Journal of Network and Computer Applications 69 (2016) 117–133

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Power management in virtualized datacenter – A survey

V.K. Mohan Raj [a,*], R. Shriram [b]

[a] Tata Research Development and Design Center (TRDDC), Tata Consultancy Services Limited, Pune 400013, India
[b] Department of Computer Science and Engineering, BS Abdur Rahman University, Chennai 600048, India

ARTICLE INFO

Article history:
Received 27 May 2015
Received in revised form
8 March 2016
Accepted 13 April 2016
Available online 22 April 2016

Keywords:
Power management approaches
Server heterogeneity controls
Server virtualization controls
Datacenter thermal controls

ABSTRACT

With growth in internet usage, increase in user demands, increase in applications or services due to technology advances (virtualization) and new set of business models (cloud computing), datacenters' are growing in numbers and size. This datacenter growth, additionally contributes to increased energy consumption and thereby contributing to increased electricity cost and the concerns about the environmental impacts.

Traditionally, computing systems were designed and developed to improve and meet performance demands of application or services. However, the increase in datacenter energy consumption and carbon footprint environmental impacts are real practical concerns. Therefore, the goal of the computing system design now is to optimally minimize power consumption and improve energy efficiency.

This paper presents a survey of energy-efficient design of computing systems covering techniques used at hardware, virtualization, server heterogeneity and thermal levels. In this paper, we highlight the need to have a holistic view of power consumption of both IT and non-IT systems to achieve energy efficiency in datacenters.

1. Introduction

Growth in use of internet and need for increased data storage have contributed to growth of cloud datacenters both in size and numbers. These datacenters employ tens of thousands of servers hosting 24×7 services and catering to demands of hundreds or thousands of users. In recent years, use of high-performance computing techniques has also increased, which require trading off energy consumption for obtaining increased performance with performance adherence considered to be the primary objective. This increase in datacenter size and numbers and need for performance centric demand processing need has led to increase in datacenter power consumption.

It has been shown that most of the time, the modern servers operate between 10% and 50% of maximum possible utilization (Barroso and Holzle, 2009; Ranganathan et al., 2006). Server when operated at low to medium utilization consumes more power and negatively impacts energy efficiency due to non-linear power proportionality characteristics of servers at these utilization levels (Annavaram and Wong, 2012; Barroso and Holzle, 2007). Many a times, the reasons to allow servers operate at low to medium utilization, with unused free resource capacities as a factor or safety buffer, is to accommodate brief bursts in future workload transactions and to meet server level agreements (*SLA*) commitments (Mittal and Zhang, 2012). Also, different server configurations exhibit different levels of power proportionality characteristics, implying that server heterogeneity as a factor also need to be considered. Thus, power management technique should arrive at the optimality level between servers' energy efficiency and applications' *SLA* commitments, either by reactive, proactive or predictive workload analysis.

In a datacenter, peak power consumption of a computation system (for e.g., server), and increasing density of server placements (example: server racks, blade servers etc.) dictates the cooling infrastructure required (cooling coverage) to maintain server temperatures within safe thermal limits. High power consumption and high concentration of nodes in datacenters might lead to increased node failures due to increase in localized temperature (Bianchini and Rajamony, 2004). It has been observed that a 15 °C rise, increases the failure rates in hard-disks by a factor of two (Anderson et al., 2003). Hence, maintaining the computer systems at proper temperature is important to ensure system reliability. For every watt of power consumed in the computing equipment, an additional 0.5–1 W of power is required to operate the cooling system itself (Patel et al., 2003), which further increases the energy cost. High levels of compute power consumption demand a costly cooling infrastructure (Mittal, 2014). Finally, large power consumption also has adverse environmental impact, e.g. large carbon emission.

