Joint Power Scaling of Processing Resources and Consolidation of Virtual Network Functions

Roberto Bruschi[2], Franco Davoli[1,2], Paolo Lago[1,2] and Jane Frances Pajo[1,2]
[1]DITEN – University of Genoa, Genoa, Italy
[2]CNIT – Research Unit of the University of Genoa, Genoa, Italy
{roberto.bruschi | franco.davoli}@cnit.it, {paolo | jane.pajo}@tnt-lab.unige.it

***Abstract*—With the advent of network "softwarization", Network Functions Virtualization (NFV) is foreseen to provide flexibility and programmability levels that would essentially help in coping with tomorrow's demands. However, energy efficiency and the resulting complexity of network/service management pose serious sustainability and scalability issues that may hinder NFV's advantages. This paper considers these aspects in the context of datacenter networks. We propose an energy-aware resource allocation scheme to manage virtual machines, dedicated to perform certain (virtualized) network functions, among a pool of energy-tunable physical resources (processors/cores). We use online measurements to periodically estimate some statistical features of the offered workloads by considering a fairly general renewal model that captures traffic burstiness and hardware operational settings. Then, resources are dynamically managed by jointly performing power scaling and in-server consolidation according to the actual workload variations. The average power consumption generated by this strategy is evaluated and compared with that of a classical bin-packing consolidation, over processors running always with the highest-performance configuration. Results show that the proposed approach can reduce the average power consumption of the datacenter by up to 10%, suggesting a considerable amount of annual savings.**

I. Introduction

In recent years, the integration of networking paradigms with Information Technology (IT) services has become a trend in the networking community due to their notable potential for supporting tomorrow's demands. Network Functions Virtualization (NFV), for instance, allows for improved time to market of network services, at lesser capital expense, by software implementation of networking functionalities [1]. However, energy efficiency and the resulting complexity of network/service management are foreseen to be major sources of sustainability and scalability issues, and this open problem, if not addressed, will definitely hinder the upcoming network "softwarization" revolution.

With regard to the energy efficiency aspect, several initiatives have focused on the adoption of power management techniques — i.e., Adaptive Rate (AR) and Low Power Idle (LPI), which are already widespread in general-purpose computer systems through the Advanced Configuration and Power Interface (ACPI) specification [2]. Such techniques allow for the reduction of the power requirements of network devices, at the cost of lower performance [3]. Therefore, to fully utilize such capabilities, it is necessary to optimize the trade-off between power saving and network performance. In this respect, efforts have been made in modelling the behaviour of energy-aware network devices, wherein power consumption models based on traffic characteristics are developed using classical principles of renewal theory (e.g., [4] and [5]).

On the other hand, while numerous studies have addressed the virtual machine (VM) consolidation problem in datacenter management, none of them has (completely) considered the AR and LPI capabilities of today's network devices. To the best of the authors' knowledge, the closest attempts only consider making underutilized servers idle (i.e. via live migration) and turning them off (e.g., [6] and [7]).

With this scenario in mind, we propose an energy- and performance-aware consolidation policy that takes into account the aforementioned capabilities and incorporates a power consumption model in the consolidation decision. Towards this end, the classical First-Fit Decreasing (FFD) bin-packing algorithm [8] is considered as a baseline for VM consolidation in this paper, although our approach can be easily adapted to other packing algorithms. Moreover, we consider a set of VMs dedicated to perform certain (virtualized) network functions (VNFs) on incoming traffic streams of various nature. For the sake of simplicity, a one-to-one correspondence between VNFs and VMs is supposed in this work; the rationale behind this is that for a VNF consisting of multiple VMs the overall VNF performance can be derived from the individual VM performances according to the chaining defined by the VNF provider. In any case, the VNF consolidation reduces to a VM consolidation problem.

The VMs are initially placed among a given set of multi-core servers through FFD based on the workloads specified in the Service Level Agreement (SLA). Since such specifications are generally derived from peak workloads, our main goal is to dynamically manage VM consolidation in each server according to actual workload variations, by jointly tuning the ACPI configuration and minimizing the number of active cores, as shown in Fig. 1. Additionally, by limiting the dynamic reconsolidation within a server, costly VM migrations in datacenters will be reduced. In more detail, we want to: (i) estimate statistical features of the offered workloads starting from easily measurable parameters (e.g., utilization, idle and busy times, etc.), (ii) find which subset of VMs should be assigned to each of their cores, and (iii) find the AR and LPI configurations of the active processors. These actions are the basis of a novel energy- and performance-aware consolidation policy, addressing the aforementioned open problem.

