Osmotic Computing: A New Paradigm for Edge/Cloud Integration

Massimo Villari and Maria Fazio
University of Messina

Schahram Dustdar
TU Wien

Omer Rana
Cardiff University

Rajiv Ranjan
Newcastle University

With the promise of potentially unlimited power and scalability, cloud computing (especially infrastructure as a service [IaaS]) supports the deployment of reliable services across several application domains. In the Internet of Things (IoT), cloud solutions can improve the quality of service (QoS), fostering new business opportunities in multiple domains, such as healthcare, finance, traffic management, and disaster management. Available mature solutions, such as Amazon IoT and Google Cloud Dataflow, demonstrate the success of cloud-centric IoT programming models and resource orchestration techniques. However, recent technological advances have disrupted the current centralized cloud computing model, moving cloud resources close to users.

This evolution is mainly required for the adaptation of the cloud paradigm to the IoT phenomenon. The increasing need for supporting interaction between IoT and cloud computing systems has also led to the creation of the edge computing model, which aims to provide processing and storage capacity as an extension of available IoT devices, without needing to move data/processing to a central cloud datacenter (such as Amazon Web Services). This reduces communication delays and the overall size of the data that needs to be migrated across the Internet and public and private datacenters.

Osmotic computing is a new paradigm that's driven by the significant increase in resource capacity/capability at the network edge, along with support for data transfer protocols that enable such resources to interact more seamlessly with datacenter-based services. It aims at highly distributed and federated environments, and enables the automatic deployment of microservices that are composed and interconnected over both edge and cloud infrastructures.

In chemistry, "osmosis" represents the seamless diffusion of molecules from a higher to a lower concentration solution. We believe this process should represent how services can be migrated across datacenters to the network edge. Hence, osmotic computing implies the dynamic management of services and microservices across cloud and edge datacenters, addressing issues related to deployment, networking, and security, thus providing reliable IoT support with specified levels of QoS. Osmotic computing inherits challenges and issues related to elasticity in cloud

2325-6095/16/$33.00 © 2016 IEEE

datacenters, but adds several features due to the heterogeneous nature of edge datacenters and cloud datacenters. Moreover, various stakeholders (cloud providers, edge providers, application providers, and so on) can contribute to the provisioning of IoT service and applications in a federated environment.

Motivations

The emerging availability and varying complexity and types of IoT devices, along with large data volumes that such devices (can potentially) generate, can have a significant impact on our lives, fueling the development of critical next-generation services and applications in a variety of application domains (healthcare, finance, disaster management, and so on). Understanding how data from such devices can be more efficiently analyzed remains a challenge, with existing reliance on large-scale cloud computing systems becoming a bottleneck over time. Transferring large datastreams to such centralized cloud datacenter environments, in a timely and reliable manner, is a key limitation of current cloud-centric IoT programming models (such as Amazon IoT and Google Cloud Dataflow). These existing IoT programming models are considered inappropriate in the context of emerging IoT applications for the principal reason that they assume that the intelligence and resource capacity necessary for data processing reside predominantly in the cloud datacenter.

Thus, to implement complex IoT-oriented computing systems, both cloud and edge resources should be exploited when setting up a hybrid virtual infrastructure, as Figure 1 shows. Cloud and edge datacenters will be managed in a federated environment, where different providers share their resources for IoT services and application support. The burden of data upload toward datacenters leads to inefficient use of communication bandwidth and energy consumption, and a recent study by Cisco (http://goo.gl/M09Ucj) shows that total datacenter traffic will triple by 2019, worsening the situation further. Store-and-process-later approaches, which can save network bandwidth, undermine real-time decision making, which is often a necessary requirement behind IoT applications in the domains of disaster management and healthcare. On the contrary, edge computing aims to lay computing needs on the resource-constrained edge devices, as Figure 1 shows. Edge applications are highly time sensitive (for example, hazard warning applications for environmental conditions such as storms, landslides, and flooding) because they perform immediate analysis of, or response to, collected sensing data.