* Corresponding author.
E-mail addresses: v.raj@tcs.com (V.K.M. Raj), shriram@bsauniv.ac.in (R. Shriram).

http://dx.doi.org/10.1016/j.jnca.2016.04.019
1084-8045/

Electricity costs for powering servers forms a major cost of operation in datacenters and it has been estimated that in near future, operational expense (energy costs) over a period may be even more than the capital expense (cost of IT infrastructures) (Bianchini and Rajamony, 2004). Ever increasing energy consumption of computing systems has started to become another important factor contributing to increase in datacenter owners' (DCO) electricity bills. For all of the above reasons, the design and use of power management techniques for modern datacenters has become a topic of importance mainly to reduce energy consumption and to reduce carbon footprint environmental impacts. In this paper, we survey several power management techniques considering controls at levels, such as hardware, software, workload etc., to improve datacenter energy efficiency.

The rest of the paper is organized as follows. In Section 2, we provide the background of power and energy models. In Section 3, we describe datacenter power management approaches with detailed taxonomy survey of the solutions. We finally conclude our work in Section 4.

2. Power and energy models

Power is the instantaneous rate of work done by the system, while energy is the total amount of work performed over a period of time (Beloglazov and Buyya, 2012). In an electrical system, power and energy are measured in watts (W) and watt-hour (Wh), respectively (Eqs. (1) and (2)). Electric current is the flow of electric charge measured in amperes (A). Ampere defines the amount of electric charge transferred by a circuit per second. Potential difference is the difference in amount of potential energy between two locations and is measured in volts (V). 1 W of work done is achieved when 1 A is transferred through a potential difference of 1 V. A kilowatt-hour (kWh) is the amount of energy equivalent to a power of 1 kW (1000 W) being applied for one hour (Beloglazov, 2013). Formally, power and energy can be defined as follows:

$$P=\frac{W}{T} \tag{1}$$

$$E=P\times T \tag{2}$$

where P is the power consumed; W is the work done during the time duration T, E is the energy consumed. Since energy depends on both power and time duration, any change in either power or time duration or both could impact energy consumption. It is not always true that decrease in power consumption would decrease energy consumption, such as, operating with the decreased power consumption but for greater time duration. This very fact drives the need to optimally control both power consumption and time duration, if we need to reduce energy consumption.

3. Power management approaches

Datacenters' energy consumption is determined by hardware efficiency, efficiency of infrastructure resource management system, efficiency of applications running in the system which includes workload processing approaches. The interdependence of different levels of computing systems (Beloglazov, 2013) with respect to energy consumption is shown in Fig. 1. Enterprise level constraints such as power usage limits, datacenter or server temperature limits, cost limits etc., and user application SLAs etc., need be considered in each of the power management level techniques to arrive at optimal power savings. Each of the power management level focuses on specific power management aspect, such as

- At Application level: whether application's architecture and usage is designed to account for energy efficiency improvement;
- At Cluster architecture level: managed by a resource management system deployed on the connected infrastructure, which caters to application specific resource capacity allocations both at start and at run-time to achieve energy efficiency; and
- At Server hardware and other infrastructure level: whether hardware architecture is designed to account for energy efficiency improvement, signifying hardware efficiency management principles such as dynamic power management (DPM).

Use of right power management technique(s) is essential to improve both energy efficiency and ultimately cost efficiency. There are varied types of power management techniques, categorized depending on specific component, type of usage, system architecture design etc. At the high-level (Fig. 2), power management techniques (Beloglazov, 2013) can be divided into static and dynamic, each of these categories has a hardware level power management control and a software level control.

Static power management (*SPM*) contains all the optimization methods that are applied at the design time at the hardware circuit, logic, architectural system levels and also at software components such as OS, compilers etc. At the Static hardware power management level, system devices can be replaced by the low-power ARM or atom based processor servers which have reduced performance capability but are more energy-efficient (Annavaram and Wong, 2012; Pedram, 2012) and the system workload can be effectively distributed to these devices. In case of dynamic power management level, various power optimization methods can be applied at hardware level such as dynamic power management (DPM), dynamic voltage frequency scaling (DVFS), and at software level using virtualization controls, thermal controls, heterogeneity controls etc.

We provide a snapshot of the related works, approach methodology, outcome strengths and weakness of each of these works, from the following aspects:

- Dynamic power management using server sleep state transition controls,
- Dynamic VM consolidation and deconsolidation controls,
- Server heterogeneity controls, and
- Server and datacenter thermal controls.