978-1-5090-5093-2/16 $31.00 © 2016 IEEE
DOI 10.1109/CloudNet.2016.20

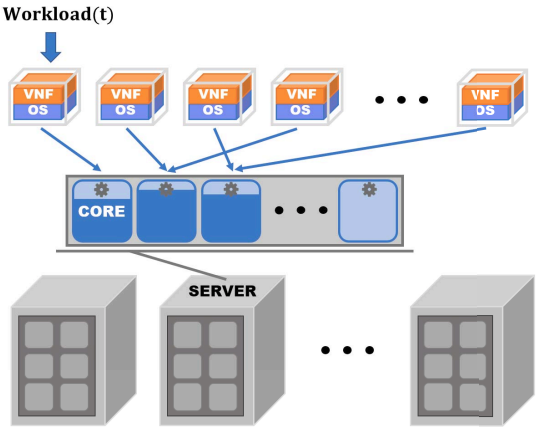

Fig. 1: Conceptual framework.

The remainder of this paper is organized as follows. Sections II and III discuss the proposed model and consolidation policy, respectively. Performance evaluation results are then presented in Section IV. Finally, conclusions are drawn in Section V.

II. The Analytical Model

In this section, we recall a model already proposed and analysed for the single core case [5], which represents the behaviour and performance of a network device with AR and LPI capabilities; then, we use it in a core sharing scenario. For the sake of simplicity, we adopt the ACPI representation of power management primitives.

We consider to have Λ servers, each containing a set of multi-core processors. The VMs are mapped among these servers, with each VM being allocated a certain number of cores, depending on the processing requirements of the applications running inside. In this work, however, we limit the VM workloads to be less than the maximum core capacity for the sake of simplicity, and each core serves a subset of VMs. Suppose that the system has Γ cores; we assume to model each core as a single server queueing system with an average packet service rate $\mu^{(j)}$, $j \in \{1, \ldots, \Gamma\}$. We further assume that packets arrive at a core in batches, with batch arrival rate $\lambda^{(j)}$ and average batch size $\beta^{(j)}$. The server utilization can be expressed as $\rho^{(j)} = \lambda^{(j)}\beta^{(j)}/\mu^{(j)}$, which is assumed to be less than 1 for system stability.

Sub-section II-A outlines how the ACPI specification models AR and LPI functionalities. The main parameters to be considered in a device equipped with AR and LPI capabilities are introduced in Sub-section II-B. Further on, Sub-sections II-C, II-D and II-E present the proposed queueing model, the traffic model and the power consumption model for the energy-aware core, respectively. Finally, the latency aspect of the model is remarked in Sub-section II-F. In order to make the equations in this section more readable, we omit the index $^{(j)}$ of the core.

A. The ACPI Specification

The ACPI specification is an open industrial standard for device configuration and power management in general-purpose computer systems. It models the AR and LPI functionalities through two sets of energy-aware states — the *performance* and *power* states ($P-$ and $C-$ states, respectively).

As regards the $C-$ states, C_0 indicates the processor operating state where instructions are executed, while C_1 through C_X are the LPI states where the processor is sleeping (hence also called sleeping states). The deeper the sleeping state (higher $C-$ state index), the less power is consumed, but the transitions between active and sleeping states require longer times.

Conversely, the $P-$ states tune the performance of the processor's cores by modifying the operating energy point via the working frequency and/or voltage. By means of these states, a core can consume different amounts of power corresponding to different processing performances at the C_0 state. Like the $C-$ states, the higher the $P-$ state index, the less power is consumed, where P_0 is the highest performance state. At a given $P-$ state, the core can transition to deeper $C-$ states when idle.

B. Energy-aware Parameters

For a generic core, let $C_1, \ldots, C_X$ and $P_0, \ldots, P_Y$ be the sets of sleeping and performance states, respectively, and C_0 be the active state in a given P_y configuration.

