the *MMGreen* framework outperforms the other considered alternatives.

5.2.2 Realistic Workload

The last comparison among multiple frameworks is based on the realistic workload introduced in Section 5.1.2. We aim to validate our previous findings on the energy consumption under a realistic scenario that is characterized by a significantly high variance in the workload intensity (as testified by the higher PMR exhibited by the realistic workload).

Fig. 5 provides a view on how $\overline{\mathcal{E}}_{tot}$ changes with the number of VMs for the various considered frameworks.

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.2016.2617367, IEEE Transactions on Cloud Computing

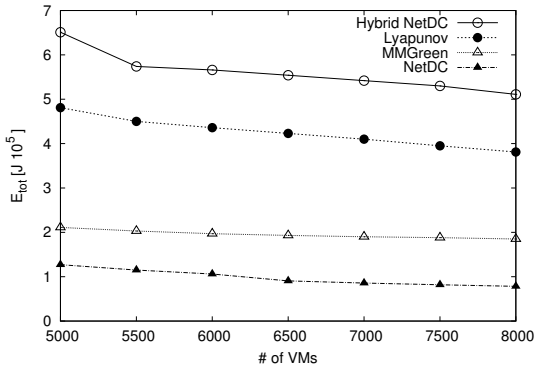


Fig. 5: Performance comparison: realistic workload

A comparison with Fig. 4 shows that the overall energy consumption tends to be lower for the realistic scenario compared with the synthetic one even if the latter has a lower peak value for L_{tot}. This counterintuitive effect can be explained considering that the realistic workload, even if it shows significant intensity peaks, has an average value for L_{tot} significantly lower compared with the synthetic one. However, even with this higher variability, the conclusion of the comparison remains the same as the previous section: the *MMGreen* is the best alternative, excluding the theoretical bound provided by the *NetDC* framework.

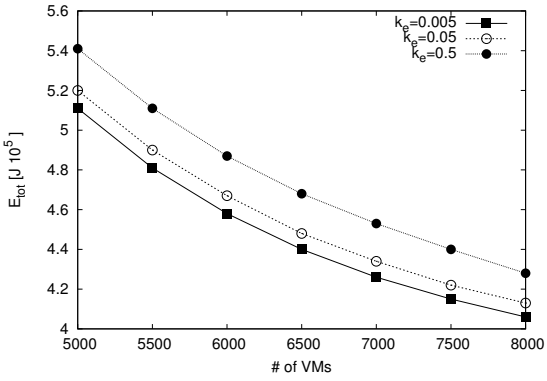
5.3 Sensitivity Analysis of *MMGreen*

In order to fully evaluate the *MMGreen* framework, we test our proposal under various operating scenarios detailed in the following subsections to understand how the data center characteristics and the framework parameters affect its performance.

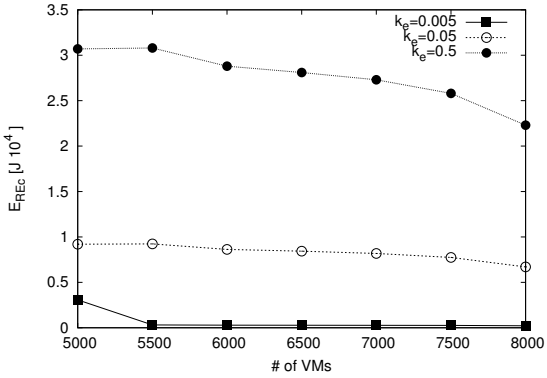
5.3.1 Sensitivity to Dynamic Reconfiguration Parameters

The first experiment focuses on the impact of dynamic frequency reconfiguration for different numbers of VMs and for the different values of the parameter k_e. We recall that k_e is the parameter affecting the weight of energy reconfiguration in DVFS-enabled CPUs. The values of k_e considered in our experiments are reported in Table 3 and span over two orders of magnitude to encompass a wide range of scenarios. Specifically, Fig. 6a shows the effects of the reconfiguration cost on the per-VM total energy $\overline{\mathcal{E}}_{tot}$, while Fig. 6b provides a detail of the reconfiguration contribution $\overline{\mathcal{E}}_{REc}$ to the total energy $\overline{\mathcal{E}}_{tot}$. The Fig. 6a shows that, as k_e grows, the power consumption grows, as testifies by the neatly stacked curves in the figure. This effect is even more evident if we look at Fig. 6b where the reconfiguration contribution $\overline{\mathcal{E}}_{REc}$ drops as k_e decreases.