3.1. DPM server sleep transition controls

Dynamic power management (*DPM*) techniques are primarily adopted to reduce power consumption or maintain/improve application performance and also improve real-time resource usage. Server systems' power consumption (P_{total}) is contributed by a static component (P_{static}) which is independent of workload resource usage and a dynamic component ($P^{f}_{dynamic}$). Dynamic component power consumption value ($P^{f}_{dynamic}$) depends mainly on a specific usage scenario, clock rates, I/O activity, short-circuiting current and switched capacitance (Elnozahy et al., 2002).

$$P^{f}_{dynamic}=aCv^2f \tag{3}$$

where a is the switching activity constant, C is the physical capacitance, v is the supply voltage, and f is the processor clock frequency.

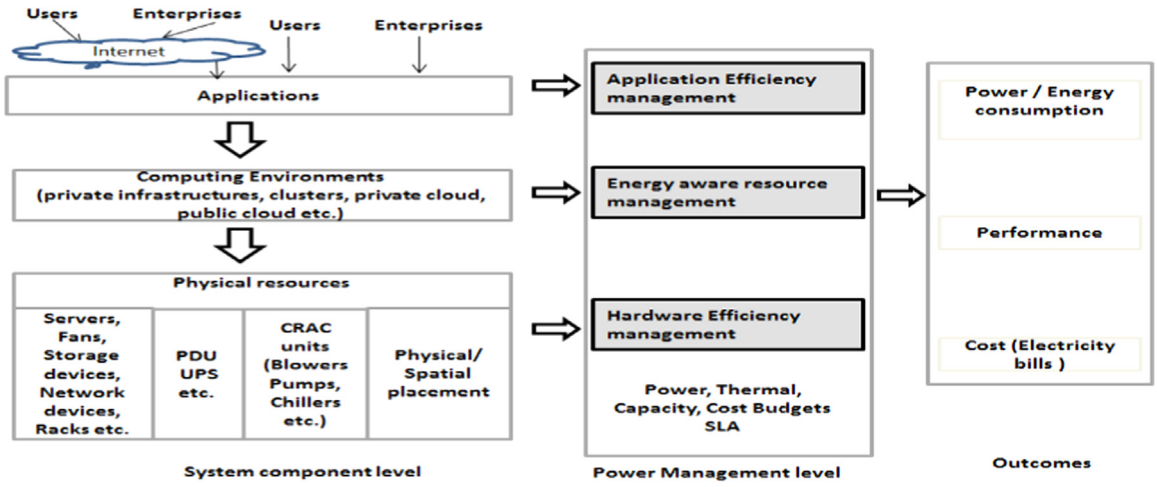

Fig. 1. Datacenter system component and energy management interdependence.

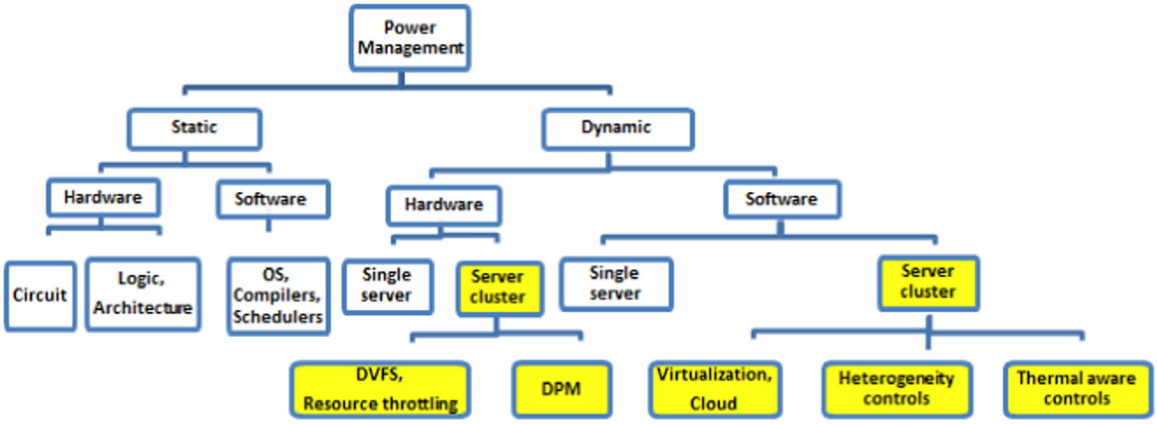

Fig. 2. A high-level taxonomy of power management.