Each sleeping state is associated to a specific value of idle power consumption Φ_{idle}, as well as different transition times τ_{off} and τ_{on}, required to enter and to wake-up from the sleeping state, respectively. We recall that Φ_{idle} decreases, while τ_{off} and τ_{on} increase with the $C-$ state index. Similarly, each performance state is associated to a specific value of active power consumption Φ_a, as well as an average packet processing capacity μ, both of which decrease with increasing $P-$ state index.

Since transitions between the active state C_0 and C_x are not instantaneous, the server is assumed to have an instantaneous power consumption equal to Φ_t during the τ_{on} and τ_{off} periods. However, the average power consumed during τ_{off} approximates Φ_{idle} [9]; hence, τ_{off} is neglected in the power consumption model. Furthermore, depending on the specific device architecture and implementation, an additional time τ_r is required to set up and to suitably reconfigure packet processing. It is worth noting that while τ_{on} (and τ_{off}) depends on C_x, τ_r depends on P_y, representing a certain number of operations that have to be performed by the server before it starts processing packets. The sum $\tau = \tau_{on} + \tau_r$ defines the setup time of the core; however, we assume $\tau \approx \tau_{on}$ in this work. In summary, the instantaneous power requirements for each state pair (C_x, P_y) can be expressed as:

$$\Phi = \begin{cases} \Phi_a(P_y) & \text{if the core is in the } C_0(P_y) \text{ state,} \\ \Phi_{idle}(C_x) & \text{if the core is in the } C_x \text{ state,} \\ \Phi_t(C_x) & \text{if the core is in } C_x \rightarrow C_0. \end{cases} \quad (1)$$

C. The Queueing Model

It has been established in [10] and [11] that a Batch Markov Arrival Process (BMAP) can effectively estimate the network traffic behaviour. Hence, we assume that the traffic incoming to the i–*th* VM is represented as a BMAP with batch arrival rate λ_i and average batch size β_i, $i \in \{1, \ldots, |\mathcal{I}|\}$, where $|\mathcal{I}|$ is the cardinality of the set $\mathcal{I}$ of VMs served by a core.

Suppose that each VM has its own queue and is allocated a virtual CPU (vCPU) that is subject to scheduling. Although this suggests a multiple-queue single-server queueing model for the core, an equivalent single-queue model can be easily derived considering the aggregate workload, as described in [12].

Based on the assumptions described up to this point, the model we propose corresponds to a M^X/G/1/SET queueing system [13]. For each core, packets arrive in batches, at exponentially distributed inter-arrival times with arrival rate λ and are served by a single server at an average service rate μ, with generally distributed service times. When the system becomes empty, the server is put to sleep. In order to take into account the case where an incoming packet experiences a wake-up delay due to sleeping-active transitions, the model includes a deterministic setup time τ, only after which service can begin.

Using classical principles of renewal theory, we can identify independent and identically distributed (iid) "cycles" of the form:

$$T_R^{(n)} = T_I^{(n)} + \left[T_B^{(n)} + \tau\right] \tag{2}$$

where $T_I^{(n)}$ and $\left[T_B^{(n)} + \tau\right]$ are the n-th idle and delay busy (i.e., actual busy period plus the setup) periods of a core, respectively. As presented in [14], the average idle and actual busy periods are given by:

$$T_I = E\{T_I^{(n)}\} = \frac{1}{\lambda} \tag{3}$$

$$T_B = E\{T_B^{(n)}\} = \frac{\rho(1+\lambda\tau)}{\lambda(1-\rho)} \tag{4}$$

Then, the average duration of a renewal cycle can be obtained as:

$$T_R = T_I + T_B + \tau = \frac{1+\lambda\tau}{\lambda(1-\rho)} \tag{5}$$

D. The Traffic Model

Since the sum of independent Poisson processes is a Poisson process with rate given by the sum of the individual rates, then the batch arrival rate at the core is given by:

$$\lambda = \sum_{i=1}^{|\mathcal{I}|} \lambda_i \tag{6}$$

which can also be easily measured by inverting (3). Moreover, the average batch size at the core is obtained as:

$$\beta = \frac{1}{\lambda} \sum_{i=1}^{|\mathcal{I}|} \lambda_i \beta_i \tag{7}$$

The summation in the right-hand side is the average offered workload $\lambda\beta$ at the core, which can also be easily obtained from measurements of the core utilization and service rate as ρ/μ.