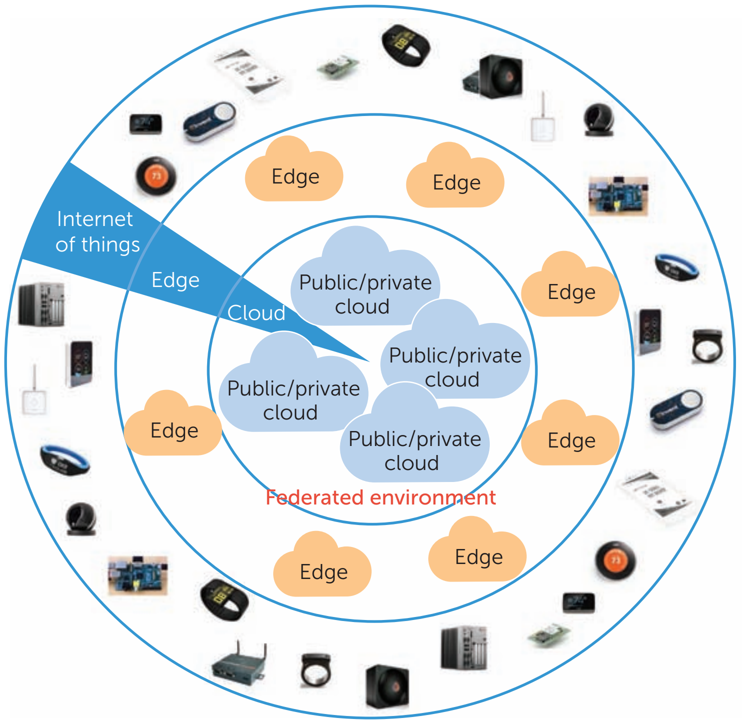

FIGURE 1. Edge and cloud computing for the Internet of Things.

However, even if cloud-based programming models can't support the desired degree of sensitivity for IoT applications, they can strongly increase computation and storage availability whenever necessary. As a result, the prevailing cloud-centric IoT programming model needs to be revised into something that's more adaptable and decentralized to meet the needs of emerging IoT applications.

Osmotic Computing

Osmotic computing aims to decompose applications into microservices and perform dynamic tailoring of microservices in smart environments exploiting resources in edge and cloud infrastructures. Application delivery follows an osmotic behavior where microservices in containers are deployed opportunistically in cloud and edge systems. Like the movement of solvent molecules through a semipermeable membrane into a region of higher solute concentration to equalize the solute concentrations on the two sides of the membrane—that is, osmosis (in the context of chemistry)—in osmotic computing, the dynamic management

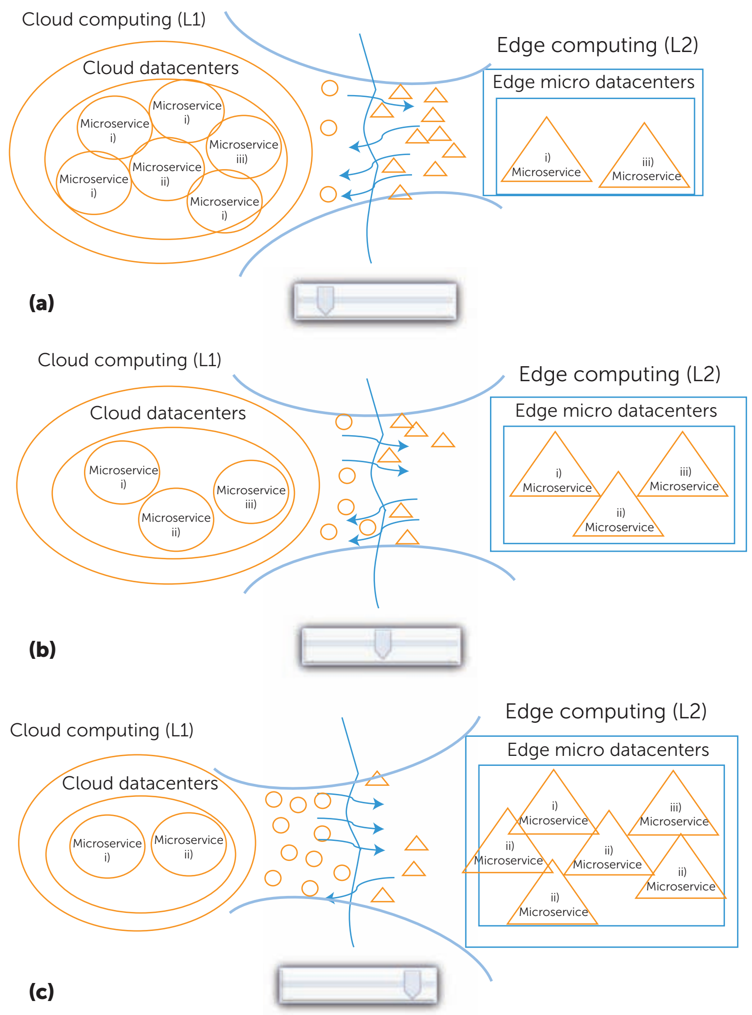

FIGURE 2. Osmotic computing in cloud and edge datacenters: (a) movement of microservices from edge to cloud, (b) optimal balance of microservices across the edge and the cloud, and (c) movement of microservices from cloud to the edge.