$$P_{total}=P_{static}+P^{f}_{dynamic} \quad (4)$$

Server when completely idle consumes about 30–60% of peak power referred to as static or idle power (Fig. 3) due to the fact that the system components consumes power to be in switched on state and also, a portion of power is consumed by the base OS, system driver tasks etc. *DPM* approach deals with containing or eliminating this static power by either turning the server off or transitioning the server to a low power sleep state. With *DPM*, to save power, it is beneficial to switch the server to off or switch the server to a reduced sleep mode, when the server is not processing workloads or is not in use. DPM enablement assumes that workload is variable over time and allows dynamic server power state transitions.

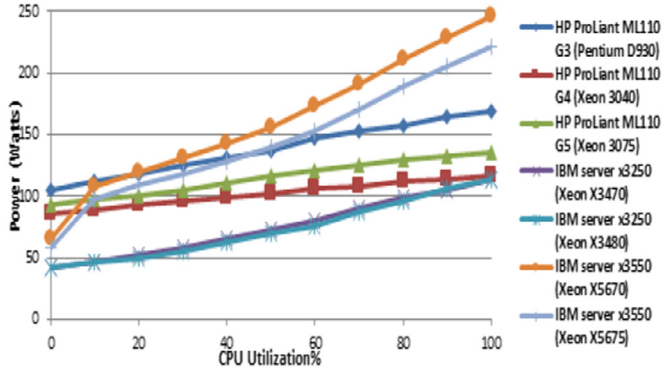

Fig. 3. Server CPU load vs. power consumption (Standard Performance Evaluation Corporation (SPEC), 2011).

Application workload characteristics is a critical factor, which determines when a server could be switched off or transitioned to a low power sleep mode or when a server should be switched on from off or low power sleep modes. In case of *DPM* controls, transitioning server between power states (could be high power state to low power state or vice-versa) has a time expend called setup or transition time (Gandhi et al., 2012), especially the servers in sleep or off state incur a setup time to get them to on state again. Another challenge is the scarcity of sleep states and support in current datacenter servers. An assumption enabling *DPM* is to allow dynamic adjustment of power state transitions is driven by the performance requirements in processing the variable workloads. *DPM* with server sleep state transition in datacenter helps save power on certain types of workload traces (Gandhi et al., 2012). For very bursty workloads, sleep state transitioning is counter-productive due to the fact that setup time from low-power off or sleep state to operational state is high and could negatively impacts both performance and energy consumption to switch on a server back again from off or low power sleep state.

Advanced Configuration and Power Interface (*ACPI*) define the power state of system processors in either active (executing) or sleep (not executing) state in commodity platform servers. The most relevant states in the context of *DPM* are S-states and P-states. System sleep states include are designated S0, S1,. Sn. For e.g., Intel Core i7 processor, can be in six different power states ranging from S0 (on or active state) to S5 (the system is in power

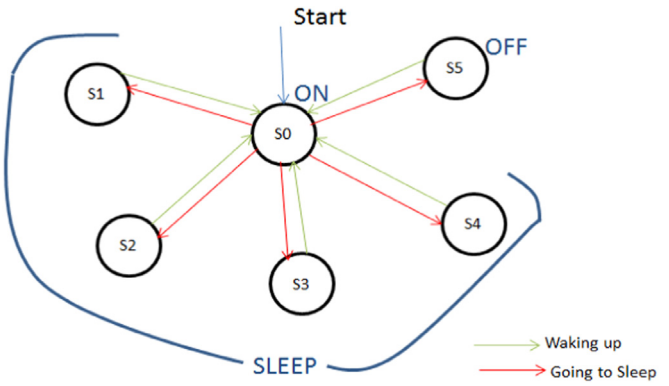

Fig. 4. Allowable system power state transitions (Intel Core i7 Nehalem processor).

off state), and the remaining intermediate states are referred to as sleep states (Fig. 4). Also, with respect to Windows OS, system sleep state S4 is referred to as *hibernate*, sleep state S4 is referred to as *sleep* state and sleep state S5 is referred to as *shutdown*. Return to active state from any one of the sleep states, enough does not require a system reboot (Intel and Core i7 (Nehalem) Dynamic Power Management, 2015). While in a sleep state, the processor does not execute any instructions. Each processor sleep state has a latency associated with entering and exiting, which in-turn influences the power savings (ACPI, 2015).