E. The Power Consumption Model

Using Eqs. (3)−(5), the average power consumption of a core in a renewal cycle can be expressed as:

$$\begin{aligned}\tilde{\Phi} &= \frac{T_B}{T_R}\Phi_a + \frac{\tau}{T_R}\Phi_t + \frac{T_I}{T_R}\Phi_{idle} \\ &= \rho\Phi_a + \frac{\lambda\tau(1-\rho)}{1+\lambda\tau}\Phi_t + \frac{(1-\rho)}{1+\lambda\tau}\Phi_{idle}\end{aligned} \tag{8}$$

Given the offered workload at a core, this model can be used to determine which ACPI configuration gives the minimum power consumption.

F. The Performance Model

As we will see in the following section, in the proposed consolidation procedure we will only impose a limit on the maximum utilization for each core to avoid unacceptable performance degradation. In this first implementation of the consolidation strategy adopted, we prefer to concentrate on the evaluation of the potential power saving, by keeping the enforcement of performance constraints in the simplest possible form. However, the performance of a VNF can be better controlled by considering the average system latency W. For a M^X/G/1/SET queueing model, W depends on the second moments of the batch size and service time, as indicated in [13]. Though the usage of the delay to impose performance requirements in the problem will be left for future work, nevertheless it is worth mentioning how the terms required in such scenario could be obtained. The expression of W derived in [13] (and also exploited in [4], [5]) requires the knowledge of the second moments of the batch size and of the packet service time. The latter can be directly obtained from measurements; the former is analytically related to the second moment of the busy period $T_B^{(n)}$, which can be also easily obtained from measurements.

III. Energy- and Performance-aware Consolidation

In this section, we introduce a novel policy that dynamically manages the VM consolidation among cores in a server according to actual workload variations. Suppose a datacenter performs a global FFD bin-packing consolidation every time period T. Considering that in typical consolidation policies significant variations in the total number of active servers and VMs might occur on a much longer timescale than the dynamics of power scaling policies, we can suppose the time interval T to be in the order of tens of minutes or even hours. Similarly, the number of VMs in each server can be considered relatively stationary over shorter time intervals ΔT. Then, we assume that the number of VMs in the system does not change in this interval (i.e. no arrival and/or departure of VMs — if any occur, they will be accounted for with some delay in the

successive interval). We propose a consolidation policy that can be performed in each active server at every sub-interval ΔT during T.

Firstly, we adopt a slightly looser provisioning than the core network capacity planning rule-of-thumb, constraining the offered workload to be less than 80% of the maximum capacity, instead of 50% [15]. This still provides a safety headroom for any fluctuation in the VMs' workload, ensuring that the required Quality of Service (QoS) constraints are not violated [16]. Additionally, we limit the choices of sleeping states $\{C_x\}$ based on their sleeping times $\tau_{off}(C_x)$ and the batch arrival rate λ at the core such that $\tau_{off}(C_x) < 1/\lambda$, in order to have a relatively low probability of arrivals during $C_0 \rightarrow C_x$ transitions.

The scheme includes energy- and performance-aware workload classification rules that define the most energy efficient configuration to be applied to the serving core. Specifically, given a certain aggregate workload characterized by λ and an average batch size β, as in (6) and (7), we evaluate the quantity $\tilde{\Phi}(C_x, P_y)$ for all possible pairs of (C_x, P_y), ensuring at the same time the satisfaction of the aforementioned constraints, in order to find the ACPI configuration that yields the minimum average power consumption. It is worth noting that this computation can be performed offline for a whole range of values of λ and $\lambda\beta$ for a specific CPU architecture, giving rise to regions in the $(\lambda, \lambda\beta)$ space that correspond to the most suitable configuration for the points in the region (a specific example of such regions will be provided in Section IV). When two or more configurations give the minimum value, the one with the better performance (i.e., greater capacity and/or lighter sleeping state) is selected.

