
# Energy Saving in QoS Fog-supported Data Centers

**by**

**Paola Gabriela Vinueza Naranjo**

A thesis submitted
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Information and Communication Engineering
at
Sapienza University of Rome

February 2018
Cycle XXX

Author .....
Department of Information Engineering, Electronics and Telecommunications

Certified by .....
Enzo Baccarelli
Scientist
Thesis Supervisor

This thesis was evaluated by the two following external referees:

Luigi Paura, Professor, Università Federico II di Napoli, Naples, Italy

Marco Aiello, Professor, University of Groningen, Groningen, Netherlands

The time and effort of the external referees in evaluating this thesis, as well as their valuable and constructive suggestions, are very much appreciated and greatly acknowledged.

I would like to dedicate this thesis to my loving parents ...

# Declaration

I hereby declare that except where specific references are made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing, which is the outcome of work done in collaboration with others, except as specified in the text and acknowledgments. This dissertation contains fewer than 150,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 250 figures.

Paola Gabriela Vinueza Naranjo
February 2018
Cycle XXX

# Acknowledgements

Three years ago, I thought "3 long years" was really a long time far away from my country, family, and friends but actually when I have been through this journey, it felt like a flash. I'm so grateful to all the wonderful people who made it so amazing.

First of all, I am grateful to The Almighty God for establishing and providing me with the ability and perseverance that was needed to complete these studies and for the opportunity to enjoy such a wonderful experience. Foremost, my sincere gratitude to my advisor Professor Enzo Baccarelli a kind person, that continuously support of my Ph.D.career. Your continuous support, passion for research, patience, motivation, and immense knowledge set an example. Above all, he has been the most understanding and supportive person throughout my stay in Sapienza University and has helped me achieve whatever I have craved for. It is only through his views, that I have learned mostly about a researcher, a professor and most importantly - an adviser, which I deeply appreciate from my heart. I am also thankful for the excellent example he has provided as a successful person and professor.

I would also like to thank the Ecuadorian government for financially supporting my research and stay in Rome, Italy through Secretaria Nacional de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) financially supported my research activities and stay in Italy. I am extremely grateful and indebted to my unforgettable laboratory colleague, fellow and friend, Mohammad Shojafar "hola-como estas-todo bien" for all their kindness, happiness and support in difficult times, thanks for all good and bad, "gracias por todo" shojijo, and the infallible bit/lot of coffee and sweets (Couldn't have done it without them!).

I cannot forget my deepest and endless gratitude for the unconditional love and support given by my parents and my brother and for always being there beside me and for there never-ending encouragement. I feel blessed with all your love, and faith in me. I would like to thank my lovely boyfriend Navin, always there showing me the right way, helping me to face situations and encourage me to be strong in life with his love and patience. He is my perfect support person, my partner in crime and my travel buddy to see the world. I can't

express enough how much they all mean to me. Without their support and love, I wouldn't have fulfilled my dreams and lifelong ambitions.

I would like to extend my thanks to mom Italian Rocío, her sons Kenny and Brandon, my roommate Camila, Hieda and Zahra lovely friends that I had the pleasure to meet here in "BELLA ITALIA". This thesis would not have happened without your help, support, and friendship. Thank you with all my heart!

Paola G. Vinueza N.

# Abstract

One of the most important challenges that cloud providers face in the explosive growth of data is to reduce the energy consumption of their designed, modern data centers. The majority of current research focuses on energy-efficient resources management in the infrastructure as a service (IaaS) model through "resources virtualization" - virtual machines and physical machines consolidation. However, actual virtualized data centers are not supporting communication–computing intensive real-time applications, big data stream computing (info-mobility applications, real-time video co-decoding). Indeed, imposing hard-limits on the overall per-job computing-plus-communication delays forces the overall networked computing infrastructure to quickly adopt its resource utilization to the (possibly, unpredictable and abrupt) time fluctuations of the offered workload.

Recently, Fog Computing centers are as promising commodities in Internet virtual computing platform that raising the energy consumption and making the critical issues on such platform. Therefore, it is expected to present some green solutions (i.e., support energy provisioning) that cover fog-supported delay-sensitive web applications. Moreover, the usage of traffic engineering-based methods dynamically keep up the number of active servers to match the current workload. Therefore, it is desirable to develop a flexible, reliable technological paradigm and resource allocation algorithm to pay attention the consumed energy. Furthermore, these algorithms could automatically adapt themselves to time-varying workloads, joint reconfiguration, and orchestration of the virtualized computing-plus-communication resources available at the computing nodes. Besides, these methods facilitate things devices to operate under real-time constraints on the allowed computing-plus-communication delay and service latency.

The purpose of this thesis is: i) to propose a novel technological paradigm, the *Fog of Everything (FoE) paradigm*, where we detail the main building blocks and services of the corresponding technological platform and protocol stack; ii) propose a dynamic and adaptive energy-aware algorithm that models and manages virtualized networked data centers Fog Nodes (FNs), to minimize the resulting networking-plus-computing average energy consumption; and, iii) propose a novel Software-as-a-Service (SaaS) Fog Computing platform to integrate the user applications over the FoE. The emerging utilization of SaaS Fog

Computing centers as an Internet virtual computing commodity is to support delay-sensitive applications.

The main blocks of the virtualized Fog node, operating at the Middleware layer of the underlying protocol stack and comprises of: i) admission control of the offered input traffic; ii) balanced control and dispatching of the admitted workload; iii) dynamic reconfiguration and consolidation of the Dynamic Voltage and Frequency Scaling (DVFS)-enabled Virtual Machines (VMs) instantiated onto the parallel computing platform; and, iv) rate control of the traffic injected into the TCP/IP connection.

The salient features of this algorithm are that: i) it is adaptive and admits distributed scalable implementation; ii) it has the capacity to provide hard QoS guarantees, in terms of minimum/maximum instantaneous rate of the traffic delivered to the client, instantaneous goodput and total processing delay; and, iii) it explicitly accounts for the dynamic interaction between computing and networking resources in order to maximize the resulting energy efficiency. Actual performance of the proposed scheduler in the presence of: i) client mobility; ii) wireless fading; iii) reconfiguration and two-thresholds consolidation costs of the underlying networked computing platform; and, iv) abrupt changes of the transport quality of the available TCP/IP mobile connection, is *numerically* tested and compared to the corresponding ones of some state-of-the-art static schedulers, under both synthetically generated and measured real-world workload traces.

# Table of contents

# List of figures

# List of tables

# Nomenclature

| | |
|---|---|
| *AP* | Access Point |
| *BD* | Big Data |
| *BDS* | Big Data Streaming |
| *C2C* | Clone-to-Clone |
| *CC* | Cloud Computing |
| *CDN* | Content Delivery Network |
| *CNT* | Container |
| *D2D* | Device-to-Device |
| *DCN* | Data Center Network |
| *EV* | Electric Vehicle |
| *F2T* | Fog-to-Thing |
| *FC* | Fog Computing |
| *FCL* | Fog Clone |
| *FDS* | Fog Data Service |
| *FN* | Fog Node |
| *FoE* | Fog of Everything |
| *FV* | Fog Virtualization |
| *IaaS* | Infrastructure-as-a-Service |
| *IoE* | Internet of Everything |
| *IoT* | Internet of Things |
| *MAN* | Metropolitan Area Network |
| *MEC* | Mobile Edge Computing |
| *MVP* | Multi-core Virtual Processor |
| *P2P* | Peer-to-Peer |
| *PaaS* | Platform-as-a-Service |
| *QoS* | Quality-of-Service |
| *RFID* | Radio-frequency identification |
| *SaaS* | Software-as-a-Service |
| *ST* | Smart Transportation |
| *T2F* | Thing-to-Fog |
| *T2T* | Thing-to-Thing |
| *V2G* | Vehicular-to-Grid |
| *VP* | Virtual Processor |
| *WAN* | Wide Area Network |
| *WLAN* | Wireless Local Area Network |
| *WSN* | Wireless Sensor Network |

# Chapter 1

# Introduction

This chapter is for introducing the basic concepts that concern Fog Computing (FC), Internet of Everything (IoE) and Data Centers (DCs), to understand the purpose of our work, justifying its value and identifying applications. Giving a short description of the structures and technological paradigms before mentioned, and describe some challenges and the proposed objectives to address them. FC is a quite novel computing paradigm that aims at moving the Cloud Computing (CC) facilities and services to the access network, involving various types of applications in a consider pervasive network of densely distributed energy and resource-limited wireless things, FC services are capable of gathering and transferring in real-time large volumes of heterogeneous environmental data, that runs both in the Cloud and in devices, such as smart gateways and routers [117] where billions of devices are interconnected using IoE to the Internet. Fog Nodes (FNs) are the architecture that provides resources for services at the edge of the network, processing data without the need to transmit through the Cloud. The Fog and IoE paradigms promise to reduce energy consumption and related operating costs by leveraging their native self-organizing and self-scaling capabilities to the difference to the overall operating costs of state-of-the-art Cloud-based data centers. As a result, Fog service providers are seeking innovative ways that allow them to reduce the amount of energy that their data centers consume. This chapter will be shifted attention to the architecture that proves the viability of real-time IoE-based environment and the problem of current interest in energy conservation with the aim of presenting the technological environment in the DCs.

## 1.1 Fog Computing and IoE paradigms

The goal of this subsection is review and compare the native attributes of standing-alone IoE, Fog and Cloud paradigms, introduce the Fog over IoE platform and its architecture, including data services. In Table 1.1, we present the Cloud and Fog Computing platform, emphasizing the main characteristics of both

architectures, reviewing the two main technologies currently utilized for the virtualization of the computing and networking physical resources of data centers, Virtual Machines (VM)-based and the Container (CNT)-based virtualization technologies, and review the service models done available by the resulting virtualized IoE-Fog-Cloud ecosystem.

### 1.1.1 IoE paradigms - Attributes and Challenges

The IoE model refers to an ecosystem of edge devices that autonomously share and self-manage their limited resources, to attain a common system-wide goal. The IoE paradigm is still searching for system-wide architectural solutions for many emerging application scenarios, one of which is embedded sensor networks. As sketched in Fig. 1.1, the goal of the IoE is to provide a spatially distributed technological platform for the pervasive support of Machine-to-Machine (M2M), People-to-Machine (P2M) and People-to-People (P2P) services [95, 97, 132, 134].

The IoE paradigm introduces some challenges that cannot be adequately addressed by Cloud and Host computing models alone, such as: i) reduced communication delays, where IoT applications such vehicle-to-vehicle communication or online games may require service-deployment latencies below a few tens of milliseconds, the latencies required the IoT applications mentioned cannot be supported by the remote Cloud alone; ii) wise usage of Internet bandwidth: the big data generated by things is growing and routing all these data to the remote Cloud would congest the Internet backbone, requiring that the processing of the data generated by the edge things is carried out as much as possible within the access network; iii) resource limitations of the IoE devices: most of the IoE devices are resource and energy limited, being not capable to rely alone on own capabilities to fulfill their computing-communication tasks. Moreover due to the constraints on the delays and usage of the Internet bandwidth, offloading all task to the remote Cloud is not a feasible option; iv) intermittent network connectivity: to support device mobility, in Device-to-Cloud and Device-to-Device the reliable network connections should be guaranteed, but the multi-hop nature of the Internet backbone and short-range capability of the network technologies currently envisioned for the support of inter-device communication, guaranteeing reliable network connection is a challenging task in the envisioned IoE realm.

Opening all of this IoE challenges the doors to the FC paradigm in the next subsection.

### 1.1.2 Fog Computing - Attributes

The Cisco [36] is promoting FC as a new paradigm for IoE data analysis by considering the key characteristics of Cloud and Fog Computing, and by encouraging the study of this new FC paradigm with the exponential

Table 1.1 Cloud-Fog computing platform, emphasizing the main characteristics of both architectures.

| Platform | Response Time and Target user | Data Storage | Geographic location | Service and Applications | Process and Benefits | Connectivity and Requirements | Data Centers |
|---|---|---|---|---|---|---|---|
| **Cloud** | – Minutes, days, weeks [36]<br>– General Internet users | – Long durations (months, years) [36, 68] | – Global and centralized [36, 68] | – Big Data analytics [36]<br>– Graphical dashboards [36]<br>– Storage metadata<br>– Long-term viability | – CC concentrates on sharing of information in the grid of nodes [56, 68]<br>– CC receives and aggregates data summaries from many FNs [36]<br>– CC performs analysis on the IoT data from multiple sources to gain business insight | – Through IP networks [66]<br>– High latency [68, 37]<br>– High delay jitter<br>– Multi-hops<br>– No location awareness<br>– Real-time interactions | Cloud is built by a large-scale virtualized DC, and computing resources are available to Internet users as VMs [65, 68, 36] |
| **Fog** | – Milliseconds, sub-seconds [36]<br>– Mobile users | – Short duration (hours, days, weeks) [36] | – Widely distributed<br>– A FN is usually is placed at the edge of the network between end devices and the remote Cloud [119, 36] | – Fog supplies computing, storage and networking services<br>– Virtualization [77, 36]<br>– Simple analytics [36] | – Greater business agility [36]<br>– Better security [36]<br>– Deeper insights, with privacy control [36]<br>– Lower operating expense [36]<br>– FP mitigrates the transport latency<br>– FC permits real-time decisions<br>– FC receives feeds from IoT devices, in real-time [36]<br>– FC runs IoT applications for real-time control and analytics [36]<br>– FC sends periodic data summaries to the Cloud<br>– FC provides mobility support | – Wireless interface [66]<br>– Low latency [68, 37]<br>– Low delay jitter<br>– One-hop<br>– Very low probability of attacks<br>– Location awareness [68, 37]<br>– Real-time interactions [68, 37] | FC is located the edge of the network, where millions to billions embedded systems and sensors are placed |

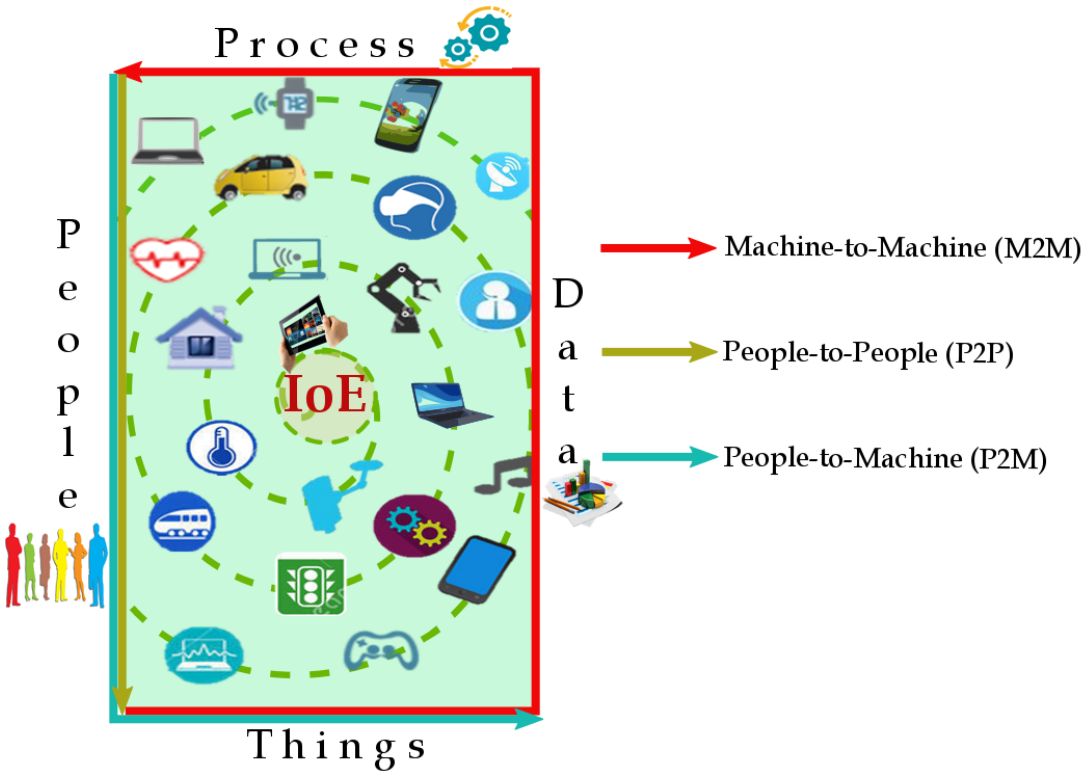



Fig. 1.1 Main services supported by the IoE paradigm

growth of IoT devices and the amount of data they produce, communication between "things" and Cloud will be costly, inefficient, and in some cases infeasible. FC serves as solution for this as it provides computation, storage, and networking resource for IoT. FC accelerates awareness and real-time responses to events by eliminating the return to the Cloud for the analysis and avoiding expending budget in bandwidth expansions. Moreover, FC protects sensitive data generated by analyzing them close to the end users.

Under the perspective of the computing-plus-bandwidth resources service latencies, the FC paradigm differs from the related Mobile Edge Computing (MEC) one, where only edge nodes are employed by device argumentation in those are one-hop away from the served IoE devices, the former paradigm integrates them with both the remote Cloud and the IoE devices as showed in Fig. 1.2, whereas the latter accounts only for the IoE devices.

We may see *Fog Computing* as a model that complements the Cloud through the distribution of the computing-plus-networking resources from remote data centers towards edge devices. Being the aim to save energy and bandwidth, while simultaneously increasing the QoS level provided to the users. The *Fog Nodes* (FNs) are virtualized networked data centers, that run atop (wireless) Access Points (APs) at the edge of

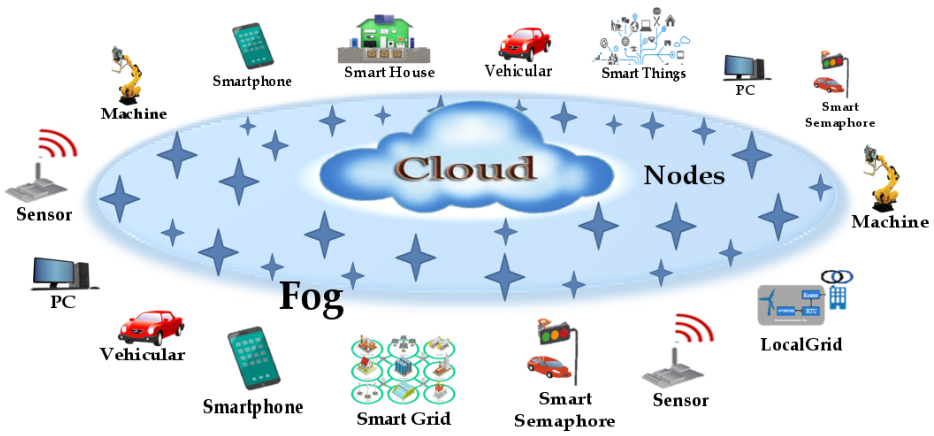



Fig. 1.2 Pictorial view of the tree-tier FC model

the access network, to give rise to a three-tier IoE-Fog-Cloud hierarchical architecture [129, 130]. The Fog paradigm is designed to provide some key features that are not retained by the Cloud [22]. They include:

- *Edge location and low latency* – FC supports endpoints by rich services, being deployed in proximity of the served IoE devices, FNs may efficiently leverage the awareness of the states of the communication links (e.g., WiFi-based single-hop TCP/IP transport-layer connections) for the support of delay and delay-jitter sensitive applications that require very low and predictable latency, like video conferences, video streaming [61, 63, 98];
- *Pervasive spatial deployment* – FNs support distributed applications, which demand for wide spatial deployments, like Wireless Sensor Network (WSN)-based applications [64];
- *Mobility support* – FNs positioned next to the users may exploit Fog-to-Thing (F2T) and Thing-to-Fog (T2F) single-hop WiFi links for data dissemination/aggregation [21, 66];
- *Heterogeneity of the served devices* – FNs must be capable to serve a large spectrum of heterogeneous devices. They require distributed computing and storage resources;
- *Low energy consumption through adaptive resource scaling* – FNs are densely distributed over the spatial domain and connected to the wireless access network, they are typically equipped with capacity-limited batteries, that can be re-charged through renewable energy sources (such as solar panels, and

micro-grids) [54, 69]. Fog paradigm is the reduction of both the computing and networking energy consumption through the adaptive horizontal (e.g., intra-Fog nodes) and vertical (e.g., inter-Fog nodes) scaling of the overall available resource pool;

- *Dense virtualization* – IoE devices are resource-limited and densely deployed over spatial domain. FNs must be capable to multiplex a large number of virtual clones with different resource demands onto a few number of physical servers [20];
- *Device isolation* – The clones must run atop Fog servers as isolated virtual machines or containers [49].

We need to incorporate Fog Data Services (FDS) on FC paradigm, to cope with the huge volume of data, to generate and transfer millions of data of things and to manage data processing over edge devices (gateways). FDS is an IoT software product that runs on the network, to transform raw data from sensors and endpoints into actionable information. While critical and time-sensitive data can go to the Cloud for long-term storage and historical analysis.

### 1.1.3 Fog Data Services

We need to incorporate Fog Data Service on FC paradigm, to cope with the huge volume of data, to generate and transfer millions of data of things and to manage data processing over edge devices (gateways).

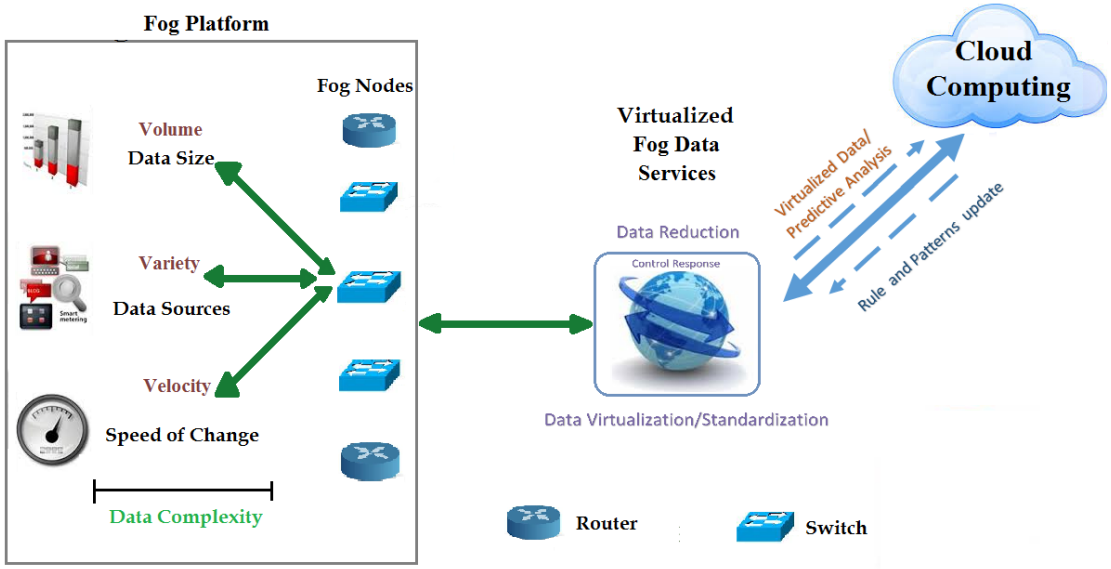



Fig. 1.3 FP-vs.-FDS-vs.-CC, FP:=Fog Platform, FDS:= Fog Data Service, CC:=Cloud Computing.

Fog Data Service (FDS) by Cisco [38] is an IoT SaaS-oriented software product that runs on the network, to transform raw data from sensors and endpoints into actionable information. While critical and time-sensitive data is stored and analyzed at the network edge, less time-sensitive data can go to the Cloud for long-term storage and historical analysis. FDS acts on data at the source and the edge of the network (see Fig. 1.3). So, their operations (i.e., data processing, data transferring, etc.) can use processed data immediately, and FDS may teach to protect against security risks of transmitting data to the Cloud, and can relieve the strain on the network, Cloud resources, and infrastructure. Moreover, FDS deploys to IOx Virtual Machines (VMs) running on devices in the Fog. FDS builds scalable IoT solutions with consistent APIs across a variety of rugged network devices and endpoints. FDS supports [38]:

- filtering of data, content-based;
- intelligent encryption of raw sensor data;
- remote reconfiguration of devices through REST-based APIs;
- local caching of sensor-acquired data;
- dynamic management of intra-Fog databases.

FDS also supports a database in motion zooming at extremely high speed and securely distributed and heterogeneous system. FDS supports Fog-based database in order to prepare atomicity, consistency, isolation and durability data in FC. The benefits of FC over FDS (see Fig. 1.4) are:

- datasets are processed and accessed reliably and more rapidly in the most logical location;
- data can be accessed more efficiently from a network perspective;
- it generates deeper insights and protects data with encryption;
- it simplifies service developments and deployments;
- it increases business agility;
- it improves security and protects sensor network from attackers.

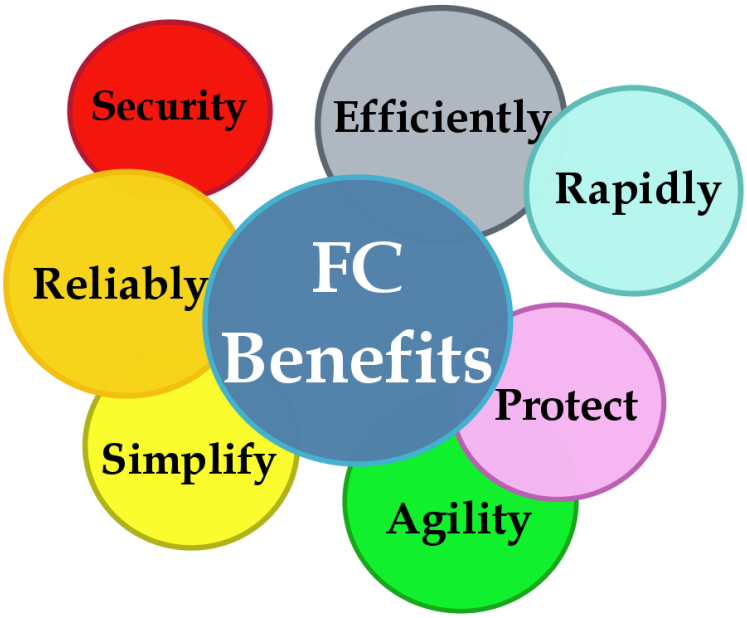



Fig. 1.4 FC Benefits over FDS.

## 1.2 Data Center Structure

Data Centers (DCs) are the pool of resources (core components are compute[1], storage, networking), that reside along with the necessary utilities such as: power, cooling, and ventilation equipment. One characteristic of the data center is the use of a dedicated, specialized network for storage traffic to bolster performance, mission-critical computing infrastructure, and operate around the clock. DCs have been touted as one of the key enabling technologies for the fast-growing IT industry and transform the economy at large (resulting in a global market size of 152 billion US dollars by 2016 [125]).

The optimization system to manage data center energy consumption uses the available knowledge about the energy demand characteristics of the applications, and characteristics of computing and cooling resources to carry out proactive optimization techniques as showed in Fig. 1.5.

Naturally, on-demand allocation of virtual resources and dynamic relocation of the assigned resources are two essential features that enable virtualization of a data center. The allocation and relocation of resources may be based on criteria such as performance requirements, load balancing, improved resilience, disaster recovery, and regulatory compliance. The mapping of physical to virtual resources is a matter of implementation. Overall, in addition to virtualization technology, it is also necessary to have a Fog and Cloud management system that manages all the resources across the underlying infrastructure and provides a uniform interface to applications.

[1]Compute (as a noun) is a new term coined in the industry to refer to computing resources as opposed to storage resources

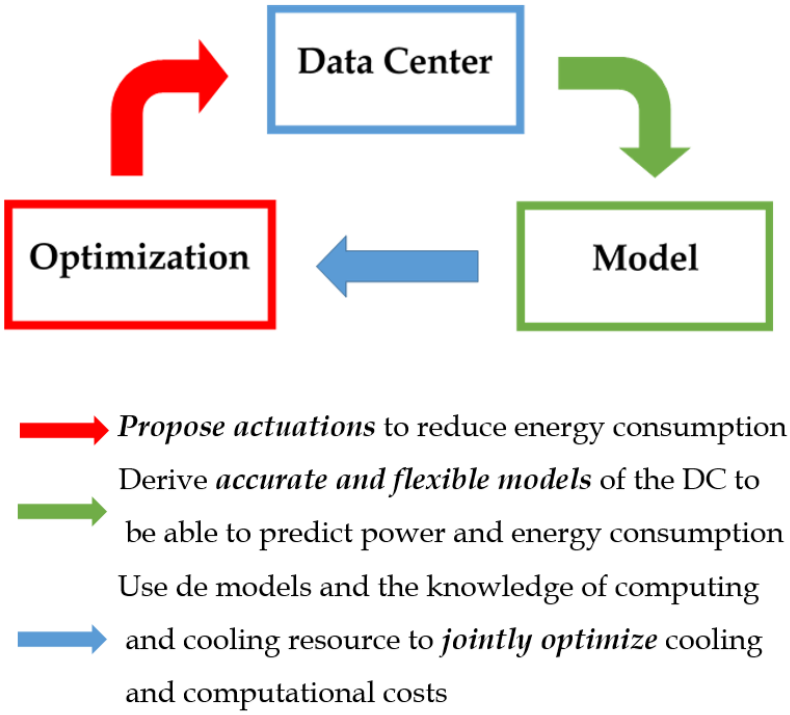



Fig. 1.5 A systematic view of the DC energy consumption modeling and solution [55].

With the proposal to reduce the energy consumption of a data center, we need to (see Fig. 1.5): i) to measure the energy consumption of its components [15, 26] also identifying where the energy is mostly employed; ii) opt input features to build the techniques used for energy consumption modeling and prediction for data centers and their components. One of the key problems which we face in this step is that certain important system parameters such as, the power consumption of a particular component in a data center which cannot be measured directly. The outcome of this step is a power model that is a formal abstraction of a real system. The models for computer systems can be represented as equations, rules, neural networks, decision trees, sets of representative examples, graphical models, etc. However, the choice of representation affects the accuracy of the models, as well as their interoperability by people [26]; iii) the model should be validated for fitness and intended purposes. Continually to be resolved using online and offline techniques. iv) the validated model and solution are used as the basis for predicting the component or system's energy consumption. Those predictions can also be used to improve the energy efficiency of the data center, such as: incorporating the model into techniques as dynamic voltage frequency scaling (DVFS), resource virtualization [82], improve the algorithms used by the applications and even completely shutting down unused servers [80], etc. Forward, we can use the solution for forecasting the trends in energy consumption, meaning in daily operations of computer systems,

users and data center operators that can understand the power usage patterns of computer systems, to maximize their energy consumption. The experimental verification using real test data is generally expensive and inflexible.

On the other hand, the Energy models are cheaper [5, 16, 52, 62, 96, 105, 122] and more adaptive to changes in operating parameters [72]. Nevertheless, many different power consumption optimization schemes have been developed on top of power consumption models and are represented as a mathematical function. In this context, the power modeling is an active area of research, study linear and nonlinear correlations between the system utilization and power consumption [120]. As a consequence, the DC energy consumption is very high since servers kept *ON* and idle do consume significant amounts of energy, even when they aren't performing useful work [44]. Hence, prediction techniques can be used to estimate future Cloud workloads to appropriately decide whether and when the physical Machines (PMs) should be put to sleep and when they should be awakened to accommodate new VM request. However, predicting Cloud workloads is a challenge due to the diversity and the sporadic arrivals of client requests, each coming at a different time and requesting different amounts of resources (such as: CPU, memory, bandwidth, etc.). The fact, that there are a lot of possibilities for the combinations of the requested amounts of resources associated with these requests, which means that requires classifying requests onto multiple categories, based on their resource demands. To each category, a separate predictor is defined to estimate the number of requests of that category, which allows estimating the number of PMs that are needed. Giving use to these predictions, and efficient power management decisions can be made, where an idle PM is switched to sleep only if it is predicted that it will not be needed for a period long enough, to compensate the overhead to be incurred due to switching it back *ON* later when it will be needed.

## 1.3 Data Center Issues and Objectives

The most important issues related to energy efficiency in DC are a major challenge because of their economic, environmental and performance impact. The numbers regarding DC power consumption are outstanding: i) 1.3% of worldwide energy production in 2010; ii) in USA alone, DCs consumed 80 mill *MWh/year* in 2011, which means the consumption of 1,5 $\times$ NYC; iii) 1 DC = 25000 houses; iv) More than 43 Million Tons of $CO2$ emissions per year (2% worldwide); v) more water consumption than many industries (paper, automotive, petrol, wood, or plastic) [55].