Performance P-States are a pre-defined set of frequency and voltage combinations at which the processor can operate when the CPU is active (Table 1). The processor utilizes *DVFS* to implement the various processor supported P-States. *DVFS* reduces the number of instructions a processor can issue in a given amount of time, thus reducing the performance. This, in turn, increases the run-time of workload tasks or segments that are significantly CPU bound. For power savings, the main idea with *DVFS* technique is to decrease processor frequency and/or voltage when it is not fully utilized with an upside i.e., reduction in power consumption but with a downside to performance. Another orthogonal use-case scenario would be to increase processor voltage and/or frequency if performance needs to be improved upon (reduce task critical path length), with a downside of increased power consumption adopting a *race-to-idle* or *race-to-sleep* methodology with a follow-up opportunity to switch off the server. Server power consumption (Fan et al., 2007) is closely related to CPU utilization and frequency (Eq. (5)). Server power and load utilization at different frequencies controlled using *DVFS* depicts a near linear relationship (Gupta et al., 2012). *DVFS* allows servers to dynamically modify the processor frequency (according to the CPU load) in order to achieve less energy consumption (Kamga, 2013)

$$P^f(u) = P^f_{idle} + \left(P^f_{peak} - P^f_{idle}\right) \times u \tag{5}$$

where $P^f(u)$ is the server power consumption at frequency f and CPU utilization u, P^f_{idle} is the power consumption when not processing any workload, P^f_{peak} is the maximum power consumption at 100% utilization. In case of multi core processors, each core can be operated at different frequencies, and to that extent, processor core utilization and core frequency would determine the power consumption. In commodity server processors such as Intel or AMD, configuration settings such as Intel's *SpeedStep* or AMD's *Cool 'n' Quiet* should be enabled to activate *DVFS* controls at different operating points such as 66.5%, 77.7%…100% of the maximum frequency (Gandhi et al., 2009). Although *DVFS* provides an efficient way of managing power consumption of the CPU, the overall dynamic power range of servers remains narrow. Even if a server is completely idle, it still consumes around 30–60% of peak power (Fan et al., 2007). Authors in Sueur and Heiser (2010) have indicated that the portion of variability or control that *DVFS* can bring to power consumption is diminishing (as static power consumption is increasing) and sleep state mode of *DPM* controls have better coverage. But with new trend processors such as Intel™ Nehalem architecture (static power consumption of 30% of peak power consumption as opposed to 60–70% with earlier processor families) and adoption of these in datacenters could provide *DVFS* controls have similar range coverage as that of *DPM* controls. With multi-core processor servers, there could be a scenario with each of the processor cores have a pre-determined frequency speed. In such a case, selecting the right processor core (in effect the frequency) would analogize *DVFS* control in itself.

Table 1
P-State operating points and power consumption (for Intel Pentium 1.6 GHz CPU) (Enhanced Intel SpeedStep Technology for the Intel Pentium M Processor, 2015).

P-State	Frequency (GHz)	Voltage (V)	Power (W)
P0	1.6	1.484	25
P1	1.4	1.420	17
P2	1.2	1.276	13
P3	1.0	1.164	10
P4	800	1.036	8
P5	600	0.956	6

Both *DPM* and *DVFS* control levers are tied to physical server. Any approach that uses *DPM* and *DVFS* in a virtualized server setup have to consider the operational status of all the VM's hosted in the server. Switching off the server or transitioning the server to low power sleep state using *DPM* levers requires all the VMs hosted in the server to be free from workload processing. In case of *DVFS*, transitioning to a lower frequency (with the aim to save power consumption) should be considered if and only if all VMs in the server that is processing workload request can still meet *SLAs* (as performance reduces with decrease in frequency). Another point to note is that increasing frequency using *DVFS* levers with the aim to improve performance may or may not help save energy (as power consumption increases with increased frequency setting. Monitoring or capturing power consumption at physical server is normally done using power models (Eqs. (3)–(5)) or by direct power-meter measurements. But in case of virtualized servers, measuring VM level power consumption is primarily done using modeling approaches proposed in Kansal et al. (2010), Bertran et al. (2010), and Bohra and Chaudhary (2010).