Taking into account that some states are set on a per-processor basis [17], for the sake of simplicity in the power management of the server farm, we suppose that all cores in a processor package have the same configuration. Hence, if the optimum configuration varies among the cores in a package, a suboptimal solution is to set them according to the one with the highest requirement.

The workload in each VM can be monitored via its vCPU's usage, and we exploit these data to perform a dynamic VM consolidation inside the server. Now, let Γ_k indicate the set of cores of the k–th server, and $\mathcal{I}_k = \bigcup_{j \in \Gamma_k} \mathcal{I}_k^{(j)}$ the set of VMs they serve (where $\mathcal{I}_k^{(j)}$ is the set of VMs served by the j–th core). Based on the actual VM workload in a sub-interval ΔT, the FFD algorithm is jointly performed with power scaling to find the minimum core capacity required to serve $\mathcal{I}_k$, obtaining $|\Gamma_k|$ updated groupings of VMs, $\{\hat{\mathcal{I}}_k^{(j)}, j \in \Gamma_k\}$. At this point, some groups may be empty — this only means that some cores will be idle. Considering the new aggregate workload of each group, the core classification rules are then applied to determine their optimum configuration. The groups are ordered with decreasing performance requirement and each one is allocated a core accordingly. Finally, the configuration of each processor package is set according to the core with the highest requirement and all idle packages are put to a deep sleep. The proposed energy- and performance-aware consolidation policy is summarized in Algorithm 1.

Algorithm 1 In–Server Consolidation Policy

$\mathcal{I}_k$, λ_i, β_i, $\forall i \in \{1, \ldots, |\mathcal{I}_k|\}$
$\mathcal{P} \leftarrow \{\}$
for $y = 0$ to Y **do**
  $\mu^{(j)} \leftarrow \mu(P_{Y-y})$, $\forall j \in \Gamma_k$
  perform the FFD algorithm to obtain $\hat{\mathcal{I}}_k^{(j)}$, $\forall j \in \Gamma_k$
  if all VMs have been allocated **then**
    $\mathcal{P} \leftarrow \{P_0, \ldots, P_{Y-y}\}$
    $\{\mathcal{I}_k^{(j)}, j \in \Gamma_k\} \leftarrow \{\hat{\mathcal{I}}_k^{(j)}, j \in \Gamma_k\}$
    break
  end if
end for
for $j = 1$ to $|\Gamma_k|$ **do**
  $\lambda^{(j)} \leftarrow \sum_{i=1}^{|\mathcal{I}_k^{(j)}|} \lambda_i$
  $\lambda^{(j)}\beta^{(j)} \leftarrow \sum_{i=1}^{|\mathcal{I}_k^{(j)}|} \lambda_i\beta_i$
  $\mathcal{C}^{(j)} \leftarrow \{\}$, $\forall j \in \Gamma_k$
  for $x = 0$ to $X - 1$ **do**
    if $\tau_{off}(C_{X-x}) < \frac{1}{\lambda_j}$ **then**
      $\mathcal{C}^{(j)} \leftarrow \{C_1, \ldots, C_{X-x}\}$
      break
    end if
  end for
  evaluate Eq. (8) considering the states in $\mathcal{P}$ and $\mathcal{C}^{(j)}$, and apply classification rules
end for
re-order $\{\mathcal{I}_k^{(j)}, j \in \Gamma_k\}$ with decreasing performance requirement
for $j = 1$ to $|\Gamma_k|$ **do**
  allocate a core to $\mathcal{I}_k^{(j)}$
end for
set processor package to most suitable configuration

IV. Performance Evaluation

In order to evaluate the performance of the proposed policy, a simulation framework for a scaled-down datacenter is implemented in Matlab, considering Intel Xeon E5-2690 2.9GHz processors [17]. Besides being configurable through the ACPI, this processor has also been used by Intel in evaluating the Data Plane Development Kit (DPDK) virtual switch (vSwitch) for NFV [18].