Considering the most of the DCs consume vast amounts of energy unnecessarily, wasting 90% or more of the electricity they draw from the grid [113] [2]. One of the noteworthy reason of such inefficiency is the under-utilization of servers[3]: typical utilization figures range from 6% to 12%. Virtualization of DCs offers a way to increase server utilization and reduce energy consumption. It also sets lose the traditional delineation of DCs by hardware, physical casing and wiring, floor space, and other physical attributes. Finally, running in the idle mode servers consume a significant amount of energy. Large savings can be made by turning off these servers. This and other measures such as workload consolidation need to be taken to reduce DC electricity usage. At the same time, these power saving techniques reduce system performance, pointing to a complex balance between energy savings and high performance. The energy consumed by a DC is categorized into two parts: energy use by IT equipment (such as: servers, networks, storage, etc.) and usage by infrastructure facilities (such as: cooling and power conditioning systems). The amount of energy consumed depends on the design of the DC as well as the efficiency of the equipment. However, modeling the exact energy consumption behavior of a DC, at the whole system or the individual component level, is not straightforward.

## 1.4 Contributions

The main contributions of this thesis are classified into 4 categories: IoT on Fog Architecture is the cornerstone for future IoE. It is a fundamental issue that provides a supporting platform for addressing other issues in IoE (such as: connectivity, compatibility and longevity, intelligent analysis and actions, etc.). Indeed, it can address the Internet of Energy and, in turn, it makes energy systems renewable, being a much efficient holistic for managing sources. In the field of IoE is also an important source of BD. In Smart Cities, IoE and BD may come from industry, agriculture, traffic transportation, medical care, public departments, families, etc., and IoE and BD are looking for various renewable energies that are available. The key contributions for the proposed FoE paradigm consist in: i) the main building blocks of the architecture in the proposed FoE technological platform; ii) the role played by the virtual containers, and; iii) the main fictions of the corresponding FoE protocol stack.

[2]Google's DCs are exceptions, being the company that has some of the largest, but also more ecological, infrastructure centers around the world. They use 50% less energy than typical DCs. This is achieved by, to name just a few steps raising the server floor temperature to 80°F, using outside air for cooling, and building custom servers that are 93% efficient. Also With the transition to Google Apps, companies have reduced the cost of their computer systems, energy consumption, and carbon dioxide emissions from 65% to 90%. In addition, companies using Gmail have reduced their environmental impact by up to 98% compared to those who manage emails on local servers. See www.google.com/about/datacenters/.

[3]The servers are running all the time (24/24 hours). See www.google.com/about/datacenters/efficiency

The second contribution is the Vehicular FoE (V-FoE), being the Fog paradigm still in its infancy, large-scale real-world Fog infrastructures are not currently available for testing purpose. Therefore, in order to corroborate the aforementioned expectations, we have emulated in software a small-scale FoE prototype, named, the Vehicular FoE (V-FoE) test-bed. With the desire to provide a proof-of-concept of the proposed FoE protocol stack by implementing the resource orchestration and management solutions around the following pillars: i) dynamic orchestration of streaming application of [85]; ii) energy-efficient bandwidth manager of [4] and Follow-Me-Cloud dynamic solution of [110, 111]; iii) cognitive solution of [39] and integrated resource management of [13]; iv) adaptive solution of [107] and dynamic reservation-based solution of [39]. For this propose, we utilize simulation toolkit called *iFogSim* that allow the simulation of FNs and IoT devices by tuning their computing, communication and storage capabilities as: i) number of computing cores and their CPU speed-vs.-computing power profiles; ii) the bandwidths of their NICs and the corresponding transmission rate-vs.communication power profiles, and; iii) the available RAM for task storage.

The third contribution is the TCP/IP virtualized data center, including: i) the considered virtualized TCP/IP computing platform; ii) the proposed joint dynamic scheduling; iii) performance optimality and QoS of the proposed scheduler; iv) implementation aspects and implementation complexity, and; v) experimental work.

The final contribution of the thesis is implementing the problem solutions using *iFogSim* [59], and *Matlab* optimization packages. Matlab is used as a modeling language, allowing constraints and objectives to be specified using standard *Matlab* expression syntax. Also, we use the *Cloudsim* toolkit [27]. *CloudSim* is developed in the Cloud Computing (CC) and Distributed Systems (CLOUDS) Laboratory, at the Computer Science and Software Engineering Department of the University of Melbourne). It has been selected as the simulation platform to model and simulate the solutions in a simulated based environment, mainly due to the fact that it natively supports a number of primitives for modeling networked SaaS data centers.

## 1.5 Thesis Structure

The remainder of the thesis is organized as follows:

- *Chapter 2−* highlights of the research issues related to Fog-based applications are aligned along the main research lines, dealing with resource management issues related to conflicting requirements of maximizing quality of services (QoS)(availability, reliability, etc.) delivered by the Cloud services the energy efficiency of Fog and Cloud-based applications while minimizing energy consumption of the data center resources.
- *Chapter 3−* proposes the FoE paradigm and a proof-of-concept case study the V-FoE testbed which are derived from [12, 40, 91, 92, 108] and [119]:
  - **Paola G. Vinueza Naranjo**, Mohammad Shojafar, Habib Mostafaei, Zahra Pooranian and Enzo Baccarelli, "P-sep: a prolong stable election routing algorithm for energy-limited heterogeneous fog-supported wireless sensor networks". The Journal of Supercomputing, pages 1–23.
  - **Paola G. Vinueza Naranjo**, Mohammad Shojafar, Ajith Abraham and Enzo Baccarelli, "New stable election-based routing algorithm to preserve aliveness and energy in fog-supported wireless sensor networks". In Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on, pages 002413–002418. IEEE
  - Nicola Cordeschi, Danilo Amendola, Mohammad Shojafar, **Paola G. Vinueza Naranjo**, and Enzo Baccarelli, "Memory and memoryless optimal time-window controllers for secondary users in vehicular networks", In Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems,pages 1–7. Society for Computer Simulation International.
  - **Paola G. Vinueza Naranjo**, Mohammad Shojafar, Leticia Vaca-Cardenas, Claudia Canali, Riccardo Lancellotti, and Enzo Baccarelli, "Big data over smartgrid − a fog computing perspective". 24rd International Conference on Software, Telecommunications and Computer Networks-Softcom.
  - Enzo Baccarelli, **Paola G. Vinueza Naranjo**, Michele Scarpiniti, Mohammad Shojafar, and Jemal H Abawajy, "Fog of Everything: Energy-efficient Networked Computing Architectures, Research Challenges, and a Case Study", IEEE Access.

  - Mohammad Shojafar, Zahra Pooranian, **Paola G Vinueza Naranjo**, and Enzo Baccarelli, "FLAPS: bandwidth and delay-efficient distributed data searching in Fog-supported P2P content delivery networks", The Journal of Super-computing, pages 1–22.

- *Chapter 4*– proposes and develops a framework for the joint characterization and optimization of TCP/IP SaaS Fog data centers that utilize a bank of queues for increasing the fraction of the admitted workload derived from [14]:

  - Michele Scarpiniti, Enzo Baccarelli, **Paola G. Vinueza Naranjo** and Aurelio Uncini "Energy Performance of Heuristics and Meta-heuristics for Real-time Joint Resource Scaling and Consolidation in Virtualized Networked Data Centers".

  - Enzo Baccarelli, **Paola G. Vinueza Naranjo**, Mohammad Shojafar, and Michele Scarpiniti, "Q*:energy and delay-efficient dynamic queue management in tcp/ip virtualized data centers". Computer Communications.

  - **Paola G. Vinueza Naranjo**, Enzo Baccarelli, and Michele Scarpiniti, "Design and energy-efficient resource management of virtualized networked Fog architectures for the real-time support of IoT applications".

  - Enzo Baccarelli, Michele Scarpiniti,**Paola G. Vinueza Naranjo**, Leticia Vaca-Cardenas, "Fog of Social IoT: when the Fog becomes social".

- *Chapter 5*– concludes the thesis with a summary of the main findings, discussion of future research directions, and final remarks.

In the next chapter, we provide a description of the state of the art strictly related to the present thesis.

# Chapter 2

# State-of-the-art

The technical contribution focuses on the design of Fog-based networked computing architecture for the energy-efficient joint management of the networking and computing resources under hard constraints on the overall tolerated computing-plus-communication delay. At this regard the paradigm FoE from of a technological point of view and take into account the communication problem, the most recent studies that had taken steps to make the point of the state of the art that has been made in several types of research. Very recently, few of research work has been done in the area of power and energy-efficient resource management in Fog/DCs and IoE. Our goal is to give an appropriate examination of the current literature supports the conclusion, the topics addressed in this thesis underline the motivation behind the latter and detail their limitations compared to the proposed methods.

## 2.1 Emerging applications areas and related QoS requirements

Therefore, we conduct an in-depth study of the existing work in principle, computing and networking-intensive applications that require real-time processing of spatially distributed environmental data may gain benefit from the integration of the pillar IoE and FoE paradigms. The characteristic are retained, indeed, by four broad applications areas of growing practical interest such as, Smart City, Big Data Streaming, Industry 4.0 and Internet of Energy. We believe that, in the future these application areas could provide "killer" use cases for the proposed FoE paradigm.

- *Smart City*: In 125 years of electric power, we have transformed our lives and cities to the point that we can build densely populated cities (cities with more than 10 million citizens living in skyscrapers). In order to keep cities power-efficient, we need to pay a close attention to two points of view: increasing the central Gross Domestic Product and obeying/posing a powerful legislation to make green environments. Planning and operation are never coordinated in cities frequently: hence, to fill this gap, Fog Computing (FC) platform is one reasonable response, FC platform, due to its intrinsic characteristics, can be close enough to the Fog Nodes (FNs), take data from hundreds or thousands of small things (IoE devices) in the whole city, in order to provide a feasible QoS. Therefore, by using the suggested platform, some smart city benefits can be listed as: a real-time traffic information, a multi-agency coordination, a generalized controlling and managing congestion alerts, a reduced latency, a less disturbance of vehicular transportation, an optimized multi-model travel and ticketing and a dynamic capacity management.

- *Big Data Streaming*: Mobile computing has been recently proposed as a paradigm that exploits the integration of the Big Data, Cloud-based stream processing, and broadband mobile Internet Networking paradigms. Its specific target is the design and implementation of novel self-organizing spatially distributed networked computing platforms, in order to enable the real-time offloading and pervasive processing of environmental big data streams gathered by energy and bandwidth-limited wireless things (e.g., sensors, tables, RFIDs, PDAs, smartphones). The Big Data Streaming paradigm relies on a "three Vs": Big data is not just about Volume (terabytes, petabytes), but also about Velocity (real-time or near-real-time) and Variety (social networks, blog posts, logs, sensors, etc.). The big data streaming is composed of five main blocks, namely, the *IoT layer, radio access network, Fog layer, Internet backbone* and *remote Cloud layer*. According to the architecture, big data streams are: i) gathered by a number of spatially heterogeneous mobile/wireless devices scattered over the environment of interest; ii) forwarded to proximate FNs over WiFi/Cellular connections for local pre-processing, and; iii) routed to Cloud-based remote data centers over Internet WANs for further post-processing. The big data streaming paradigm is capable to support three main classes of IoE-oriented applications, called: i) *spatial sensing*: provide Internet access to a number of heterogeneous sensors, in order to enable the real-time exchange of data about the monitored environment, requiring, in turn, that a huge number of simultaneous transport connections is sustained without inducing time-consuming traffic congestion phenomena; ii) *crowd-sourcing*: populations of non-professional users acquire environmental data streams such as typical, video/audio data streams through own phones and share in real-time by building up P2P transport connections, and; iii) *data caching & nomadic computing*: rely on context-aware services for data caching and personal computing on-the-go, allowing to the users to share information

in real-time by leveraging Fog assisted social network platforms such as Dropbox, YouTube, Facebook, etc. requiring massive sets of inter-stream cross-correlation analytic, in order to detect as faster possible the occurrence of new social trends or anomalies.

- *Industry 4.0*: has been quite recently introduced to indicate the fourth industrial revolution, is a rather vast vision and, increasingly, vast reality. In fact, the Industrie 4.0 as the digital transformation of manufacturing, leveraging third platform technologies, such as Big Data/Analytics and innovation accelerators, such as the (Industrial) Internet of Things, Fog/Cloud computing; and requiring the convergence of IT (Information Technology) and OT (Operational Technology), robotics, data and manufacturing processes to realize connected factories, smart decentralized manufacturing, self-optimizing systems and the digital supply chain in the information-driven, cyber-physical environment of the fourth industrial revolution.

  Therefore, the bridging of digital and physical, cyber-physical production systems and the Industrial IoT as parts that describe the fourth industrial revolution. Although the term Industry 4.0 and the reference architecture model behind it originated from Germany "Industrie 4.0", it's clear that the vision and reality of the fourth industrial revolution have caught the attention of organizations across the globe. Moreover, Industrie 4.0 is not just about manufacturing anymore. The global diffusion of the Industrie 4.0 vision and technologies, at different speeds, is related to the universal challenges and possibilities across the globe and with the cross-fertilization, enabled by collaborations with the US, Japanese, EU industries initiatives and so forth. Still, there are several hurdles to take before the Industry 4.0 vision is realized in more companies than is the case today.

  The Industry 4.0 typically are automation, (manufacturing) process improvement and productivity/production optimization. Industry 4.0 is also called *smart industry* or *smart manufacturing*. In many senses, it is related to the Industrial Internet and since 2016 the Industrial Internet Consortium and Industry 4.0 platform, "Plattform Industrie 4.0", indeed started collaborating.

  By design, a *smart* industry requires a vertical integration of a four main subsystems, namely, i) *IoE-based Physical Resource layer*: is IoE-based and comprises smart things, like smart products, smart machines, and smart conveyors, that self-establish Thing-to-Thing (T2T) communication links by exploiting the corresponding Network layer, being capable to self-collaborate and self-organize to attain a system-wide; ii) *Network layer*: provides the communication services that are required by the underlying physical smart things, to implement the needed inter-thing negotiation mechanisms and communicate with the Fog layer, the topology of smart industry is expected to be highly time-varying

due to the presence of mobile entities such as: robots and automated guided vehicles. The Network layer will rely on short/medium-range wireless networking technologies such as: like WiFi, UWB, and Bluetooth; iii) *Proximate Fog layer*: this is capable to provide the scalable processing environment required by big data applications in the case of computing, storage and networking resources. The smart industry things at the Physical layer may produce massive streams of data, that required being transferred to the Fog layer for further filtering and analytic, and; iv) *Remote Control layer*: allows remote people to access to the smart industry through Web-based portals and Internet gateways. In principle, it may also comprise a remote Cloud layer, in order to perform offline complex analytic on massive semi-permanent datasets. By doing so, people can access to the statistics provided by the Cloud and perform maintenance/diagnostic operations, even remotely through the Internet.

- *Internet of Energy*: is based on the establishment of a large number of applications of distributed renewable energy generation, and it can achieve the plug and play technology in the grid that can accommodate a variety of different power generation, such as renewable energy, including energy storage devices. FNs are able to establish a well-defined real-time communication between user terminals and power generations, energy storage devices, monitoring and managing load equipment. The Internet of Energy will change the classical methods of response to raising loads, to the energy consumption, in order to improve the power system reliability and flexibility. The Internet of Energy will be formed as a unified global network and it could be an umbrella for the temporary Internet.

Table 2.1 Expected networking QoS requirements for the application fields.

| | **Smart City** | **Big Data Streaming** | **Industry 4.0** | **Internet of Energy** |
|---|---|---|---|---|
| **Latency** | $\leq 10$ $(ms)$ | $\leq 100$ $(ms)$ | $\leq 5$ $(ms)$ | $\leq 200$ $(ms)$ |
| **Latency jitter** | $\leq 3$ $(ms)$ | $\leq 10$ $(ms)$ | $\leq 0.5$ $(ms)$ | $\leq 15$ $(ms)$ |
| **Packet loss rate** | $\leq 10^{-3}$ | $\leq 10^{-2}$ | $\leq 10^{-4}$ | $\leq 10^{-2}$ |
| **Bandwidth** | $\geq 2$ $(Mb/s)$ | $\geq 10$ $(Mb/s)$ | $\geq 200$ $(Kb/s)$ | $\geq 50$ $(Kb/s)$ |

To recap, since it is expected that the energy-saving support of the applications described and accurate characterization of the corresponding per-service resource usages, presenting in the Table 2.1 a synoptic indication of the networking QoS requirements that are expected to stem from the considered application fields.

## 2.2 Data Center Energy Consumption: A System Perspective

Hence, we conduct the study of the existing works in data center power models and present the models using a coherent layer-wise abstraction as shown in Figure 2.1. In general, we can categorize the constituents of a data center as belonging to one of two layers, software, and hardware. The software layer can be further divided into two subcategories, the OS/virtualization layer, and the application layer. In this chapter, we describe the power consumption modeling work in the software layer.

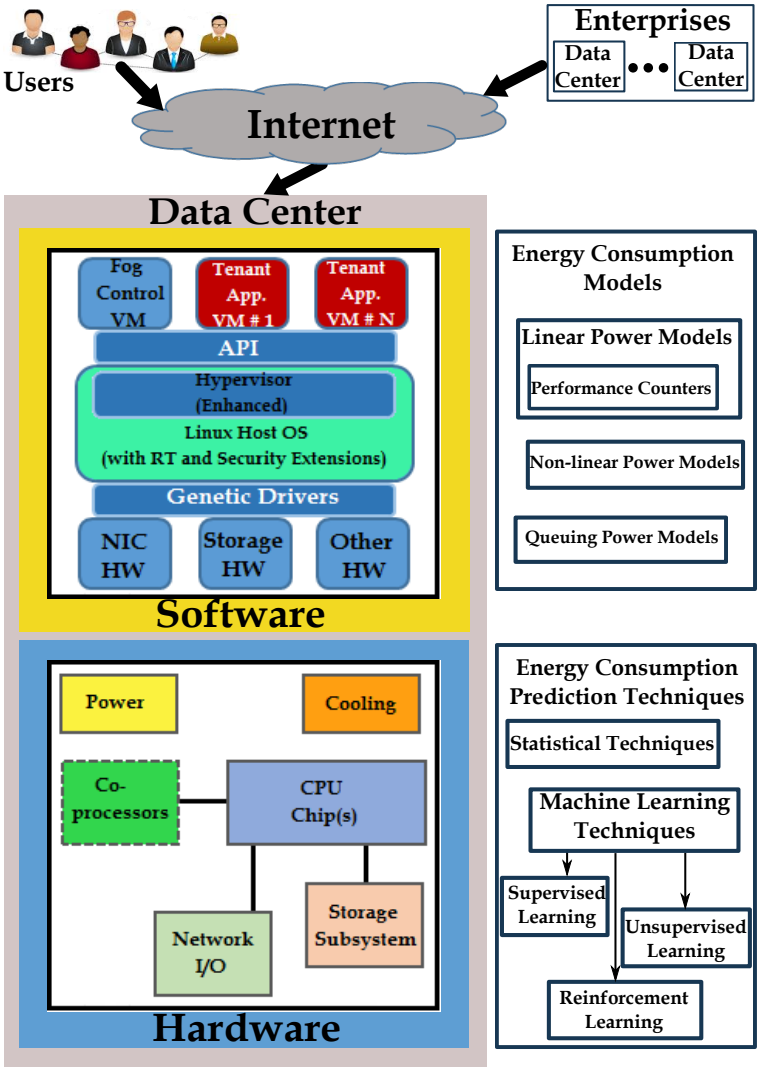



Fig. 2.1 A holistic view of the context for energy consumption modeling and prediction in data centers

Throughout this process, we embossing various energy consumption models and prediction techniques during this process which are applied at various different levels of the data center of systems. Power models play a fundamental role in the energy-efficiency research of which the goal is to improve the components' and systems' design or to efficiently use the existing hardware. Some energy-provisioning models use non-linear power models and some use queuing power models [50, 106, 114]. For brevity, authors in certain works the server's CPU usage and operation frequency are used for modeling a server's power consumption. Whilst the approach is promising, but power consumption of each core must be known beforehand which could be resolved in the proposed methods in this thesis.

## 2.3 Related Works

The past years have been characterized by two seemingly contrasting technological trends.

The first one regarded the surging of the Cloud model as ubiquitous computing paradigm and the resulting shift of computing, control and data storage capabilities towards remote and large-size data centers [125]. Since these data centers are far away from the network edge, end-users connect them through the Internet backbone.

The second one trend concerned the surfing of a number of heterogeneous user-oriented access and sensor devices, like tablets, smartphones, smart home appliances, access points, edge routers, roadside-placed cabinets for the smart control of the vehicular traffic, connected vehicles, smart meters for power grids, smart control systems for Industry 4.0 factories, just to name a few. Additional smart edge devices, like industrial and home robots, computers on a stick, RFID-based frequency tuners, are currently gaining momentum. The common feature of all these devices is that they are things that operate at the network edge. This is, indeed, the realm of the so-called IoE paradigm [24, 25, 45, 116], which context-aware things autonomously setup and manage self-organizing networks. Interestingly, these networks are no longer human networks empowered by the presence of things. In [24] explain how the IoE is self-orchestrating eco-systems, that aim at providing services to humans by empowering the performance of the underlying IoT physical infrastructures. This is attained by improving the functions of thing discovery and service composition, while suitably self-managing the limited computing-plus-communication resources of the involved things. Passing to consider the contributions [45, 116] that focus on the IoE networks, that autonomously attain right energy consumption-vs.-attained performance tradeoffs without any human supervision.

The authors [23, 103, 104] give IoE vision to implement the technological paradigm of FC that is expected to provide the needed networking-plus-computing support by allowing the on-the-fly instantiation of software clones (e.g., virtual surrogates) of the physical things atop nearby resource-equipped cloudlets.

The authors [7, 20, 101, 109], roughly speaking, in virtualized data centers, each served physical is mapped into a virtual clone that acts as a virtual processor and executes the programs on behalf of the cloned device. In principle, two main virtualization technologies cloud be used to attain device virtualization, namely, the (more traditional) Virtual Machine (VM)-based technology [7, 101] and the (emerging) Container (CNT)-based technology [20, 109]. In a nutshell, their main architectural differences are that [109, 128], the VM technology relies on a Middleware software layer (e.g., the so-called Hypervisor) that *statically* performs hardware virtualization, while the CNT-technology uses an Execution Engine, in order to *dynamically* carry out resource scaling and multiplexing, and, a VM is equipped with an own (typically, heavy-weight) Guest Operating System (GOS), while a container comprises only application-related (typically, light-weight) libraries and shares with the other containers the Host Operating System (HOS) of the physical server. Noteworthy advantages of CNT-based virtualization technology are that, the containers are light-weight and can be deployed significantly quicker than VMs, and, the physical resources required by a container can be scaled up/down in real-time by the corresponding Execution Engine, while, in general, physical resources are statically assigned to a VM during its bootstrapping. However, since all containers running on the same physical server share the same HOS, the main disadvantages of the CNT-based virtualization technology are that, the level of inter-application isolation (e.g., the level of trustworthiness) guaranteed by the container-based virtualization is typically below than the corresponding one offered by the VM-based technology, and, all the application libraries stored by the instantiated containers must be compliant with the HOS equipping the host physical server. However, due to the expected large number of devices to be virtualized in IoE application environments, resorting to the CNT-based virtualization would allow increasing the number of virtual clones per physical server (e.g., the so-called virtualization density) [128].

The main challenge in DCs is the minimization of the energy usage, while still meeting the QoS requirements of the supported applications. For this purpose, adaptive and scalable energy-aware scheduling algorithms are required that jointly perform the allocation of the networking and computing resources on the Cloud and over the TCP/IP mobile connections which link the DCs to the mobile clients. There have been numerous works in the area of the Internet DCs which aim at providing various models and techniques to seamlessly integrate the management of computing-communication virtualized platforms, in order to provide QoS [43], robustness and reduced energy consumption. Among the contributions that mainly deal with the adaptive scaling of the server resources, the work in [124] focuses on the impact of the dynamic scaling of the CPU computing frequencies on the energy consumption experienced by the MapReduce-type jobs. The goal in [73] is the attainment of an optimized delay-vs.-energy trade-off under bag-of-task type applications that are executed on Dynamic Frequency Voltage Scaling (DVFS)-enabled data centers [89]. DVFS technique uses

in DCs for energy provisioning by dynamically adjusted processors. DVFS is applied in most of the modern computing units, such as cluster computing and supercomputing, to reduce power consumption and achieve high reliability and availability. The resource virtualization refers to instantiate several VMs on a same physical server, in order to reduce the numbers of hardware, while improving the utilization of resources. Therefore, we should take into account both virtualization and DVFS techniques, which are energy-aware tools in cloud data-centers to manage resource provisioning and energies.

Furthermore, the important requirement by a Cloud computing environment is provisioning of reliable Service Level Agreements (SLAs). It can be managed *global* or *local* by the cloud providers, adding some policies for each VM in each server or in the data centers which include several servers. At the *local level*, the system leverages the power of each VM based on the SLA management policies locally, while, at *global level*, this policy can be handled by general SLA policies among servers or data centers.

Another important issue of growing concern in cloud environments is the evenly distribute a huge amount of workloads over various servers, which is called *load balancing*. These algorithms seek to distribute workloads across a number of servers so that the average executions are minimized [123]. Load balancing schemes can be classified as static or dynamic. In static schemes, the current states of the servers are not considered when dispatching the workloads; examples of such schemes include Random Selection of servers and Round Robin policies. Dynamic schemes involve direct notification or indirect inference of server states by the load balancer.

The authors [88] pursued an approach that formulates the afforded minimum-cost resource allocated problem as a sequential optimization problem and, then, solve it by using limited look-ahead control. Hence, the effectiveness of this approach relies on the ability to accurately predict the future workload and the performance degrades when the workload exhibits almost unpredictable time fluctuations. In order to avoid the prediction of future workload, resorts to a Lyapunov-based technique, that dynamically optimizes the provisioning of the computing resources by exploiting the available queue information [114]. Although the pursued approach is of interest, it relies on an inherent delay-vs.-utility the tradeoff, that does not allow us to account for hard deadline constraints.

Passing to consider the contributions that focus on the consolidation of underutilized server resources as the main means for attaining energy reduction, the authors of [19] rely on Virtual Machine (VM) migration as a primitive function, in order to implement server consolidation. Furthermore, they present two heuristic algorithms that exploit server heterogeneity for the reduction of the energy consumption of deadline-constrained batch workloads. The contribution of the topic in [29] is the optimization of VM placement in heterogeneous data centers. For this purpose, the approach [29] deals with the trade-off between two contrasting goals such as:

i) the efficient spatial placement of VMs on the servers to minimize the number of turned ON servers, and; ii) the balanced time placement of VMs that exhibit similar resource demands, to avoid server overload.

Roughly speaking, the common approach pursued by [30] is the developed a resource management approach to attain energy saving in heterogeneous data centers subject to burst input workload with short-time forecasting of the future task arrivals is periodically performed and resource provisioning carries out on the basis of the forecast peak workload. The authors [35] with the purpose to attain a similar goal, they resort to a threshold-based approach, whose target is the reduction of the energy consumed by underutilized servers, which are put into a sleep mode and, stay sleeping til the number of queued pending tasks exceeds a given possibly threshold.

Passing to consider the research area on the mobile communication, a first research line focuses on the cross-layer analysis and optimization of TCP/IP traffic control mechanisms for single-antenna and multi-antenna mobile connections [83]. These contributions support the conclusion that an optimal control of the energy employed by the wireless transmission is an effective means to improve the resulting TCP good-put. However, this conclusion neglect the computing aspects. Mobile Communication mainly deals with the design of systems equipped with several communication and (possibly) computing resources, whose combined utilization aims at providing seamless ubiquitous services to mobile (possibly, vehicular) clients [60].

Another research direction is focused on the resource management of Cloud/Fog-based distributed computing architectures for the energy-efficient supporting real-time big data streaming applications run by resources-limited wireless devices [75, 94, 102, 131]. Specially, the S4 and DS-treams management frameworks in [94, 131] perform dynamic resource scaling of the virtualized resources hosted by Fog data centers by explicitly accounting for the delay-sensitive nature of the streaming workload offloaded by proximate wireless devices, while the Time Stream and PLAstiCC resource orchestrators in [75, 102] also perform dynamic server consolidation and inter-server live VM migration.

Consolidation of physical servers in virtualized data centers is the main focus of [18, 48, 126]. Roughly speaking, after recognizing that the optimal VM-to-physical server mapping is an *NP*-hard problem, these works propose various reduced-complexity greedy-type heuristics, that aim at minimizing the energy costs of the performed mappings. However, unlike our contribution, all these works do not consider time-fluctuations of the input workload and, then: i) do not perform dynamic VM placements, and; ii) do not carry out queue management.

Dynamic management of the computing resources hosted by queue-equipped SaaS data centers is, indeed, the topic of [81, 114], that solve approach pursued by these contributions relies on the exploitation of the Lyapunov's criterion and aims at attaining good energy-vs.-delay trade-offs. Therefore, both these works do not

consider the networking aspects, do not perform network flow control and focus only on the management of the computing resources, also not consider server consolidation and performs server consolidation by directly resorting to an offline approach, that applies exhaustive search and, then, presents a (worst-case) computational complexity that exponentially scales up with the number of VMs. Furthermore, [81, 114] do not provide closed-form expressions for the dynamic scaling of the processing frequencies of the running VMs. The work in [121] deals, indeed with the network aspects of the IaaS data centers. Specifically, by leveraging the topology properties of the fat-tree networks, the authors of [121] developed a traffic engineering-based heuristic for the VM placement. It aims at reducing the inter-pod network traffic by exploiting the priori knowledge of the traffic flows generated by the running applications and focuses on the reduction of the network energy in virtualized data centers.

Another research direction is focused on the optimization of the congestion control algorithm of the TCPNewReno protocol, in order to cope with the so-called "in-cast" traffic congestion induced by MapReduce-type applications [3, 46, 115]. For this purpose, [3] proposes to modify the multiplicative decrement policy of the standard TCP protocol, in order to account for the messages of network congestion generated by the switches. Being further refined in [115], that proposes a so-called gramma-correction factor, to adjust the TCP congestion window by simultaneously accounting for the experienced congestion level and deadline of the supported connection. The latest improvement proposed in [46] includes also a mechanism for the dynamic tuning of the times for the generation of suitably delayed ACK messages. Overall, TCP-oriented contributions focus on the flow control and consider the TCP/IP protocol as an effective means for managing the end-to-end transport connections in the virtualized data center. However, the deals with the energy-efficient distributed resource management in federated clouds. At this regard, the recent contributions in [70, 71] develop a promising approach that aims at maximizing the management profit through a suitable energy-saving distribution of the workload over the available set of geographically dispersed data centers. Passing to consider the research area on the mobile communication, a first research line focuses on the cross-layer analysis and optimization of TCP/IP traffic control mechanisms for single-antenna and multi-antenna mobile connections [51, 83]. These contributions support the conclusion that an optimal control of the energy employed by the wireless transmission is an effective means to improve the resulting TCP goodput. However, this conclusion is partially offset by the fact that neglect the computing aspects [51, 83]. Analogous conclusion holds for the works in [90, 107], in which optimized schedulers are derived by exploiting nonlinear optimization and queuing theory. Specifically, the scheduler developed in [90] does not present adaptive capability, while, the scheduler in [107] does not account for the limitation on the energy budget of the underlying mobile TCP/IP connection.

Without a doubt, server's underutilization in Cloud-based large-scale remote data centers is a common phenomenon, mainly due to an over-provisioning of network and computing resources for handling workload peaks. As a consequence, electricity costs cover a large fraction of the overall operating costs of state-of-the-art Cloud-based data centers [125]. From this point of view, the Fog and IoE paradigms promise to reduce energy consumptions and related operating costs by leveraging their native self-organizing and self-scaling capabilities, as well as their pervasive spatial deployment [23].

In the next chapters, firstly, we propose the FoE paradigm, where the IoE model could be adequately addressed by the native attributes of the FC model, the complementary features of these two pillar paradigms, and points out how the Fog could provide support to the IoE. This is the focus of the first part of the next chapter, whose main contributions may be so summarized. First, the role played by the virtual containers. Second, the main functions of the corresponding FoE protocol stack. Third, a proof of concept case study, the V-FoE testbed. Secondly, we propose and develop a framework for the joint characterization and optimization of TCP/IP SaaS Fog data center that utilize a bank of queues for increasing the fraction of the admitted workload. Specifically, the aim is two-fold:

- Maximize the average workload admitted by the data center;
- Minimize the resulting networking-plus-computing average energy consumption.

Lastly, in this thesis, we develop and test a new scheduler for minimizing the energy consumption induced by computing, communication and reconfiguration costs in Internet-based virtualized DCs which utilize end-to-end TCP/IP mobile energy-constrained connections under hard limits on the per-job total processing time. The salient features of the resulting scheduler are that it admits distributed and scalable implementation, it provides deterministic bounds on the instantaneous queue backlogs, it avoids queue overflow phenomena, and, it effectively track the time-fluctuations of the input workload, in order to perform joint resource consolidation without requiring any *a priori* information and/or forecast of the input workload.