of resources in cloud and edge datacenters evolves toward the balanced deployment of microservices satisfying well-defined low-level constrains and high-level needs. However, unlike the chemical osmotic process, osmotic computing allows a tunable configuration of the resource involvement, following resource availability and application requirements (see Figure 2). This is an important distinction—that is, how the difference in configuration (very much infrastructure and application dependent) can determine whether microservices should migrate from cloud to edge or vice versa.

Osmotic computing goes beyond simple elastic management of deployed resources, because deployment strategies are related to requirements of both infrastructure (such as load balancing, reliability, and availability) and applications (such as sensing/actuation capabilities, context awareness, proximity, and QoS) requirements, and they can also change over time. Because of the high heterogeneity of physical resources, the microservice deployment task needs to adapt the virtual environment to the involved hardware equipment. Thus, a bidirectional flow of adapted microservices from cloud to edge (and vice versa) must be managed. Moreover, the migration of microservices in the edge/cloud system implies the need for dynamic and efficient management of virtual network issues to avoid application breakdown or degradation of QoS.

A breakthrough approach to address these issues is to decouple the management of user data and applications from the management of networking and security services. Osmotic computing moves in this direction, providing a flexible infrastructure by offering an automatic and secure microservice deployment solution. Specifically, osmotic computing is based on an innovative application-agnostic approach, exploiting lightweight container-based virtualization technologies (such as Docker and Kubernetes), for the deployment of microservices in heterogeneous edge and cloud datacenters.

Osmotic Ecosystem

As Figure 3 shows, osmotic computing spans two main infrastructure layers. The L1 layer consists of cloud datacenters, which provide several types of services and microservices. For osmotic computing purposes, at this layer, microservices are composed according to users' high-level requirements. The L2 layer identifies the edge computing environment, which includes data capture points and gateway nodes, able to perform operations (average, min, max, filtering, aggregation, and so on) on local data. These devices capture data with a predefined frequency (often dictated by the rate of change of the phenomenon being observed), depending on the device's capacity to record or collect data and on the specific system requirements needing to be satisfied. Devices at L2 can perform various more advanced operations on the raw data collected in the environment, such as encryption of an incoming datastream or encoding/transcoding operations before forwarding this data for subsequent analysis to L1. Due to different properties of systems