For the sake of simplicity, but without loss of generality, in this work we only consider the following subset of configurations: $\{(C_1, P_0), (C_3, P_0), (C_1, P_8), (C_3, P_8)\}$. The C_1 and C_3 power states correspond to the *Halt* and *Sleep* modes, while the P_0 and P_8 performance states are supposed to correspond to the maximum and minimum core frequencies, respectively. Based on the data presented in [18], [9] and [19], the parameter values listed in Table I can be derived. Φ_{OH} is the power consumption overhead for each active server, and Φ_{DS} is the power consumption of a core in deep sleep.

TABLE I: Configuration parameters.

Parameter	Configuration			
	(C_1, P_0)	(C_3, P_0)	(C_1, P_8)	(C_3, P_8)
Φ_a (W)	16.75	16.75	10.5	10.5
Φ_t (W)	40	25	17	13
Φ_{idle} (W)	7	5.4	7	5.4
μ (pps)	891875	891875	515539	515539
τ (μs)	100	200	100	200
τ_{off} (μs)	5	10	5	10
Φ_{OH} (W)	195			
Φ_{DS} (W)	2			

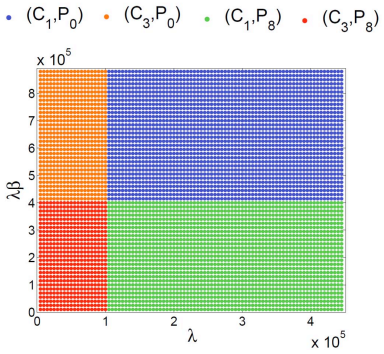

Fig. 2: Regions defined by the core classification.

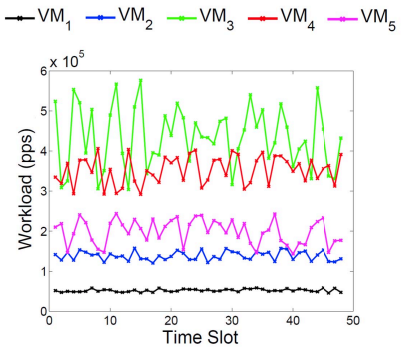

Fig. 3: VM workload variations.

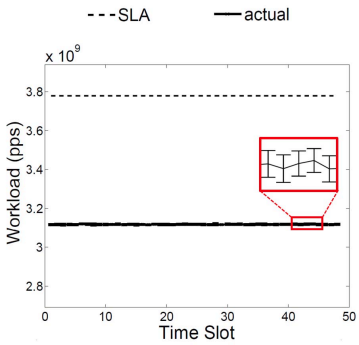

Fig. 4: Datacenter workload.

With this, Sub-section IV-A illustrates how the optimum configuration varies with the workload. Then, as an initial evaluation of the proposed policy, a numerical example is presented in Sub-section IV-B.

A. Core Classification

We express the power consumption model as a function of λ and $\lambda\beta$ by substituting the given configuration parameters into (8). By varying the workload and applying the core classification rules, regions in the $(\lambda, \lambda\beta)$ space are obtained. Fig. 2 shows that the resulting regions are simply defined by constant discriminant functions. However, as we add more variables (e.g., latency constraint, trade-off parameters, etc.) into the rules, we also expect to add complexity into these functions.

B. Numerical Example

We consider a system with 500 servers and 10000 VMs. Each server is supposed to have 2 octa core processors, as in [18].

For this example, the maximum workload $[\lambda_i\beta_i]^{max}$ of each VM is generated from the uniform distribution $\boldsymbol{U}(0.05\mu_{max}, 0.8\mu_{max})$. To consider the time variability of the VMs' workloads, the minimum workload $[\lambda_i\beta_i]^{min}$ is also generated from the uniform distribution $\boldsymbol{U}(0.5[\lambda_i\beta_i]^{max}, 0.8[\lambda_i\beta_i]^{max})$, $i \in \{1, \ldots, 10000\}$. In addition, the minimum β_i^{min} and maximum β_i^{max} batch sizes are generated from $\boldsymbol{U}(1, 15)$ and $\boldsymbol{U}(\beta_i^{min} + 1, 50)$, respectively.