# Chapter 3

# The FoE paradigm

In this chapter, we present the FoE paradigm and details regarding its technological platform and supporting protocol stack. By design, the FoE paradigm aims at implementing the Fog-IoE integration fostered by: i) *Deployment in the proximity of the devices*: Operating at the network's edge, FNs may provide infrastructure-based support to IoE applications that demand spatially distributed device deployment; ii) *Context-awareness*: Being located near to the served devices, FNs may acquire context awareness in real-time, and exploit it for the support of latency and latency-jitter sensitive services, like interactive and/or monitoring services; iii) *Support to the device mobility*: FNs may be arranged into spatial clusters, to serve mobile devices through single-hop links; iv) *Reduced energy consumption*: Inter-device communication may occur through (stable and energy-efficient) Device-Fog-Device up/down links in place of intermittent and energy-hungry) D2D links; v) *Container-based dense virtualization*: Virtual clones of the served devices may be dynamically packed into Fog servers as light-weight containers. Where the goal is to provide device argumentation on an on-demand basis, to dynamically support the resource-limited IoE devices.

We observe that all the (application-specific) technological platforms of Figs. 3.1, 3.2 and 3.3 retain the three following common features:

1. They rely on three-tier Device-Proximate Fog-Remote Cloud architectures;
2. They exploit single-hop WLANs and multi-hop WANs, to implement Device-Fog and Fog-Cloud connectivity, respectively; and
3. When there are multiple proximate Fog nodes, these platforms are typically equipped with (possibly, wireless) backbones, to provide inter-Fog communication.

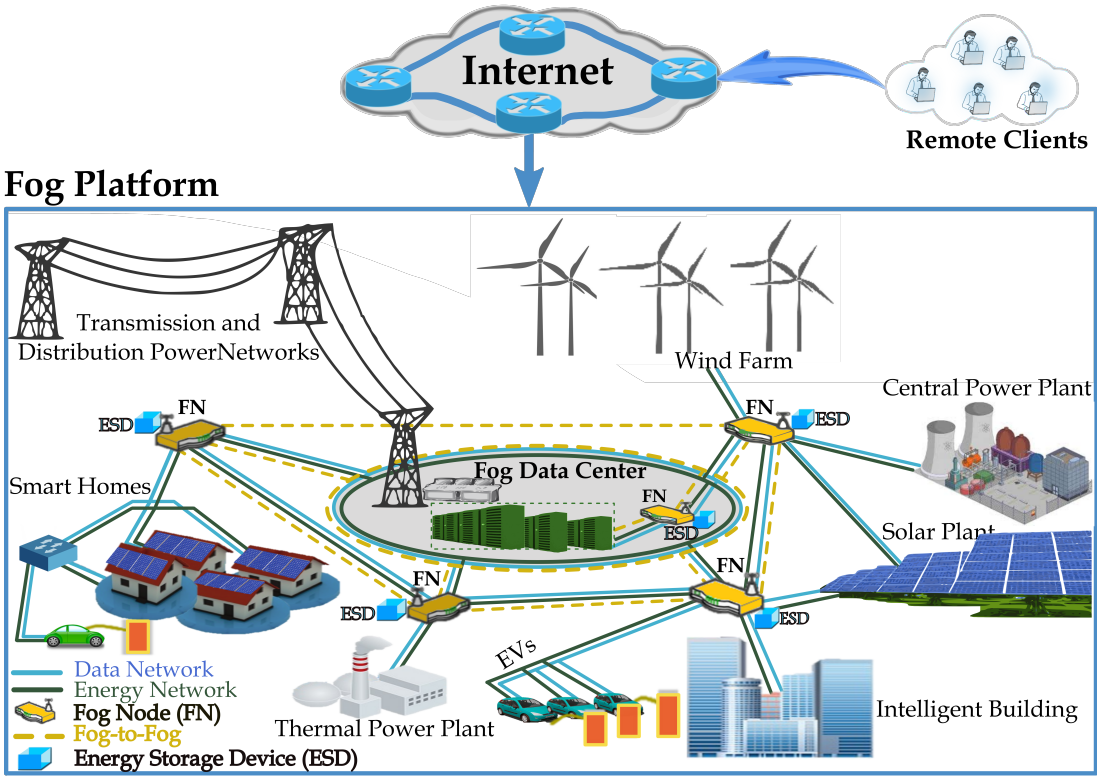



Fig. 3.1 IoE/SmartGrid Infrastructure on Fog Computing EV:=Electric Vehicles.

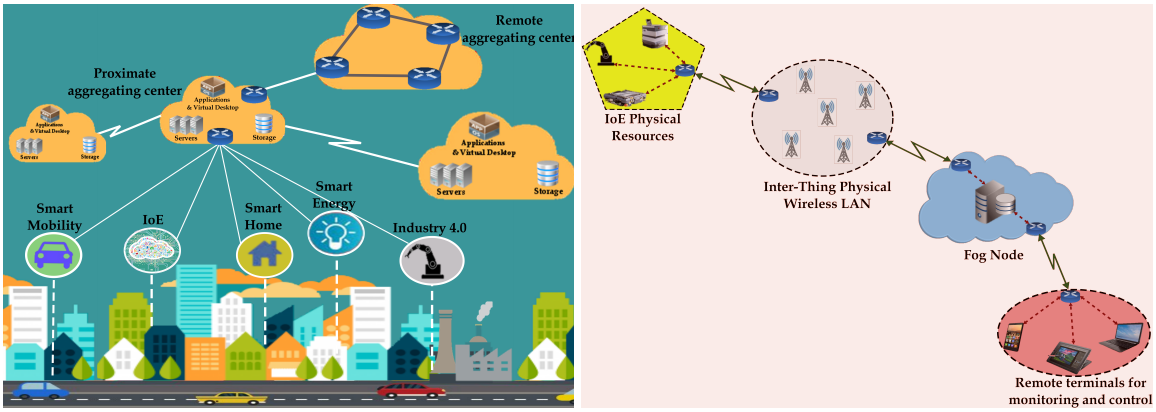
Fig. 3.4 shows the basic architecture of the virtualized technological platform for the support of the proposed FoE paradigm. Roughly speaking, The FoE architecture is composed by the integration of the six main building blocks, which are:

1. ***IoE layer***, a number of heterogeneous things operate over multiple spatial clusters, when *a thing* is a resource-limited user device that needs additional resources, to execute its workload and may be fixed, nomadic or even mobile, as we show in Fig. 3.4;
2. ***Wireless access network*** that supports Fog-to-Thing (F2T) and Thing-to-Fog (T2F) communication through TCP/IP connections running atop *IEEE802.11/15* single-hop links;
3. ***Inter-connected FNs***, a set that acts as virtualized cluster headers;
4. ***Inter-Fog backbone*** that (can be possibly wireless) provides inter-Fog connectivity and makes inter-Fog resource pooling feasible;
5. ***Virtualization layer*** that allows each thing to augment its limited resources by exploiting the computing capability of a corresponding virtual clone. The latter runs atop a physical server of the FN that at that moment serves the cloned thing, and;

6. ***Overlay inter-clone virtual network*** that allows P2P inter-clone communication by relying on TCP/IP end-to-end transport connection.

In Fig. 3.4 we observe that the remote Cloud is interconnected (multi-hop) by an Internet WAN to a set of Virtualized FNs that are distributed over a wireless access network, where each FN is equipped with a limited number of virtualized physical servers, which are inter-connected by an intra-Fog wired network such Ethernet type. Therefore, a FN covers a spatial area of diameter *Da (m)* that serves a cluster of things. Being the things a resource-limited, each one is augmented by a software clone, which runs in the serving FN and acts as a virtual server.

The main function of the inter-Fog backbone showed in Fig. 3.4 wireless and multi-antenna [8] is two-fold. First, it makes feasible the aforementioned horizontal dynamic scaling and pooling of the computing-plus-communication resources of the FNs. Second, it allows each clone to migrate from a FN to another by tracking the spatial trajectory of the corresponding mobile thing.




(a) (b)

Fig. 3.2 Emerging application areas (a) Architecture for Smart city application scenario; (b) Building block of Smart factory.

The overall set of clones running atop all Fog servers constitutes an overall P2P virtual network, which is composed of Clone-to-Clone (C2C) TCP/IP connections. Where intra-Fog wired Ethernet links and backbone-supported inter-Fog wireless links. Specially, the former (resp., the latter) are used to sustain the end-to-end transport connections among clones that run atop the same FN (resp., atop different FNs).

The service models supported by the FoE platform, with two main observations that are: First, since the FNs showed in Fig. 3.4 play the two-fold role of offloading and aggregating points of the traffic generated by the underlying things, being the FoE platform capable to support, by design, **all** the *Up/Down* offloading,

aggregation and P2P service models. Second, the *main peculiar feature* of the proposed FoE paradigm is that the overlay network allows to move the implementation of inter-things links from the device-based physical bottom layer to clone-based virtual upper layer, shown in Fig. 3.4. It makes, in turn, feasible to replace unreliable, intermittent and mobility-affected D2D-based inter-thing physical links with reliable, static and TCP/IP-based inter-clone virtual transport connections. The numerical test and performance comparisons are presented in Subsection 3.3.3 which bring evidence to support the actual effectiveness of this feature of the FoE platform.

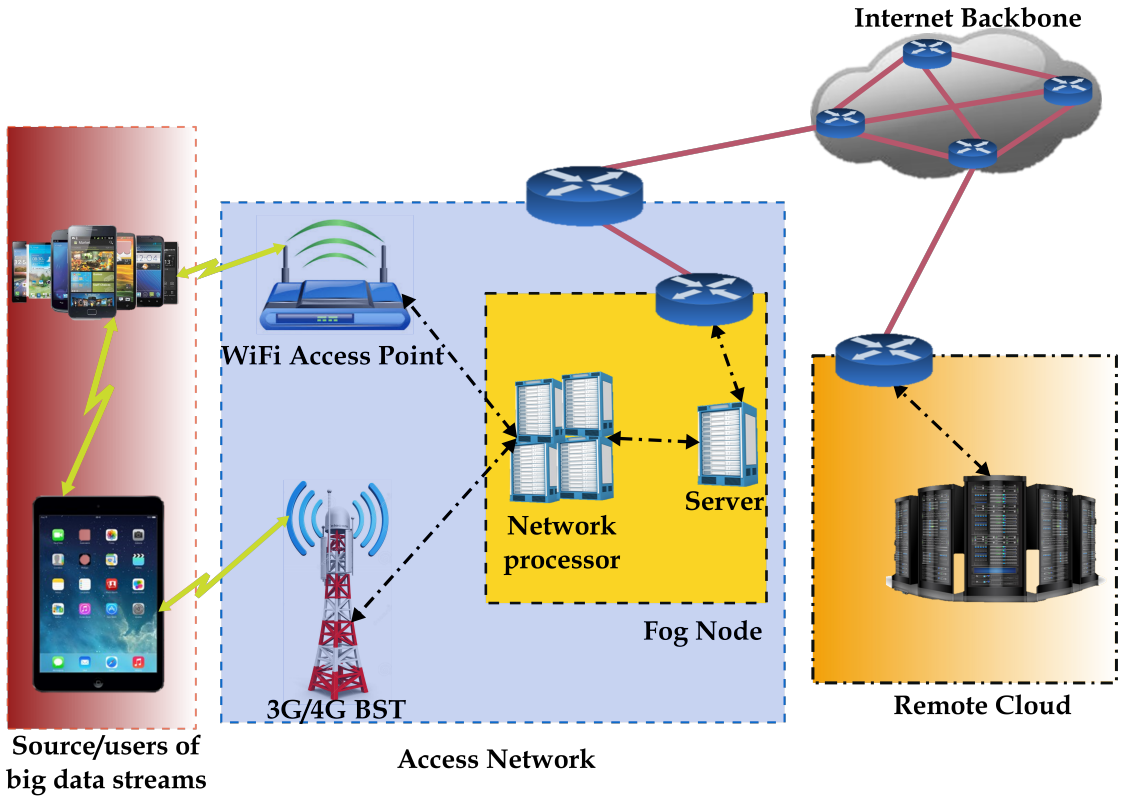



Fig. 3.3 IoE/SmartGrid Infrastructure on Fog Computing EV:=Electric Vehicles.

## 3.1 Container-based Virtualization of the IoE devices

Light-weight and fine-grain dynamic resource scaling is the key feature that makes appealing to resort to the container-based technology, to perform the virtualization of the FoE technological platform showed in Fig. 3.4.

Motivated by the considerations shown in Fig. 3.5(a), where we report the main functional blocks [20, 109, 128] of the virtualized architecture of the physical servers at the FNs.

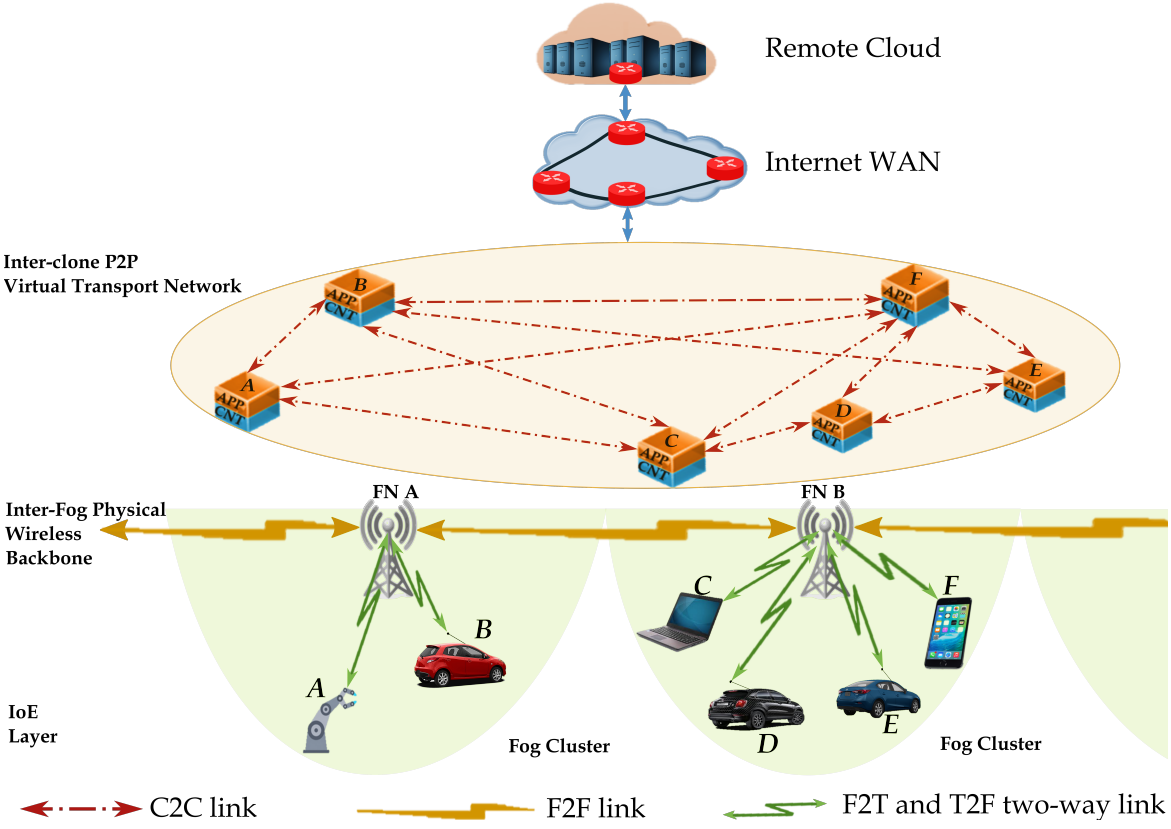
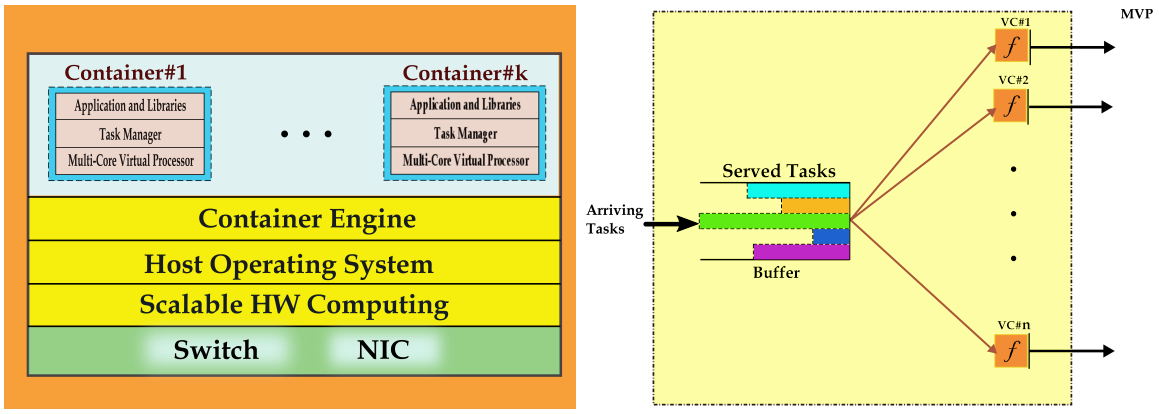



Fig. 3.4 Architecture for the FoE technological platform. Fog node A (resp., Fog node B) hosts FCL-A and FCL-B (resp., FCL-C, FCL-D, FCL-E and FCL-F) clones. FN:=Fog Node; FCL:=Fog Clone; P2P:=Peer-to-Peer; C2C:=Clone-to-Clone; F2F:=Fog-to-Fog; T2F:=Thing-to-Fog; F2T:=Fog-to-Thing; App:=Application code and libraries; CNT:=Container.

For the main explicative, we have four remarks classified as: First, each server hosts a number: $N_{CNT} \geq 1$ of containers. Where those share: i) the server's Host Operating System, and; ii) the pool of computing (e.g., CPU cycles) and networking (e.g., I/O bandwidth) physical resources done available by the CPU and Network Interface Card (NIC) equipping the host server. The *Container Engine's*, shown in Fig. 3.5(a), task is to allocate dynamically to the requiring containers, bandwidth, and computing resources where are available by the host server. For this purpose, the *Weighted Processor Sharing (WPS)* scheduling discipline is typically implemented by the Container Engine [20, 128].




(a) (b)

Fig. 3.5 Container-based virtualization of a physical server equipped a Fog node. (a) Virtualized server architecture; (b) Architecture of a multi-core virtual processor. HW:=CPU Hardware; NIC:=Network Interface Card; $k$:=Number of the Containers; MVP:=Multi-core Virtual Processor; VC:= Virtual Core; $n$:=Number of virtual cores; $f$:=Per-core processing frequency.

Second, each container plays the role of a virtual clone to associate physical things. The container acts as a virtual processor and executes the tasks offloaded by the thing on behalf of it. Being equipped each container with a *Multi-core Virtual Processor (MVP)* showed in Fig. 3.5(b), which comprises of: i) a buffer, that stores the currently processed application tasks, and; ii) a number $n \geq 1$ of homogeneous *Virtual Cores (VCs)*, that run at the processing frequency $f$ dictated by the Container Engine. Hence, the *Task Manager* of Fig. 3.5(a) is allocate the pending (which are in queue) application tasks over the set of virtual cores of Fig. 3.5(b) in a balanced and dynamic way. This is still done according to the aforementioned WPS scheduling discipline, so that the average frequency $f_i$ which is processed the $i-$th task equates to [128]:

$$f_i = \frac{\varphi_i}{(\sum_{j \in \mathcal{T}} \varphi_j)} \times n \times f. \tag{3.1}$$

In Eq. 3.1, by definition we have: i) $n$ and $f$ are the per-container number of virtual cores and the corresponding per-core processing frequency, respectively; ii) $\mathscr{T}$ which is the (time-varying) set of the tasks which are currently processed by the container, and; iii) $\varphi_i$ (resp., $\varphi_j$) that is a positive coefficient, that fixes the relative priority of the $i-$th (resp., $j-$th) processed task. Therefore, according to the Eq. 3.1, the Task Manager can increase the processing frequency $f_i$ of the $i-$th task by increasing the corresponding weight $\varphi_i$ and decreasing the number of the simultaneously served tasks showed in Fig. 3.5(b). Finally, we assume that per-core and per-task processing frequencies $f$ and $f_i$ in Eq. 3.1, where are measured in bit-per-second $(b/s)$, and the task sizes are measured in bit.

In fact, according [76] the corresponding numbers $s$ and $s_i$ of CPU cycles-per-second $(cycles/s)$ can be directly computed by:

$$s = \delta \times f, \quad \text{and} \quad s_i = \delta \times f_i. \tag{3.2}$$

In Eq. 3.2, $\delta$ (CPU $cycles/b$) is the processing density of the running application, which fixed the average of CPU cycles per processed bit, which one increases with the computing intensity of the considered application. We considered a list of the range values of some test applications of the processing density [76] such as: Face recognition $-$ 2339-31680 (CPU $cycles/b$), 400 frame video game $-$ 2640 (CPU $cycles/b$), Virus scanning $-$ 32946-36992 (CPU $cycles/b$), Video trans-coding $-$ 200-1200 (CPU $cycles/b$).

## 3.2 Management of Virtualized FoE Technological Platform: Protocol stack and Implemented QoS Services

In order to appropriately orchestrate the overall technological FoE platform showed in Fig. 3.4, we designed and implemented in software the FoE protocol stack of *Cloud Layer, Overlay Layer, Fog Layer, IoE Layer*, depending on the suitable integration of some QoS resource managers, that recently were proposed in the literature in the distributed self-management of multi-tier virtualized networked computing platform.

Specifically, in the Fig. 3.4 the FoE platform, the corresponding protocol stack comprises the following four hierarchically layers before mentioned which are explained following:

- **IoE Layer**$-$ which, provides T2F access and F2T broadcast services. Where the T2F access services are implemented by resorting to the reservation-based access protocol developed in [39], which exploits a Network Utility Maximization approach, to provide collision-free access to the things served by

a FN, implementing the protocol dynamically allocates access time-windows. Also, access rates to the requiring things fundamentally on: volume of data (uploaded), per-thing available energy, and, per-connection fading level. The F2T broadcast service is implemented according to [107]. Where, periodically profiles the throughput sustained by the ongoing F2T TCP/IP connections and this dynamically adjust to the corresponding transmission parameters, to maximize the energy and bandwidth efficiency's, under hard constraints per-connection minimum throughput and maximum tolerated delay-jitter.

- **Fog Layer**– which, performs the energy-efficient management of the networking and computing physical resources equipping on each FN. Also, the energy-efficient management of the inter-Fog traffic conveyed by the wireless backbone shown in Fig. 3.4. The integrated resource manager described in [13] is implemented. It jointly performs traffic admission control, load balancing, flow control and dynamic CPU speed scaling. With the objective to minimize the overall energy consumed by each FN, under hard upper limits on the resulting per-task processing delays. In other hand, the context-aware scheduler developed in [39] is also implemented, to control the bidirectional inter-Fog traffic over the wireless backbone showed in Fig. 3.4, operating on a Time Division Duplex (TDD) way and resorting to a cognitive data-fusion approach, to maximize the utilization of the backbone bandwidth under hard constraints per-connection packet collision rates.

- **Fog Layer**– which, supports the overlay inter-clone P2P network shown in Fig. 3.4, for sustaining the inter-Fog clone migration, and, manage dynamically the required migration bandwidth. The Clone migration is supported by an implementation called Follow-Me-Cloud framework [110, 111], that comprises the signaling protocol and associated logic, to allow *live* inter-Fog clone migration to respond to the thing mobility. And the dynamic management of the required migration bandwidth is accomplished to implement the bandwidth manager deployed in [4], minimizing the energy consumed by the clone migrations under hard bounds in the corresponding migration times and service downtimes.

- **Cloud Layer**– which, orchestrates the overall *Cloud-Fog-IoE* platform showed in Fig. 3.4 based on the specific features and QoS requirements to run applications. In order, to implement the solution at this layer must be "ad-hoc" tailored over the expected attributes of the supported applications. The tested FoE prototype implements the *VTube* services mentioned in [85], providing a set of *YouTube* like service primitives for real-time P2P sharing of streaming contents (such as: games, videos, multimedia books, etc.) over Fog-supported mobile content delivery networks [134]

In the next Section 3.3 corroborates the actual effectiveness of the adopted solutions for the implementation of the protocol stack by presenting the tested performance of a FoE-based prototype.

## 3.3 A Proof of Concepts Case Study: The V-FoE testbed

According. the overall FoE technological platform is composed of two inter-connected distinct subsystems. Having as the first one, which comprises the virtualized networked computing technological platform that equips that each FN equips, such as intra-Fog platform sketched in Fig. 3.5. Moreover, the IoE devices, the FNs and the remote Cloud, this subsystem comprises also the underlying networking infrastructure, which is, mobile access network, inter-Fog wireless backbone, overlay inter-clone virtual network and Internet WAN. The energy and delay-efficient management of the intra-Fog platform have been a specific focus in a number of quite recent contributions as some authors mention [7, 10, 13, 41, 42, 92, 106, 107, 119]. In our setting, explore various solutions for the adaptive orchestration of the intra-Fog virtualized resources, under a number of computing and networking setups. Therefore, in the following of this section, we focus on the performance tests and the comparisons of the inter-Fog subsystem of the Fig. 3.4, to check the actual effectiveness of the FC paradigm, supporting resource-limited wireless/mobile IoE devices. In the next subsections, we discuss the motivations for the performed test and presents the considered testing scenario, describes the main features of a small-scale FoE prototype which is the V-FoE prototype, that is implemented under the umbrella of the current ongoing research project called "GAUCHO" [53], test the energy and delay performance of the V-FoE prototype under various mobility scenarios, and, the compares the obtained V-FoE performances against the corresponding ones of a benchmark platform, that does not exploit Fog and Cloud infrastructures.

### 3.3.1 The performed test and Comparisons

The inter-thing communication could be implemented by entirely relying on the P2P service model that proximate devices make available their computing and storage capabilities, to share tasks and cooperate for workload execution, the cooperating devices build up D2D links, that typically rely on short-range wireless communication technology, such as: UWB, WiFi/Bluetooth [6, 9]. For this purpose, D2D single-hop physical links among the communicating things may be built up at the IoE layer by exploiting short-range *IEEE802.11/15* transmission technologies above mentioned. The physical links operate in the "ad-hoc" mode, and, no require the support of Fog or Cloud infrastructures. Therefore, in [6] due to fading and path-loss, the energy consumption of D2D links increases with inter-thing distance in a cubic way, to thing mobility, D2D mobile links are

intermittent, and their average failure rates typically increase with the average thing speeds [11], and the D2D model, the initiator thing needs to discover proximate things, and must perform task distribution, thing synchronization, and task retrieval. These operations can induce large service delays, particularly when, the intermittent nature of D2D connections, these abort several times before them completing.

The inter-clone overlay network in Fig. 3.4 that can be used, to cope with aforementioned limitations of D2D "ad-hoc" communication model. The overlay network allows to move the implementation of the inter-thing links *from* the unreliable, D2D-based and energy-hungry IoE physical layer *up* to the reliable, TCP/IP-based and energy-efficient virtual overlay layer. So, since the overlay C2C communication platform replaces the corresponding underlay D2D one, we await that, in principle, the following two can benefit are attained:

- the migration of the aforementioned limitations of the "ad-hoc" D2D communication platform through the utilization of stable (no intermittent) and energy-efficient (no mobility affected) intra-Fog Ethernet and inter-Fog backbone links, and;
- the reduction of the delays for the service discovery and setup.

### 3.3.2 Modeling the Simulated Framework: V-FoE testbed

Being, the Fog paradigm that is still in infancy, large-scale real-world Fog infrastructures are not currently available for test purpose. Therefore, to carry out the aforementioned expectations, we have emulated it in software a small-scale FoE prototype, which we called *Vehicular FoE (V-FoE)* testbed, which provides a proof of concept of the proposed FoE protocol stack by implementing in software) the resource orchestration and management solutions such as: i) *IoE Layer*: adaptive solution of [107], and, dynamic reservation-based solution of [39]; ii) *Fog Layer*: cognitive solution of [39], and, integrated resource management of [13]; iii) *Overlay Layer*: energy-efficient bandwidth manager of [4], and, Follow-Me-Cloud dynamic solution of [110, 111]; iv) *Cloud Layer*: dynamic orchestration of streaming applications of [85].

***Utilized simulation toolkit***– For this purpose, we adopted the recently deployed *iFogSim* toolkit [59], a short description is explained in the sequel, it natively retains the main features that allow a quite direct integration of the FoE protocol stack.

*iFogSim* toolkit allows simulations of FNs and IoE devices by tuning their computing, communication and storage capabilities, such as: i) number of computing cores and their CPU speed-vs.-computing power profiles; ii) bandwidths of NICs and them corresponding transmission rate-vs.-communication power profiles; and, iii) the available RAM for task storage.

*Edge-ward placement* mode [59], where the *iFogSim* toolkit allows to implement and tune various resource orchestration policies, to attain the most energy-efficient allocation of the workload over the overall spectrum of the available IoE devices, FNs, and remote Clouds.

### Test scenario

The considered test scenario refers to a crowd-sourcing application, which involves end-users on broad of vehicles. Specially, in [85], where this scenario considers the real-time sharing of environmental video sequences, which are acquired on-the-fly by non-professional users, equipped with smartphones and move on board of vehicles over urban areas.

The smartphones we assumed to be equipped with *VTube*- type APIs [85], and launch P2P video streaming sessions when the vehicles come in contact. The scenario simulated, by design, we have: i) two vehicles come in contact while they are moving over the same cluster, being served by the same FN 3.4; ii) after they becoming in contact, the vehicles may establish a new P2P session with probability 0.5, providing that they are not already involved in other ongoing P2P sessions, and; iii) the time is slotted, where $T_{SLT}$(s) is the slot time.

So, after launching, a session goes on, even if the involved vehicles move away to a different cluster (go to other FN). The session duration represented by $T_{SED}$ and the Inter-Session time Interval represented by $T_{ISI}$ are randomly distributed over the time intervals: $(600-1000)\ T_{SLT}$, and $(1100-1400)\ T_{SLT}$ respectively.

Having as the maximum vehicle speed $v_{MAX} = 50\ (Km/h)$, the average speeds are $\overline{v} = 5-15-25-35$ and $45\ (Km/h)$. The number of simulated vehicles in total is $N_{VHC}^{(TOT)} = 260$, distributed over $N_{CLS} = 13$ hexagonal spatial cluster of diameter $Da = 650\ (m)$, which are arranged over concentric spatial rings.

### Simulated mobility model

In [86], vehicle mobility is simulated according to the so-called *Markovian random walk with positioning*. At the beginning of time slot, each vehicle moves to a randomly selected neighborhood target cluster with probability $\alpha$, or also stays in the current cluster with probability $(1-\alpha)$. After that select of the target cluster, a point inside it is randomly chosen and the vehicle moves to it. By doing so, in the numerically ascertained that, in each time slot, one-half of the simulated inter-things such as: inter-vehicle TCP/IP connections involves vehicles, which are traveling over different clusters. Moreover, according to [86], the inter-cluster transition probability $\alpha$ and per-cluster average number of vehicle $\overline{N}_{VHC}$ may be accurately approximated by the following formulas:

Table 3.1 Power reduction factors from [17].

| Per-connection parameters | Power reduction factor $\rho_{COR}^{(SER)}$ | | |
|---|---|---|---|
| | 2 Cores | 3 Cores | 4 Cores |
| Intel SpeedStep @ 2.0$GHz$ | 6 % | 7 % | 8 % |
| Intel SpeedStep @ 2.5$GHz$ | 30 % | 30 % | 30 % |
| AMD Cool 'n' Quiet @ 2.5$GHz$ | 6 % | 7 % | 8 % |

$$\alpha = \overline{v}/v_{MAX}, \tag{3.3}$$

and

$$\overline{N}_{VHC} = \frac{1}{2} \times A_{JAM} \times (1-\alpha) \times \mathscr{S}u, \tag{3.4}$$

Where $\mathscr{S}u(m^2)$) represent the cluster area, and $A_{JAM}$ $vehicle/m^2$) represent per-cluster the maximum spatial density of vehicles when congestion vehicular occur.

### Power profiles of the simulated computing nodes

In Fig. 3.4 as we know each spatial cluster is served by a FN, where this last comprises $N_{SER} = 7$ homogeneous quad-core DELL Power Edge-type physical servers, these are equipped with 3.06 $GHz$ Intel Xeon CPU and 8 $GB$ of RAM. Where per-server the maximum and static power consumption are [125]: $P_{SER}^{(MAX)} = 228(W)$, and, $P_{SER}^{(STATIC)} = 118(W)$, respectively. A commodity wired *Giga-Ethernet* switch provides intra-Fog connectivity. Each server may host up to $N_{CNT}^{(MAX)}$ Docker-type containers [20] of size: $S_{CNT} = 30$ $(Mb)$. Each container clones a user smartphone that is a thing, and, as we show in Fig 3.5(b), which is equipped with a virtual processor with $n_{COR}$ homogeneous virtual cores. So, according to the general model which one is reported in [17] for the power consumption of virtualized multi-core processors, the average computing $P_{CMP}^{(SER)}(W)$ wasted by a multi-core container can be modeled as:

$$P_{CMP}^{(SER)} = \frac{P_{SER}^{(STATIC)}}{N_{CNT}^{(MAX)}} + (1-\rho_{COR}^{(SER)}) \times \frac{(P_{SER}^{(MAX)} - P_{SER}^{(STATIC)})}{N_{CNT}^{(MAX)}} \times n_{COR} \times \left(\frac{f_{SER}}{f_{SER}^{(MAX)}}\right)^{\gamma}. \tag{3.5}$$

In Eq. (3.5), we have that: i) $f_{SER}$ $(bit/s)$ (resp., $f_{SER}^{(MAX)}$ $(bit/s)$) is the per-virtual core average (resp., maximum) processing frequency; ii) $\gamma \cong 3$ is a dimension-less power exponent; and, iii) $\rho_{COR}^{(SER)}$ is the so-called power reduction factor. According to [17], it is formally defined as the fraction of the total consumed power

that is shared by the processing cores for common target operations. As illustrated in Table 3.1, this fraction depends on both the power features of the considered multi-core processor and the number $n_{COR}$ of processing cores. Its typical values fall into the range $6\%-30\%$ and tend to somewhat increase with the number $n_{COR}$ of utilized cores.