3.1.1. DPM server sleep transition controls—classification

A classification of the related research works specifically to *DPM* with sleep state transitions and a combined interplay of *DVFS* and *DPM* approaches is shown in Table 2. Also, we improve upon this taxonomy with a detailed study of the contribution and areas for improvements in these works.

Napsac (Krioukov et al., 2010) approach considers three types of server's—(a) server class, (b) mobile class, and (c) embedded processor class. Napsac uses prediction to arrive at number of servers in of the server types that should be awake in future to process the workload and meet SLAs. Prediction policies considered are Last Arrival (*LA*), moving window average (*MWA*), exponentially weighted average (*EWA*), and linear regression (*LR*). Each server's request handling capacity i.e., maximum number of requests that can be handled by the server for the current load is empirically determined. Prediction function predicts the future request rate, which in-turn along with server's request handling capacity drives the number of active servers required in future. Server start-up and shut-down algorithms kicks in every second. The difference between current number of active servers and predicted future number of active servers implies whether to start a server or shutdown a server. The authors show that *MWA* and *LR*

Table 2
Server DPM with sleep state transition and DVFS level research.

Literature	Target system	System resources	Approach/techniques	Evaluation		
				Methodology	Workload	Results
NapSac (Krioukov et al., 2010)	Server cluster	CPU	Uses Greedy KnapSack algorithm in order to provision a near optimal number of servers at any given point in time. Uses low power atom processor servers to handle workload bursts.	Simulation Physical Heterogeneous servers	Wikipedia http traffic	energy savings is still 27% when compared to a cluster provisioned
Dreamweaver (Meisner et al., 2009) PowerNap (Wang et al., 2011)	Server	CPU	Dreamweaver: multi-core server framework uses Weave scheduling policy to coalesce idle and busy periods across server cores. PowerNap: approach to power management that enables the entire system to transition rapidly into and out of a low-power state where all activity is suspended until new work arrives	Simulation Physical Homogeneous servers	traces of departmental servers Google Web Search test cluster traces Web 2.0 application server traces	DreamWeaver can offer roughly 25% better power savings than PowerNap and nearly 30% more than Voltage Frequency Scaling (VFS)
PowerSleep (Meisner and Wenisch, 2012)	server	CPU	Uses Linear programming controlled DVS mechanism to select execution speed for the server. It also uses DPM to put the system in sleep power mode. Server is modeled to be follow M/G/1/PS queuing model	Simulation Physics Homogeneous servers	Web 2.0 application server traces	Power savings of 33% over PowerNap approach
Virtualmapping (Wang et al., 2010)	server	CPU	Uses control theoretic technique to request batching to delay instant usage of VMs, request batching and DVFS integration to control performance, and CPU resource allocation to coalesce VMs execution improving server idle periods	Testbed and simulation Virtualized Homogeneous servers	SDSCHTTP, EPA-HTTP and ClarkNet-HTTP	Virtual Batching can achieve the desired performance with 63% more energy savings on average than DVFS only approach
Autoscale (Gandhi et al., 2012)	Server cluster	CPU	Uses dynamic power management considering setup time state transition overhead and delayed switch off approach	Testbed and simulation physical Homogeneous servers	ITA 1998, NLANR 1995 SAP 2011	AutoScale approach gives 40% savings on power consumption when compared with AlwaysOn approach.
Dhiman (Dhiman and Rosing, 2009)	Server	CPU Disk RAM	This approach uses the interplay of both DPM and DVFS techniques to select the best server and best voltage–frequency (v–f) settings for a given workload	Testbed and simulation physical Heterogeneous servers	Synthetic workloads for CPU and RAM evaluation; and server traces from HP system for HDD evaluation	achieves an overall performance comparable to the best-performing expert at any point in time, with energy savings as high as 61% and 49% for HDD and CPU.
HyPowMan (Bhatti et al., 2011)	Embedded system	CPU	This approach uses the interplay of both DPM and DVFS techniques to select the best server and best voltage–frequency (v–f) settings for a given workload	simulation physical Homogeneous servers	Synthetic workloads	proposed scheme adapts well to the changing workload and quickly converges to the best-performing technique within the selected expert (policy) set by a margin of 1.5–11% on energy savings