Moreover, the number of sub-intervals is set to 48 — this corresponds to the number of times the consolidation policy is performed in a simulation run. In each sub-interval, the workloads vary according to $\boldsymbol{U}([\lambda_i\beta_i]^{min}, [\lambda_i\beta_i]^{max})$, while the batch sizes according to $\boldsymbol{U}(\beta_i^{min}, \beta_i^{max})$. Fig. 3 shows the workload variations of five representative VMs in the system, indicating the diversity among the VM workloads and behaviours in this implementation. As regards the reliability of the results, 10 runs with different seeds are performed for a given set of VMs to show the 95% confidence intervals of the data through error bars.

Taking a look at the entire system, Fig. 4 illustrates the average workload of the scaled-down datacenter. Despite the workload variations among the VMs, the total workload in the system remains stable at around 3.1 Gpps, which is approximately 18% lower than the one specified in the SLA.

In this evaluation, we want to compare the average datacenter power consumptions generated by the proposed policy (**C+PS**) and two baseline scenarios — (**xC+xPS**) and (**C+xPS**). **C** and **PS** denote in-server consolidation and power scaling, respectively, and **x** is appended to indicate the absence of the specified capability. Fig. 5 shows that simply performing in-server consolidation reduces the power consumption by around 4%, which can be further improved to 10% when jointly performed with power scaling.

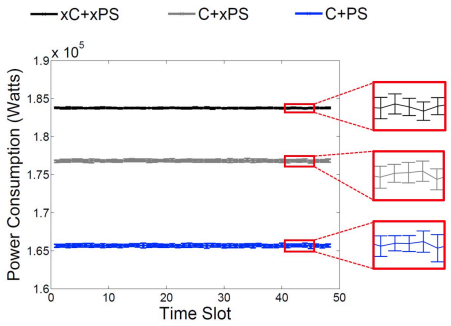

Fig. 5: Datacenter power consumption.

To better grasp the impact of this improvement, we can put it into figures by deriving the annual savings S of the datacenter as:

$$S = \phi \cdot \left[\tilde{\Phi}_{(xC+xPS)} - \tilde{\Phi}_{(C+PS)}\right] \cdot \frac{1}{1000} \cdot 24 \cdot 365 \qquad (9)$$

where ϕ is the energy cost per kilowatt hour (kWh). Based on the values reported in [20], electricity prices for industrial consumers in the European Union averaged 0.12 €/kWh during the second half of 2014. With this, we obtain around 19000 € of annual savings for the scaled-down datacenter considered in this example, which is quite a good number given its size. Now, imagine the potential savings for a datacenter operator like Google that has been estimated to run over 1 million servers since a couple of years back.

V. Conclusions

Energy efficiency and the complexity of network/service management pose sustainability and scalability issues on the upcoming network "softwarization" revolution. To address these aspects in the context of datacenter networks, a novel energy- and performance-aware consolidation policy that dynamically manages ACPI-enabled resources according to actual workload variations is proposed.

Starting from an M^X/G/1/SET queueing model, the traffic and power consumption models for a core serving multiple VMs are derived considering the aggregate workload. The queueing and traffic models are used to estimate some statistical features of the offered workloads, and together with the power consumption model, core classification rules are defined, and are used to perform joint power scaling and in-server consolidation of VNFs according to actual workload variations. To evaluate the performance of the proposed approach, a simulation framework for a scaled-down datacenter is implemented. Results show that the average datacenter power consumption can be reduced by up to 10%, suggesting a considerable amount of annual savings. Moving forward, the policy can better capture the network performance by incorporating the latency aspect, which requires an extension to the proposed model. In particular, it is necessary to derive the second moments of the batch size and service time, and a $M^X/D_M/1/SET$ queueing model, as adopted in [4], can come in handy for this concept.

Acknowledgment

This work was supported by the INPUT (In-Network Programmability for next-generation personal cloUd service supporT) project, funded by the European Commission under the Horizon 2020 Programme (Grant no. 644672).

References

[1] M. Chiosi, et al., "Network Functions Virtualization: An Introduction, Benefits, Enablers, Challenges & Call For Action," White Paper, 2012. [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper.pdf

[2] "Advanced Configurationa and Power Interface Specification." [Online]. Available: http://www.acpi.info/DOWNLOADS/ACPIspec40a.pdf

[3] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-Aware Fixed Network Infrastructures," *IEEE Commun. Surveys Tuts.*, vol. 13, no. 15, pp. 223–244, 2011.