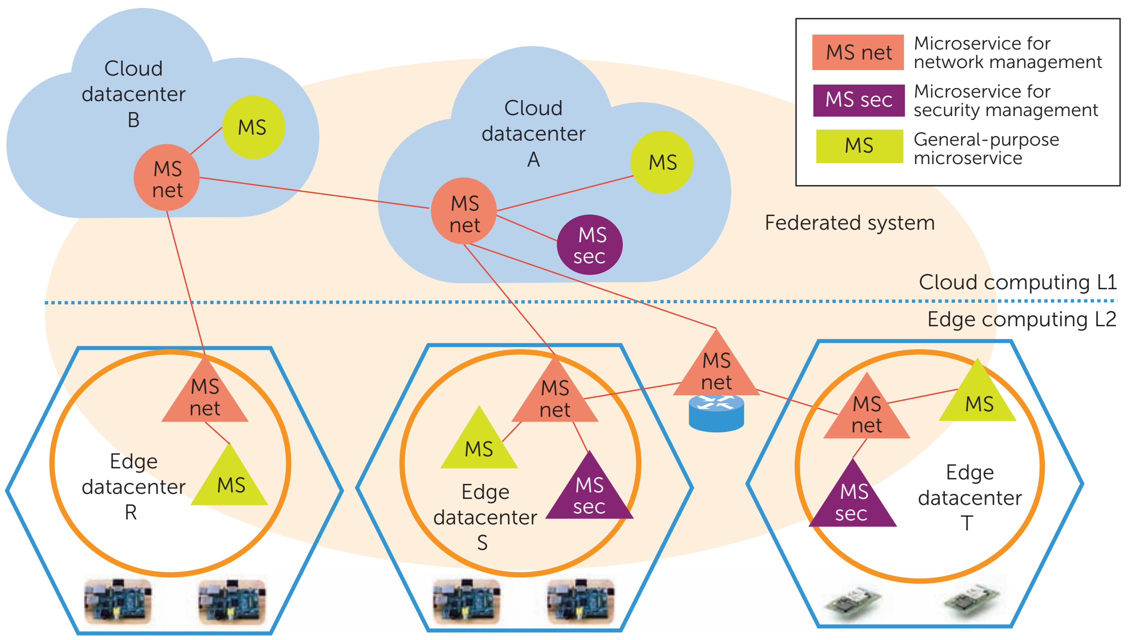

FIGURE 3. A two-layer (L1/L2) federated cloud environment in osmotic computing.

at L1 and L2, we envision a distributed heterogeneous cloud composed of different types of resources located at each of the two layers. Understanding how a microservice hosted on a cloud at L1 can interact and coordinate with a microservice in L2 is a key research challenge in such systems. Each level has its own objective functionalities that influence the types of operations performed. For instance, L2 generally consists of resource-constrained devices (limited battery power, network range, and so on) and network elements, which must perform tasks without overloading available resources.

Datacenters at L1 and microdatacenters at L2 can belong to different providers. However, in a federated scenario, providers can establish relationships and cooperate to share resources and services, thus increasing their business opportunities.[1,2] In this scenario, an osmotic computing framework is application agnostic, offering user applications with runtime environments working in a distributed and secure way. Thus, the main types of microservices that the osmotic computing framework must orchestrate and deploy into cloud and edge infrastructure are general-purpose microservices, which are strictly related to the specific applicative goal; microservices for network management for setting up virtual networks among microservices deployed in the distributed and federated cloud/edge system; and microservices for security management to support cross-platform development of security-enabled microservices.

The microservice provisioning solution can benefit from aggregating different types of resources in the L1 and L2 deployment environments. Understanding how these systems could be aggregated to support application requirements (particularly nonfunctional requirements, such as latency, throughput, security, and budget) remains an important challenge. In particular, the proposed solution follows an advanced approach where microservices are opportunistically deployed in virtual components, called containers. Container-based virtualization technologies (for example, Linux Containers, Docker, Preboot Execution Environment, Google Container, and Amazon Compute Cloud Container) have emerged as a lightweight alternative to hypervisor-based approaches (such as Xen and Microsoft Hyper-V) used in the cloud. A container permits only well-defined software components (such as a database server) to be encapsulated, which leads to significant reduction of deployment overhead and much higher instance density on a single device than a hypervisor. Hence, the new container-based approaches permit deployment of lightweight microservices on resource-constrained and programmable smart devices on the network edge such as gateways (Raspberry Pi and Arduino), network switches (HP OpenFlow), and routers (such as Cisco IOx), but also increase performance in the dynamic management of microservices in cloud datacenters.

Osmotic computing attempts to characterize how composed microservices must be automatically adapted to the deployment sites, considering deployment location and context, since containers are strictly related to the physical host's capabilities. In addition, a decision maker must map microservices to the relevant location. Such a decision is influenced by constraints identified by the specific application and the infrastructure provider, such as utilization of specialist resources (such as a GPU cluster), improving revenue, or reducing management overheads (for example, system administration and/or energy costs). Adaptation of microservices to fluctuations in the computing environment must be performed over time, during the execution of microservices. Therefore, a feedback-driven orchestration is necessary to detect changes in infrastructure performance and QoS metrics.

Research Directions

To make most effective use of the osmotic computing paradigm, we propose the following research directions.

Microservice Configuration

Existing work in the cloud datacenter context supports provider evaluation methods but lacks microservice and edge datacenter configuration support. Multiple approaches have applied optimization[3] and performance measurement techniques[4] for selecting cloud datacenter resources for deploying virtual machine (VM) images according to QoS criteria (throughput, availability, cost, reputation, and so on). While doing so, existing configuration selection techniques have largely ignored the need for VM images and a migration process with transparent decision support and adaptability to custom criteria; hence, for example, they lack flexibility in terms of selection constraints and objectives that can model configurations of edge cloud resources and microservices. However, the configurations and QoS criteria for selecting and ranking microservices and datacenter resources on the network edge differ from VM deployment on cloud datacenters.