A second important result is that, as the number of VMs grows, the VMs tend to operate at lower frequencies with a twofold effect. First, the per-VM energy is reduced due to the non-linear relationship between frequency and power consumption (this effect is clearly shown by the total energy curves in Fig. 6a). Second, less frequency switches are required (with a consequent reduction of $\overline{\mathcal{E}}_{REc}$ shown in Fig. 6b), because the VMs do not have to explore the full spectrum of available frequencies but can operate just at the lowest values of f_j.



(a) Per-VM total energy



(b) Per-VM reconfiguration energy

Fig. 6: Sensitivity to dynamic reconfiguration parameters

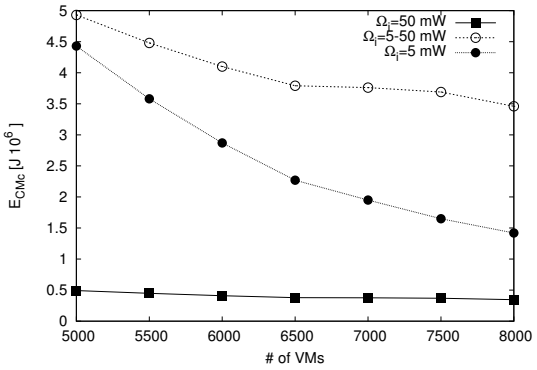


Fig. 7: Sensitivity to communication parameters

5.3.2 Sensitivity to Communication Parameters

The second experiment focuses on the impact of communication parameters over the energy consumption in *MMGreen*.

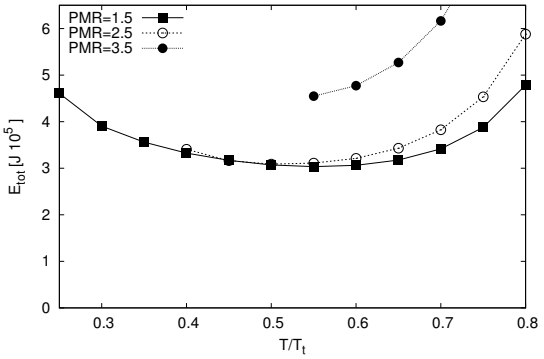
Fig. 7 provides an analysis of per-VM $\overline{\mathcal{E}}_{CMc}$ for different Ω_i values; we recall that Ω_i is the power consumption to transmit one bit of data every RTT in the data center network. We consider both an homogeneous data center (with two different values of Ω_i) and an heterogeneous data center where Ω_i can change across different VMs, as described in Tab 3. From Fig. 7 we observe that the parameter Ω_i has a major impact on the energy consumption of the data center for communication, with the two homogeneous nodes achieving the highest and lowest energy consumptions and the heterogeneous scenario obtaining a value in between

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.2016.2617367, IEEE Transactions on Cloud Computing

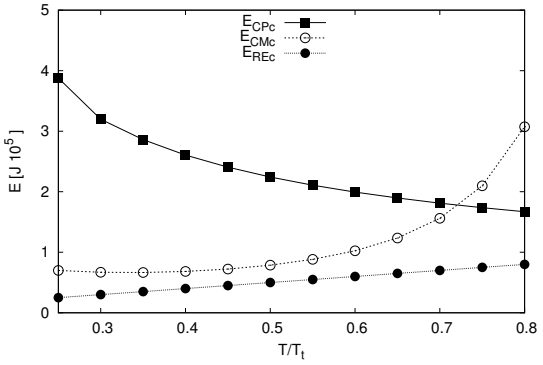
the two extremes. Furthermore, we observe that, as the computational tasks can be distributed over multiple VMs, the energy consumption is reduced due to the sharing of the incoming load over multiple VMs. This results in a double effect of reducing the amount of per-VM data to exchange and reducing the exchange data rate of the VMs network interfaces with a consequent benefit on the energy consumption.