[4] R. Bolla, R. Bruschi, A. Carrega, F. Davoli, and P. Lago, "A Closed-Form Model for the IEEE 802.3az Network and Power Performance," *IEEE J. Sel. Areas Commun.*, vol. 32, pp. 16–27, Jan. 2014.

[5] R. Bolla, R. Bruschi, A. Carrega, and F. Davoli, "Green Networking with Packet Processing Engines: Modeling and Optimization," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, pp. 110–123, Feb. 2014.

[6] M. H. Ferdaus, M. Murshed, R. Calheiros and R. Buyya, "Virtual Machine Consolidation in Cloud Data Centers Using ACO Metaheuristic," in *Proc. of the 2014 Euro-Par Internat. Conf. on Parallel Processing*, F. Silva, I. Dutra and V. Santos Costa, Ed., Porto, Portugal, Aug. 2014, pp. 306–317.

[7] W. Song, Z. Xiao, Q. Chen and H. Luo, "Adaptive Resource Provisioning for the Cloud Using Online Bin Packing," *IEEE Trans. Comput.*, vol. 63, no. 11, pp. 2647–2660, Nov. 2014.

[8] E. G. Coffman, Jr., M. R. Garey and D. S. Johnson, "Approximation Algorithms for Bin Packing: A Survey," in *Approximation Algorithms for NP-Hard Problems*, D. S. Hochbaum, Ed. USA: PWS Publishing, Boston, 1996.

[9] R. Bolla, R. Bruschi and P. Lago, "The Hidden Cost of Network Low Power Idle," in *Proc. of the 2013 IEEE Internat. Conf. on Communications (IEEE ICC 2013)*, Budapest, Hungary, 2013, pp. 4148–4153.

[10] A. Klemm, C. Lindemann, and M. Lohmann, "Modeling IP Traffic using the Batch Markovian Arrival Process," *Performance Evaluation*, vol. 54, pp. 149–173, Oct. 2003.

[11] P. Salvador, A. Pacheco, and R. Valadas, "Modeling IP Traffic: Joint Characterization of Packet Arrivals and Packet Sizes using BMAPs," *Computer Networks*, vol. 44, pp. 335–352, Feb. 2004.

[12] J. W. Cohen, "A Two-queue, One-server Model with Priority for the Longer Queue," *Queueing Systems: Theory and Applications*, vol. 2, no. 3, pp. 261–283, Nov. 1987.

[13] G. Choudhury, "An M^X/G/1 Queueing System with a Setup Period and a Vacation Period," *Queueing Systems*, vol. 36, no. 1-3, pp. 23–38, 2000.

[14] J. Medhi, *Stochastic Models in Queueing Theory*, 2nd ed. Elsevier Science USA, 2003.

[15] Cisco, "Best Practices in Core Network Capacity Planning," White Paper, 2013. [Online]. Available: http://www.cisco.com/c/en/us/products/collateral/routers/wan-automation-engine/white_paper_c11-728551.pdf

[16] Z. Huang and D. Tsang, "SLA Guaranteed Virtual Machine Consolidation for Computing Clouds," in *Proc. of the 2012 IEEE Internat. Conf. on Communications (ICC)*, 2012, pp. 1314–1319.

[17] "Intel® Xeon® Processor E5-Product Family Datasheet." [Online]. Available: http://www.intel.com/content/dam/www/public/us/en/documents/datasheets/xeon-e5-1600-2600-vol-1-datasheet.pdf

[18] "Network Function Virtualization: Intel® Data Plane Development Kit vSwitch with Linux Virtualization and Intel® Architecture." [Online]. Available: https://networkbuilders.intel.com/docs/Network_Builders_RA_DPDK_vSwitch_Final.pdf

[19] "Intel® Xeon® Processor E5-2690: Enterprise Workload Performance while Running Storage Efficiency Tasks." [Online]. Available: http://www.intel.fr/content/dam/www/public/us/en/documents/reports/performance-xeon-e5-2690-pt-report.pdf

[20] "Energy Price Statistics." [Online]. Available: http://ec.europa.eu/eurostat/statistics-explained/index.php/Energy_price_statistics