According to [17], where define the fraction of the total consumed power that shared by the processing cores for common target operations, as showed in Table 3.1, the fraction depends on both the power features of the considered multi-core processor and number $n_{COR}$ of the processing cores, which values fall into the range $(6-30)\%$, and, tend to some which increase with the number $n_{COR}$ of utilized cores.

At first, the power model of Eq. (3.5), may be used for the evaluation of the computing power: $P_{CMP}^{(MOB)}(W)$ consumed by each mobile user device. Due, since the most part of the current IoE devices, is still single-core and no virtualized, where the Eq. (3.5) simplifies to [17]:

$$P_{CMP}^{(MOB)} = P_{MOB}^{(STATIC)} + \left(P_{MOB}^{(MAX)} - P_{MOB}^{(STATIC)}\right) \times \left(\frac{f_{MOB}}{f_{MOB}^{(MAX)}}\right)^{\gamma}. \tag{3.6}$$

In a formal point of view, the Eq. (3.6) is obtained by posing: $N_{CNT}^{(MAX)} = n_{COR} = 1$, and: $\rho_{COR}^{(SER)} = 0$ into Eq. (3.5).

## Power profiles of the simulated TCP/IP connections

The simulated *Vehicle-to-Fog*, *Fog-to-Vehicle* and *Fog-to-Fog* wireless channels as we show in Fig. 3.4, where are assumed to be affected by frequency-flat block-type Rice fading and, as mentioned in [34], where assumed to be supported by *IEEE802.11b WiFi* technology. The Rice factor of the mobile *Vehicle-to-Fog* and *Fog-to-Vehicle* channels is *7.4 (dB)*, while in the static inter-Fog wireless backbone (*Fog-to-Fog*) is *17 (dB)*.

Moreover, we assume that the resulting wireless/wired end-to-end transport-layer connections showed in Fig. 3.4 implement the TCP NewReno protocol, to guarantee reliability, through the presence of fading/mobility/traffic congestion induced connection failures [34]. Accordingly, to the results of the power analysis reported in [10, 107], the average power $P_{NET}(W)$ consumed by TCP connection is related to the corresponding average transport $R_{NET}(b/s)$ as in the following formula:

$$P_{NET} = \Lambda \times (R_{NET} \times \overline{RTT})^{\eta} + P_{NET}^{(SETUP)}. \tag{3.7}$$

Table 3.2 Main default parameters of the simulated V-FoE testbed. The subscripts WD, BB and WL denote Wired (e.g., intra-Fog), BackBone-supported and WireLess (e.g., Vehicle-to-Fog, Fog-to-Vehicle and Vehicle-to-Vehicle) TCP/IP connections, respectively.

| **Parameter setting** | | |
|---|---|---|
| $T_{SLT} = 500\ (ms)$ | $v_{MAX} = 50\ (Km/h)$ | $N_{VHC}^{TOT} = 260$ |
| $OVH = 0.2$ | $N_{CLS} = 13$ | $Da = 650\ (m)$ |
| $\eta_{WD} = 1.1$ | $\eta_{BB} = 2.1$ | $\eta_{WL} = 3.5$ |
| $\Lambda_{WL} = \Lambda_{BB} = 11.5\ (mW/Mb)$ | $\Lambda_{WD} = 4.5\ (mW/Mb)$ | $\overline{RTT}_{WD} = 0.7\ (ms)$ |
| $R_{WD}^{(MAX)} = 4 \times R_{BB}^{(MAX)} = 500\ (Mb/s)$ | $R_{WL}^{(MAX)} = 8.5\ (Mb/s)$ | $\overline{RTT}_{BB} = 12\ (ms)$ |
| $P_{SER}^{(STATIC)} = 118\ (W)$ | $P_{SER}^{(MAX)} = 228\ (W)$ | $P_{NET,WD}^{(SETUP)} = 18\ (mW)$ |
| $f_{SER}^{(MAX)} = 9.5\ (Mb/s)$ | $f_{SER} = 2.4\ (Mb/s)$ | $P_{NET,WL}^{(SETUP)} = P_{NET,BB}^{(SETUP)} = 525\ (mW)$ |
| $\rho_{COR}^{(SER)} = 0.06$ | $\delta = 500\ (CPU cycles/b)$ | $S_{CNT} = 30\ (Mb)$ |
| $N_{SER} = 7$ | $N_{CLS} = 13$ | $\overline{v} = 5, 15, 25, 35, 45\ (Km/h)$ |
| $P_{MOB}^{(MAX)} = 0.2\ (W)$ | $P_{MOB}^{(STATIC)} = 0.12\ (W)$ | $\gamma = 3$ |
| $f_{MOB}^{(MAX)} = 0.9\ (Mb/s)$ | $f_{MOB} = 0.25\ (Mb/s)$ | $n_{COR} = 2$ |

The Eq. (3.7), we have: i) $\eta$ is a dimension-less positive exponent; ii) $P_{NET}^{(SETUP)}\ (W)$ is the static power consumed by the connection setup; iii) $\overline{RTT}\ (s)$ is the average round-trip-time of our considered connection; and, iv) $\Lambda\ (W/b)$ represent the average dynamic power consumed by the connection on a per-bit basis. Table 3.2 show the actual values of $\Lambda$, $\overline{RTT}$, $\eta$, and $P_{NET}^{(SETUP)}$ this depend on the power-delay features of the utilized wireless/wired transmission technologies [34].

## Energy wasted by the live migration of clones

By definition, the average energy $\mathscr{E}_{CLONE}^{(MIG)}\ (J)$ consumed by the inter-Fog migration of a clone over the wireless backbone as we showed in Fig. 3.4 equates the product: (network power) by (migration time). So, by leveraging Eq. (3.7), we have that [7, 127]:

$$\mathscr{E}_{CLONE}^{(MIG)} \stackrel{\text{def}}{=} P_{NET} \times T_{MIG} = P_{NET} \times \left( \frac{S_{CNT} \times (1 + OVH)}{R_{NET}} \right) \times (1 + \overline{FN}_{CON}), \quad (3.8)$$

where:

- *the dimension-less and positive coefficient*: $OVH$ accounts for the migration-induced traffic overhead [7, 127], and;
- *the non-negative factor*: $\overline{FN}_{CON}$ is the per-connection average number of failures, (e.g., the average number of times that an on-going connection fails before completing). We can anticipate that $\overline{FN}_{CON}$ depends on the power-delay profiles of the considered wireless/wired transmission technologies, as well as on the considered service and mobility models, which will be described in the following Section 3.3.3.

### 3.3.3 Performance results and Comparisons

We evaluate the numerical results of the simulated V-FoE testbed report per-connection average consumed energies and the resulting round-trip-times of P2P inter-clone overlay virtual network showed in Fig. 3.4.

Specifically, in order to stress the effect of the reported energy values account to:

- support of the instantiated *Vehicle-to-Fog*, *Fog-to-Vehicle* and *Clone-to-Clone* wireless/wired links;
- processing of the workload to all involved mobile/fixed computing nodes, and;
- support of the inter-Fog mobility-induced clone migrations.

#### Reference benchmark

For comparison, we have implemented in software and simulated a benchmark testbed (e.g., *Vehicular D2D (V-D2D)* testbed), which operates under the same scenario vehicular scenario), which was describe previously for the *V-FoE* testbed, according [6, 79], utilizes only "ad-hoc" D2D *IEEE802.11b* single-hop links for the support of the *Vehicle-to-Vehicle* TCP/IP transport connections.

At this regard, we will mention the main remarks:

- the general power-vs.-rate model in Eq. 3.7 also apply to WiFi-supported D2D transport connections. Therefore, in the resulting per-connection round-trip-time $\overline{RTT}$ becomes quite sensitive on the corresponding (time-varying) inter-vehicle distance *d (m)* and tends to scale *up/down* proportionally to it, where is [6, 34, 79]:

$$\overline{RTT} \propto d. \tag{3.9}$$

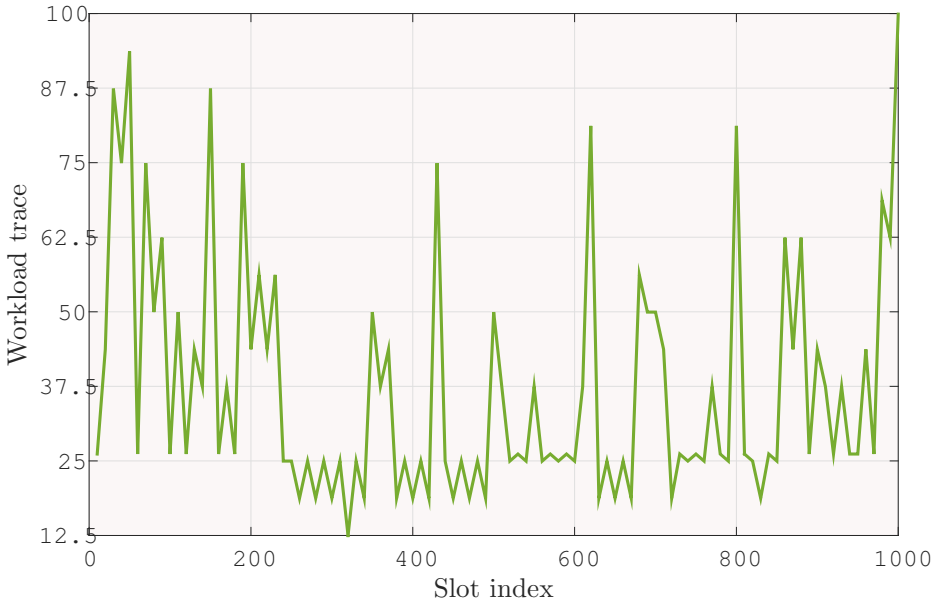



Fig. 3.6 Normalized sample trace of an I/O traffic flow from an enterprise data center in Microsoft [18].

- the CPU power consumed by a device engaged into a D2D connection which can be still evaluated through Eq. (3.6).
- to carry out fair performance comparisons, the traffic flows conveyed by all the simulated TCP/IP connections which are randomly scaled and cyclically delayed versions of the mother traffic trace in Fig. 3.6, which report the normalized I/O traffic flow actually measured from four RAID volumes of an enterprise data center in Microsoft [18]. In the carried out tests, actually this is the peak traffic values, these are set to 80% of the maximum throughput $R_{NET}^{(MAX)}$ of the corresponding TCP/IP connections.
- in the FoE paradigm, the T2F access is managed by the context-aware reservation-based protocol proposed in [39] that guarantees collision-free access, to perform fair comparisons, the simulation of WiFi-supported *V-D2D* benchmark testbed carried out under the assumption that the utilized Carrier Sensing Multiple Access with Collision Avoidance (CSMA/CA) protocol guarantees collision-free (multiple access interference-free) communication. With all of above mentioned may tend, indeed, to over-estimate the actual performance of the benchmark *V-D2D* testbed, we can anticipate that the numerical plots shown in Figs. 3.7 that corroborate the performance superiority of the proposed V-FoE platform.

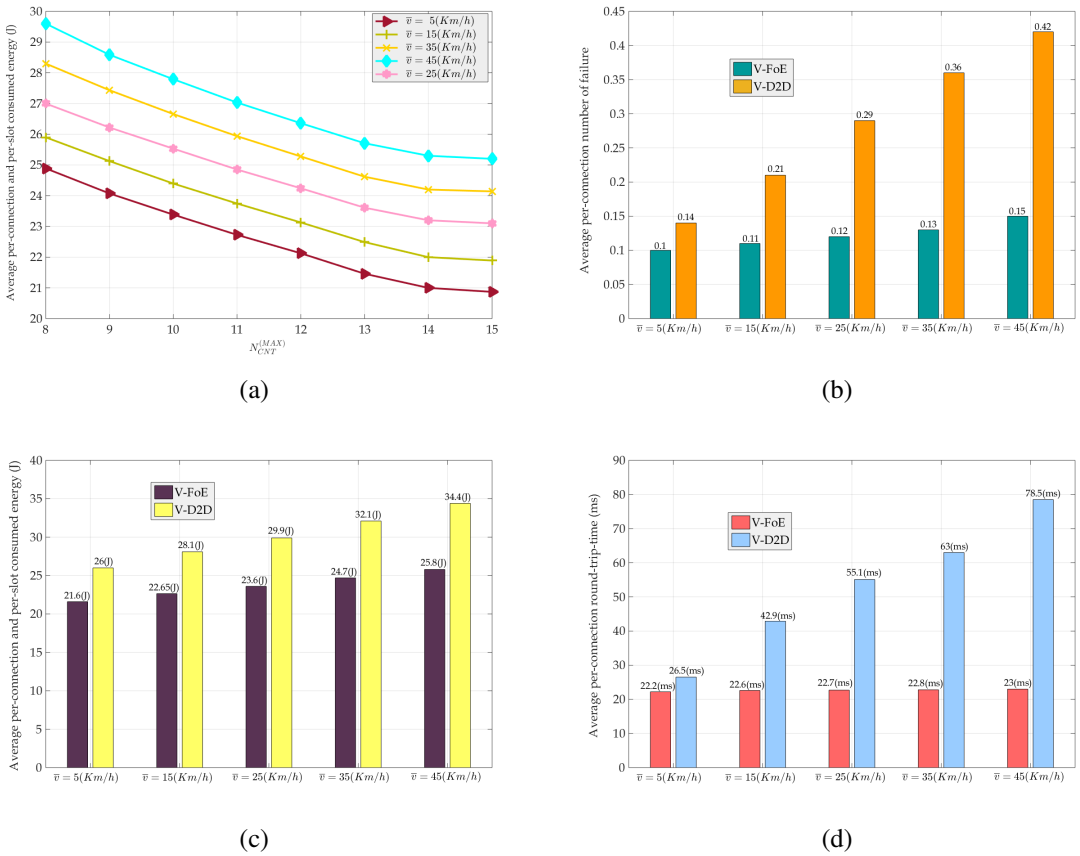



Fig. 3.7 Performance results and comparisons. (a) Per-connection and per-slot average energies consumed by the *V-FoE* testbed at $\overline{v}$ =5, 15, 25, 35, 45 *(Km/h)*; (b) Per-connection average number of failures: $\overline{F}_{CON}$ of the *V-FoE* and *V-D2D* testbed at $\overline{v}(Km/h)$ =5, 15, 25, 35, 45, and, $N_{CNT}^{(MAX)} = 13$; (c) Per-connection and per-slot average energies consumed by the *V-FoE* and *V-D2D* testbed at $\overline{v}(Km/h)$ =5, 15, 25, 35, 45, and, $N_{CNT}^{(MAX)} = 13$; (d) Per-connection average round-trip-times of the *V-FoE* and *V-D2D* testbed at $\overline{v}(Km/h)$ =5, 15, 25, 35, 45, and, $N_{CNT}^{(MAX)} = 13$;

## Obtained numerical results

An examination of the V-FoE energy curves in Fig. 3.7(a) gives rise to two main remarks.

1. the per-Fog number of the turned ON physical servers decreases to growing values of the per-server virtualization capacity $N_{CNT}^{(MAX)}$ represented in Eq. 3.5, all the V-FoE energy plots in Fig. 3.7(a) decrease in $N_{CNT}^{(MAX)}$ at fixed average vehicle speed $\overline{v}$. So, the virtualization density of the container-based virtualization technology is higher than the corresponding one of the VM-based technology, we can say that the former technology is more energy-efficient than the latter one.

2. we have numerically ascertained in the simulation scenario, where the average number of performed clone migrations increases of about 4.2 times by passing from $\overline{v} = 5 - 45$ $(Km/h)$, for this the V-FoE energy curves showed in Fig. 3.7(a) scale-up of about 21% for increasing values of the average speed of the simulated vehicles.

In the next step, we pass to consider the V-FoE - vs.- V-D2D performance comparison showed in Fig. 3.7(b) where we indicate: i) V-D2D testbed exhibits values of the average number of per-connection failures that are higher than the corresponding ones of V-FoE platform, and; ii) the rates of the increment of the average number of the per-connection failures with the vehicle speed. Specifically, in the Fig. 3.7(b) shows the average number of per-connection failures of V-D2D (resp., V-FoE) testbed increases about 3 times (resp., 1.5 times) by passing from $\overline{v} = 5 - 45$ $(Km/h)$, carrying out test to the following conclusions: i) thanks to the Fog infrastructure, the connection failures suffered by the V-FoE testbed are mainly due to sporadic traffic congestion phenomena in the access network, and; ii) due to the "ad-hoc" nature, the benchmark V-D2D testbed is very sensitive on the fading and path-loss impairments, and, the mobility-induced increments of the average distances of the sustained D2D connections.

We confirm the performance trend by bar plots showed in Figs, 3.7(c) and 3.7(d), opening the doors to the follow considerations:

- the V-FoE testbed is more energy efficient of the benchmark V-D2D one, the measured per-connection average energy gaps are around 20%, 24%, 26.7%, 29.9% and 33% at $\overline{v}$ =5, 15, 25, 35, and 45 *Km/h* (see Fig. 3.7(c));

- the increment of the energy consumed by the V-FoE connections is almost entirely induced by the increment of the average number of clone migration, induced by the increment of the average vehicle speed.

- the live migrations of clones involves, very limited service interruptions [7, 127], the corresponding average-round-trip times of V-FoE connections are almost insensitive on the vehicle speed and remain around $22-26$ $(ms)$ as we can see in Fig. 3.7(d). And due, to the increased propagation delays and failure-induced TCP re-transmission, the Fig. 3.7(d) shows the corresponding average round-trip-time of V-D2D connections quickly scales up with the values of the average vehicles speed, that passes from $26.5-78.5$ $(ms)$ at corresponding $\overline{v}=(5-45)$ $(Km/h)$.

The reported comparative performance results confirm the aforementioned expectation about the improved delay and energy efficiencies of the proposed FoE technological platform of Fig. 3.4

# Chapter 4

# Energy and delay-efficient dynamic queue management in TCP/IP virtualized data centers

In this chapter, we propose and test an efficient dynamic queue management scheduler that allows us to dynamically allocate tasks size, rate of computing, communication rate, in TCP/IP virtualized data centers which are connected to (possibly, mobile) clients operating under tight constraints of maximum delay per-job.

The quickly emerging utilization of Software-as-a-Service (SaaS) Fog Computing centers as an Internet virtual computing commodity is raising concerns over the energy consumptions of networked data centers for the support of delay-sensitive applications. In addition to that, the energy consumed by the servers, the energy wasted by the network devices, which support TCP/IP reliable inter-Virtual Machines (VMs) connections is becoming a quite significant challenge. We propose and develop a framework for the joint characterization and optimization of TCP/IP SaaS Fog data centers which utilize a bank of queues to increase the fraction of the admitted workload.

The goal is to maximize the average workload admitted by the data center; and, minimize the resulting networking-plus-computing average energy consumption while meeting hard QoS requirements on the delivered transmission rate and processing delay. For this purpose, we exploit the **Lyapunov** stochastic optimization approach, to analyze an optimal (yet practical) online joint resource management framework, which dynamically performs: i) admission control of the offered input traffic; ii) balanced control and dispatching of the admitted

workload; iii) dynamic reconfiguration and consolidation of the Dynamic Voltage and Frequency Scaling (DVFS)-enabled Virtual Machines (VMs) instantiated onto the parallel computing platform; iv) flow control of the inter-VM TCP/IP connections; v) queue control; vi) up/down scaling of the processing frequencies of the instantiated VMs; and, vii) adaptive joint consolidation of both physical servers and TCP/IP connections. Necessary and sufficient conditions for the feasibility and optimality of the proposed scheduler are also provided in closed-form. The salient features of the proposed scheduler are that: i) it is adaptive and admits distributed scalable implementation; ii) it is capable to provide hard QoS guarantees, in terms of minimum/maximum instantaneous rate of the traffic delivered to the client, instantaneous rate-jitter, and total processing delay; iii) it provides deterministic bounds on the instantaneous queue backlogs; iv) it avoids queue overflow phenomena, and v) it effectively tracks the possibly unpredictable time-fluctuations of the input workload, to perform joint resource consolidation without requiring any *a priori* information or forecast of input workload. The energy and delay performances of the proposed scheduler are numerically evaluated and compared against the corresponding ones of some competing and state-of-the-art schedulers, under: i) Fast-Giga-10Giga Ethernet switching technologies; ii) several settings of reconfiguration-consolidation costs, and; iii) synthetic, real-world workloads.

## 4.1 Virtualized TCP/IP computing platform

The Fig. 4.1 reports the main functional blocks of emerging virtualized networked data centers, that work under the SaaS model that is mentioned in [100, 101, 107], which operate at the Middleware layer and those are composed by: i) Admission Control Server (ACS), that acts also as gateway Internet router; ii) Input queue, that temporarily buffers the admitted workload; iii)Load Balance, which dynamically reconfigures and consolidates the available computing-plus-networking physical resources and dispatches the workload buffered by the input queue to the turned ON VMs; iv) Virtual Switch, which performs network flow control and manages the TCP/IP end-to-end connections, and; v) Bank of VMs, each VM equipped with a local buffer for temporarily storing the assigned workload. With the help of the emerging next generation of broadband Fog computing data centers as mentioned in [100, 125], considering a time-slotted system, where the duration $T_s$ $(s)$ in each slot can range from tens of milliseconds to tens of minutes. Where $t$ is the discrete slot index, with $t-$th slot spanning the semi-open interval $[tT_s,\ (t+1)T_s),\ t \geq 0$.

In the infrastructure layer of the the considered data center this is composed by a set $\{PS(k),\ k = 1,2,\ldots,N_{SE}\}$ of heterogeneous physical servers. $PS(k)$, with $k-$th servers can host up to $M_{max}(k)$ heterogeneous VMs, Hence, $M_v \triangleq \sum_{k=1}^{N_{SE}} M_{max}(k)$, where is the resulting maximum number of VMs hosted by the considered

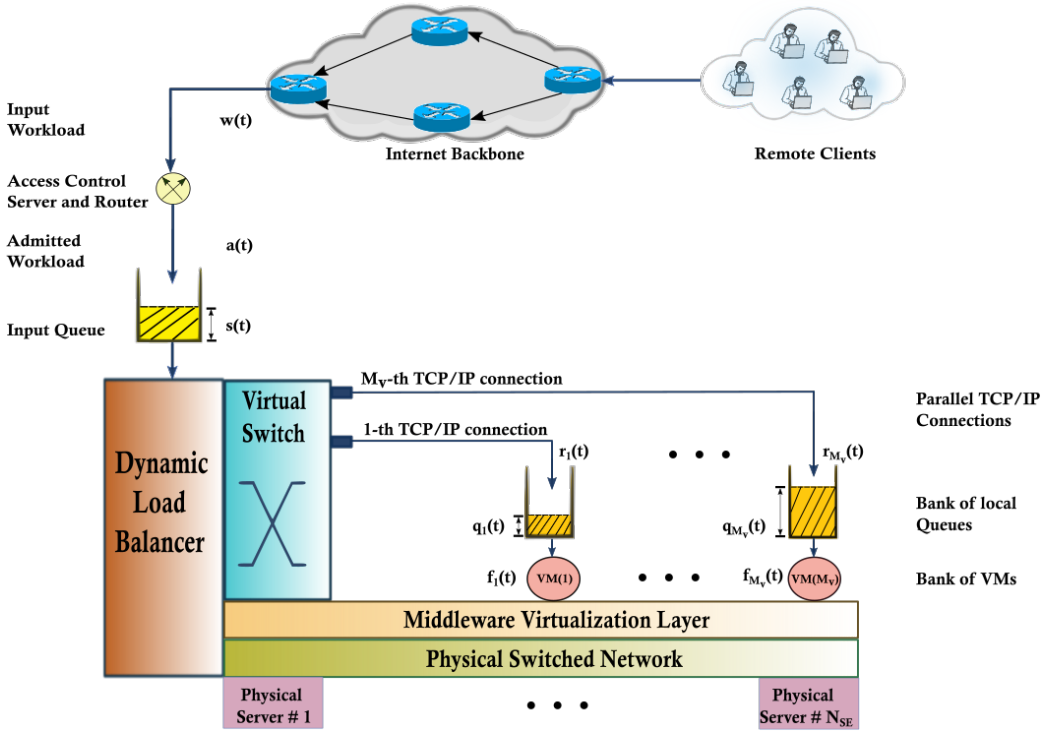



Fig. 4.1 The considered virtualized networked computing platform. Light blue boxed are virtual network interface cards.

data center, while $\{VM(j),\ j = 1,\ldots,M_v\}$, being the resulting set of available VMs. The wired physical intra-data center network is composed by a set $\{PW(m),\ m = 1,2,\ldots,N_{SW}\}$ of heterogeneous physical switches, where each one is equipped with multiple ports and each port hosts a Ethernet-type Physical Network Interface Card (PNIC).

We considered some important remarks about the considered networked architecture which are: i) the physical network in the framework can be single or multi-hop and of the arbitrary topology. The wired type, we assume that the multi-hop route spanned by each end-to-end TCP/IP connection as we show in Fig. 4.1, this connection is stable, and not change during the time duration of the supported connection; ii) the set of turned ON VMs and associated TCP/IP connections as a consequence of the performed consolidation actions, can vary over the time. So, in the sequel, $\mathscr{S}(t) \subseteq \{1,2,\ldots,M_v\}$, that indicates the set of VMs and related connections when turned ON at slot $t$; iii) the admission control and load balancing actions can be implemented by either centralized front-end servers or distributed back-end servers. The pursued solving approach and resulting optimal scheduling actions apply, regardless of the specific implementation choice, and; iv) in the case when each physical server hosts homogeneous VMs, the maximum number $M_{max}(k)$ of VMs hosted by $k-$th physical server is evaluated by:

$$M_{max}(k) = \left\lfloor \min\left\{ \frac{Co_{PS}(k)}{co_{VM}(k)}; \frac{Ra_{PS}(k)}{ra_{VM}(k)}; \frac{Di_{PS}(k)}{di_{VM}(k)}; \frac{Bw_{PS}(k)}{bw_{VM}(k)} \right\} \right\rfloor,$$

where, we have $Co_{PS}(k)$ (resp., $Ra_{PS}(k)$, $Di_{PS}(k)$ and $Bw_{PS}(k)$) being the number of physical cores (resp., RAM, disk and I/O bandwidth capacity) of $k-$th physical server, and $co_{VM}(k)$, $ra_{VM}(k)$, $di_{VM}(k)$ and $bw_{VM}(k)$ correspond to the per-VM demands.

### 4.1.1 Input workload and dynamic of the input queue

In Internet virtualized DCs, each processing unit executes the currently assigned task by self-managing own local virtualized storage/computing resources. When a request for a new job is submitted from the Cloud providers to the virtualized cloud which contains virtualized networked data center, the resource controller dynamically performs both admission control and allocation of the available virtual resources [84].

Specifically, at the end of slot $t$, new input requests arrive at the input of the ACS of Fig. 4.1. This happens according to a random real-valued arrival process $\{w(t) \in \mathbb{R}_0^+,\ t \geq 0\}$, that is limited up to $\{w_{max} \in \mathbb{R}_0^+\}$ Information Units (IUs) per slot (e.g., $w(t) \leq w_{max},\ t \geq 0$)[1]. The arrival process is assumed to be independent from the current backlogs of the input/output queues of Fig. 4.1 and its statistics can vary in an *unpredictable* way. However, we do not assume any a *priori* knowledge about the statistical behavior of $\{w(t)\}$. For example, $\{w(t)\}$ could be a Constant Bit Rate (CBR) input traffic, or it could be a Markov-modulated process with the time-varying instantaneous rate. Hence, after indicating by $a(t) \in \mathbb{R}_0^+(IU/slot)$ the number of $IUs$ out of $w(t)$ that are admitted into the input queue in Fig. 4.1 at the end of slot $t$, the following constraints holds[2]:

$$0 \leq a(t) \leq w(t) \leq w_{max}, \quad \forall\, t \geq 0. \tag{4.1}$$

This models a general scenario with unpredictable and (possibly) time-varying input workloads. Let $r_j(t) \in \mathbb{R}_0^+\ (IU/slot),\ j = 1,\ldots,M_V$, be the workload that is drained by the input queue during slot $t$ and, then, forwarded to $VM(j)$ over the $j-$th TCP/IP connection. Since only the turned ON VMs may receive new workload, we must have: $r_j(t) = 0,\ j \notin \mathscr{S}(t)$. Moreover, the summation of the forwarded workloads cannot exceed the current backlog $s(t) \in \mathbb{R}_0^+$ of the input queue, for that we must have: $\sum_{j \in \mathscr{S}(t)} r_j(t) \leq s(t)$. Finally, since the maximum flow conveyed by each TCP/IP connection is upper limited by the maximum size of the

[1]The meaning of an IU is application dependent. It may represent a bit, byte, segment or even an overall large-size application task (for example, a large image). We anticipate that, in the carried out tests of Section 4.5, IUs are understood as *Mbit*.

[2]Regarding the reject fraction: $1 - (a(t)/w(t))$ of the workload, the data center can return a negative feedback to the clients, who, in turn, may resend later their request to the data center

corresponding congestion window [3], we introduce the additional constraint: $\sum_{j\in\mathscr{S}(t)} r_j(t) \leq r_{max}$. Above all, we must have:

$$r_j(t) \geq 0,\ j = 1,\ldots,M_V, \qquad \forall\, t \geq 0, \tag{4.2.1}$$

$$r_j(t) = 0,\ j \notin \mathscr{S}(t), \qquad \forall\, t \geq 0, \tag{4.2.2}$$

$$\sum_{j\in\mathscr{S}(t)} r_j(t) \leq \min\{s(t);\ r_{max}\}, \qquad \forall\, t \geq 0, \tag{4.2.3}$$

so that the time-evolution of the backlog $\{s(t) \in \mathbb{R}_0^+,\ t \geq 0\}$ of the input queue reads as in (see Fig. 4.1):

$$s(t+1) = \left[s(t) - \left(\sum_{j\in\mathscr{S}(t)} r_j(t)\right)\right]_+ + a(t), \quad t \geq 0,\ s(0) = 0. \tag{4.3}$$

### 4.1.2 Networking-plus-computing energy consumptions

The goal of the Transport-layer connection in Fig. 4.1 is provided end-to-end links from the input queue to the local ones by exploiting the multi-hop routes done available by the underlying switched physical network. As mentioned in [3, 46, 115], where there are at least two reasons to resort to the TCP/IP mainly, the TCPNewReno protocol to model the managed end-to-end intra-DC transport connections, to implement these connections. First, it guarantees, loss and error-free transport of data. Second, it performs congestion control, to match the per-connection flows to the aggregate traffic conveyed by the overall network. Furthermore, we note that the energy $\mathscr{E}_{net}(j,t)$ (*Joule*) consumed by the $j-$th TCP/IP connection is the summation:

$$\mathscr{E}_{net}(j,t) = \mathscr{E}_{net}^{setup}(j) + \mathscr{E}_{net}^{dyn}(j,t), \tag{4.4}$$

a static $\mathscr{E}_{net}^{setup}(j)$, and a dynamic $\mathscr{E}_{net}^{dyn}(j,t)$ portion, the static portion does not depend on the conveyed flow and accounts for the summation of the setup energies of the PNICs actually crossed by the $j-$th connection. While the dynamic portion accounts for the additional flow-depending energy consumed by the crossed PNICs. At this regard, we point out that the analysis detailed in [41] leads to the conclusion that the dynamic energy wasted by TCPNewReno connections working in the steady-state (e.g., In agreement with the behavior of legacy TCP/IP connection working in the Congestion Avoidance state) is well captured by the following closed-form expression:

$$\mathscr{E}_{net}^{dyn}(j,t) = \sigma_j\ (r_j(t))^{\gamma}, \tag{4.5}$$

where the state $\sigma_j$ $((Joule) \times (slot/IU)^{\gamma})$ of the $j-$th connection is, in turn, given by:

$$\sigma_j = G_j \left( \frac{\overline{RTT}_j}{1.22\, MSS} \right)^{\gamma}. \tag{4.6}$$

In the Eq. (4.6), we have [41]: i) $MSS$ $(IU)$ being the maximum size of a TCP segment; ii) $\overline{RTT}_j$ is the average round-trip-time in multiple of the slot period) of the $j-$th connection; iii) $\gamma > 1$ is a dimension-less shaping exponent which depends on the utilized switching technology; and, iv) $G_j$ $(Joule)$ is the summation of the dynamic energy consumptions of the PNICs crossed by the $j-$th connection. We point out that, after performing the routing operation, both $\mathscr{E}_{net}^{setup}(j)$ in (4.4) and $G_j$ in (4.6) may be profiled online.