work best for the Wikipedia.org traffic traces, and provide significant power savings over conservative static approaches. With respect to challenges, this work, considers mobile and embedded processors alongside servers as part to build heterogeneous server system setup. This work is done using physical systems and does not consider virtualization aspects. Also, server provisioning algorithm is application dependent.

Powersleep scheme (Meisner and Wenisch, 2012) uses both *DPM* with sleep state and dynamic voltage scaling (*DVS*) to reduce power consumption and meet response time constraints. This work has been done considering a single physical server. Goal of this work is to minimize mean power consumption of server under given mean response time. Authors have highlighted the need to account for power state transition times from either running power mode into the sleep power mode or wake-up transition time, both of these could impact cost and hit performance response times. Authors have considered three types of threshold values: (a) idle period threshold, which is the minimum length of idle queue duration (δ_h) before server is transitioned into sleep; (b) sleep period threshold, which is the maximum length of period (δ_e) during which server can stay in sleep mode; and (c) procrastination sleep period, implying a wait for this period (δ_x) to expire and only then schedule jobs which arrived during this period. Authors have accounted for mode transition overheads to transition between the running power mode and the sleep power mode i.e., time overheads for the suspend transition (δs) from the running power mode to the sleep power mode and the wake-up transition (δ_w) from the sleep power mode back to the running power mode. These time thresholds and overheads acts as controls and needs to be considered in processing new request(s)—activating servers or threshold durations a server can stay in active mode of sleep modes. Under *PowerSleep*, the server stays in four different modes: running, transition, sleep, and idle. With sleep and transition modes, *PowerSleep* uses queuing with starter model to manage server modes in processing requests. The server is "turned off" whenever the request queue becomes empty. When a job arrives at an empty queue, it is not served immediately; rather the server consumes an additional amount of transition time TX to start from off state before it can serve the new first job. Jobs which arrive are tracked in a queue and are processed by a server in on state (i.e., server with at least one job either in service or in the queue). Starter TX under *PowerSleep* includes the wake-up transition time, the procrastination sleep period (δ_x) and may also include the remaining portion of a suspend transition. With respect to challenges, this work, considers only one server and operates on known application workload.

VirtualBatching (Wang et al., 2010) is an approach to improve energy efficiency, by coalescing requests so that VMs' of the server processing the requests are controlled to complete at same time, and put the server into sleep between such batches. This completion-time balanced VM request processing is achieved using control-theoretic use of VM specific CPU time allocations, server level DVFS and request arrival-rate controls. When the group of VMs completes processing the request, the server is switched off or transitioned to low power sleep mode to improve energy efficiency. *VirtualBatching* technique works mainly with light workloads. *VirtualBatching* dynamically allocates the CPU resource such that all the VMs can exhibit approximately the same performance level relative to their allowed peak values. *VirtualBatching* then determines the time length for periodically batching incoming requests and transitioning the processor into deep sleep state. When the workload intensity changes from light to medium, request batching is automatically switched to *DVFS* to increase processor frequency to meet performance criteria. The authors have shown empirically that *VirtualBatching* achieves the desired performance with energy savings of more than 63% on an average against the baseline test runs with *DVFS* controls alone.

With respect to solution weakness, this work uses one virtualized server with multiple VMs. This approach does not scale well with bursty workloads.