5.3.3 Sensitivity to Maximum Computation Time

We now evaluate how the T parameter (that is the time reserved for computation) affects the performance of the *MMGreen* framework. Specifically, we evaluate energy consumption for different values of the T/T_t ratio and for different values of Peak-to-Mean Ratio (PMR) according to Tab. 3. In order to allow for a large value of PMR, in this specific experiment we reduce the average workload size to $\overline{L}_{tot} = 45Gbit$.



(a) Per-VM total energy



(b) Contributions to the total energy (PMR=1.5)

Fig. 8: Sensitivity to T/T_t and to PMR

Fig. 8a shows the per-VM total energy consumption $\overline{\mathcal{E}}_{tot}$ for different PMR and T/T_t values. If we compare the three curves for the different considered PMR, we observe that increasing the PMR, and thus the variance of the incoming workload, has a twofold effect. First, it increases the overall power consumption, because the data center must use for longer periods of time the highest CPU frequencies to cope with the workload peaks. For example, if we compare the curve of PMR=3.5 with the curve for 1.5 (the two extreme cases), we observe that the first is characterized by an energy consumption that is at least 60% higher with respect to the less variable workload. Second, the viability range, that is the set of values of T/T_t for which our framework can find a solution is reduced as PMR grows. Indeed, for high values of PMR, the highest peaks of data cannot be processed satisfying constraint 10.3 when the T is too low. For example, for PMR=3.5 we have that $T/T_t < 0.55$ results in SLA violation for the largest values of L_{tot}. On the other hand, as PMR=1.5 does not involve the service of large chunk of data, we can reach values of T/T_t as low as 0.25.

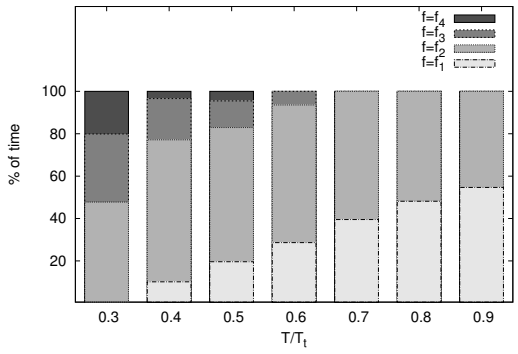


Fig. 9: Time breakdown for different operating frequencies

If we focus on the curve marked with the black squares (PMR=1.5) we observe that energy consumption follows a concave behavior as a function of T/T_t. The reasons for this behavior can be better understood focusing on Fig. 8b, that reports the three components of $\overline{\mathcal{E}}_{tot}$. As expected, the computational contribution $\overline{\mathcal{E}}_{CPc}$ tends to be higher than the other components, however, it is affected by the value of T/T_t: specifically, $\overline{\mathcal{E}}_{CPc}$ is higher for low values of T/T_t and then decreases as the computation can be spread over longer times and the CPU can run the highest frequencies for less time. On the other hand, the communication contribution $\overline{\mathcal{E}}_{CMc}$ is not negligible and has an exactly opposite behavior, because for lower values of T/T_t the data center can spread communication over long time periods, while as T/T_t grows the data rate must increase, resulting in higher energy consumption for data transfer. Finally, the reconfiguration cost remains almost constant for the different considered values of T/T_t. The contribution of these three terms explains the resulting total energy curve because energy consumption is dominated by the computation contribution at the lowest extreme of T/T_t and by the communication contribution at the other end. The default value used throughout the experiments is an intermediate value that corresponds to the best time division between computation and communication.

As a final analysis, we focus on evaluating the impact of the reduction of T/T_t on the computation-related component of the energy. To this aim, Fig. 9 provides a breakdown of the time spent operating at each discrete frequency for different values of T/T_t (again we refer to the standard case where PMR=1.5). We observe that, as T/T_t is reduced, the time spent operating at the highest frequencies grows. As T/T_t exceeds 0.5 the highest frequency is no longer used, and for $T/T_t > 0.6$ only the two lowest frequencies are used to serve the incoming requests. This insight is a further confirmation of our initial claim that reducing the time for computation forces the framework to have the VMs operating at higher frequencies for longer times, thus

resulting in higher energy consumption.