Under the SaaS model, clients submit their computing requests as variable-size VMs and, then, the data center provider must charge them on a per-VM basis [74]. Hence, according to the VM-centric perspective, as mentioned in [67], we proceed to model the overall computing energy $\mathscr{E}_{com}(j,t)$ $(Joule)$ consumed by $VM(j)$ at slot $t$ as in:

$$\mathscr{E}_{com}(j,t) = \mathscr{E}_{com}^{idle}(j,t) + \left( \mathscr{E}_{com}^{max}(j,t) - \mathscr{E}_{com}^{idle}(j,t) \right) \left( \frac{f_j(t)}{f_j^{max}} \right)^{\alpha}. \tag{4.7}$$

In the Eq. (4.7), the $f_j(t)$ $(IU/slot)$ is the number of $IUs$ processed by $VM(j)$ at slot $t$, while $f_j^{max}$ $(IU/slot)$ that is the corresponding maximum processing capability, and having:

$$0 \leq f_j(t) \leq f_j^{max}(t), \quad \forall\, t \geq 0. \tag{4.8}$$

Furthermore, $\mathscr{E}_{com}^{idle}(j,t)$ $(Joule)$ is the energy consumed by $VM(j)$ at slot $t$ in the idle state (e.g., when $VM(j)$ is turned ON but its processing frequency vanishes), while $\mathscr{E}_{com}^{max}(j,t)$ is the is the maximum energy consumed by $VM(j)$ runs at $f_j^{max}$, which is the maximum processing speed of $VM(j)$ $(bit/s)$. Finally, $\alpha \geq 2$ is a dimension-less exponent which depends on the energy profile of the hosting physical server [67]. Moreover, due to the performed server consolidation actions, the number of turned ON VMs per-server is time-varying, so that both the *idle* and *maximum* energies presented in (4.7) may vary over the time [67], and their actual values should be periodically profiled.

Continue, to consider the dynamic of the $j-$th local queue of Fig. 4.1, let $q_j(t) \in \mathbb{R}_0^+$ be its current backlog. Hence, since $VM(j)$ acts as the virtual) server of the $j-$th local queue and its service rate is $f_j(t)$, where, we have:

$$q_j(t+1) = [q_j(t) - f_j(t)]_+ + r_j(t), \quad t \geq 0,\ q_j(0) = 0,\ j \in \mathscr{S}(t), \tag{4.9.1}$$

and

$$q_j(t+1) \equiv q_j(t),\ q_j(0) = 0, \quad j \notin \mathscr{S}(t), \tag{4.9.2}$$

In the Eq. (4.9.2) accounts for the fact that turned OFF VMs do not process workload. Furthermore, we consider a (time-slotted) *G/G/1* service for modeling the input and output queues of Fig. 4.1, respectively.

Overall, the total networking-plus-computing energy $\mathscr{E}_{tot}(t)$ (*Joule*) consumed by the data center of Fig. 4.1 at slot $t$ is the summation of the corresponding computing and networking energies and, then, equates:

$$\mathscr{E}_{tot}(t) \triangleq \mathscr{E}^{tot}_{com}(t) + \mathscr{E}^{tot}_{net}(t) \equiv \sum_{j \in \mathscr{S}(t)} \left(\mathscr{E}_{com}(j,t) + \mathscr{E}_{net}(j,t)\right). \tag{4.10}$$

## 4.2 Optimization problem and dynamic solving approach

The main goal of the data center provider is to attain the best trade-off among two contrasting targets, which are, the maximization and minimization of the average networking-plus-computing energy consumption. Being this the goal of the optimization problem, which we will introduce . Here, we let $\overline{w} \triangleq \lim_{t\to\infty}(1/t)\left(\sum_{\tau=0}^{t-1} E\{a(\tau)\}\right)$ $(IU/slot)$ that is the time-average expected input workload. Moreover, the $\Pi$ is the set of all feasible scheduling policies, these policies at each slot, select the admitted workload, the network flows, the VMs processing frequencies and the set of turned ON VMs without violating the constraints in the Eqs. (4.1), (4.2.1), (4.2.2), (4.2.3) and (4.8). Furthermore, after consider any feasible scheduling policy $\pi \in \Pi$ which is taking per-slot scheduling decisions: $\mathscr{S}^{\pi}(t)$, $a^{\pi}(t)$, $r^{\pi}_j(t)$ and $f^{\pi}_j(t)$ for all $j = 1,\ldots,M_V$, let:

$$\overline{a}^{\pi} \triangleq \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} E\{a(\tau)\}, \tag{4.11}$$

being the average expected throughput under the policy $\pi$, and let: $\overline{r}^{\pi}_j$, $\overline{f}^{\pi}_j$, $\overline{\mathscr{E}}^{\pi}_{com}(j)$, and $\overline{\mathscr{E}}^{\pi}_{net}(j)$, $j = 1,\ldots,M_V$, is the analogously defined time-average expected network flows, the VM processing frequencies, the VM computing energies and per-connection networking energies, respectively. After indicating by:

$$g : \mathbb{R}^+_0 \to \mathbb{R}^+_0, \tag{4.12}$$

Table 4.1 Main taxonomy of the paper.

| Symbol | Meaning/Role |
|---|---|
| $\{PS(k),\ k=1,\ldots,N_{SE}\}$ | Set of physical servers |
| $\{PW(m),\ m=1,\ldots,N_{SW}\}$ | Set of physical switches |
| $\{VM(j),\ j=1,\ldots,M_V\}$ | Set of VMs |
| $f_j(t)\ (IU/slot)$ | Processing frequency of $VM(j)$ at $t$ |
| $r_j(t)\ (IU/slot)$ | Flow of the $j$-th TCP/IP connection at $t$ |
| $w(t)\ (IU/slot)$ | Input workload at $t$ |
| $a(t)\ (IU/slot)$ | Admitted workload at $t$ |
| $s(t),\ z(t),\ \{q_j(t)\}\ (IU)$ | Queue backlogs at $t$ |
| $\mathscr{E}_{com}(j,t),\ \mathscr{E}_{net}(j,t)\ (Joule)$ | $j$-th computing and networking consumptions at $t$ |
| $\mathscr{E}_{tot}(t)\ (Joule)$ | Total data center energy consumption at $t$ |
| $\overline{w},\ \overline{a},\ \overline{r_j},\ \overline{f_j}\ (IU/slot)$ | Time averages of $w(t)$, $a(t)$, $r(j)$ and $f_j(t)$ |
| $\overline{\mathscr{E}}_{com}(j),\ \overline{\mathscr{E}}_{net}(j)\ (Joule)$ | Time averages of $\mathscr{E}_{com}(j,t)$ and $\mathscr{E}_{net}(j,t)$ |
| $V$ | Lyapunov control parameter |
| $\alpha,\ \gamma$ | Per-VM and per-connection power exponents |
| $\mathscr{E}_{com}^{idle}(j,t),\ \mathscr{E}_{com}^{max}(j,t)\ (Joule)$ | Per-VM idle and maximum energies |
| $\sigma_j,\ \mathscr{E}_{net}^{setup}(j)\ (Joule)$ | Per-connection dynamic and setup energies |

in the revenue function adopted by the data center provider, and $\beta\ (Joule)^{-1}$ the corresponding Energy Usage Efficiency (EUE) coefficient [125], our aim is solve the following stochastic optimization problem:

$$\max_{\pi\in\Pi}\left\{g(\overline{a}^{\pi})-\beta\sum_{j=1}^{M_V}\left(\overline{\mathscr{E}}_{com}^{\pi}(j)+\overline{\mathscr{E}}_{net}^{\pi}(j)\right)\right\}, \tag{4.13.1}$$

s.t.:

$$0\leq\overline{a}^{\pi}\leq\overline{w}, \tag{4.13.2}$$

$$\overline{a}^{\pi}\leq\sum_{j=1}^{M_V}\overline{f}_j^{\pi}. \tag{4.13.3}$$

The Table 4.1 reports the main introduced taxonomy.

The optimization problem in (4.13), we have three explicative remarks. First, the revenue function in (4.12) prices the data center throughput according to the billing policy of the data center provider [125]. Hence, since its analytical expression is strongly provider-depending, according to the law of diminishing marginal revenue in economics [74], we limit to assume that $g(\cdot)$ in (4.12) is a non-negative, strictly increasing and concave

function, that vanishes in the origin and admits continue first derivative [3]. Second, several reports conclude that, in state-of-the-art data centers, the energy consumption of non-Information Technology (non-IT) equipment (cooling) can be considered almost proportional to that consumed by physical switches and servers [125]. So, the coefficient $\beta$ in (4.13.1) is numerically equal to the ratio of the total energy consumed by the data center to that by the IT equipment. We can mention that old energy-inefficient data centers can have values of $\beta$ larger than 2 while emerging energy-proportional data centers exhibit values of $\beta$ as low as 1.2 [125]. Third, we observed the constraint in (4.13.2) on the average throughput is compliant with the per-slot constraint in (4.1), while (4.13.3) limits the average throughput up to the average aggregate processing capability of all available VMs.

Let $\pi_{opt}$ (resp., $\overline{\mathscr{X}}_{opt}$) be the solution of (4.13) (resp., the value of the objective function in (4.13.1) under $\pi_{opt}$). Although $\pi_{opt}$ well meets the (aforementioned) contrasting targets of the data center provider, from an application point of view, its computation presents at least three main challenges. As First, the evaluation of $\pi_{opt}$ generally resists closed-form expression, and it's analytical characterization relies on suitable arguments of randomized scheduling as demonstrate in [93] for an in-depth formal examination of this topic. Second, the characterization of $\pi_{opt}$ requires the *a priori* knowledge of the average input workload $\overline{w}$ in (4.13.2). Since the workload offered to large-scale production data centers is typically nonstationary, its statistics may vary over the time and are not known in advance. Third, even if $\pi_{opt}$ can be computed for given workload statistics, the obtained expression would not be adaptive to unpredictable changes in the workload statistics and, then, it should be re-computed from scratch when the workload statistics change.

Motivated by these considerations, in the next sub-Section, where we present a dynamic solving approach that bypasses these drawbacks.

### 4.2.1 The pursued dynamic solving approach

Considering the recent contributions on the dynamic control of queue systems mentioned in [81, 114], the pursued solving approach relies on a suitable application of the *Lyapunov* optimization technique. So, the average performance of the developed *dynamic* scheduler approaches arbitrarily close the optimal one $\overline{\mathscr{X}}_{opt}$. Which is archived *without* requiring any *priori* knowledge or forecasting of $\overline{w}$ in (4.13.2), where QoS guarantees in terms of *deterministically limited* queue backlogs. The used formal framework is similar to that detailed, such as in [93] in the sequel, we limit to report only three steps which are more peculiar of our scenario. We have.

[3] Just as an example, the $\log-$function $g(r) = \log(1+r)$ meets the above assumptions and is, indeed, a popular revenue function [125].

First, after introducing the non-negative auxiliary variable $\overline{c}$, we recast the problem in (4.13) in the following *equivalent* form[4]:

$$\max\left\{g(\overline{c})-\beta\sum_{j=1}^{M_V}\left(\overline{\mathscr{E}}_{com}(j)+\overline{\mathscr{E}}_{net}(j)\right)\right\}, \tag{4.14.1}$$

s.t.:

$$\overline{c}\leq\overline{a}, \tag{4.14.2}$$

$$0\leq\overline{a}\leq\overline{w}, \tag{4.14.3}$$

$$\overline{a}\leq\sum_{j=1}^{M_V}\overline{f}_j. \tag{4.14.4}$$

Since $g(\cdot)$ is, by assumption, an increasing function, the constraint in (4.14.2) is attained at the optimum, so that the problems in (4.13) and (4.14) are equivalent, admitting the same solution.

Second, we have the constraint in (4.14.2) that can be viewed as a stability constraint on a *virtual* (e.g., dummy) queue with average arrival (resp., service) rate $\overline{c}$ (resp., $\overline{a}$). Specifically, the dynamic of the backlog $\{z(t),\ t\geq 0\}$ of this virtual queue reads as in:

$$z(t+1)=[z(t)-a(t)]_+ + c(t),\quad t\geq 0,\ z(0)=0, \tag{4.15}$$

The $\{c(t)\in\mathbb{R}_0^+,\ t\geq 0\}$ is a virtual arrival process whose time-average expectation equates $\overline{c}$ and its peak value is limited up to $w_{max}$, demonstrated in Eqs. (4.1) and (4.14.2),

$$\overline{c}=\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}E\{c(\tau)\}, \tag{4.16.1}$$

and

$$0\leq c(t)\leq w_{max},\quad \forall\, t\geq 0. \tag{4.16.2}$$

[4]In order to speed up the notation, we omit the explicit dependence on the policy $\pi$.

Third, the constraint in (4.14.2) on the stability of the virtual queue in (4.15) can be, in turn, *automatically* enforced by applying the Lyapunov criterion. So, after noting that the backlogs of the queues of turned OFF VMs do not change (see Eq. (4.9.2)), let:

$$L(t) \triangleq \frac{1}{2}\left(s^2(t)+z^2(t)+\sum_{j\in\mathscr{S}(t)} q_j^2(t)\right), \tag{4.17}$$

being the Lyapunov function at slot $t$. In Eq. (4.17) measures the overall queue congestion, in order to stabilize the queue system by persistently pushing the Lyapunov function towards zero, we introduce the one-step conditional Lyapunov drift function, formally defined as in [93]:

$$D(t) \triangleq E\left\{L(t+1)-L(t)\,\middle|\, \overrightarrow{h}(t)\right\}, \tag{4.18}$$

The $\overrightarrow{h}(t) \triangleq [s(t),\, z(t),\, \{q_j(t),\, j\in\mathscr{S}(t)\}]^T$ is the (column) vector of the queue backlogs in (4.17). Hence, as in [93], the goal of the optimal (Lyapunov sense) scheduler, to minimize the following drift-minus-utility function by acting on a per-slot basis:

$$D(t) - V\, E\left\{g(c(t)) - \beta \sum_{j\in\mathscr{S}(t)} (\mathscr{E}_{com}(j,t)+\mathscr{E}_{net}(j,t))\,\middle|\, \overrightarrow{h}(t)\right\}. \tag{4.19}$$

As in [93], the (dimension-less, non-negative) scalar parameter $V$ is a knob that allows the data center provider to tune the queue stability-vs.-utility trade-off. Specifically, large (resp., low) values of $V$ put the emphasis on the maximization (resp., minimization) of the utility (resp., queue delays).

### 4.2.2 Upper bounding drift-minus-utility

Unfortunately, due to the presence of the maximum operator in the Eqs. (4.3), (4.9), (4.15), per-slot the minimization of (4.19) with respect to the set of variables:

$$\left\{c(t),\, a(t),\, \mathscr{S}(t),\, \{r_j(t),\, f_j(t),\, j\in\mathscr{S}(t)\}\right\}, \tag{4.20}$$

resists closed-form expression. Hence, to tackle with this challenge, we apply the same bounding technique of [93]. Where, we square both sides of Eqs. (4.3), (4.9), (4.15), and, then, apply the following algebraic inequality:

$$(\max\{0;\, a-b\}+d)^2 \leq a^2+b^2+d^2-2a(b-d),\quad \forall\, a,b,d\geq 0, \tag{4.21}$$

To bound the *Lyapunov* drift in (4.18). So, after introducing the attained bound on $D(t)$ into (4.19), arriving at the following final upper bound on the drift-minus-utility function:

$$\begin{aligned} D(t) - V\ E\left\{ g(c(t)) - \beta \sum_{j\in\mathscr{J}(t)} (\mathscr{E}_{com}(j,t)+,\mathscr{E}_{net}(j,t)) \,\middle|\, \overrightarrow{h}(t) \right\} \\ \leq B_0 + \sum_{m=0}^{3} E\{\psi_m(t) \mid \overrightarrow{h}(t)\}, \end{aligned} \tag{4.22.1}$$

with the dummy positions:

$$B_0 \triangleq \frac{1}{2}\left(2\ r_{max}^2 + 3\ w_{max}^2 + \sum_{j=1}^{M_V} (f_j^{max})^2\right), \tag{4.22.2}$$

$$\psi_0(t) \triangleq a(t)\,(s(t) - z(t)), \tag{4.22.3}$$

$$\psi_1(t) \triangleq c(t)\ z(t) - V\ g\,(c(t)), \tag{4.22.4}$$

$$\psi_2(t) \triangleq \sum_{j\in\mathscr{J}(t)} (r_j(t)\,(q_j(t) - s(t)) + V\ \beta\ \mathscr{E}_{net}(j,t)), \tag{4.22.5}$$

$$\psi_3(t) \triangleq \sum_{j\in\mathscr{J}(t)} (V\ \beta\ \mathscr{E}_{com}(j,t) - f_j(t)\ q_j(t)). \tag{4.22.6}$$

Then, we remark. i) it is proved in [93] that the upper bound in (4.22.1) is tight at the optimum, so that, without loss of optimality, we can directly minimize it in place of the drift-minus-utility function. Moreover, since $B_0$ in (4.22.2) is a constant and the expectations in (4.22.1) are conditioned on the queue backlogs, the minimization reduces to minimize on a per-slot basis the $\psi$'s functions in (4.22.3) – (4.22.6) [93]. ii) since these functions do not depend on the statistics of the input workload, all the results presented in the sequel apply, regardless of the (*a priori* unknown) workload statistics, and; iii) since the queue backlogs in (4.22.3) – (4.22.6) play the role of know parameters, the resulting dynamic scheduler is of clairvoyant-type. So, in our framework, there is no reason to consider dynamic migration of running VMs or dynamic re-routing of the on-going TCP/IP connections, meaning that the placement of both VMs and TCP/IP connections can be assumed static and captured by the following sets of binary variables:

$$d_{kj} = \begin{cases} 1, & \text{if } VM(j) \text{ is hosted by } PS(k), \\ 0, & \text{otherwise}, \end{cases} \tag{4.23.1}$$

and

$$b_{mj} = \begin{cases} 1, & \text{if } j\text{-th connection crosses } PW(m), \\ 0, & \text{otherwise.} \end{cases} \tag{4.23.2}$$

In our framework, they play the role of known binary constants.

## 4.3 Joint dynamic scheduler

The dynamic scheduler interleaves reconfiguration and consolidation slots. Specifically, at each reconfiguration slot, the set $\mathscr{S}(t)$ of turned ON TCP/IP remains unchanged, while the scheduler performs the next reconfiguration actions: i) admission control and updating of the virtual queue, and; ii)flow control and scaling of the processing frequencies. At each consolidation slot, the set $\mathscr{S}(t)$, is updated, then mentioned reconfiguration actions take place.

Having the analytical expressions:

$$\left\{c^*(t),\, a^*(t),\, \mathscr{S}^*(t),\, \{r_j^*(t), f_j^*(t),\, j \in \mathscr{S}^*(t)\}\right\}, \tag{4.24}$$

of the control actions taken by the proposed dynamic scheduler, where is detailed in the remaining part of this section.

### 4.3.1 Updating of the virtual queue and access control

The $g_r(r) \triangleq \partial g(r)/\partial r$, is the derivative of the revenue function in the (4.12), and the $g_r^{-1}(r) : \mathbb{R}_0^+ \to \mathbb{R}$, is the resulting inverse function. So, the minimization of $\psi_0(\cdot)$ in (4.22.3) (resp., $\psi_1(\cdot)$, (4.22.4)) with respect to $c(t)$ (resp., $a(t)$) under the constraint in Eq. (4.16.2) (resp., in Eq. (4.1)), leads to the next results showed in the Appendix A for the proof.

**Proposition 1 (Dynamic virtual queue updating and access control)** *The dynamic scheduler updates the admitted traffic and arrivals of the virtual queue according to:*

$$a^*(t) = \begin{cases} w(t), & \textit{for } s(t) < z(t), \\ 0, & \textit{otherwise}, \end{cases} \tag{4.25.1}$$

*and*

$$c^*(t) = \max\left\{0; \min\left\{w_{max}; g_r^{-1}\left(\frac{z(t)}{V}\right)\right\}\right\}. \tag{4.25.2}$$

To gain insight about the behavior of the action control in (4.25.2), we can see under the previously reported assumptions, the function $g_r^{-1}(r)$ is strictly decreasing and can also assume negative values. This is a consequence, that we have: i) $c^*(t)$ in (4.25.2) decreases and, vanishes for increasing values of the ratio $(z(t)/V)$; ii) at $V = 0$, we have (see Eqs. (4.15) and (4.25.2)):

$$c^*(t) \equiv z(t) \equiv 0, \quad \forall\, t \geq 0\,; \tag{4.26}$$

and; iii) the sensitivity of $c^*(t)$ on the ratio $(z(t)/V)$ increases for increasing values of the slope of $g_r^{-1}(\cdot)$. By passing to consider the admission control in (4.25.1), we can see that it implements a threshold-based flow control, that admits all the current input workload only if the backpressure: $(s(t) - z(t))$ is negative, for example, the input queue is less congested than the virtual one, and this implies that, at $V = 0$, we have (see Eqs. (4.3) and (4.26)):

$$a^*(t) \equiv s(t) \equiv 0, \quad \forall\, t \geq 0. \tag{4.27}$$

### 4.3.2 Per-connection flow control and per-VM scaling of the processing frequencies

The minimization of $\psi_2(\cdot)$ in the Eq. (4.22.5) (resp., $\psi_3(\cdot)$ in Eq. (4.22.6)) under the constraints in (4.2.1) – (4.2.3) (resp., in (4.8)) leads to the following flow (resp., frequency scaling) control action showed in the Appendix B for the proof.

**Proposition 2 (Dynamic network flow control and VM frequency scaling)** *For $j \in \mathscr{S}(t)$, the dynamic scheduler updates the per-connection network flows and per-VM processing frequencies as in:*

$$r_j^*(t) = \begin{cases} \left(\frac{s(t) - \left(q_j(t) + \zeta^*(t)\right)}{\gamma\, V\, \beta\, \sigma_j}\right)^{\frac{1}{\gamma - 1}}, & for\, (q_j(t) + \zeta^*(t)) \leq s(t), \\ 0, & otherwise, \end{cases} \tag{4.28}$$

*and*

$$f_j^*(t) = \min\left\{ f_j^{max};\ \left( \frac{(f_j^{max})^{\alpha} q_j(t)}{\alpha\, V\, \beta\, (\mathscr{E}_{com}^{max}(j,t) - \mathscr{E}_{com}^{idle}(j,t))} \right)^{\frac{1}{\alpha-1}} \right\}. \tag{4.29}$$

*For $j \notin \mathscr{S}(t)$, we have:*

$$r_j^*(t) = f_j^*(t) = 0. \tag{4.30}$$

*Furthermore, $\zeta^*(t)$ in* (4.28) *is the non-negative Lagrange multiplier of the constraint in* (4.2.3)*. It is the (unique non-negative) root of the following nonlinear algebraic equation:*

$$\sum_{j \in \mathscr{S}(t)} r_j^*(\zeta(t)) = \min\{s(t);\ r_{max}\}, \tag{4.31}$$

*where $r_j^*(\cdot)$ is given by* (4.28)*, with $\zeta^*(t)$ replaced by the dummy variable $\zeta(t)$.*

The network flow control in (4.28), we remark: i) in the presence of network cost, for example in the case of positive values of $\sigma_j$, where the performed flow control *does not* longer follow the usual Join-to-the-Shortest-Queue policy reported, such as in [81, 114]. In fact, the flow $r_j^*(\cdot)$ conveyed by the $j-$th connection gently scales down for increasing values of the network cost $\sigma_j$, also the connections with larger energy costs tend to transport layer flows; ii) larger values of $\zeta^*(\cdot)$ tend to stronger reduce the network flows dispatched to more congested local queues.

We consider the frequency scaling control action in (4.29), we see that the $j-$th processing frequency scales up for increasing values of the ratio: $q_j(t)/V$, and approaches $f_j^{max}$ at large enough values of this ratio, being in agreement with the following properties: First, in order to stabilize the most congested local queues, higher values of the processing frequencies are required showed in Eq. (4.9). Second, vanishing values of $V$ push the overall queue system to be *less* energy conserving, while *enforcing* queue stability showed in Eq. (4.19).

Finally, we see that, at $V = 0$, the term proportional to the networking (resp., computing) energy in (4.22.5) (resp., in (4.22.6)) vanishes, the consequence, since also $s(t)$ vanishes at $V = 0$ showed in Eq. (4.27), the minimization of (4.22.5) and (4.22.6) directly leads to:

$$r_j^*(t) \equiv q_j(t) \equiv 0, \ \text{ and } \ f_j^*(t) \equiv f_j^{max}, \quad \forall\, t \geq 0, \ \text{ at } V = 0. \tag{4.32}$$

### 4.3.3 Joint consolidation of the networking-plus-computing resources

At each consolidation slot, the set $\mathscr{S}(t)$ should be selected over discrete collection $\Omega$ of all feasible configurations of turned ON VMs. Then, observing that only $\psi_2(\cdot)$ and $\psi_3(\cdot)$ in (4.22.5), (4.22.6) depend on $\mathscr{S}(t)$, the optimal set $\mathscr{S}^*(t)$ of turned ON VMs is individuated by solving the following optimization problem:

$$\mathscr{S}^*(t) = \arg \min_{\mathscr{S}(t)\in\Omega} \{\psi_2(t) + \psi_3(t)\}, \quad \text{s.t.: Eqs. (4.2.1), (4.2.2), (4.2.3) and (4.8).} \tag{4.33}$$

At $V = 0$, all network flows vanish showed in Eq. (4.32), that (4.33) reduces to the maximization of: $\sum_{j\in\mathscr{S}(t)} f_j(t)\, q_j(t)$, under the constraint in (8). This maximum is obtained, in turn, by running all the available VMs at their maximum frequencies, so that we have:

$$\mathscr{S}^*(t) \equiv \{1,\ldots,M_V\}, \quad \text{and} \quad f_j^*(t) \equiv f_j^{max}, \quad \forall j, \text{ at } V = 0. \tag{4.34}$$

While, at $V > 0$, the solution of (4.33) resists closed-form evaluation and, in principle, it should be computed by the exhaustive search over discrete set $\Omega$. This introduces to two main challenges. First, being the implementation complexity of the exhaustive search proportional to the size of $\Omega$, it scales up in an exponential (resp., linear) way for increasing $M_V$ in the case of heterogeneous (resp., homogeneous) data centers[5]. In practice, this limits the maximum affordable values of $M_V$ up to: 11 – 12 (resp., 2000 – 4000) in the case of heterogeneous (resp., homogeneous) data centers. Second, the exhaustive search-based approach is inherently offline and, then, requires that, at each consolidation slot, the solution of (4.33) is re-computed from scratch.

To tackle these challenges, we develop a dynamic two-step approach for the joint consolidation of the computing-plus-networking resources. Specifically, in the firstly, we transform the discrete optimization problem in (4.33) into an equivalent one that involves only *continuous* variables. The obtained continuous problem is non-convex and, it resists closed-form solution. So, in the next step, we develop a suitable set of adaptive iterations that approach the solution of the afforded consolidation problem by starting from the resource allocation configuration already computed at the previous slot.

#### The proposed dynamic approach for the joint adaptive consolidation

The $\mathscr{P}_{SE}^{idle}(k)$ (resp., $\mathscr{P}_{SE}^{max}(k)$) is the idle (resp., maximum) power measured in (*Watt*) consumed by the $k-$th physical server of Fig. 4.1. At each consolidation slot, we have: i) $VM(j)$ is turned ON off its processing

[5]Specially, in the homogeneous case, we have: $\Omega = \{\emptyset, \{1,2\}, \{1,2,3\}, \ldots, \{1,2,\ldots,M_V\}\}$, so that the size of $\Omega$ is: $(1+M_V)$. Moreover, in the heterogeneous case, the size of $\Omega$ scales as the binomial coefficient: $\binom{M_V}{N_{SE}}$.

frequency is positive; ii) the $k-$th physical server is turned OFF only when all the hosted VMs are turned OFF (see Eq. (4.23.1)); iii) the $j-$th TCP/IP connection is turned OFF iff $VM(j)$ is turned OFF showed in Eq. (4.2.2), and; iv) all the available VMs and TCP/IP connections are candidate to be turned ON/OFF.

As a direct consequence, after indicating by $u_{-1}(x)$ the unit-step Heaviside's function (that is, $u_{-1}(x)=0$ for $x\leq 0$, and $u_{-1}(x)=1$ for $x>0$), the total computing and networking energies in (4.10) consumed by all physical servers and PNICs are given by:

$$\mathscr{E}_{com}^{tot}(t)=T_S\sum_{k=1}^{N_{SE}}\left\{\mathscr{P}_{SE}^{idle}(k)\;u_{-1}\left(\sum_{j=1}^{M_V}d_{kj}f_j(t)\right)+\frac{\mathscr{P}_{SE}^{max}(k)-\mathscr{P}_{SE}^{idle}(k)}{M_{max}(k)}\sum_{j=1}^{M_V}d_{kj}\left(\frac{f_j(t)}{f_j^{max}}\right)^{\alpha}\right\}, \tag{4.35.1}$$

and

$$\mathscr{E}_{net}^{tot}(t)=\sum_{j=1}^{M_V}\left(\sigma_j\left(r_j(t)\right)^{\gamma}+\mathscr{E}_{net}^{setup}(j)\;u_{-1}\left(f_j(t)\right)\right), \tag{4.35.2}$$

where the coefficients $\{d_{kj}\}$ in (4.35.1) account for the performed VM-to-physical server mapping showed in Eq. (4.23.1). So, after introducing the dummy position:

$$\varphi(t)\triangleq\left\{\sum_{j=1}^{M_V}r_j(t)\left(q_j(t)-s_j(t)\right)-q_j(t)f_j(t)\right\}+V\;\beta\;\left(\mathscr{E}_{com}^{tot}(t)+\mathscr{E}_{net}^{tot}(t)\right), \tag{4.36}$$

the consolidation problem in (4.33) may be recast in the following equivalent form:

$$\min_{\{r_j(t),f_j(t),\;j=1,\ldots,M_V\}}\varphi(t), \tag{4.37.1}$$

s.t.:

$$0\leq f_j(t)\leq f_j^{max},\quad j=1,\ldots,M_V, \tag{4.37.2}$$

$$0\leq r_j(t)\leq r_{max}\;u_{-1}\left(f_j(t)\right),\quad j=1,\ldots,M_V, \tag{4.37.3}$$

$$\sum_{j=1}^{M_V}r_j(t)\leq\min\left\{s(t);\;r_{max}\right\}. \tag{4.37.4}$$

So, in (4.37), some explicative remarks. i) the terms proportional to the unit-step function account for the increments/decrements of the static energies that stem from turning ON/OFF physical servers and TCP/IP connections; ii) the $2M_V$ optimization variables in (4.37) are continuous-valued. So, unlike (4.33), (4.37)

is a continuous optimization problem; iii) as a consequence of the (aforementioned) structural properties in i)-iv), the problem formulations in (4.33) and (4.37) are equivalent. Specifically, after computing the solution: $\left\{r_j^*(t),\ f_j^*(t)\right\}$ of (4.37), the corresponding optimal set of the turned ON VMs is evaluated as in:

$$\mathscr{S}^*(t) = \left\{ j:\ f_j^*(t) > 0, \quad j = 1,\ldots,M_V \right\}. \tag{4.38}$$

iv) according to (4.2.2), the constraint in (4.37.3) guarantees that the turned OFF VMs do not receive workload, while the constraint in (4.37.4) is compliant with (4.2.3).

In the presence of the unit-step function, the consolidation problem in (4.37) is not convex and, then, it resists closed-form solution. Hence, as a second step of the proposed solving approach, we proceed to develop a set of iterations, to approach its solution. Therefore, we leverage the primal-dual iteration-based algorithm recently revised in [99]. It builds up a set of parallel gradient-based iterations for the adaptive tracking of both the primal variables, for example, the flow rates and processing frequencies in (4.37) and dual variables, for example the Lagrange multipliers of the considered constraints. Specially, in the proposed dynamic implementation of the $Q^*$ scheduler, primal-dual iterations run: i) at the beginning of each reconfiguration slot, in order to compute the solution $\zeta^*(t)$ of the nonlinear algebraic equation in (4.31); and, ii) at the beginning of each consolidation slot, to iteratively approach the solution of the consolidation problem in (4.37). So, after indicating by $y(t)$ a (scalar) generic resource variable to be updated at slot $t$ and by $\mathscr{L}(t)$ the Lagrangian function of the considered optimization problem, the $n-$th iteration assumes the following general form [99]:

$$y^{(n+1)}(t) = \left[ y^{(n)}(t) - \rho_y^{(n)}(t)\ \nabla_y \mathscr{L}^{(n)}(t) \right]_+, \quad n \geq 0, \ \text{ with: } y^{(0)}(t) = y^*(t-1), \tag{4.39}$$

when $n = 1,2,\ldots,$ is a discrete iteration index which runs at the beginning of the considered slot $t$. Furthermore, according to [99], in the Eq. (4.39) we have: i) $\nabla_y \mathscr{L}^{(n)}(t)$ that is the scalar derivative of the considered Lagrangian function, this is done with respect to the variable $y$ and is evaluated at the resource configuration setting attained at the $n-$th iteration; ii) $\rho_y^{(n)}(t)$ is a suitable $n-$varying adaptive step-size, according to [99], is set as in: $\rho_y^{(n)}(t) = 0.5\ \left(y^{(n)}(t)\right)^2$; and, iii) the projection in (4.39) accounts for the non-negative values of all involved variables. At each slot $t$, the iterations in (4.39) start from the solution $y^*(t-1)$ attained at the previous slot. That allows the scheduler to effectively track over time the solution of the afforded consolidation problem *without* re-computing it from scratch, while *escaping* from the local minim of the objective function in (4.36).