In osmotic computing, it's necessary to develop holistic decision-making frameworks that automate configuration selection across microservices and resources in cloud and edge datacenters to meet QoS constraints. To this end, novel decision-making techniques based on multicriteria optimization (for example, genetic algorithms) and multicriteria decision making (for example, analytic network process) techniques should be investigated.

Microservice Networking

Osmotic computing is based on an abstraction of networks that spawn from cloud to edge and vice versa for improving the performance of the communication among microservices.

The network here represents an enabler that allows us to dynamically adjust the overall microservices behavior according to user requirements. Both software-defined networking (SDN) and network function virtualization (NFV)[5] offer useful solutions for supporting in-network/in-transit processing of data (between edge and datacenter) and providing network management abstraction independent of the underlying technology.

Future network management advances in osmotic computing should include the development of an interoperability layer enabling interdomain, federated networks for remote orchestration of heterogeneous edge devices (for example, exploiting SDN and NFV capabilities) accessible through an API. Moreover, the characterization of federated networks in the domain of cloud and edge is missing from the scientific literature. In osmotic computing, a specific metadata ontology for overcoming this issue should be assessed.

Microservice Security

A previous "Blue Skies" column outlined the security challenges and threats of integrating edge computing devices (IoT devices, in transit network devices) with a cloud datacenter.[6] An osmotic computing framework needs a coherent security policy that's supported within both a cloud datacenter and an edge computing environment to enable microservice execution and migration. Ensuring that the same security considerations are observed for a particular microservice across both environments remains a challenge. Such security features will enable self-identification processes that will make the deployment of microservices inside cloud and edge devices easier and more secure, also facilitating the wide adoption of osmotic computing technology. In addition, another objective of osmotic computing is to add security capabilities to the container engine to enable the secure deployment of containers including microservices on IoT devices. More specifically, an osmotic computing framework should allow developers to build chains of trust involving both edge devices and cloud systems by means of a transversal security process.

Edge Computing

Recent efforts to create an open source "IoTCloud" (providing sensors-as-a-service) and middleware-oriented efforts in the European Open IoT project indicate significant interest in this area from the academic community. In the same context, HTTP/REST-based APIs, such as Xively, Open Sen.se, and Think Speak, indicate strong commercial interest, in applications ranging from smart cities to intelligent homes. This also aligns with the fog computing efforts involving cloudlets (from Cisco), which involve small clouds that are geographically scattered across a network and act as small datacenters at the network edge.[7]

The related approach of "mobile offloading" is centered on the need to offload complex and long-running tasks from mobile devices to cloud-based datacenters.[8] To reduce potential battery power consumption and application delay due to intermittent network connectivity, tasks from mobile devices (which generally have lower computation and storage capabilities than a datacenter) are executed at a datacenter, with periodic synchronization between the edge device and the datacenter. An alternative approach (to achieve the same outcome) involves creating a mobile device clone within a datacenter as a VM. Examples include CloneCloud[9] and Moitree.

Our osmotic computing approach suggests the need to combine mobile offloading with datacenter offloading—that is, we offload computation initially carried out within a datacenter to a mobile device. This "reverse" offloading enables computation to be undertaken closer to the phenomenon being measured (overcoming latency and data transfer costs). The osmotic computing approach therefore focuses on understanding the types of microservices that would be more relevant to execute at the edge than within a datacenter environment, and vice versa.

Microservice Workload Contention and Interference Evaluation

Recently, research activities in cloud-based solutions for IoT and edge devices presented container-based virtualization as an alternative to VMs in the cloud.[10] For example, Docker Swarm (https://docs.docker.com/swarm) provides a native orchestration framework (container engine) for multiple Docker deployments, and Kubernetes (http://kubernetes.io/v1.1/docs/user-guide/horizontal-pod-autoscaler.html) is an open source system for automating deployment, operations, and management of clusters of containerized microservices on edge devices and cloud datacenter resources. However, codeployed, containerized microservices leads to workload contention. Workload (generated by containerized microservices) resource consumption and QoS aren't additive, so understanding the nature of their composition is critical to deciding which microservices can be deployed together (that is, can coexist). Recent work has investigated several approaches to minimize the impact of workload interference on the QoS of hosted applications on cloud datacenters.