6 Conclusions

In this paper, we proposed the *MMGreen* optimization framework for a joint adaptive allocation of computing and communication rates in energy-efficient data centers for multimedia data processing. Although the resulting optimization problem is inherently non convex, we exploited its loosely coupled structure to achieve an analytical solution of the problem. Our experiments with both synthetic and realistic workloads confirm that *MMGreen* outperforms any other alternative that can be applied to a DVFS-enabled data center, with an energy saving up to more than 60% compared to the static solutions often used in data centers, where the CPU operating frequency is not adjusted to match the incoming workload. Our performance analysis provides also a detailed insight on the impact of the algorithm and model parameters on the overall performance of the proposed framework.

Appendix A
Derivations of the OP solution

Since the constraint in (10.3) is already accounted for by the feasibility condition (B.6), without loss of optimality, we may directly focus on the solution of the optimization problem in (10.1) under the constraints in (10.2), (10.4), (10.5). Since this problem is strictly convex and all its constraints are linear, the Slater's qualification conditions hold [43, chap.5]. We observe that each power-rate function in the third term of (10.1) is non decreasing for: $F_{ij}t_{ij} \geq 0$, so that, without loss of optimality, we may replace the equality constraint in (10.2) by the following equivalent one: $\sum_{i=1}^{M}\sum_{j=1}^{Q} F_{ij}t_{ij} \geq L_{tot}$. Moreover, the frequencies for each VM is known so it plays the role of a coefficient. Therefore, the *OP* may be simplified as follows:

$$\min_{t_{ij}} \sum_{i=1}^{M}\sum_{j=0}^{Q}\left(AC_{eff}f_j^3 t_{ij}\right) + \sum_{i=1}^{M}\mathcal{E}_{REc}(i) \tag{A.1.1}$$
$$+\sum_{i=1}^{M}\sum_{j=1}^{Q}(T_t-T)P_i^{CMc}\left(\frac{2F_j t_{ij}}{T_t-T}\right),$$
$$\text{s.t.: } \sum_{j=0}^{Q} t_{ij} - T = 0, \quad i=1,\ldots,M, \tag{A.1.2}$$
$$L_{tot} - \sum_{i=1}^{M}\sum_{j=1}^{Q} F_j t_{ij} = 0, \tag{A.1.3}$$
$$t_{ij} - T \leq 0, \;\; i=1,\ldots,M,\; j=0,\ldots,Q. \tag{A.1.4}$$

After denoting the objective function in (A.1.1) by $\mathcal{Z}(\{t_{ij}\})$, we have: $\mathcal{Z}(\{t_{ij}\}) \triangleq$ (A.1.1). The partial derivative of $\mathcal{Z}(.)$ with respect to t_{ij} is given by

$$\frac{\partial \mathcal{Z}(.)}{\partial t_{ij}} = AC_{eff}f_j^3 + (T_t-T)\frac{\partial P_i^{CMc}(.)}{\partial t_{ij}}, \quad i=1,\ldots,M,\; j=0,\ldots,Q. \tag{A.2}$$

Hence, the $\mathcal{Z}(.)$ is linear and by equating the partial derivatives of (A.2) to zero, we can solve the resulting algebraic equation with respect to t_{ij}. In this way, we calculate the $M(Q+1)$ variables by solving the aforementioned linear problem, which is the same as the Gauss-Jordan system which is produced by the M equations in (A.1.2) and the equations in (A.1.3) and (A.2).

Appendix B
Proof of Proposition 1

Let $\{R_i^*\left(\overrightarrow{F_j t_{ij}}\right), i=1,\ldots,M\}$ be the optimal solution of the eq. (12), and let

$$\mathcal{C} \triangleq \left\{ \left(\overrightarrow{F_j t_{ij}}\right) \in (\mathbb{R}_0^+)^M : \left(\sum_{j=1}^{Q} F_j t_{ij}/R_i^*\left(\overrightarrow{F_j t_{ij}}\right)\right) \leq (T_t-T)/2, i=\{1,2,\ldots,M\}, j=\{1,2,\ldots,Q\}; \sum_{i=1}^{M}\sum_{j=1}^{Q} R_i^*\left(\overrightarrow{F_j t_{ij}}\right) \leq R_t \right\}, \tag{B.1}$$