At the beginning of each reconfiguration slot, the root $\zeta^*(t)$ of the nonlinear algebraic equation in (4.31) for the optimal network flow control is computed by running the following iterations in the $n-$index:

$$\zeta^{(n+1)}(t) = \left[\zeta^{(n)}(t) - 0.5\left(\zeta^{(n)}(t)\right)^2 \left(\min\{s(t);\, r_{max}\} - \sum_{j\in\mathscr{J}(t)} r_j^*\left(\zeta^{(n)}(t)\right)\right)\right]_+, \quad n \geq 0, \tag{4.40}$$

where $r_j^*(\cdot)$ is given by Eq. (4.28), with $\zeta^*(t)$ replaced by $\zeta^{(n)}(t)$.

# 4.4 Performance optimality and QoS of the proposed scheduler

In this Section we present the performance of the proposed scheduler of the last Section, where $T_{max} \geq 1$ is the maximum expected delay in multiple of the slot period, between two sequential consolidation slots and the $g_r(0)$ is the positive value of the derivative of the revenue function in (4.12) at $r = 0$. So, after indicate the $\mathscr{E}_{tot}^*(t)$ that is the total energy in (4.10) consumed by the proposed scheduler of the last Section at slot $t$, So:

$$\overline{\chi}^* \triangleq \liminf_{t\to\infty} \left\{ g\left((1/t)\left(\sum_{\tau=0}^{t-1} E\{a^*(\tau)\}\right)\right) - \beta\,(1/t)\left(\sum_{\tau=0}^{t-1} E\{\mathscr{E}_{tot}^*(\tau)\}\right)\right\}, \tag{4.41}$$

be the achieved time-average expected utility, and the following performance results hold (see the Appendix C for the proof).

**Proposition 3** *(Hard QoS guarantees). Under arbitrary input workload (possibly, exceeding the processing capability of the overall data center), the proposed dynamic scheduler guarantees that:*

*a) all the queues are strongly stable at each slot t and for any $V \geq 0$, that is,*

$$z(t) \leq g_r(0)V + w_{max},\ \forall t \geq 0, \tag{4.42.1}$$

$$s(t) \leq g_r(0)V + 2w_{max},\ \forall t \geq 0, \tag{4.42.2}$$

$$q_j(t) \leq g_r(0)V + 2w_{max} + r_{max},\ \forall t \geq 0,\ j = 1,\ldots,M_V; \tag{4.42.3}$$

*b) for any $V \geq 0$, the average utility in* (4.41) *attained by the proposed scheduler is lower bounded as in:,*

$$\overline{\chi}^{*} \geq \overline{\chi}_{opt} - \frac{B_0 T_{max}}{V}, \tag{4.43}$$

*where $B_0$ is the constant in* (4.22.2) *and $\overline{\chi}_{opt}$ is the value of the objective function in* (4.13.1) *under the (possibly unknown) optimal policy $\pi_{opt}$.*

In the following, the above results on the performance of the proposed scheduler merit some main remarks. i) the Eqs. (4.42.1) - (4.42.3) guarantee deterministic upper bounds on all queue backlogs, that, by the Little's Theorem, translate into *hard* QoS guarantees on the total queue delays experienced by the clients of the data center of Fig. 4.1, this because to the access control policy in (4.25.1), that is capable to stabilize the queues, even when the input workload exceeds the processing capability of the data center. The consequence, to avoid queue overflow phenomena, it suffices that the buffering capacity of each queue equates the corresponding upper bound in (4.42); ii) to attain limited queue backlogs, the value assumed by $g_r(0)$ in (4.42) must be finite and (hopefully) as low as possible. Which supports the utilization of log-like revenue functions, while forbids to resort to power-like functions, for example, $g(r) = r^m$, with $0 < m < 1$; iii) the $\overline{\chi}^{*}$-vs.-$\overline{\chi}_{opt}$ gap in (4.43) is limited up to: $(B_0 T_{max})/V$. Nevertheless, it vanishes for $V \to \infty$, but, doing in so, all the bounds in (4.42) on the queue backlogs linearly grow unbounded, confirming this in [81, 114], even in our framework, we have $\mathcal{O}(1/V,\ V)$ utility-vs.-delay tradeoff; iv) at fixed $V$, the optimality gap in (4.43) linearly increases with the inter-consolidation delay $T_{max}$. So, we expect that the performance of the proposed scheduler degrades under low rates of occurrence of the consolidation actions, and we anticipate that our numerical results confirm, indeed, this expectation.

To finally, at vanishing values of $V$, the control actions of the proposed scheduler are given by Eqs. (4.26), (4.32) and (4.34), so that the resulting utility in (4.41) is directly computable as in the follow equation:

$$\overline{\chi}^{*} = -\beta\ T_S(\sum_{k=1}^{N_{SE}} \mathscr{P}_{SE}^{max}(k)) \tag{4.44}$$

## 4.5 Implementation aspects and implementation complexity

In this Section, we address several aspects related to the adaptive and distributed implementation of the proposed scheduler, together with the resulting implementation complexity. In fact, we also point out some possible generalizations of the considered application scenario.

### 4.5.1 Dynamic selection of the consolidation slots and implementation complexity of the proposed scheduler

In agreement with the dynanic nature of the proposed scheduler, as mentioned in [18], we adopt a dynamic approach for the selection of the consolidation slots, that is based on the online evaluation of the per-slot average utilization of the turned ON VMs. Specially, after indicating by $|\mathscr{S}(t)|$ the size of the set $\mathscr{S}(t)$ that is the number of turned ON VMs at slot $t$, the current average utilization:

$$\overline{\mathscr{U}}(t) \triangleq \frac{1}{|\mathscr{S}(t)|}\left(\sum_{j\in\mathscr{S}(t)}\left(\frac{f_j^*(t)}{f_j^{max}}\right)\right), \tag{4.45}$$

the turned ON VMs is computed, together with the exponentially weighted moving average prediction of the next average utilization [126]:

$$\widehat{\overline{\mathscr{U}}}(t+1) = 0.7\,\widehat{\overline{\mathscr{U}}}(t) + 0.3\,\overline{\mathscr{U}}(t), \tag{4.46}$$

Afterwards, the following $l$-out-$m$ decision rule is applied: the next slot $(t+1)$ is flagged as a consolidation slot if at least $l$ out $m$ most recently measured utilization in (4.45) as the predicted one in (4.46) fall out of a target interval: $\zeta \triangleq [Th_L,\ Th_U]$, where $Th_L$ (resp., $Th_U$) is the lower (resp., upper) desired value of the average utilization. At this regard, we also note that it may happen that some VMs are turned OFF, even if they have positive backlogs showed in the Eq. (4.9.2). This phenomenon can verify when the queue backlogs of some VMs are so small that the networking-plus-computing energy cost of leaving these VMs ON is larger than the corresponding revenue increment showed in the Eq. (4.37.1). Where the right action that leads to the maximization of the objective function in (4.14.1), we can anticipate the carried out experimental work, we take a more practical although formally sub-optimal strategy, that evenly splits the aggregate backlogs of the turning OFF VMs over the turned ON VMs.

### 4.5.2 Managing the turning OFF/ON delay

Turning OFF a VM may require some time, specially when the corresponding TCP/IP connection and hosting physical server must be also turned OFF [125, 126]. Let $T_{VM}^{ON}$ be the (dimensionless) fraction of the slot time needed to turn ON a VM and the associated TCP/IP connection. Hence, since the corresponding time available for the data transport is reduced by a fraction equal to $T_{VM}^{ON}$, the constraint in (4.37.4) modifies as in:

$$\sum_{j\in\mathscr{S}(t-1)} r_j(t) + \left(1 - T_{VM}^{ON}\right)\left(\sum_{j\notin\mathscr{S}(t-1)} r_j(t)\right) \leq \min\{s(t);\ r_{max}\}, \tag{4.47}$$

The constraint in the Eq. (4.47) applies when $t$ is a consolidation slot, so that the first (resp., second) summation is over the set of VMs that are turned ON (resp., turned OFF) at the previous slot $(t-1)$.

### 4.5.3 Managing multiple applications and multi-tier data centers

The performance optimality of the proposed scheduler is retained also in the more general cases of multi-application such in [81, 114] and multi-tier such in [31, 32, 112] data centers.

In multi-application data centers, $N_{AP} \geq 2$ applications generate parallel input workloads that are processed by a bank of networked computing platforms [81]. Since each per-application computing platform retains the architecture of Fig. 4.1 and manages own dedicated networking-plus-computing resources, as is mentioned in [81, 114], even in our framework, the overall optimal scheduler reduces to $N_{AP}$ parallel sub-schedulers that operate on a per-application basis and apply the control actions.

A similar conclusion holds also for multi-tier data centers for emerging Web 2.0 applications. Which are composed by the cascade of $N_T \geq 2$ networked computing platforms, that sequentially process the input workload in a pipelined fashion [31, 32, 112]. So, have indicated by: $w^{i+1)}(t) \triangleq \sum_{j\in\mathscr{S}^{i)}(t)} f_j^{i)}(t)$, (resp., $a^{i+1)}(t)$), $i \geq 1$, the workload processed by the $i-$th tier (resp., the workload admitted at the input of the $i+1)-$th tier), the backlog of the input queue of the $i+1)-$th tier evolves:

$$s^{i+1)}(t+1) = \left[s^{i+1)}(t) - \min\left\{r_{max};\ \sum_{j\in\mathscr{S}^{i+1)}(t)} r_j^{i+1)}(t)\right\}\right]_+ + a^{i+1)}(t), \tag{4.48}$$

The corresponding dynamic of the backlog of the $j-$th local queue is given by:

$$q_j^{i+1)}(t+1) = \left[q^{i+1)}(t) - f_j^{i+1)}(t)\right]_+ + r_j^{i+1)}(t),\ j \in \mathscr{S}^{i+1)}(t), \tag{4.49.1}$$

and

$$q_j^{i+1)}(t+1) \equiv q^{i+1)}(t),\ j \notin \mathscr{S}^{i+1)}(t), \tag{4.49.2}$$

where: $\mathscr{S}^{i+1)}(t)$ is the set of turned ON VMs of the $i+1)-$th tier at slot $t$. Therefore, under the common assumption that each tier is equipped with own dedicated networking-plus-computing resources such indicate in [100], each tier is managed by a local scheduler that still applies the control actions.

### 4.5.4 Managing Virtual-to-Physical resource mapping

The task of the virtualization layer of Fig. 4.1 is to map the demands for the per-connection flows in (4.28) and per-VM processing frequencies in (4.29) done by the Middleware layer into adequate channel bandwidths and CPU cycles at the underlying Network and Server layers of Fig. 4.1. These mappings may be performed by equipping the Virtualization layer of Fig. 4.1 by the so-called *mClock* and *SecondNet* mappers in [57, 58], respectively. Specially, Table 4.1 of [57] points out that the *mClock* mapper is capable to guarantee CPU cycles on a per-VM basis by adaptively managing the computing power of the underlying DVFS-enabled of possibly, multi-core physical servers. Consequently so, the output of the *mClock* mapper is the set of binary coefficients $\{d_{kj}\}$ in (4.23.1). Likewise, the *SecondNet* network mapper provides Ethernet-type contention-free links atop any set of TCP-based (possibly, multi-hop) end-to-end connections by resorting to a suitable Port-Switching based Source Routing [58]. Being the output of the set of binary coefficients $\{b_{mj}\}$ in (4.23.2). As pointed out in [57, 58], both of this *mClock* and *SecondNet* mappers may be implemented by exploiting the primitive functionalities provided by (commodity) Xen hypervisors [101].

### 4.5.5 Dynamic profiling of computing-networking energy consumptions

The maximum and idle energies in Eq. (4.7) consumed by each VM may be dynamically profiled on a per-slot basis by equipping the Virtualization layer of Fig. 4.1 with the *Joulemeter* tool [67]. Being this a software tool that provides per-slot and per-VM energy metering functionalities as currently exist in hardware for physical servers. For this purpose, *Joulemeter* uses hypervisor-observable hardware power states to track the VM energy usage on each hardware component more detailed description in [67]. The field trials reported in [67] support the conclusion that, at least when the VMs hosted by each physical server is homogeneous, the per-slot maximum and idle energies wasted by a turned ON VM are proportional to the maximum and idle powers consumed by the hosting physical server, that is showed in Eq. (4.7),

$$\mathscr{E}_{com}^{idle}(j,t) = \sum_{k=1}^{N_{SE}} d_{k,j} \left( \frac{T_S \mathscr{P}_{SE}^{idle}(k)}{\mathscr{M}_k(t)} \right), \; j \in \mathscr{S}(t), \tag{4.50.1}$$

and

$$\mathscr{E}_{com}^{max}(j,t) = \sum_{k=1}^{N_{SE}} d_{k,j} \left( \frac{T_S \mathscr{P}_{SE}^{idle}(k)}{\mathscr{M}_k(t)} \right),\ j \in \mathscr{S}(t) \tag{4.50.2}$$

Being $\mathscr{M}_k(t)$ the number of VMs hosted running atop $k-$th physical server at slot $t$.

So, passing to consider the online profiling of the setup and dynamic energies in (4.4) and (4.5) of the $j-$th TCP/IP connection, we can see that, in emerging broadband data centers [121], each PNIC typically supports a single connection, to reduce the resulting average round-trip-time and, then, saving energy such Eqs. (4.5) and (4.6). So, after indicating by: $\mathscr{P}_{SW}^{setup}(m)$ (resp., $\mathscr{P}_{SW}^{dyn}(m)$) the setup (resp., dynamic) energy consumed by each PNIC hosted by the $m-$th physical switch, the resulting setup and dynamic energies in (4.4) and (4.5) of the $j-$th connection may be profiled online as the corresponding summations of the setup and dynamic energies consumed by the crossed PNICs, that is showed in Eq. (4.23.2),

$$\mathscr{E}_{setup}^{net}(j) = \sum_{m=1}^{N_{SW}} b_{m,j} T_S \mathscr{P}_{SW}^{setup}(m),\ j \in \mathscr{S}(t), \tag{4.51.1}$$

and

$$G(j) = \sum_{m=1}^{N_{SW}} b_{m,j} T_S \mathscr{P}_{SW}^{dyn}(m),\ j \in \mathscr{S}(t), \tag{4.51.2}$$

Finally, we evaluated Eqs. (4.51.1) and (4.51.2), the $\gamma$ exponent in (4.5) is profiled as in:

$$\gamma = \left( \frac{\log\left( \left( \mathscr{E}_{net}^{max}(j) - \mathscr{E}_{net}^{setup}(j) \right) / \sigma_j \right)}{\log(r_{max})} \right),\ j \in \mathscr{S}(t), \tag{4.51.3}$$

where: $\mathscr{E}_{max}^{net}(j)$ is the profiled setup-plus-dynamic energy in (4.4) consumed by the $j-$th connection when it works at $r_j(t) = r_{max}$. We anticipate that, in the experimental work, we use Eqs. (4.50) and (4.51), in order to evaluate the involved energies.

### 4.5.6 Managing discrete processing frequencies

Dynamic scaling of the VM processing frequencies relies on DVFS-enabled physical servers that, in general, do available only a finite set: $\mathscr{F} \triangleq \{\widehat{f}^{(0)},\ \widehat{f}^{(1)}, \ldots, \widehat{f}^{(H-1)}\}$ of $H \geq 2$ discrete CPU processing speeds. To deal with continuous and discrete frequency settings under a unified framework, we borrow the time-sharing approach developed in [78]. For this purpose, we have: $\widehat{f}^{(s)}$, and: $\widehat{f}^{(s+1)}$ be the discrete allowed frequencies that surround the continuous processing frequency $f_j^*(t)$ in Eq. (4.29), that is, $\widehat{f}^{(s)} \leq f_j^* \leq \widehat{f}^{(s+1)}$. Hence, according to [26], $VM(j)$ runs at $\widehat{f}^{(s)}$ (resp.,$\widehat{f}^{(s+1)}$) during a fraction $k$ (resp., $(1-k)$)

of slot $t$. In order to leave unchanged the resulting per-slot energy consumption, we set (see Eq. (4.7)): $k = \left(\left(\widehat{f}^{(s+1)}\right)^{\alpha} - \left(f_j^*\right)^{\alpha}\right) / \left(\left(\widehat{f}^{(s+1)}\right)^{\alpha} - \left(\widehat{f}^{(s)}\right)^{\alpha}\right)$. In practice, the frequency hopping mechanism required for performing the time-sharing operation may be implemented by equipping the physical servers with commodity delta modulators [78].

### 4.5.7 Implementation complexity of the proposed scheduler

Overall, Table 4.2 reports a pseudo-code for the implementation of the proposed joint dynamic scheduler, namely, the $Q^*$ scheduler.

Table 4.2 A pseudo-code of the proposed $Q^*$ scheduler

**The Q* Scheduler (Q*S)**

**for** $t \geq 1$ **do**
  Perform access control through Eq. (4.25.1)
  Update the virtual arrivals through Eq. (4.25.2)
  Update $z(t+1)$ through Eq. (4.15), with $a(t)$ and $c(t)$ replaced by $a^*(t)$ and $c^*(t)$
  Apply the $l$-out-$m$ decision rule and flag $t$ as reconfiguration or consolidation slot;
  **if** $t$ is a consolidation slot, **then**
    Perform resource consolidation by running the iteration in Eq. (4.39);
    Update $\mathscr{S}(t)$ through Eq. (4.38)
  **end if**
  **if** $t$ is a reconfiguration slot, **then**
    Compute $\zeta^*(t)$ through the iteration in (4.40);
    Evaluate Eqs. (4.28) and (4.29);
  **end if**
  Update $s(t+1)$ through Eq. (4.3), with $\{r_j(t)\}$ and $a(t)$ replaced by $\{r_j^*(t)\}$ and $a^*(t)$;
  Update $q_j(t+1)$ through Eq. (4.9), with $\{f_j(t)\}$ and $\{r_j(t)\}$ replaced by $\{f_j^*(t)\}$ and $\{r_j^*(t)\}$;
  Update $\overline{\mathscr{U}}(t)$ and $\widehat{\overline{\mathscr{U}}}(t+1)$ through Eqs. (4.45) and (4.46);
**end for**

Regarding the resulting implementation complexity, three main remarks are in order.

1. since the access control in (4.25.1) and the evaluation of the virtual arrivals in (4.25.2) require the per-slot measurements of the input workload and the backlogs of the input and virtual queues, these

control actions may be implemented by the Access Server and Load Balancer of Fig. 4.1, respectively. Likewise, the iterations in (4.40) for updating the (global) Lagrange multiplier $\zeta^*(.)$ may be carried out by the Load Balancer, that also broadcasts to the turned ON VMs the current backlog of the input queue;

2. each VM may update its own network flow and processing frequency in a distributed way by directly measuring the current backlog of its local queue (see Eqs. (4.28) and (4.29)). Hence, the total per-slot implementation complexity of the $Q^*$ scheduler of Table 4.2 is $\mathcal{O}(M_V)$, but the corresponding per-VM implementation complexity does not depend on the (possibly, very large) number $M_V$ of the VMs hosted by the data center, that is, it is of the order of $\mathcal{O}(1)$, and;

3. since the iterations in (4.39) are carried out at the beginning of each consolidation slot, the iteration index n must run faster than the slot time $T_S$. Hence, the actual time duration: $T_I(s)$ of each n-indexed iteration must be so small so to allow the iterations in (4.39) to converge within a limited fraction of the slot time. At this regard, the formal results of *Theorem* 3.3 of [99] assure the asymptotic convergence of the iterations in (4.39). Furthermore, we have numerically ascertained that, at least in the carried out tests, about 20-25 n-indexed steps suffice, in order to achieve the convergence of (4.39) with a final accuracy below 1%. Hence, in the carried out tests, we pose $T_I = T_S/300$.

Overall, the above remarks lead to the conclusion that the implementation of the $Q^*$ scheduler is *distributed* over the available VMs, while the resulting $per-slot$ execution time is of the order of about $25-30$ iteration periods, *regardless* from the possibly large size of the considered data center.

## 4.6 Experimental work

Being the targeted data center of Fig. 4.1 a generic SaaS cloud, its performance is evaluated by considering a large-scale infrastructure. Therefore, since it is challenging to carry out repeatable large-scale field trials on real-world data centers, we selected numerical simulations as a means to evaluate and compare the performance of the $Q^*$ scheduler, to assure the repeatability of the performed tests. *CloudSim* toolkit [27] is selected as the simulation platform, mainly due to the fact that it natively supports a number of primitives for modeling networked SaaS data centers.

The simulated data center comprises $N_{SE} = 200$ homogeneous servers and each server can host up to: $M_{max} = 10$ virtual machines, being the total number of VMs: $M_V = 2,000$. According to the power profile of commodity *Power Edge R610* servers, the idle (resp., maximum) power consumed by each server is 152 (*Watt*) (resp., 256 (*Watt*)). The topology of the simulated network is the fat-tree one [2], and the set of servers is

Table 4.3 Default setup of the main simulated parameters

| **Parameters** | | |
|---|---|---|
| $T_S = 1(s)$ | $T_{VM}^{ON} = 10^{-2}$ | $P_{SE}^{idle} = 120,\ 152,\ 180\ (Watt)$ |
| $f^{max} = 12.5\ (Mbit/slot)$ | $H = 6$ | $P_{SE}^{max} = 224,\ 265,\ 284\ (Watt)$ |
| $r_{max} = w_{max} = 17.5\ (Gbit/slot)$ | $Th_L = 0.25$ | $P_{SW}^{setup} = 130\ (Watt)$ |
| $N_{SE} = 200,\ N_{SW} = 500$ | $Th_U = 0.75$ | $P_{SW}^{max} = 250\ (Watt)$ |
| $M_V = 2000$ | $\beta = 1.5\ (Joule)^{-1}$ | $M^{max} = 10$ |
| $\alpha = 2,\ \gamma = 1.1$ | $V = 5500$ | $MSS = 1500\ (Byte)$ |

partitioned into 20 equal-size pods. So, the resulting total number of physical switches is: $N_{SW} = 500$ (e.g., 200 edge switches, 200 aggregation switches and 100 core switches), while each 20-port switch is equipped with Ethernet-type PNICs. According to the power profile of *Cisco Nexus 3548* commodity switches, the per-switch setup (resp., maximum) power is 130 $(Watt)$ (resp., 250 $(Watt)$).

Both the ACS and Load Balancer are co-allocated at the access router, they act as the root of the fat-tree network and are connected to each core switch by dedicated links. Minimum-hop static routing is applied to compute the Load Balancer-to-VM shortest routes and each per-connection flow is evenly split over the multiple end-to-end paths done available by the fat-tree network.

Both real-world and synthetic workloads are considered for the tests, with the peak-workload set to 70% of the overall processing capacity of the data center, which is:

$$r_{max} = w_{max} = 0.70 \times M_V \times f^{max},\ (Mbit/slot) \tag{4.52}$$

with $f^{max} = 12.5\ (Mbit/slot)$. So, each simulated VM is equipped with $H = 6$ discrete processing frequencies, the simulated revenue function is the logarithmic one (e.g., $g(r) = \log!\,(1+r)$), while the $3-out-5$ decision rule is implemented, to dynamically select the consolidation slots. Unless otherwise stated, the default values for the upper and lower consolidation thresholds are $Th_L = 0.25$ and $Th_U = 0.75$, the (normalized) delay for turning ON a VM is: $T_{ON}^{VM} = 10^{-2}$, while the default value of the EUE coefficient in (4.13.1) is $\beta = 1.5\ (Joule)^{-1}$.

Finally, the actual average total delay $\overline{T}^*_{tot}$ in multiple of the slot period) of the $Q^*$ scheduler is measured as in:

$$\overline{T}^*_{tot} = \frac{\lim_{t\to\infty}\left(\frac{1}{t}\sum_{\tau=0}^{t-1} s^*(\tau)\right)}{\lim_{t\to\infty}\left(\frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{j=1}^{M_V} r^*_j(\tau)\right)} + \frac{1}{M_V}\sum_{j=1}^{M_V}\left(\frac{\lim_{t\to\infty}\left(\frac{1}{t}\sum_{\tau=0}^{t-1} q^*_j(\tau)\right)}{\lim_{t\to\infty}\left(\frac{1}{t}\sum_{\tau=0}^{t-1} f^*_j(\tau)\right)}\right) \tag{4.53}$$

where, according to the Little's formula, the first ratio is the average delay induced by the input queue of Fig. 4.1, while each term of the second $j-$indexed summation is the average delay of the $j-$th local queue.

### 4.6.1 Performance tests under synthetic workload

The first part of simulations is carried out under a synthetic workload. This means that the input workload $\{w(t)\}$ is an independent and identically distributed random sequence, whose samples are evenly distributed over the set $[0,\ w_{max}]$. Each simulated point is averaged over $10^6$ slots.

The goal of the first set of simulations is to unveil the impact of the Ethernet switching technologies on the energy performance of the $Q^*$ scheduler. For this purpose, Table 4.4 reports the numerically evaluated per-connection profiles of the tested Fast, Giga and 10G Ethernet technologies, while Fig. 4.2 reports the corresponding simulated average per-slot and per-VM energy consumptions. In order to better stress the tested networking technologies and carry out fair comparisons, the energy curves of Fig. 4.2 refer to the case of disabled consolidation.

Table 4.4 Numerically evaluated profiles of the tested switching technologies

| **Per-connection parameters** | **Fast Ethernet** | **Giga Ethernet** | **10G Ethernet** |
|---|---|---|---|
| $\sigma\ (Joule)\times(s/Mbit)^{\gamma})$ | $1.8\times10^{-3}$ | $1.3\times10^{-3}$ | $8.5\times10^{-4}$ |
| $\gamma$ | 1.3 | 1.25 | 1.1 |
| $\mathcal{E}^{setup}_{net}\ (Joule)$ | $1.9\times10^{-4}$ | $1.1\times10^{-4}$ | $8\times10^{-5}$ |
| $\overline{RTT}\ (ms)$ | 4.2 | 0.45 | 0.04 |

An examination of these energy curves leads to two main conclusions. First, all plots of Fig. 4.2 decrease for increasing values of $M_V$ and approach asymptotical minimum values. This is due to the fact that, since both

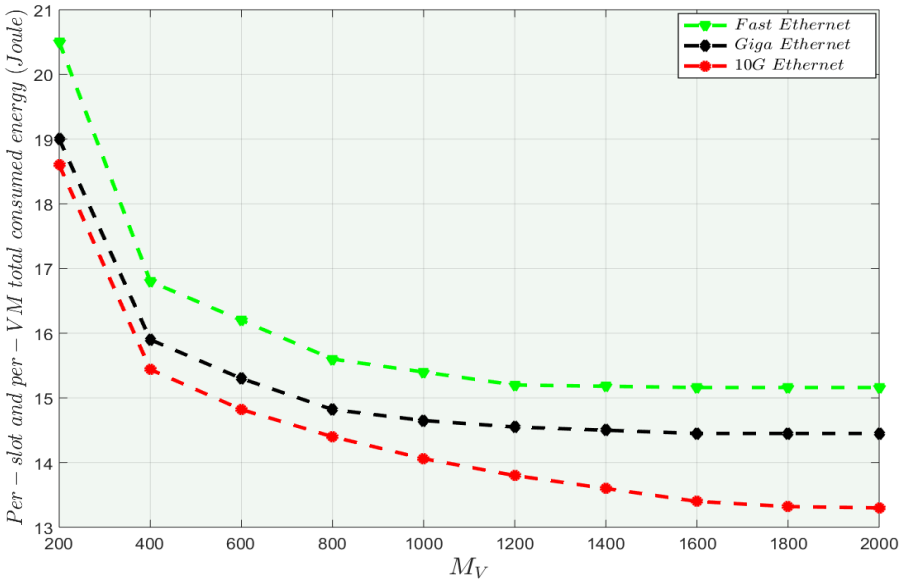



Fig. 4.2 Per-slot and per-VM total energy consumptions under the tested switching technologies for the application scenario at $V = 5500$ and $\beta = 1.5\ (Joule)^{-1}$.

the exponents $\gamma$ and $\alpha$ in Eqs. (4.5) and (4.7) are larger than the unit, the resulting energy functions are convex. So, at least when consolidation is not performed, lower energy consumptions are obtained by splitting the input workload over larger numbers of VMs and TCP/IP connections. Second, since the simulated peak workload in (4.52) is quite large, the energy consumptions of the tested switching technologies are mainly dominated by the corresponding round-trip-times in (4.6). Hence, since the (profiled) average round-trip-time of the 10*G Ethernet* technology is the lowest one (see the last row of Table 4.4), the resulting consumed energies of Fig. 4.2 reduce of about 15% by passing from the Fast Ethernet technology to the 10G one. For this reason, all the numerical results reported in the sequel subsume 10*G Ethernet* technology and, furthermore, they also account for the effect of resource consolidation.

### 4.6.2 Utility-vs.-delay performance

In this Section, we test the performance sensitivity of the $Q^*$ scheduler on the Lyapunov parameter $V$ and EUE coefficient $\beta$ in terms of: i) attained average utility (see Fig. 4.3); ii) average queue delay (see Fig. 4.4); and, iii) fraction of the admitted workload (see Fig. 4.5).

An examination of the plots of Figs. 4.3, 4.4 and 4.5 shows that, at fixed $\beta$, they increase with $V$. In particular, according to the bound in (4.43), Fig. 4.3 points out that the achieved utility approaches a limit

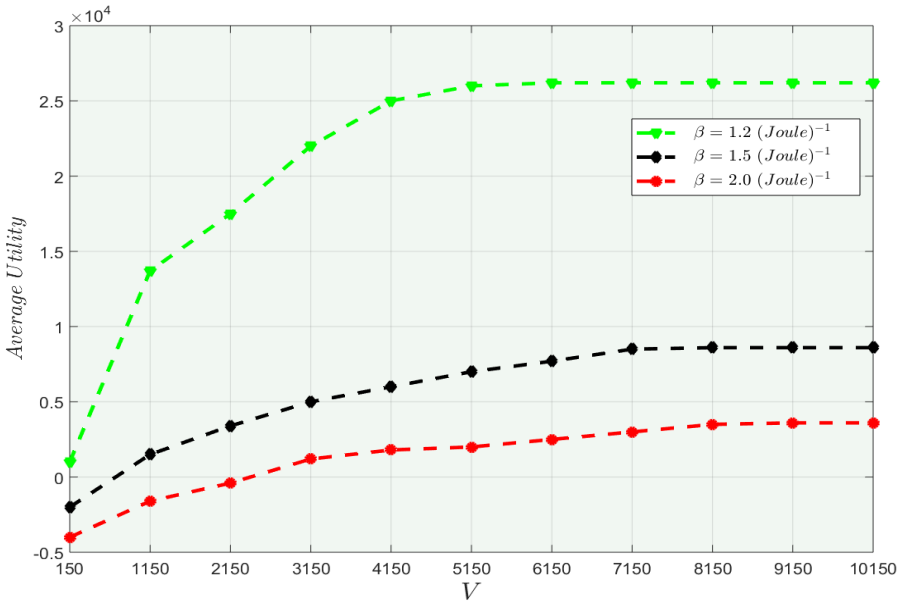



Fig. 4.3 Average utility-vs-$V$ of the $Q^*$ proposed scheduler under the application scenario for different values of $\beta$.