Hardware-based approaches add complexity to the processor architecture and are difficult to manage over time. Sriram Govindan and his colleagues developed a scheme to quantify the effects of cache contention between consolidated workloads.[11] However, these techniques focus on the contention issues of only one hardware resource type (that is, cache) while ignoring others. Mohammad Nathuji and his colleagues present a control theory-based approach to consolidation that mitigates the effects of cache, memory, and hardware prefetching contention of coexisting workloads.[12] However, they consider only CPU-bound or compute-intensive applications.

To the best of our knowledge, none of the existing academic approaches or the container engines such as Open-Shift Origin, Amazon EC2 Container Service, Docker Swarm, and Kubernetes can automatically detect and handle resource contentions among codeployed microservices across cloud and edge datacenter resources. Hence, research in osmotic computing should focus on novel microservice consolidation techniques that can dynamically detect and resolve resource contention via microservice performance characterization, workload prioritization, and coordinated deployment.

Monitoring

Much of the difficulty in monitoring activities originates from the inherent scale and complexity of the infrastructure considered by the osmotic computing paradigm for deployment of microservices. This infrastructure includes hardware resources in the datacenter (CPU, storage, and network), in-transit network (SDN/NFV-enabled routers and switches), and resources on the network edge (for example, gateways). In such microservice deployment scenarios, detecting problems (for example, in end-to-end request processing latency) and pinpointing the source as one or more culprit components (microservice or datacenter resources or in-transit network) is difficult in such complex systems. The heterogeneity and scale of microservices and infrastructure resources (datacenter, in-transit, and network edge) make it difficult to implement robust monitoring techniques for diagnosing the root cause of QoS degradation.

Monitoring frameworks and techniques used by Amazon Container Service (Amazon CloudWatch) and Kubernetes (Heapster) typically monitor CPU, memory, filesystem, and network usage statistics, so they can't monitor microservice-level QoS metrics (query processing latency of database microserver, throughput of data compression microserver, and so on).

To the best of our knowledge, none of the approaches proposed in academic literature and commercial monitoring tools/frameworks can monitor and instrument data (workload input and QoS metrics, disruptive event) across microservices, cloud datacenter, in-transit network, and edge datacenter, or detect root causes of QoS violations and failures across the infrastructure based on workload and QoS metrics logs. Researchers should investigate scalable methods (based on self-balanced trees) to monitor QoS and security metrics across multiple levels of osmotic computing, including microservices and cloud and edge datacenters.

Microservice Orchestration and Elasticity Control

The runtime orchestration of microservices in a scalable edge/cloud system is a complex research problem due to the difficulty of estimating microservice workload behavior in terms of data volume to be analyzed, data arrival rate, query types, data processing time distributions, query processing time distributions, I/O system behavior, and number of users connecting to different types and mixes of microservices. Without knowing the workload behaviors of microservices, it's difficult to make decisions about the types and scale of cloud and edge datacenter resources to be provisioned to microservices at any given time. Kubernetes and OpenShift Origin (www.openshift.org) offer a microservice container reconfiguration feature, which scales by observing CPU usage ("scaling is agnostic to the workload behavior and QoS targets of a microservice"). Amazon's autoscaling service (https://aws.amazon.com/autoscaling) employs simple threshold-based rules or scheduled actions based on a timetable to regulate infrastructural resources (for example, if the average CPU usage is above 40 percent, use an additional microservice container).

Osmotic computing should extend the traditional notion of runtime control and reconfiguration that only considers resources hosted in cloud datacenters to resources that are deployed and available at the edge. Researchers should investigate machine learning techniques for developing predictive models to forecast workload input and performance metrics across multiple, collocated microservices on cloud and edge datacenter resources. Additionally, intelligent, QoS-aware, and contention-aware resource orchestration algorithms should be developed based on the described models, monitoring systems, and configuration selection techniques.

Whereas significant emphasis has been placed on (mobile) cloud offloading (whereby software applications can be offloaded from a mobile device to a datacenter), there's also a need for the reverse offloading—that is, movement of functionality from the cloud to the edge devices, to counter for latency-sensitive applications and to minimize data sizes that must be transferred over a network. Osmotic computing provides a useful basis for providing a unifying paradigm for this purpose.

References

1. M. Giacobbe et al., "Toward Energy Management in Cloud Federation: A Survey in the Perspective of Future Sustainable and Cost-Saving Strategies," *Computer Networks*, vol. 91, 2015, pp. 438–452.