be the region of nonnegative M-dimensional Euclidean space constituted by all $\overrightarrow{F_j t_{ij}}$ vectors meeting the constraints in (10.4) and (10.5). For the feasibility of (12) we have that:

i) the CMc in (12) is feasible *if and only if* the vector $\overrightarrow{F_j t_{ij}}$ meets the following condition:

$$\sum_{i=1}^{M}\sum_{j=1}^{Q} F_j t_{ij} \leq (R_t(T_t-T))/2; \tag{B.2}$$

ii) the solution of the CMc is given by the following closed-form expression:

$$R_i^*\left(\overrightarrow{F_j t_{ij}}\right) \equiv R_i^*\left(\sum_{j=1}^{Q} F_j t_{ij}\right) \equiv \left(\sum_{j=1}^{Q} 2F_j t_{ij}/(T_t-T)\right), i=1,\ldots,M. \tag{B.3}$$

For any assigned $\overrightarrow{F_j t_{ij}}$, the objective function in (12) is the summation of $M(Q+1)$ nonnegative terms, where the ij-th term depends only on R_i for all j. Thus, being the objective function in (12) separable, its minimization may be carried out component-wise. Since the ij-th term in (12) is increasing in R_i and the constraints in (10.4) and (10.5) must be met, the ij-th minimum is attained when the constraints in (10.4) and (10.5) are binding, and this proves the validity of (B.2). Finally, the set of rates in (B.3) is feasible for the CMc *if and only if* the constraint in (10.5) is met, and this proves the validity of the feasibility condition in (B.3). Moreover, the end-to-end links power cost: $\sum_{j=1}^{Q} 2P_i^{CMc}(R_i)\left(F_j t_{ij}/R_i\right)$ is continuous, nonnegative and nondecreasing for $R_i > 0$, $\forall i \in \{1,\ldots,M\}$, with the multi-variable coefficient which can be feasible if only the following equation holds (i.e., we use "$\rightarrow$" which means *implies*):

$$\sum_{j=1}^{Q}\frac{2F_j t_{ij}}{R_i} + T \leq T_t \rightarrow \left(\sum_{j=1}^{Q}\frac{F_j t_{ij}}{R_i}\right) \leq \frac{(T_t-T)}{2}. \tag{B.4}$$

Equation (B.4) is obtained by manipulating equation (10.4). To make the optimization problem easier to solve, we recast

the second control variable by rewriting R_i as a function of t_{ij} as follows:

$$\sum_{j=1}^{Q} \frac{2F_j t_{ij}}{R_i} + T \leq T_t \rightarrow R_i \geq \sum_{j=1}^{Q} \left(\frac{2F_j t_{ij}}{T_t - T} \right). \tag{B.5}$$

So, we can introduce equations (B.4) and (B.5) into the third term of the objective function in (10.1), in order to attain the following formula:

$$\begin{aligned} &\sum_{i=1}^{M} 2P_i^{CMc}(R_i) \left(\sum_{j=1}^{Q} \frac{F_j t_{ij}}{R_i} \right) = \\ &(T_t - T) \sum_{i=1}^{M} P_i^{CMc} \left(\sum_{j=1}^{Q} \frac{2F_j t_{ij}}{T_t - T} \right). \end{aligned} \tag{B.6}$$

To recap, the end-to-end link function $\mathcal{E}^{CMc}$ which is based on two control variables ($\mathcal{G}(R_i; t_{ij})$) can be written as in:

$$\mathcal{E}^{CMc}(i) = \mathcal{G}\left(R_i; t_{ij}\right) \triangleq \mathcal{H}(t_{ij}). \tag{B.7}$$

The new formula for energy-aware communication end-to-end link just depends on the summation of time variables for each VM and the main function ($\mathcal{H}(.)$) can be written according to the equation (7). Thus, this proves the third term in (10.1) is *convex*.