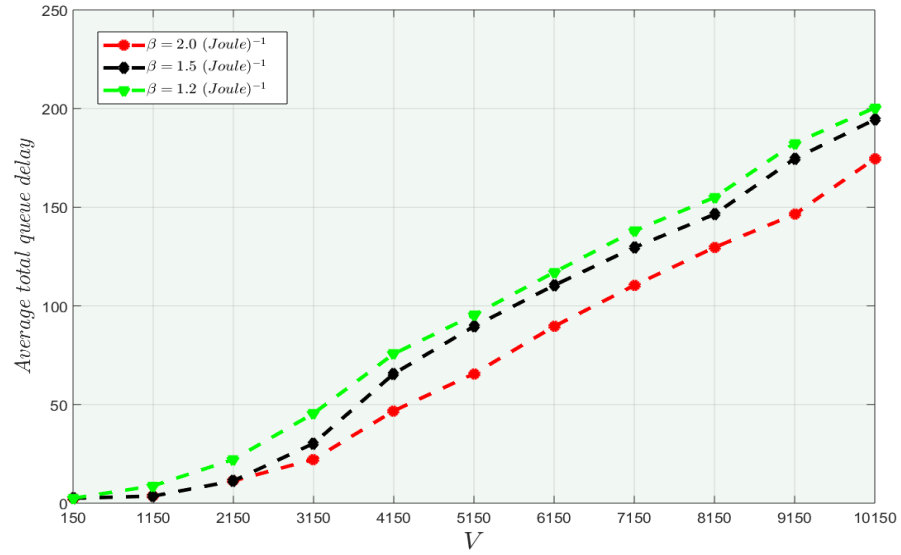



Fig. 4.4 Average queue delay $\overline{T}^*_{tot}(slot)$ of the $Q^*$ scheduler under the application scenario for different values of $\beta$.

value that, in turn, depends on $\beta$. At the same time, the corresponding delay curves of Fig. 4.4 increase almost linearly with $V$, as predicted by the bounds in (4.42). The increasing behaviors of the plots of Fig. 4.5 points

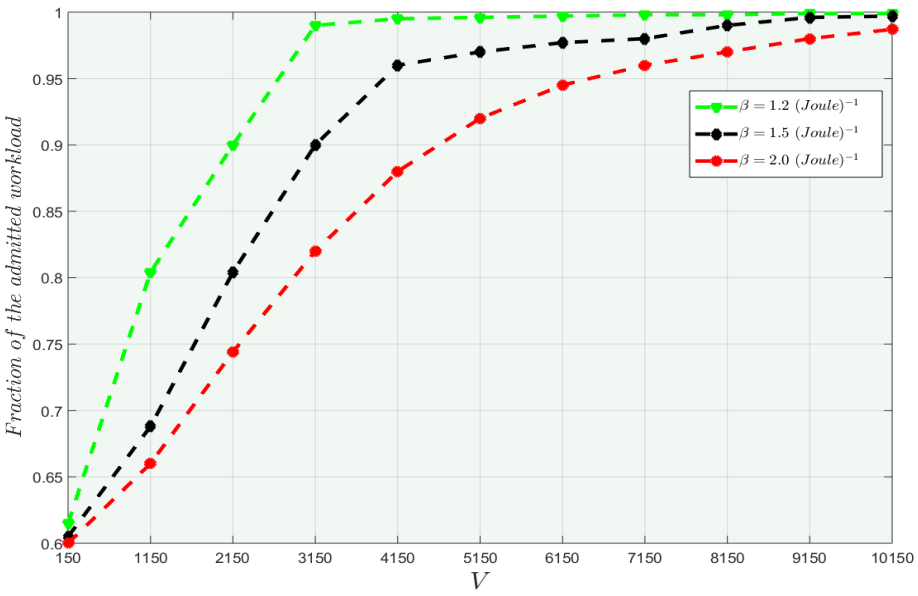



Fig. 4.5 Average fraction of the input workload admitted by the $Q^*$ scheduler under the application scenario for different values of $\beta$.

out that the delay increments in Fig. 4.4 are induced by the corresponding increments of the fraction of the admitted workload. This confirms, indeed, the utility-vs.-delay tradeoff predicted by Proposition 3. However, at fixed $V$, all the curves of Figs. 4.3, 4.4 and 4.5 decrease for increasing values of $\beta$. Intuitively, this is due to the fact that large values of $\beta$ penalize the objective function in (4.14.1). This leads to a reduction of the admitted workload that, in turn, reduces the resulting queue delays.

### 4.6.3 Performance tests under real-world workload

In order to test the performance of the $Q^*$ scheduler under time-correlated workload, we consider the arrival sequence of Fig. 4.6. It reports the real-world trace in Fig. 4.2 of [133] and refers to the I/O workload sampled from four RAID volumes of an enterprise storage cluster in Microsoft. To meet the peak workload in (4.52), each arrival of Fig. 4.6 conveys a workload of 1.1 (*kbit*).

#### Performance tests under real-world workload

The plots of Fig. 4.7 report the simulated per-slot and per-VM average total, networking, and computing energy consumptions of the $Q^*$ scheduler at $\beta = 1.5\ (Joule)^{-1}$ under the real-world trace of Fig. 4.6.

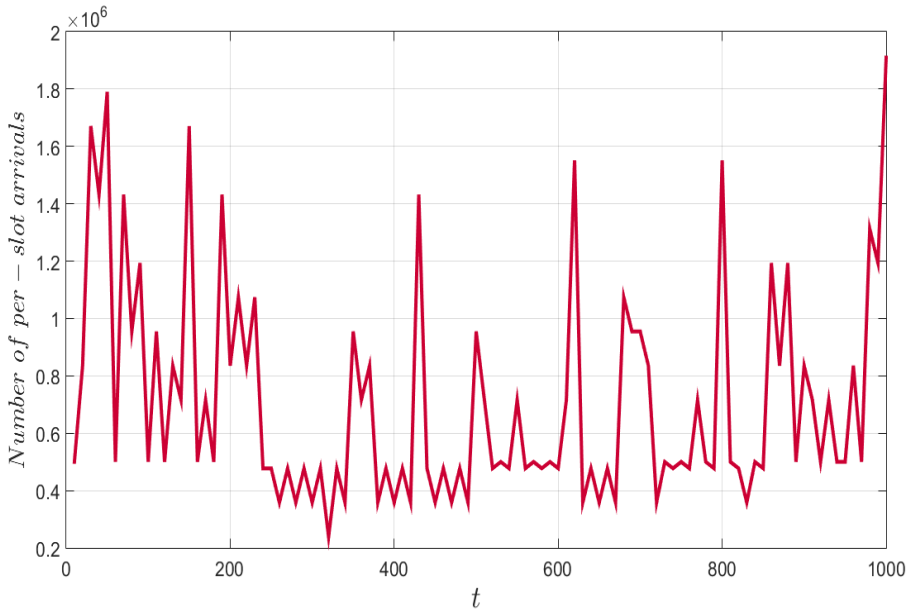



Fig. 4.6 Sampled trace of an I/O arrival trace from an enterprise cluster in Microsoft [133]. The (numerically evaluated) *PMR* and time-correlating coefficient are 2.49 and 0.85, respectly.

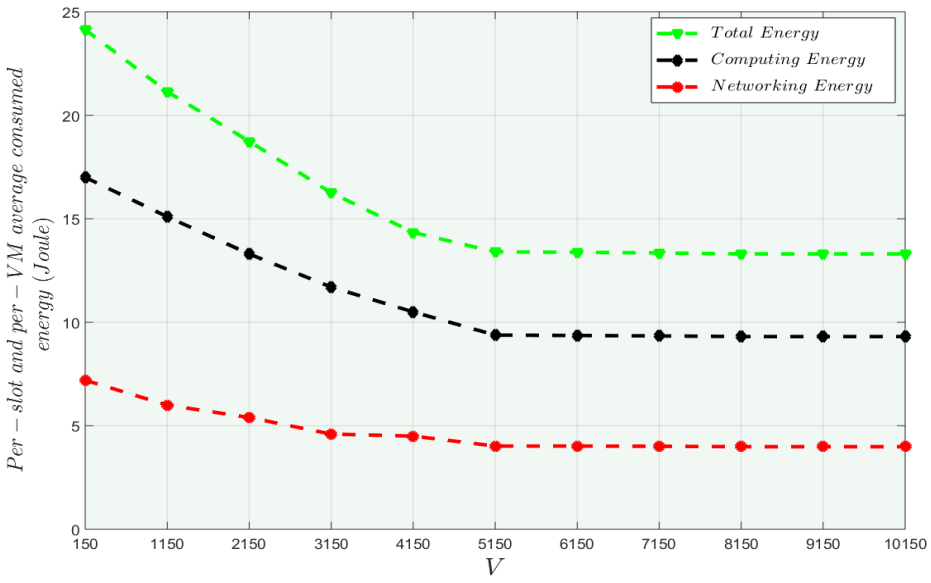



Fig. 4.7 Per-slot and per-VM average total, computing and networking energy consumptions under the application scenario at $\beta = 1.5\ (Joule)^{-1}$

An examination of these plots leads to two main conclusions. First, the drift-minus-utility function in (4.19) becomes more and more dominated by the terms proportional to the consumed networking and computing

energies under large values of $V$. As a consequence, the main target of the scheduler becomes the minimization of these energies, so that the curves of Fig. 4.7 decrease for increasing values of $V$. Second, the contribution of the networking energy to the total consumed energy is around (28-30)% over the overall spectrum of the considered values of $V$. This confirms, indeed, the conclusion drawn by [1], namely, the network component of the total energy consumed by emerging energy-proportional data centers is not longer negligible, specially when the server utilization is below 70%.

### 4.6.4 Consolidation thresholds and tracking capability

The goal of this Section is twofold. First, it aims at investigating the effect of the consolidation thresholds on the energy performance of the $Q^*$ scheduler. Second, it aims at the testing actual capability of the adaptive consolidation algorithm of Table 4.2 to track the time-fluctuations of the input workload. For this purpose, three different event-driven policies are simulated. They trigger consolidation at slot $t$ when at least $l = 3$ out last $m = 5$ values of the utilization factor in (4.45) plus its predicted value in (4.46) fall out of the following target intervals: i) $\mathscr{I}^1 \triangleq [0.1,\ 0.9]$ (*Case* 1); ii) $\mathscr{I}^2 \triangleq [0.25,\ 0.75]$ (*Case* 2); iii) $\mathscr{I}^3 \triangleq [0.35,\ 0.65]$ (*Case* 3). The corresponding simulated energy loss in %) suffered by the iterative consolidation algorithm of Table 4.2 with respect to the exhaustive search-based optimal one is reported in Table 4.5.

Table 4.5 Percent average energy loss of the proposed dynamic consolidation algorithm against the exhaustive-search-based one under the application scenario is considered at $V = 5500$ and $\beta = 1.5\ (Joule)^{-1}$

| | **Case 1** | **Case 2** | **Case 3** |
|---|---|---|---|
| **Syntetic workload** | 1.3% | 1.1% | 0.85% |
| **Real-world workload** | 1.1% | 0.9% | 0.72% |

An examination of the results of Table 4.5 leads to two main conclusions. First, the energy loss suffered by the proposed consolidation algorithm increases by passing from *Case* 3 to *Case* 1 under both the synthetic and real-world workloads. Intuitively, this is due to the fact that, by design, the rate of occurrence of the consolidation events increases when we pass from *Case* 1 to *Case* 3 and this increases, in turn, the chances of convergence of the iterations in Eq. (4.39) to the optimal consolidated configuration. This behavior is, indeed, in agreement with the lower bound in (4.43), that decreases for increasing values of the inter-consolidation

interval $T_{max}$. Second, the energy penalty suffered by the proposed consolidation algorithm remains limited up to 1.3%, even when the time behavior of the input workload is, by design, fully unpredictable, as in the case of the synthetic workload. In the plots of Fig. 4.8 report the time-behaviors of the numbers of VMs and physical servers that are dynamically turned ON by the $Q^*$ scheduler under the real-world workload of Fig. 4.6.

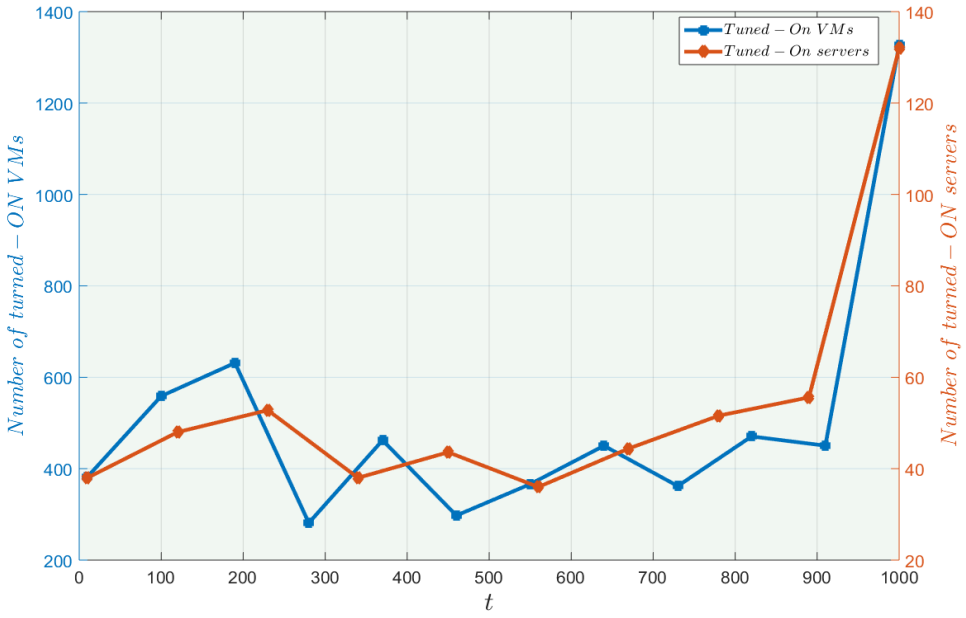



Fig. 4.8 Time-behaviors of the number of turned-ON VMs and physical servers under the application scenario at $V = 5500$, $\beta = 1.5\ (Joule)^{-1}$. The peak-workload is 70% of the maximum processing capacity of the simulated data center and $T_{max} = 1(slot)$ (e.g, consolidation is performed on a per-slot basis).

A comparison of the plots of Fig. 4.8 with the arrival trace of Fig. 4.6 supports the conclusion that the consolidation action performed by the proposed scheduler is capable to track the abrupt (and unpredicted) changes of the input workload, with a delay limited up to $1-2$ slot periods.

### 4.6.5 Energy performance comparisons

Goal of a last set of simulations is to compare the energy performance of the proposed $Q^*$ scheduler $(Q^*\mathscr{S})$ against the corresponding ones of some state-of-the-art schedulers, namely, the StaTic Scheduler ($STS$) [41], the Modified –Best-fit-decreasing Scheduler ($MBS$) [18], the Maximum-Density-consolidation Scheduler ($MDS$) [28], the First-Fit-decreasing Scheduler ($FFS$) [118] and the Exhaustive-Search-based Scheduler ($ESS$) [114]. Shortly, the $STS$ does not perform resource reconfiguration and consolidation actions. It assumes that a fixed

number of TCP/IP connections and VMs constantly run at the maximum flow rates and processing frequencies, in order to constantly provide the networking and computing capacities needed to process the peak workload. Hence, the total energy consumed by the *STS* provides, by design, an upper benchmark. At the opposite side of the performance spectrum, the *ESS* performs Lyapunov-based dynamic resource reconfiguration and consolidation under the following (over-optimistic) assumptions: i) networking energy consumption vanishes; and, ii) server consolidation is carried out offline through the (computationally expensive) exhaustive search. Hence, the (optimistic) energy performance of the *ESS* acts a lower benchmark. The *MBS* and *MDS* perform Lyapunov-based resource reconfiguration by also accounting for the networking energy consumption. However, *MBS* (resp., *MDS*) resorts to the Modified-Best-Fit-Decreasing (resp., Maximum Density Consolidation) greedy heuristic in [9] (resp., in [30]), in order to perform resource consolidation. Hence, by design, the *MBS* implements the minimum-incremental-energy greedy policy, while the target of *MDS* is the per-slot minimization of the number of turned ON physical servers.

In the Table 4.6, reports the simulated average total energy consumptions of the considered schedulers under the (previously defined) three settings of the consolidation thresholds.

Table 4.6 Per-slot and per-VM average total energy consumptions of the tested schedulers under the application scenario at $V$=5500 and $\beta = 1.5\ (Joule)^{-1}$. The previously defined three cases of threshold settings are considered.

**Per-slot and per-VM average total consumed energy** $(Joule)$

| | **STS** | **MDS** | **MBS** | **Q*S** | **ESS** |
|---|---|---|---|---|---|
| **Case 1** | 21.8 | 18.42 | 17.51 | 13.90 | 10.6 |
| **Case 2** | 21.8 | 17.45 | 16.62 | 13.30 | 10.2 |
| **Case 3** | 21.8 | 16.65 | 15.87 | 12.80 | 9.8 |

An examination of Table 4.6 leads to three main conclusions. First, since the *STS* does not perform resource reconfiguration/consolidation, its energy performance does not depend on the setting of the consolidation thresholds. However, the energy consumption of all other schedulers decreases for increasing rate of the occurrence of the consolidation events, that is when we pass from *Case* 1 to *Case* 3. Second, the average energy savings of the proposed $Q^*\mathscr{S}$ over the *STS* is of the order of 57% under *Case* 1 but increases up to about 70% under *Case* 3. The corresponding energy savings of $Q^*\mathscr{S}$ over *MDS* and *MBS* fall into the intervals 29-31 percent and (24-26)%, respectively. Third, as it could be expected, the (optimistic) energy consumptions of the *ESS* are lower than the corresponding ones of the $Q^*\mathscr{S}$ of about (29-31)%. This is compliant with the

(previously reported) networking energy consumption of Fig. 4.7 and average energy loss of Table 4.5, whose aggregate impact on the overall energy consumption is, indeed, around 30%.

# Chapter 5

# Conclusion and Future Research

## 5.1 Conclusions

The IoE enables innovations that improve the human life quality, but it generates unprecedented amounts of data for traditional systems, like Cloud and mobile computing systems. Fog Computing (FC) is designed to overcome these limitations. Indeed, FC enables the seamless integration of edge and cloud resources. In Fog environments, resource management systems should be able to dynamically determine which analytic task is being pushed to which cloud or edge-based resource in order to minimize latency and energy of the devices (e.g., FDC) and to maximize the throughput. In order to cope with these complex issues, it is essential to propose a holistic framework.

The main lesson stemming from the results that reported in this thesis is that a key challenge for coping with the unpredictable large volume of data generated by IoE-based applications is the design of a spectrum of hierarchically-organized networked computing nodes, namely, proximate Fog and remote Cloud data centers. The final goal is the adaptive energy-efficient reconfiguration and orchestration of the virtualized computing-plus-communication resources available at the computing nodes and things devices under real-time constraints on the allowed computing-plus-communication delay and service latency. In order to attain this goal, the performance results detailed in Chapter 3 suggest that three main research directions could be further pursued under the FoE realm. First, stems from the consideration that, in the next years, IoE devices will be equipped with multiple heterogeneous wireless network interface cards. This opens the doors to the design of energy-efficient transport protocols, that rely on the emerging Multipath TCP paradigm [33]. The target should be the increment of the per-connection throughput while limiting the energy overhead induced by the parallel

utilization of multiple radio interfaces. Second, research direction is motivated by the consideration that the native self-organizing feature of the IoE model induces hierarchical relationships among the involved things [45]. This should require the design of new Network-layer communication primitives for IoE-based ecosystems, to implement suitable forms of selective multicast that account for the relative roles of the involved IoE devices [45]. Third, research direction relies on the consideration that the proposed FoE architecture is inherently multi-tier and distributed (showed in the envisioned FoE technological platform and the supported service models), and exploits the inter-networking of local (e.g., IoE devices), proximate (e.g., FNs) and remote (e.g., Cloud nodes) computing entities. On the basis of this consideration, the design of distributed and adaptive resource orchestrators that jointly perform the energy and delay-efficient allocation and scheduling of the offered workload over the full spatial spectrum of the available computing nodes is a still challenging issue as mentioned in [87, 47, 129].

The rapid growth in demand for computational power driven by modern service applications combined with the shift to the Cloud computing model has led to the establishment of large-scale virtualized data centers. Dynamic consolidation of virtual machines (VMs) using online job scheduling, intelligence resource provisioning and switching idle nodes to the sleep mode allows Cloud providers to optimize resource usage and reduce energy consumption. This Dissertation is focused on introducing some state-of-the-art methods to minimize the communication-plus-computing energy which is wasted by processing streams of Big Data under hard real-time constraints on the per-job computing-plus-communication delays. In Chapter 4, we studied and developed a Lyapunov-based dynamic scheduler for the combined: (i) access control; (ii) queue control; (iii) network flow control; (iv) processing frequency scaling, and; (v) joint resource consolidation, in virtualized TCP/IP-based data centers that operate under the SaaS model. Where the overall goal is the dynamic achievement of an optimized energy-vs.-delay tradeoff under the unpredictable time-fluctuations od the input workload. Notable, features of the proposed $Q^*S$ are: (i) its implementation is distributed and scalable. and; (ii) it is capable to track the time-fluctuations of the input workload and provide hard QoS bounds on the attained utility-vs.-delay performance, without requiring any *a priori* information or forecasting of the statistics of the input workload. The carried out performance tests and performance comparisons corroborate the actual effectiveness of the proposed scheduler under both synthetic and real-world workloads.

## 5.2 Future Directions of the Research

Since the FoE model stems from two emerging paradigms FC and IoE, it is in the infancy and, then, is continuously evolving. In this thesis, we have provided an outlook on some main research areas and challenges

that are mainly related to the required computing networked architectures and energy efficiency. However, several other related and tangential research fields, that have been not covered by this thesis, can be identified, such as, since FoE relies on distributed networked computing architectures by design, it is expected that innovative solutions tackling distributed security, trustworthy and thing authentication will be needed, to allow the migration of the FoE paradigm from the theory to the practice. This opens the doors to further work.

The work of the Chapter 4, can be extended in some directions of potential interest. Just as an additional example, emerging Big Data Stream applications [10] present so rapid time-fluctuations of the input workload that the here considered assumption of static virtual-to-physical resource mapping and routing could fall short. Including dynamic virtual-to-physical resource mapping and routing into the considered optimization framework would lead to a mixed-integer non-convex optimization problem, that would involve all layers of the network plus computing protocol stacks. This (seemingly very challenging) cross-layer optimization problem is currently under investigation by the authors.

# Chapter 6

# Accomplishments

The papers that have been published are an important metric to measure the progress and to highlight the accomplishments achieved so far. We then draw the conclusions achieved so far. The papers and documents that have been published based on our working progress are listed as follows:

## Conference Papers:

- **Paola G. Vinueza Naranjo**, Mohammad Shojafar, Leticia Vaca-Cardenas, Claudia Canali, Riccardo Lancellotti, and Enzo Baccarelli, "Big data over smartgrid − a fog computing perspective". 24rd International Conference on Software, Telecommunications and Computer Networks-Softcom.
- **Paola G. Vinueza Naranjo**, Mohammad Shojafar, Ajith Abraham and Enzo Baccarelli, "New stable election-based routing algorithm to preserve aliveness and energy in fog-supported wireless sensor networks". In Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on, pages 002413–002418. IEEE
- Nicola Cordeschi, Danilo Amendola, Mohammad Shojafar, **Paola G. Vinueza Naranjo**, and Enzo Baccarelli, "Memory and memoryless optimal time-window controllers for secondary users in vehicular networks", In Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems,pages 1–7. Society for Computer Simulation International.

## Journal Papers:

- Michele Scarpiniti, Enzo Baccarelli, **Paola G. Vinueza Naranjo** and Aurelio Uncini "Energy Performance of Heuristics and Meta-heuristics for Real-time Joint Resource Scaling and Consolidation in Virtualized Networked Data Centers", The Journal of Supercomputing.
- **Paola G. Vinueza Naranjo**, Enzo Baccarelli, and Michele Scarpiniti, "Design and energy-efficient resource management of virtualized networked Fog architectures for the real-time support of IoT applications", The Journal of Supercomputing.
- Mohammad Shojafar, Zahra Pooranian, **Paola G Vinueza Naranjo**, and Enzo Baccarelli, "FLAPS: bandwidth and delay-efficient distributed data searching in Fog-supported P2P content delivery networks", The Journal of Supercomputing, pages 1–22.
- Enzo Baccarelli, **Paola G. Vinueza Naranjo**, Michele Scarpiniti, Mohammad Shojafar, and Jemal H Abawajy, "Fog of Everything: Energy-efficient Networked Computing Architectures, Research Challenges, and a Case Study", IEEE Access.
- Enzo Baccarelli, **Paola G. Vinueza Naranjo**, Mohammad Shojafar, and Michele Scarpiniti, "Q*:energy and delay-efficient dynamic queue management in tcp/ip virtualized data centers". Computer Communications.
- **Paola G. Vinueza Naranjo**, Mohammad Shojafar, Habib Mostafaei, Zahra Pooranian and Enzo Baccarelli, "P-sep: a prolong stable election routing algorithm for energy-limited heterogeneous fog-supported wireless sensor networks". The Journal of Supercomputing, pages 1–23.

## Magazine:

- Enzo Baccarelli, Michele Scarpiniti, **Paola G. Vinueza Naranjo**, Leticia Vaca-Cardenas, "Fog of Social IoT: when the Fog becomes social". **IEEE Network Magazine**

# References


[1] Abts, D., Marty, M. R., Wells, P. M., Klausler, P., and Liu, H. (2010). Energy proportional datacenter networks. In *ACM SIGARCH Computer Architecture News*, volume 38, pages 338–347. ACM.

[2] Al-Fares, M., Loukissas, A., and Vahdat, A. (2008). A scalable, commodity data center network architecture. In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 63–74. ACM.

[3] Alizadeh, M., Greenberg, A., Maltz, D. A., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., and Sridharan, M. (2010). Data center tcp (dctcp). In *ACM SIGCOMM computer communication review*, volume 40, pages 63–74. ACM.

[4] Amendola, D., Cordeschi, N., and Baccarelli, E. (2016). Bandwidth management VMs Live Migration in Wireless Fog Computing for 5G Networks. In *Cloud Networking (Cloudnet), 2016 5th IEEE International Conference on*, pages 21–26. IEEE.

[5] Arjona Aroca, J., Chatzipapas, A., Fernández Anta, A., and Mancuso, V. (2014). A measurement-based analysis of the energy consumption of data center servers. In *Proceedings of the 5th international conference on Future energy systems*, pages 63–74. ACM.

[6] Asadi, A., Wang, Q., and Mancuso, V. (2014). A survey on device-to-device communication in cellular networks. *IEEE Communications Surveys & Tutorials*, 16(4):1801–1819.

[7] Baccarelli, E., Amendola, D., and Cordeschi, N. (2015). Minimum-energy bandwidth management for QoS live migration of virtual machines. *Computer Networks*, 93:1–22.

[8] Baccarelli, E. and Biagi, M. (2004). Performance and optimized design of space-time codes for MIMO wireless systems with imperfect channel estimates. *IEEE Transactions on Signal Processing*, 52(10):2911–2923.

[9] Baccarelli, E., Biagi, M., Bruno, R., Conti, M., and Gregori, E. (pp.2515-240, 2005). *"Broadband wireless access networks: a roadmap on emerging trends and standards", in: Broadband services: Business models and technologies for community networks*. Wiley.

[10] Baccarelli, E., Cordeschi, N., Mei, A., Panella, M., Shojafar, M., and Stefa, J. (2016a). Energy-efficient dynamic traffic offloading and reconfiguration of networked data centers for big data stream mobile computing: review, challenges, and a case study. *IEEE Network*, 30(2):54–61.

[11] Baccarelli, E., Cusani, R., and Galli, S. (1998). A novel adaptive receiver with enhanced channel tracking capability for TDMA-based mobile radio communications. *IEEE Journal on Selected Areas in Communications*, 16(9):1630–1639.

[12] Baccarelli, E., Naranjo, P. G. V., Scarpiniti, M., Shojafar, M., and Abawajy, J. H. (2017a). Fog of everything: Energy-efficient networked computing architectures, research challenges, and a case study. *IEEE Access*.

[13] Baccarelli, E., Naranjo, P. G. V., Shojafar, M., and Scarpiniti, M. (2017b). Q*: Energy and delay-efficient dynamic queue management in TCP/IP virtualized data centers. *Computer Communications*, 102:89–106.

[14] Baccarelli, E., Naranjo, P. G. V., Shojafar, M., and Scarpiniti, M. (approved to publish 15-Dec-2016b). Q*:energy and delay-efficient dynamic queue management in tcp/ip virtualized data centers. *Computer Communications*.

[15] Ballani, H., Costa, P., Karagiannis, T., and Rowstron, A. (2011). Towards predictable datacenter networks. In *ACM SIGCOMM Computer Communication Review*, volume 41, pages 242–253. ACM.

[16] Bartalos, P. and Blake, M. B. (2012). Green web services: Modeling and estimating power consumption of web services. In *Web Services (ICWS), 2012 IEEE 19th International Conference on*, pages 178–185. IEEE.

[17] Basmadjian, R. and de Meer, H. (2012). Evaluating and modeling power consumption of multi-core processors. In *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet*, page 12.

[18] Beloglazov, A., Abawajy, J., and Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future generation computer systems*, 28(5):755–768.

[19] Beloglazov, A. and Buyya, R. (2010). Energy efficient resource management in virtualized cloud data centers. In *Proceedings of the 2010 10th IEEE/ACM international conference on cluster, cloud and grid computing*, pages 826–831. IEEE Computer Society.

[20] Bernstein, D. (2014). Containers and cloud: From lxc to docker to kubernetes. *IEEE Cloud Computing*, 1(3):81–84.

[21] Bonomi, F. (2011). Connected vehicles, the internet of things, and fog computing. In *The Eighth ACM International Workshop on Vehicular Inter-Networking (VANET), Las Vegas, USA*, pages 13–15.

[22] Bonomi, F., Milito, R., Natarajan, P., and Zhu, J. (2014). Fog computing: A platform for internet of things and analytics. In *Big Data and Internet of Things: A Roadmap for Smart Environments*, pages 169–186. Springer.

[23] Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

[24] Borgia, E. (2014). The internet of things vision: Key features, applications and open issues. *Computer Communications*, 54:1–31.

[25] Bradley, J., Loucks, J., Macaulay, J., and Noronha, A. (2013). Internet of everything (IoE) value index. *White Paper CISCO and/or its affiliates*.

[26] Brown, D. J. and Reams, C. (2010). Toward energy-efficient computing. *Communications of the ACM*, 53(3):50–58.

[27] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1):23–50.

[28] Cao, Z. and Dong, S. (2014). An energy-aware heuristic framework for virtual machine consolidation in cloud computing. *The Journal of Supercomputing*, 69(1):429–451.

[29] Cardosa, M., Singh, A., Pucha, H., and Chandra, A. (2012). Exploiting spatio-temporal tradeoffs for energy-aware mapreduce in the cloud. *IEEE Transactions on Computers*, 61(12):1737–1751.

[30] Çavdar, D., Chen, L. Y., and Alagöz, F. (2014). Green mapreduce for heterogeneous data centers. In *Global Communications Conference (GLOBECOM), 2014 IEEE*, pages 1120–1126. IEEE.

[31] Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., and Doyle, R. P. (2001). Managing energy and server resources in hosting centers. *ACM SIGOPS operating systems review*, 35(5):103–116.

[32] Chen, G., He, W., Liu, J., Nath, S., Rigas, L., Xiao, L., and Zhao, F. (2008). Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *NSDI*, volume 8, pages 337–350.

[33] Chen, Y.-C., Lim, Y.-s., Gibbens, R. J., Nahum, E. M., Khalili, R., and Towsley, D. (2013). A measurement-based study of multipath tcp performance over wireless networks. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 455–468. ACM.

[34] Cheng, N., Lu, N., Zhang, N., Shen, X. S., and Mark, J. W. (2014). Vehicular WiFi offloading: Challenges and solutions. *Vehicular Communications*, 1(1):13–21.

[35] Chiang, Y.-J., Ouyang, Y.-C., and Hsu, C.-H. R. (2015). An efficient green control algorithm in cloud computing for cost optimization. *IEEE Transactions on Cloud Computing*, 3(2):145–155.

[36] CISCO (2015a). Fog computing and the internet of things. [online] https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf.

[37] CISCO (2015b). IoT, from cloud to fog computing. [online] http://blogs.cisco.com/perspectives/iot-from-cloud-to-fog-computing.

[38] CISCO (2016). Cisco solutions. [online] http://www.cisco.com/c/dam/en/us/solutions/collateral/trends/at-a-glance-c45-734964.pdf.

[39] Cordeschi, N., Amendola, D., Shojafar, M., and Baccarelli, E. (2015a). Distributed and adaptive resource management in cloud-assisted cognitive radio vehicular networks with hard reliability guarantees. *Vehicular Communications*, 2(1):1–12.

[40] Cordeschi, N., Amendola, D., Shojafar, M., Naranjo, P. G. V., and Baccarelli, E. (2015b). Memory and memoryless optimal time-window controllers for secondary users in vehicular networks. In *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems*, pages 1–7. Society for Computer Simulation International.

[41] Cordeschi, N., Shojafar, M., Amendola, D., and Baccarelli, E. (2015c). Energy-efficient adaptive networked datacenters for the QoS support of real-time applications. *The Journal of Supercomputing*, 71(2):448–478.

[42] Cordeschi, N., Shojafar, M., and Baccarelli, E. (2013). Energy-saving self-configuring networked data centers. *Computer Networks*, 57(17):3479–3491.