2. A. Celesti et al., "Characterizing Cloud Federation in IoT," *Proc. 30th Int'l Conf. Advanced Information Networking and Applications Workshops* (WAINA), 2016, pp. 93–98.
3. M.K. Qureshi and Y.N. Patt, "Utility-Based Cache Partitioning: A Low-Overhead, High-Performance, Runtime Mechanism to Partition Shared Caches," *Proc. 39th Ann. IEEE/ACM Int'l Symp. Microarchitecture* (MICRO 06), 2006, pp. 423–432.
4. Q. Zhu and T. Tung, "A Performance Interference Model for Managing Consolidated Workloads in QoS-Aware Clouds," *Proc. 5th IEEE Int'l Conf. Cloud Computing* (CLOUD), 2012, pp. 170–179.
5. S. Jain et al., "B4: Experience with a Globally-Deployed Software Defined WAN," *Proc. ACM SIGCOMM*, 2013, pp. 3–14.
6. D. Puthal et al., "Threats to Networking Cloud and Edge Datacenters in the Internet of Things," *IEEE Cloud Computing*, vol. 3, no. 3, 2016, pp. 64–71.
7. M. Satyanarayanan et al., "Edge Analytics in the Internet of Things," *IEEE Pervasive Computing*, vol. 14, Apr. 2015, pp. 24–31.
8. S. Abolfazli et al., "Cloud-Based Augmentation for Mobile Devices: Motivation, Taxonomies, and Open Challenges," *IEEE Comm. Surveys Tutorials*, vol. 16, First 2014, pp. 337–368.
9. B.-G. Chun et al., "CloneCloud: Elastic Execution between Mobile Device and Cloud," *Proc. 6th Conf. Computer Systems* (EuroSys 11), 2011, pp. 301–314.
10. W. Felter et al., "An Updated Performance Comparison of Virtual Machines and Linux Containers," *Proc. IEEE Int'l Symp. Performance Analysis of Systems and Software* (ISPASS), 2015, pp. 171–172.
11. S. Govindan et al., "Cuanta: Quantifying Effects of Shared On-Chip Resource Interference for Consolidated Virtual Machines," *Proc. 2nd ACM Symp. Cloud Computing* (SOCC 11), 2011, pp. 22:1–22:14.
12. R. Nathuji and A. Kansal, "Q-Clouds: Managing Performance Interference Effects for QoS-Aware Clouds," *Proc. 5th European Conf. Computer Systems* (EuroSys 10), 2010, pp. 237–250.

MASSIMO VILLARI *is an associate professor of computer science at the University of Messina. His research interests include cloud computing, Internet of Things, big data analytics, and security systems. Villari has a PhD in computer engineering from the University of Messina. He's a member of IEEE and IARIA boards. Contact him at mvillari@unime.it.*

MARIA FAZIO *is an assistant researcher of computer science at the University of Messina. Her research interests include distributed systems and wireless communications, especially with regard to the design and development of cloud solutions for IoT services and applications. Fazio has a PhD in advanced technologies for information engineering from the University of Messina. Contact her at mfazio@unime.it.*

SCHAHRAM DUSTDAR *is a full professor of computer science heading the Distributed Systems Group at TU Wien, Austria. His work focuses on Internet technologies. He's an IEEE Fellow, a member of the Academy Europeana, and an ACM Distinguished Scientist. Contact him at dustdar@dsg.tuwien.ac.at or dsg.tuwien.ac.at.*

OMER RANA *is a full professor of performance engineering in the School of Computer Science and Informatics at Cardiff University, where he also leads the Internet of Things (IoT) laboratory. His research interests include performance modelling, simulation, and scalable algorithms for cloud computing, IoT, and edge analytics. Contact him at o.f.rana@cs.cardiff.ac.uk.*

RAJIV RANJAN *is a reader in the School of Computing Science at Newcastle University, UK; chair professor in the School of Computer, Chinese University of Geosciences, Wuhan, China; and a visiting scientist at Data61, CSIRO, Australia. His research interests include grid computing, peer-to-peer networks, cloud computing, Internet of Things, and big data analytics. Ranjan has a PhD in computer science and software engineering from the University of Melbourne (2009). Contact him at raj.ranjan@ncl.ac.uk or http://rajivranjan.net.*