APPENDIX C
PROOF OF PROPOSITION 2

Eq. (14.1) stems from the constraint in equation (10.4) as detailed below:

$$\begin{aligned} &\sum_{j=1}^{Q} \frac{2F_j t_{ij}}{R_i} + T \leq T_t \overset{(a)}{\rightarrow} \sum_{j=1}^{Q} F_j t_{ij} \leq \frac{(T_t - T)}{2} R_i \overset{(b)}{\rightarrow} \\ &\sum_{i=1}^{M} \sum_{j=1}^{Q} F_j t_{ij} \leq \frac{(T_t - T)}{2} \sum_{i=1}^{M} R_i \overset{(c)}{\rightarrow} L_{tot} \leq R_t \frac{(T_t - T)}{2}, \end{aligned} \tag{C.1}$$

where in (a), we swap the equation positions and calculate $\sum_{j=1}^{Q} F_j t_{ij}$ based on R_i; in (b), the left term (which is positive) is less than the right term (which is positive too). Therefore, we derive the summation for all discrete time fractions, and this equation can be derived and expanded; finally, in (c), the left hand of the inequality: $\sum_{i=1}^{M} \sum_{j=1}^{Q} F_j t_{ij}$ is equal to L_{tot} and the right hand inequality, $\sum_{i=1}^{M} R_i$, stems from the equation (10.5) which is less than R_t and it is obvious that left hand of the equation is less than $R_t(T_t - T)/2$. We next prove the second part of (14).

To prove the eq. (14.2), we start from (10.3). Thus, we have:

$$\begin{aligned} &\sum_{j=0}^{Q} t_{ij} = T \overset{(d)}{\rightarrow} \sum_{j=1}^{Q} t_{ij} \leq T \overset{(e)}{\rightarrow} F_i^{max} \sum_{j=1}^{Q} t_{ij} \leq TF_Q \overset{(f)}{\rightarrow} \\ &\sum_{i=1}^{M} \left(F_Q \sum_{j=1}^{Q} t_{ij} \right) \leq \sum_{i=1}^{M} (TF_Q) \overset{(g)}{\rightarrow} \\ &\sum_{i=1}^{M} \sum_{j=1}^{Q} (F_j t_{ij}) \leq \sum_{i=1}^{M} \left(F_Q \sum_{j=1}^{Q} t_{ij} \right) \leq \sum_{i=1}^{M} (TF_Q) \overset{(h)}{\rightarrow} \\ &\sum_{i=1}^{M} \sum_{j=1}^{Q} (F_j t_{ij}) \leq \sum_{i=1}^{M} TF_Q \overset{(i)}{\rightarrow} L_{tot} \leq \sum_{i=1}^{M} TF_Q, \end{aligned} \tag{C.2}$$

In eq. (C.2), we have that: (d) holds, because the summation of the VM's time fractions should be equal or less than the total hard-limit assigned for each server; (e) and (f) represent the summation over M VMs for the calculated inequality; (g) shows that the left hand side of the inequality achieved after (f) is higher than the constraint in (10.2) and (h) indicates that this inequality can be simplified as the result of (i). This proves the validity of (14.2).

ACKNOWLEDGEMENT

The first three authors acknowledge the support of the University of Modena and Reggio Emilia through the project *SAMMClouds: Secure and Adaptive Management of Multi-Clouds*.

REFERENCES

[1] C. Canali, M. Colajanni, and R. Lancellotti, "Performance evolution of mobile web-based services," *Internet Computing, IEEE*, vol. 13, no. 2, pp. 60–68, 2009.

[2] EPA, "Report to congress on server and data center energy efficiency," US Environmental Protection Agency, Tech. Rep., 2007.

[3] J. Whitney and P. Delforge, "Data center efficiency assessmenty – scaling up energy efficiency across the data center industry: Evaluating key drivers and barriers," NRDC, Anthesis, Tech. Rep., 2014, – http://www.nrdc.org/energy/files/data-center-efficiency-assessment-IP.pdf.

[4] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, F. Matus, R. Pan, N. Yadav, G. Varghese *et al.*, "Conga: Distributed congestion-aware load balancing for datacenters," in *SIGCOMM'14*. ACM, 2014, pp. 503–514.

[5] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 3551–3558.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[7] M. S. Hossain, G. Muhammad, M. F. Alhamid, B. Song, and K. Al-Mutib, "Audio-visual emotion recognition using big data towards 5g," *Mobile Networks and Applications*, pp. 1–11, 2016.

[8] J. W. Jiang, T. Lan, S. Ha, M. Chen, and M. Chiang, "Joint VM placement and routing for data center traffic engineering," in *INFOCOM'12*. IEEE, 2012, pp. 2876–2880.