[43] Cugola, G. and Margara, A. (2012). Processing flows of information: From data stream to complex event processing. *ACM Computing Surveys (CSUR)*, 44(3):15.

[44] Dabbagh, M., Hamdaoui, B., Guizani, M., and Rayes, A. (2015). Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment. *IEEE network*, 29(2):56–61.

[45] DaCosta, F. (2013). *Rethinking the Internet of Things: a scalable approach to connecting everything*. Apress.

[46] Das, T. and Sivalingam, K. M. (2013). Tcp improvements for data center networks. In *Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on*, pages 1–10. IEEE.

[47] Dastjerdi, A. V., Gupta, H., Calheiros, R. N., Ghosh, S. K., and Buyya, R. (2016). Fog computing: Principals, architectures, and applications. *arXiv preprint arXiv:1601.02752*.

[48] Divakaran, D. M., Le, T. N., and Gurusamy, M. (2014). An online integrated resource allocator for guaranteed performance in data centers. *IEEE Transactions on Parallel and Distributed Systems*, 25(6):1382–1392.

[49] Dsouza, C., Ahn, G.-J., and Taguinod, M. (2014). Policy-driven security management for fog computing: Preliminary framework and a case study. In *Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on*, pages 16–23. IEEE.

[50] Enokido, T. and Takizawa, M. (2012). An extended power consumption model for distributed applications. In *Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on*, pages 912–919. IEEE.

[51] Faruque, S. (2008). Traffic engineering for multi rate wireless data. In *Electro/Information Technology, 2008. EIT 2008. IEEE International Conference on*, pages 280–283. IEEE.

[52] Gamell, M., Rodero, I., Parashar, M., and Poole, S. (2013). Exploring energy and performance behaviors of data-intensive scientific workflows on systems with deep memory hierarchies. In *High Performance Computing (HiPC), 2013 20th International Conference on*, pages 226–235. IEEE.

[53] GAUCHO– (2016). A green adaptive fog computing and networking architecture. [online] http://www.gaucho.unifi.it.

[54] Ghamkhari, M. and Mohsenian-Rad, H. (2013). Energy and performance management of green data centers: A profit maximization approach. *Smart Grid, IEEE Transactions on*, 4(2):1017–1025.

[55] GreenLSI (2016). Greenlsi research. [online] http://greenlsi.die.upm.es/research/.

[56] Guerraoui, R. and Yabandeh, M. (2010). Independent faults in the cloud. In *Proceedings of the 4th International Workshop on Large Scale Distributed Systems and Middleware*, pages 12–17. ACM.

[57] Gulati, A., Merchant, A., and Varman, P. J. (2010). mclock: handling throughput variability for hypervisor io scheduling. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, pages 437–450. USENIX Association.

[58] Guo, C., Lu, G., Wang, H. J., Yang, S., Kong, C., Sun, P., Wu, W., and Zhang, Y. (2010). Secondnet: a data center network virtualization architecture with bandwidth guarantees. In *Proceedings of the 6th International COnference*, page 15. ACM.

[59] Gupta, H., Dastjerdi, A. V., Ghosh, S. K., and Buyya, R. (2016). iFogSim: A toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments. *arXiv preprint arXiv:1606.02007*.

[60] Hansmann, U., Merk, L., Nicklous, M. S., and Stober, T. (2003). *Pervasive computing: The mobile world*. Springer Science & Business Media.

[61] Hassan, M. A., Xiao, M., Wei, Q., and Chen, S. (2015). Help your mobile applications with fog computing. In *Sensing, Communication, and Networking-Workshops (SECON Workshops), 2015 12th Annual IEEE International Conference on*, pages 1–6. IEEE.

[62] Heller, B., Seetharaman, S., Mahadevan, P., Yiakoumis, Y., Sharma, P., Banerjee, S., and McKeown, N. (2010). Elastictree: Saving energy in data center networks. In *Nsdi*, volume 10, pages 249–264.

[63] Intharawijitr, K., Iida, K., and Koga, H. (2016). Analysis of fog model considering computing and communication latency in 5G cellular networks. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pages 1–4. IEEE.

[64] Iyengar, S. S. and Brooks, R. R. (2012). *Distributed Sensor Networks: Sensor Networking and Applications*. CRC press.

[65] Jhawar, R., Piuri, V., and Santambrogio, M. (2013). Fault tolerance management in cloud computing: A system-level perspective. *IEEE Systems Journal*, 7(2):288–297.

[66] Kai, K., Cong, W., and Tao, L. (2016). Fog computing for vehicular ad-hoc networks: paradigms, scenarios, and issues. *The Journal of China Universities of Posts and Telecommunications*, 23(2):56–96.

[67] Kansal, A., Zhao, F., Liu, J., Kothari, N., and Bhattacharya, A. A. (2010). Virtual machine power metering and provisioning. In *Proceedings of the 1st ACM symposium on Cloud computing*, pages 39–50. ACM.

[68] Kaur, R. and Mahajan, M. (2015). Fault tolerance in cloud computing. *International journal of Science Technology & Management (IJSTM)*.

[69] Kempton, W. and Tomić, J. (2005). Vehicle-to-grid power implementation: From stabilizing the grid to supporting large-scale renewable energy. *Journal of power sources*, 144(1):280–294.

[70] Kiani, A. and Ansari, N. (2015). Toward low-cost workload distribution for integrated green data centers. *IEEE Communications Letters*, 19(1):26–29.

[71] Kiani, A. and Ansari, N. (2016). Profit maximization for geographical dispersed green data centers. *IEEE Transactions on Smart Grid*.

[72] Kilper, D. C., Atkinson, G., Korotky, S. K., Goyal, S., Vetter, P., Suvakovic, D., and Blume, O. (2011). Power trends in communication networks. *IEEE Journal of Selected Topics in Quantum Electronics*, 17(2):275–284.

[73] Kim, K. H., Buyya, R., and Kim, J. (2007). Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters. In *CCGrid*, volume 7, pages 541–548.

[74] Kumar, K. K. M. (2014). Software as a service for efficient cloud computing. *environment*, 7:10.

[75] Kumbhare, A. G., Simmhan, Y., and Prasanna, V. K. (2014). Plasticc: Predictive look-ahead scheduling for continuous dataflows on clouds. In *Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on*, pages 344–353. IEEE.

[76] Kwak, J., Kim, Y., Lee, J., and Chong, S. (2015). DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems. *IEEE Journal on Selected Areas in Communications*, 33(12):2510–2523.

[77] Lakshmi, S. S. (2013). Fault tolerance in cloud computing. *IJESR International Journal Of Engineering Sciences Research*, 4.

[78] Li, K. (2008). Performance analysis of power-aware task scheduling algorithms on multiprocessor computers with dynamic voltage and speed. *IEEE Transactions on Parallel and Distributed Systems*, 19(11):1484–1497.

[79] Li, Y., Sun, L., and Wang, W. (2014). Exploring device-to-device communication for mobile cloud computing. In *Communications (ICC), 2014 IEEE International Conference on*, pages 2239–2244. IEEE.

[80] Lin, M., Wierman, A., Andrew, L. L., and Thereska, E. (2013). Dynamic right-sizing for power-proportional data centers. *IEEE/ACM Transactions on Networking (TON)*, 21(5):1378–1391.

[81] Liu, F., Zhou, Z., Jin, H., Li, B., Li, B., and Jiang, H. (2014). On arbitrating the power-performance tradeoff in saas clouds. *IEEE Transactions on Parallel and Distributed Systems*, 25(10):2648–2658.

[82] Liu, H., Jin, H., Xu, C.-Z., and Liao, X. (2013). Performance and energy modeling for live migration of virtual machines. *Cluster computing*, 16(2):249–264.

[83] Liu, Q., Zhou, S., and Giannakis, G. B. (2004). Cross-layer combining of adaptive modulation and coding with truncated arq over wireless links. *IEEE Transactions on wireless communications*, 3(5):1746–1755.

[84] Lu, T., Chen, M., and Andrew, L. L. (2013). Simple and effective dynamic provisioning for power-proportional data centers. *IEEE Transactions on Parallel and Distributed Systems*, 24(6):1161–1171.

[85] Luan, T. H., Cai, L. X., Chen, J., Shen, X., and Bai, F. (2011). VTube: Towards the media rich city life with autonomous vehicular content distribution. In *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society Conference on*, pages 359–367. IEEE.

[86] Luan, T. H., Ling, X., and Shen, X. S. (2012). Provisioning QoS controlled media access in vehicular to infrastructure communications. *Ad Hoc Networks*, 10(2):231–242.

[87] Mao, Y., You, C., Zhang, J., Huang, K., and Letaief, K. B. (2017). Mobile Edge Computing: Survey and Research Outlook. *arXiv preprint arXiv:1701.01090*.

[88] Mathew, V., Sitaraman, R. K., and Shenoy, P. (2012). Energy-aware load balancing in content delivery networks. In *Proc. of IEEE INFOCOM*, pages 954–962.

[89] Mishra, A., Jain, R., and Durresi, A. (2012). Cloud computing: networking and communication challenges. *IEEE Communications Magazine*, 50(9).

[90] Mitra, D. and Wang, Q. (2005). Stochastic traffic engineering for demand uncertainty and risk-aware network revenue management. *IEEE/ACM Transactions on Networking (TON)*, 13(2):221–233.

[91] Naranjo, P. G. V., Shojafar, M., Abraham, A., and Baccarelli, E. (2016a). A new stable election-based routing algorithm to preserve aliveness and energy in fog-supported wireless sensor networks. In *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on*, pages 002413–002418. IEEE.

[92] Naranjo, P. G. V., Shojafar, M., Mostafaei, H., Pooranian, Z., and Baccarelli, E. (2016b). P-sep: a prolong stable election routing algorithm for energy-limited heterogeneous fog-supported wireless sensor networks. *The Journal of Supercomputing*, pages 1–23.

[93] Neely, M. J. (2010). Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1):1–211.

[94] Neumeyer, L., Robbins, B., Nair, A., and Kesari, A. (2010). S4: Distributed stream computing platform. In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 170–177. IEEE.

[95] Niyato, D., Xiao, L., and Wang, P. (2011). Machine-to-machine communications for home energy management system in smart grid. *Communications Magazine, IEEE*, 49(4):53–59.

[96] Orgerie, A.-C., Lefèvre, L., and Guérin-Lassous, I. (2012). Energy-efficient bandwidth reservation for bulk data transfers in dedicated wired networks. *The Journal of Supercomputing*, 62(3):1139–1166.

[97] Peng, G. (2004). CDN: Content distribution network. *arXiv preprint cs/0411069*.

[98] Peng, M., Yan, S., Zhang, K., and Wang, C. (2015). Fog computing based radio access networks: Issues and challenges. *arXiv preprint arXiv:1506.04233*.

[99] Peng, Q., Walid, A., Hwang, J., and Low, S. H. (2016). Multipath tcp: Analysis, design, and implementation. *IEEE/ACM Transactions on Networking*, 24(1):596–609.

[100] Peter, N. (2015). Fog computing and its real time applications. *International Journal of Emerging Technology and Advanced Engineering*, 5(6).

[101] Portnoy, M. (2012). *Virtualization essentials*. John Wiley & Sons.

[102] Qian, Z., He, Y., Su, C., Wu, Z., Zhu, H., Zhang, T., Zhou, L., Yu, Y., and Zhang, Z. (2013). Timestream: Reliable stream computation in the cloud. In *Proc. of 8th ACM European Conf. on Computer Systems*, pages 1–14.

[103] Satyanarayanan, M., Bahl, P., Caceres, R., and Davies, N. (2009). The case for VM-based cloudlets in mobile computing. *Pervasive Computing, IEEE*, 8(4):14–23.

[104] Satyanarayanan, M., Chen, Z., Ha, K., Hu, W., Richter, W., and Pillai, P. (2014). Cloudlets: at the leading edge of mobile-cloud convergence. In *Mobile Computing, Applications and Services (MobiCASE), 2014 6th International Conference on*, pages 1–9. IEEE.

[105] Shao, Y. S. and Brooks, D. (2013). Energy characterization and instruction-level energy model of intel's xeon phi processor. In *Proceedings of the 2013 International Symposium on Low Power Electronics and Design*, pages 389–394. IEEE Press.

[106] Shojafar, M., Cordeschi, N., Amendola, D., and Baccarelli, E. (2015). Energy-saving adaptive computing and traffic engineering for real-time-service data centers. In *Communication Workshop (ICCW), 2015 IEEE International Conference on*, pages 1800–1806. IEEE.

[107] Shojafar, M., Cordeschi, N., and Baccarelli, E. (2016). Energy-efficient adaptive resource management for real-time vehicular cloud services. *IEEE Transactions on Cloud Computing*, PP:1–14.

[108] Shojafar, M., Pooranian, Z., Naranjo, P. G. V., and Baccarelli, E. (2017). Flaps: bandwidth and delay-efficient distributed data searching in fog-supported p2p content delivery networks. *The Journal of Supercomputing*, pages 1–22.

[109] Soltesz, S., Pötzl, H., Fiuczynski, M. E., Bavier, A., and Peterson, L. (2007). Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors. In *ACM SIGOPS Operating Systems Review*, volume 41, pages 275–287. ACM.

[110] Taleb, T. and Ksentini, A. (2013a). An analytical model for follow me cloud. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 1291–1296. IEEE.

[111] Taleb, T. and Ksentini, A. (2013b). Follow me cloud: interworking federated clouds and distributed mobile networks. *IEEE Network*, 27(5):12–19.

[112] Tang, C., Steinder, M., Spreitzer, M., and Pacifici, G. (2007). A scalable application placement controller for enterprise data centers. In *Proceedings of the 16th international conference on World Wide Web*, pages 331–340. ACM.

[113] Times, T. (2014). The cloud factories: Power, pollution and the internet.

[114] Urgaonkar, R., Kozat, U. C., Igarashi, K., and Neely, M. J. (2010). Dynamic resource allocation and power management in virtualized data centers. In *Network Operations and Management Symposium (NOMS), 2010 IEEE*, pages 479–486. IEEE.

[115] Vamanan, B., Hasan, J., and Vijaykumar, T. (2012). Deadline-aware datacenter tcp (d2tcp). *ACM SIGCOMM Computer Communication Review*, 42(4):115–126.

[116] Vandebroek, S. V. (2016). 1.2 three pillars enabling the internet of everything: Smart everyday objects, information-centric networks, and automated real-time insights. In *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 14–20. IEEE.

[117] Vemula, D., Setz, B., Rao, G. S. V., Gangadharan, G., and Aiello, M. (2017). Metrics for sustainable data centers. *IEEE Transactions on Sustainable Computing*.

[118] Verma, A., Ahuja, P., and Neogi, A. (2008). pmapper: power and migration cost aware application placement in virtualized systems. In *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware*, pages 243–264. Springer-Verlag New York, Inc.

[119] Vinueza Naranjo, P. G., Shojafar, M., Vaca, L., Canali, C., Lancellotti, R., and Baccarelli, E. (2016). Big data over smartgrid – a fog computing perspective. *24rd International Conference on Software, Telecommunications and Computer Networks-Softcom*.

[120] Wang, D., Ren, C., Govindan, S., Sivasubramaniam, A., Urgaonkar, B., Kansal, A., and Vaid, K. (2013). Ace: Abstracting, characterizing and exploiting datacenter power demands. In *Workload Characterization (IISWC), 2013 IEEE International Symposium on*, pages 44–55. IEEE.

[121] Wang, L., Zhang, F., Aroca, J. A., Vasilakos, A. V., Zheng, K., Hou, C., Li, D., and Liu, Z. (2014). Greendcn: A general framework for achieving energy efficiency in data center networks. *IEEE Journal on Selected Areas in Communications*, 32(1):4–15.

[122] Wang, Z., Tolia, N., and Bash, C. (2010). Opportunities and challenges to unify workload, power, and cooling management in data centers. In *Proceedings of the Fifth International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks*, pages 1–6. ACM.

[123] Warneke, D. and Kao, O. (2011). Exploiting dynamic resource allocation for efficient parallel data processing in the cloud. *IEEE transactions on parallel and distributed systems*, 22(6):985–997.

[124] Wirtz, T. and Ge, R. (2011). Improving mapreduce energy efficiency for computation intensive workloads. In *Green Computing Conference and Workshops (IGCC), 2011 International*, pages 1–8. IEEE.

[125] Wu, C. and Buyya, R. (2015). *Cloud Data Centers and Cost Modeling: A complete guide to planning, designing and building a cloud data center*. Morgan Kaufmann.

[126] Xiao, Z., Song, W., and Chen, Q. (2013). Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE transactions on parallel and distributed systems*, 24(6):1107–1117.

[127] Xu, F., Liu, F., Jin, H., and Vasilakos, A. V. (2014a). Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions. *Proceedings of the IEEE*, 102(1):11–31.

[128] Xu, X., Yu, H., and Pei, X. (2014b). A novel resource scheduling approach in container based clouds. In *Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on*, pages 257–264. IEEE.

[129] Yan, S., Peng, M., and Wang, W. (2016). User access mode selection in fog computing based radio access networks. *arXiv preprint arXiv:1602.00766*.

[130] Ye, D., Wu, M., Tang, S., and Yu, R. (2016). Scalable fog computing with service offloading in bus networks. In *Cyber Security and Cloud Computing (CSCloud), 2016 IEEE 3rd International Conference on*, pages 247–251. IEEE.

[131] Zaharia, M., Das, T., Li, H., Shenker, S., and Stoica, I. (2012). Discretized Streams: An Efficient and fault-tolerant model for stream processing on large clusters. *HotCloud*, 12:10–10.

[132] Zhang, Y. and Zhou, Y. (2006). Transparent computing: a new paradigm for pervasive computing. *Ubiquitous Intelligence and Computing*, pages 1–11.

[133] Zhou, Z., Liu, F., Xu, Y., Zou, R., Xu, H., Lui, J. C., and Jin, H. (2013). Carbon-aware load balancing for geo-distributed cloud services. In *2013 IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 232–241. IEEE.

[134] Zhu, J., Chan, D. S., Prabhu, M. S., Natarajan, P., Hu, H., and Bonomi, F. (2013). Improving web sites performance using edge servers in fog computing architecture. In *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*, pages 320–323. IEEE.

# Appendix A

## A.1 Proof of Proposition 1

The minimum of $\psi_0(t)$ in Eq. (4.22.3) under the constraint in (4.1) is attained at $a^*(t) = 0$ (resp., $a^*(t) = w(t)$) when $s(t) \geq z(t)$ (resp., $s(t) < z(t)$). This leads to the all-nothing threshold-type access control in Eq. (4.25.1). In order to derive Eq. (4.25.2), we observe that the maximization of $\psi_1(t)$ in Eq. (4.22.4) under the box constraint in (4.16.2) is a convex optimization problem. As a consequence, its solution may be obtained through the orthogonal projection of the solution: $\widetilde{c}(t) = g_r^{-1}(z(t)/V)$ of the unconstrained minimization of $\psi_1(t)$ onto the admissible set: $[0,\ w_{max}]$. So doing, we obtain Eq. (4.25.2). At this regard, we observe that, after expanding the *max*/*min* operators, Eq. (4.25.2) may be recast in the following equivalent form:

$$c^*(t) = \begin{cases} 0 & for\ z(t) > V g_r(0), \\ g_r^{-1}(z(t)/V), & for\ V g_r(w_{max}) \leq z(t) \leq V g_r(0), \\ w_{max}, & for\ 0 \leq z(t) \leq V g_r(w_{max}), \end{cases} \tag{A.1}$$

where $g_r(w_{max})$ (resp., $g_r(0)$) is the derivative of $g(r)$ at $r = w_{max}$ (resp., $r = 0$).

# Appendix B

## B.1 Proof of Proposition 2

The expression in (4.28) for the optimizes network flows is the solution of the following constrained optimization problem:

$$\min_{\{r_j(t), j\in\mathscr{S}(t)\}} \psi_2(t)),\ s.t.:\ Eqs.\ (4.2.1),\ and\ (4.2.3), \tag{B.1}$$

After retaining the non negativity constraint in (4.2.1) as implicit, the Lagrangian function of (B.1) reads as in:

$$\mathscr{L} = \psi_2(t) + \zeta(t)\left(\left(\sum_{j\in\mathscr{S}(t)} r_j(t)\right) - \min\{s(t);\ r_{max}\}\right\}, \tag{B.2}$$

where $\zeta(t)$ is the non-negative Lagrange multiplier of the constraint in (4.2.3). Hence, after equating to zero the derivative of (B.2) with respect to $r_j(t)$ and solving the resulting algebraic equation, we arrive at Eq. (4.28). Furthermore, the algebraic equation in (4.31) for the evaluation of $\zeta(t)$ follows by equating to zero the derivative of (B.2) with respect to $\zeta(t)$.

Passing to prove the validity of Eq. (4.29), we observe that it is the solution of the following constrained optimization problem:

$$\min_{\{f_j(t), j\in\mathscr{S}(t)\}} \psi_3(t)),\ s.t.:\ Eqs.\ (4.8), \tag{B.3}$$

Since (B.3) is a convex optimization problem, its solution may be obtained by projecting the solution:

$$\widetilde{f}_j^*(t) = \left(\frac{q_j(t)(f_j^{max})^\alpha}{\alpha V\beta(\mathscr{E}_{com}^{max}(j,t) - \mathscr{E}_{com}^{idle}(j,t))}\right)^{1/\alpha-1}. \tag{B.4}$$

of the corresponding unconstrained optimization problem onto the admissible set: $[0,\ f_j^{max}]$. So doing, we directly arrive at Eq. (4.29).

# Appendix C

## C.1 Proof of Proposition 3

*Proof of Part a)* – The proof of the bound in (4.42.1) is by induction over $t$. Hence, since $z(0) = 0$, and, then, the bound is met a $t = 0$, let us assume that Eq. (4.42.1) holds at $t$. We must prove that it also holds at $(t+1)$. For this purpose, in the sequel, we separately consider the three cases of Eq. (A.1).

(i) In the case of $Vg_r(0) < z(t) \leq Vg_r(0) + w_{max}$, we have (see Eq. (A.1)): $c^*(t) = 0$, so that from Eq. (4.15), we have that:

$$z(t+1) \leq z(t) + c^*(t) \equiv z(t) \leq Vg_r(0) + w_{max}. \tag{C.1}$$

(ii) In the case of $Vg_r(w_{max}) \leq z(t) < Vg_r(0)$, the following chain of inequalities is obtained by exploiting Eqs. (4.15) and (A.1), together with the decreasing behavior of $g_r^{-1}(r)$:

$$z(t+1) \leq z(t) + g_r^{-1}(z(t)/V) \leq z(t) + g_r^{-1}(Vg_r(w_{max})/V) \equiv z(t) + w_{max} \leq Vg_r(0) + w_{max}. \tag{C.2}$$

(iii) In the case of $z(t) < Vg_r(w_{max})$, the following developments arise by exploiting Eq. (A.1) and the decreasing behavior of $g_r(r)$:

$$z(t+1) \leq z(t) + c^*(t) \equiv z(t) + w_{max} \leq Vg_r(w_{max}) + w_{max} \leq Vg_r(0) + w_{max}. \tag{C.3}$$

This concludes the proof of the bound in Eq. (4.42.1).

Passing to consider the bound in (4.42.2), its validity is proved by the following chain of inequalities, that follows from the combined exploitation of Eqs. (4.2.3), (4.25.1) and (4.42.1):

$$\begin{aligned} s(t+1) &= max\left\{0;\ s(t)-\min\left\{r_{max};\ \sum_{(j\in\mathscr{S}(t))} r_j^*(t)\right\}\right\}+a^*(t) \leq s(t)+a^*(t) \\ &\leq z(t)+w(t) \leq z(t)+w_{max} \leq V g_r(0)+2w_{max}. \end{aligned} \tag{C.4}$$

Finally, the proof of the bound in (4.42.3) starts from Eq. (4.9.1) and, then, exploits Eqs. (4.2.3), (4.28), (4.42.2) and the non-negativity of $\zeta^*(t)$ for performing the following developments:

$$\begin{aligned} q_j(t+1) &\leq q_j(t)+r_j(t) \leq (s(t)-\zeta^*(t))+r_j^*(t) \leq (s(t)-\zeta^*(t))+(\sum_{k\in\mathscr{S}(t)} r_k^*(t) \leq (s(t)-\zeta^*(t)) \\ &+\min\{s(t);\ r_{max}\} \leq (s(t)-\zeta^*(t))+r_{max} \leq s(t)+r_{max} \leq V g_r(0)+2w_{max}+r_{max}. \end{aligned} \tag{C.5}$$

This completes *the proof of Part a)* of Proposition 3.

*Proof of Part b)* – Let us consider the policy $\pi_0$ that always rejects all the input workload and turns OFF all the physical servers and switches. Since this policy (obviously) meets both constraints in (4.13.2) and (4.13.3), the set $\prod$ of the admissible policies is not empty, so that the constrained problem in (4.13) is feasible. Since the bound in (4.22.1) on the drift-minus-utility function holds under any feasible policy, it must hold under the (possibly unknown) optimal policy $\pi_{opt}$ that solves the constrained problem in (4.13). Hence, after applying this bound under $\pi_{opt}$ and, then, carrying out the unconditional expectations of both sides of Eq. (4.22.1), we arrive at the following lower bound :

$$\frac{1}{t}\sum_{\tau=0}^{t-1} E\{\chi^*(\tau)\} \geq \overline{\chi}_{opt} - ((B_0 T_{max})/V). \tag{C.6}$$

In Eq. (C.6), we have that:

$$\chi^*(\tau) \triangleq g(c^*(\tau)) - \beta \mathscr{E}_{tot}^*(\tau), \tag{C.7}$$

is the value of the utility in (4.14.1) at slot $\tau$ under the dynamic Lyapunov-based policy $\pi^*$ of *proposed joint dynamic scheduler*, while $\overline{\chi}_{opt}$ is the (previously defined) average value achieved by the objective function

in (4.13.1) under the optimal policy $\pi_{opt}$. Since $g(r)$ is concave, an application of the Jensen's inequality allows us to write:

$$\frac{1}{t}\sum_{=0}^{t-1} E\{g(c^*(\tau))\} \leq g\left(\frac{1}{t}(\sum_{\tau=0}^{t-1} E\{c^*(\tau)\})\right). \tag{C.8}$$

Hence, after introducing (C.8) into (C.6) and letting $t \to \infty$, we have that:

$$g(\overline{c}^*) - \beta\ \overline{\mathscr{E}}^*_{tot} \geq \overline{\chi}_{opt} - ((B_0 T_{max})/V), \tag{C.9}$$

where $\overline{c}^*$ (resp., $\mathscr{E}^*_{tot}$) is the time-average expected value of $\{c^*(\tau)\}$ (resp., $\{\mathscr{E}^*_{tot}(\tau)\}$). Since the virtual queue in (4.15) is (strongly) stable (see Eq. (4.42.1)), the constraint in (4.14.2) is guaranteed to hold. Therefore, since $g(.)$ is (strictly) increasing, we have that: $g(\overline{a}^*) \geq g(\overline{c}^*)$, so that Eq. (C.8) leads to:

$$g(\overline{a}^*) - \beta\ \overline{\mathscr{E}}^*_{tot} \geq \overline{\chi}_{opt} - ((B_0 T_{max})/V). \tag{C.10}$$

Eq. (C.10) proves the validity of Eq. (4.43) and concludes the proof of Proposition 3.

# Appendix D

## D.1 Expressions of the gradients in Eq.(4.39)

Due to the presence of the unit-step function, the consolidation problem in (4.37) is non-convex, so that the corresponding box constraints cannot be longer retained as implicit. Therefore, the Lagrangian function associated to (4.37) reads as in:

$$\begin{aligned}\mathscr{L} =&\varphi(t)+\mu_0(t)\left(\left(\sum_{j=1}^{M_V} r_j(t)\right)-\min\{s(t);\ r_{max}\}\right)+\sum_{j=1}^{M_V}\mu_{1j}(t)\left(r_j(t)-r_{max}u_{-1}\left(f_j(t)\right)\right)\\ &-\sum_{j=1}^{M_V}\mu_{2j}(t)r_j(t)+\sum_{j=1}^{M_V}\mu_{3j}(t)\left(f_j(t)-f_j^{max}\right)-\sum_{j=1}^{M_V}\mu_{1j}(t)f_j(t),\end{aligned} \tag{D.1}$$

where $\varphi(t)$ is the objective function in (4.36) and $\{\mu_{1j}\}$ are the non-negative Lagrange multipliers of the constraints in (4.37.2) – (4.37.4). Since the unit-step function does not admit continue first derivative, we replace it by the (quite usual) sigmoidal function:

$$sg_\delta(x)\triangleq\frac{2}{(1+e^{-\{(x/\delta)\})}}-1,\ \ x\geq 0,\ \delta>0. \tag{D.2}$$

It converges to the unit-step function for $\delta\to 0^+$, and admits the following continue first derivative:

$$\dot{sg}_\delta(x)\triangleq\partial sg_\delta(x)/\partial x=(1/2\delta)\left(1-(sg_\delta(x))^2\right),\ x\geq 0 \tag{D.3}$$

Hence, after replacing $u_{-1}(.)$ by $sg_{\delta}(.)$ the partial derivatives of the Lagrangian function in (D.1) with respect to primal variables: $\{r_j(t), f_j(t)\}$ and dual variables: $\{\mu_{ij}(t)\}$ assume the following explicit expressions[1]:

$$\nabla_{r_j}\mathscr{L} = \nabla_{r_j}\varphi + \mu_0 + \mu_{1j} - \mu_{2j}, \tag{D.4}$$

$$\nabla_{f_j}\mathscr{L} = \nabla_{f_j}\varphi - \mu_{1j}\frac{r_{max}}{2\delta}\left(1 - (sg_{\delta}(f_j))^2\right) + \mu_{3j} - \mu_{4j}, \tag{D.5}$$

$$\nabla_{\mu_0}\mathscr{L} = \left(\sum_{j=1}^{M_V} r_j\right) - \min\{s;\ r_{max}\}, \tag{D.6}$$

$$\nabla_{\mu_1 j}\mathscr{L} = r_j - r_{max}\ sg_{\delta}(f_j), \tag{D.7}$$

$$\nabla_{\mu_2 j}\mathscr{L} = -r_j;\ \nabla_{\mu_3 j}\mathscr{L} = f_j - f_j^{max};\ \nabla_{\mu_4 j}\mathscr{L} = -f_j, \tag{D.8}$$

with the following six dummy positions (see Eqs. (4.35.1), (4.35.2) and (4.36)):

$$\nabla_{r_j}\varphi = (q_j - s) + V\ \beta\ \nabla_{r_j}\mathscr{E}_{tot}^{net}, \tag{D.9}$$

$$\nabla_{f_j}\varphi = -q_j + V\ \beta\ \left(\nabla_{f\,j}\mathscr{E}_{com}^{tot} + \nabla_{f\,j}\mathscr{E}_{net}^{tot}\right), \tag{D.10}$$

$$\nabla_{f\,j}\mathscr{E}_{com}^{tot} = T_S\sum_{k=1}^{N_{SE}}\left\{d_{kj}\mathscr{P}_{SE}^{idle}(k)\frac{1}{2\delta}\left(1 - \left(sg_{\delta}\left(\sum_{l=1}^{M_V} d_{kl}f_l\right)\right)^2\right) + \frac{\alpha d_{kj}}{\left(f_j^{max}\right)^{\alpha}}\left(\frac{\mathscr{P}_{SE}^{max}(k) - \mathscr{P}_{SE}^{idle}(k)}{M_{max}(k)}\right)(f_j)^{\alpha-1}\right\}, \tag{D.11}$$

$$\nabla_{r_j}\mathscr{E}_{net}^{tot} = \gamma\sigma_j(r_j)^{\gamma-1}, \tag{D.12}$$

[1] In order to simplify the notation, we omit the $t$ index.

$$\nabla_{f_j} \mathscr{E}_{net}^{tot} = \mathscr{E}_{net}^{setup}(j) \, \frac{1}{2\delta} \, \left(1 - (sg_{\delta}(f_j))^2\right) \tag{D.13}$$

Finally, in the presence of the turning OFF/ON delay $T_{ON}^{VM}$, Eqs. (D.4) and (D.6) modify as follows (see Eq. (4.47)):

$$\nabla_{r_j} \mathscr{L} = \nabla_{r_j} \varphi + \mathscr{J}_j \mu_0 + \mu_{1j} - \mu_{2j}, \tag{D.14}$$

$$\nabla_{\mu_0} \mathscr{L} = \left( \sum_{j \in \mathscr{S}(t-1)} r_j \right) + \left(1 - T_{ON}^{VM}\right) \left( \sum_{j \notin \mathscr{S}(t-1)} r_j \right), \tag{D.15}$$

with the dummy position:

$$\mathscr{J}_j \triangleq \begin{cases} 1, & if \ j \in \mathscr{S}(t-1) \\ 1 - T_{ON}^{VM}, & otherwise. \end{cases} \tag{D.16}$$

This completes the list of the gradients involved by the proposed dynamic consolidation algorithm in *the proposed dynamic approach for the joint adaptive consolidation*