[9] R. Brown *et al.*, "Report to congress on server and data center energy efficiency: Public law 109-431," *Lawrence Berkeley National Laboratory*, 2008.

[10] R. Urgaonkar, U. C. Kozat, K. Igarashi, and M. J. Neely, "Dynamic resource allocation and power management in virtualized data centers," in *NOMS'10*. IEEE, 2010, pp. 479–486.

[11] W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia Cloud Computing," *IEEE Signal Processing Magazine*, vol. 28, no. 3, pp. 59–69, May 2011.

[12] S. Wang and S. Dey, "Adaptive Mobile Cloud Computing to Enable Rich Mobile Multimedia Applications," *IEEE Transactions on Multimedia*, vol. 15, no. 4, pp. 870–883, 2013.

[13] Y.-W. Ma, J.-L. Chen, C.-H. Chou, and S.-K. Lu, "A Power Saving Mechanism for Multimedia Streaming Services in Cloud Computing," *IEEE Systems Journal*, vol. 8, no. 1, pp. 219–224, March 2014.

[14] V. Mathew, R. K. Sitaraman, and P. Shenoy, "Energy-aware load balancing in content delivery networks," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 954–962.

[15] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster computing*, vol. 12, no. 1, pp. 1–15, 2009.

[16] A. Beloglazov, R. Buyya, Y. C. Lee, A. Zomaya *et al.*, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," *Advances in computers*, vol. 82, no. 2, pp. 47–111, 2011.

[17] S. Wang, A. Zhou, C. H. Hsu, X. Xiao, and F. Yang, "Provision of data-intensive services through energy- and qos-aware virtual machine placement in national cloud data centers," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 290–300, 2016.

[18] M. Malekimajd, D. Ardagna, M. Ciavotta, A. M. Rizzi, and M. Passacantando, "Optimal map reduce job capacity allocation in cloud systems," *SIGMETRICS Perform. Eval. Rev.*, vol. 42, no. 4, pp. 51–61, Jun. 2015.

[19] L. Wang, F. Zhang, J. Arjona Aroca, A. V. Vasilakos, K. Zheng, C. Hou, D. Li, and Z. Liu, "Greendcn: a general framework for achieving energy efficiency in data center networks," *Selected Areas in Communications, IEEE Journal on*, vol. 32, no. 1, pp. 4–15, 2014.

[20] N. Cordeschi, D. Amendola, F. De Rango, and E. Baccarelli, "Networking-computing resource allocation for hard real-time green cloud applications," in *Wireless Days IFIP*. IEEE, 2014, pp. 1–4.

[21] N. Cordeschi, M. Shojafar, and E. Baccarelli, "Energy-saving self-configuring networked data centers," *Computer Networks*, vol. 57, no. 17, pp. 3479–3491, 2013.

[22] D. Gmach, J. Rolia, and L. Cherkasova, "Selling t-shirts and time shares in the cloud," in *Proc. of 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2012, Ottawa, Canada, May 13-16, 2012*, 2012, pp. 539–546.

[23] C. Canali and R. Lancellotti, "Exploiting Classes of Virtual Machines for Scalable IaaS Cloud Manage ment," in *Proc. of the 4th Symposium on Network Cloud Computing and Applications (NCCA)*, Munich, Germany, Jun. 2015.

[24] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future generation computer systems*, vol. 28, no. 5, pp. 755–768, 2012.

[25] N. B. Rizvandi, J. Taheri, A. Y. Zomaya, and Y. C. Lee, "Linear combinations of dvfs-enabled processor frequencies to modify the energy-aware scheduling algorithms," in *Proc. of CCGRID*, 2010.

[26] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters," in *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. IEEE Computer Society, 2005, p. 34.

[27] G. Von Laszewski, L. Wang, A. J. Younge, and X. He, "Power-aware scheduling of virtual machines in dvfs-enabled clusters," in *CLUSTER'09*. IEEE, 2009, pp. 1–10.

[28] J. Zhuo and C. Chakrabarti, "Energy-efficient dynamic task scheduling algorithms for dvs systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 7, no. 2, p. 17, 2008.

[29] H. Kimura, M. Sato, Y. Hotta, T. Boku, and D. Takahashi, "Emprical study on reducing energy of parallel programs using slack reclamation by dvfs in a power-scalable high performance cluster," in *CLUSTER'06*. IEEE, 2006, pp. 1–10.

[30] K. Li, "Performance analysis of power-aware task scheduling algorithms on multiprocessor computers with dynamic voltage and speed," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 19, no. 11, pp. 1484–1497, 2008.

[31] K. H. Kim, R. Buyya, and J. Kim, "Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters." in *CCGRID*, vol. 7, 2007, pp. 541–548.

[32] M. Portnoy, *Virtualization essentials*. John Wiley & Sons, 2012, vol. 19.

[33] J. Baliga, R. W. Ayre, K. Hinton, and R. Tucker, "Green cloud computing: Balancing energy in processing, storage, and transport," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 149–167, 2011.

[34] M. Pióro and D. Medhi, *Routing, flow, and capacity design in communication and computer networks*. Elsevier, 2004.

[35] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center tcp (dctcp)," *ACM SIGCOMM review*, vol. 41, no. 4, pp. 63–74, 2011.

[36] T. Das and K. M. Sivalingam, "Tcp improvements for data center networks," in *(COMSNETS) 2013*. IEEE, 2013, pp. 1–10.

[37] C. Gunaratne, K. Christensen, B. Nordman, and S. Suen, "Reducing the energy consumption of ethernet with adaptive link rate (alr)," *IEEE Transactions on Computers*, vol. 57, no. 4, pp. 448–461, 2008.

[38] M. Grant and S. Boyd, "Cvx: Matlab software for disciplined convex programming."

[39] M. Shojafar, C. Canali, R. Lancellotti, and S. Abolfazli, "An Energy-aware Scheduling Algorithm in DVFS-enabled Networked Data Centers," in *Proc. of 6th International Conference on Cloud Computing and Services Science (CLOSER 2016)*, Rome, Italy, Apr. 2016.

[40] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Transactions on Cloud Computing*, 2016.

[41] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[42] T. Benson, A. Akella, and D. A. Maltz, "Network Traffic Characteristics of Data Centers in the Wild," in *Proc. of SIGCOMM Conference on Internet Measurement (IMC)*, Melbourne, Australia, Nov. 2010.

[43] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear programming: theory and algorithms*. John Wiley & Sons, 2013.

Mohammad Shojafar is currently is a researcher at the University of Modena and Reggio Emilia since January 2016. Recently, he finished his Ph.D. in ICT at the Sapienza University of Rome. He received his Msc in software engineering in Qazvin Islamic Azad University, Qazvin, Iran in 2010. Also, he received his Bsc. in computer engineering-software major in Iran university science and technology, Tehran, Iran in 2006. His current research focuses on distributed computing, wireless communications, and mathematical/AI optimization.

Claudia Canali is currently an Assistant professor at the University of Modena and Reggio Emilia since 2008. She received Laurea degree summa cum laude in computer engineering from the same university in 2002, and Ph.D. in Information Technologies from the University of Parma in 2006. Her research interests include cloud computing, social networks, and wireless systems for mobile Web access. She is a member of IEEE Computer Society. For additional information: http://weblab.ing.unimo.it/people/canali

Riccardo Lancellotti is currently an Assistant professor in the Department of Information Engineering at the University of Modena, Italy. He received the Laurea and the Ph.D. degrees in computer engineering from the University of Modena and from the University of Roma "Tor Vergata", respectively. His research interests include scalable architectures for Web content delivery and adaptation, peer-to-peer systems, distributed systems and performance evaluation. He is a member of IEEE Computer Society. For additional information: http://weblab.ing.unimo.it/people/lancellotti

Jemal Abawajy (SM12) is a full professor at Faculty of Science, Engineering and Built Environment, Deakin University, Australia. Prof. Abawajy has delivered more than 50 keynote and seminars worldwide and has been involved in the organization of more than 300 international conferences in various capacity including chair and general co-chair. He has also served on the editorial-board of numerous international journals including IEEE Transaction on Cloud Computing. Prof. Abawajy is the author/coauthor of more than 250 refereed articles and supervised numerous PhD students to completion.