Autoscale's (Gandhi et al., 2012) goal is to reduce number of servers needed in datacenters driven by unpredictable, time-varying load, while meeting response time SLAs. The author's propose two approaches *Autoscale–* and *Autoscale*. *AutoScale* scales the datacenter capacity, by adding or removing servers as needed. *AutoScale* has two key features: (a) autonomically maintains just the right amount of spare capacity to handle bursts in the request rate; and (b) handles changes in request size and server efficiency. *Autoscale–* is a reactive approach. It does not shutdown servers immediately if there is no workload, but rather allows the server to stay in idle state and delays the server shutdown process by a T-wait duration which accounts for the server setup times. *Autoscale–* handles un-predictable changes in request rate, and the full *AutoScale* policy builds upon *AutoScale–* to handle all forms of changes in load recklessly. *AutoScale* uses a capacity inference algorithm, using which it determines the appropriate capacity. *AutoScale* does not require knowledge of the *request rate* or the *request size* or the *server efficiency. Autoscale*, load per server is derived from key-value datastore, where *number of jobs per server* is the key and value is *load per server. Request size* (work associated with each request) can change, if new features or security checks are added to the application, server efficiency can change, if any abnormalities occur in the system, such as internal service disruptions, slow networks, or maintenance cycles. *AutoScale* tries to infer the *amount of work* in the system by monitoring *system load*. The *amount of work* in the system is proportional to both the *request rate* and the *request size* (the *request size* in turn depends also on the *server efficiency*), and thus, tries to infer the product of *request rate* and *request size*, which is the *system load*. Capacity inference algorithm has few rules such as *number of jobs per server* is the ratio of *number of jobs* in the system and *current number of servers*, from *number of jobs per server*, *load per server* can be inferred from a key-value datastore, and *System load* is the product of *load per server* and *current number of servers.* Assumption here is that *jobs per server* and *load per server* metrics are not dependent on *request size* or *server efficiency*. Select metrics such as load per server, jobs per server etc. have to be benchmarked once for each server in the system. With respect to solution weakness, this work, considers only homogeneous physical server configurations and works with known application workloads.

Goal of work (Dhiman and Rosing, 2009) is to achieve energy savings and reduce performance delays by using both *DPM* and *DVFS* techniques to select the best server and best voltage–frequency (v–f) settings for a given workload. Server selection is done by a set of experts, which refers to a set of *DPM* policies and *DVFS* voltage–frequency settings. Different experts outperform each other under different workloads and device power characteristics. The selection takes into account energy savings, performance delay, and the user-specified energy performance trade-off. The authors use an online-learning algorithm that adapts to changes in the workload characteristics to arrive at the best-performing expert. Experts can be in one of the two possible states: dormant or operational states. Dormant experts are the default expert. When a controller event selects an expert referred to as operational expert, that has the highest probability to perform based on current workload model. Operational expert can either be a *DPM* policy or a v–f setting depending upon the event. The amount of time for which the expert stays in the operational state is referred to as the operative period, after which it returns to its default dormant state. The operative period for an operational expert in case of *DPM* is the length of the idle period, while in case of *DVFS*, it is the length of the scheduler.

With respect to solution weakness, this work, considers only physical servers in a cluster environment. It is a supervised approach, with need to build expert policies and *DVFS* definitions a-priori as part of training requirements.

Hybrid Power Management (*HyPowMan*) scheme (Bhatti et al., 2011) considers the problem of power and energy minimization for periodic real-time tasks that are scheduled over multiprocessor platforms that have *DPM* and *DVFS* capabilities. *HyPowMan* scheme takes a set of *DPM* and *DVFS* policies for a given set of conditions, and adapts at runtime to the best-performing policy for any given workload using policy selection mechanism. *HyPowMan* scheme implements this policy selection mechanism through a machine-learning algorithm. Machine-learning algorithm provides theoretical guarantee on overall performance converging to that of the best-performing power management policy (expert) among the available ones (expert set). Experts could be either dormant or working. Any expert, which is currently active, is said to be working expert. A working expert makes the power management decisions on processors under the control of applied scheduling policy. Working expert returns to dormant state, which is the default state for all experts, once it finishes its job or another working expert replaces it. The most critical task for *HyPowMan* scheme is to select an appropriate DPM or DVFS expert for a given power management need. *HyPowMan* considers processing tasks to be preemptive and support migration between processors.

With respect to solution weakness, this work, considers embedded systems with *DPM* and *DVFS* controllability provisions. It is a supervised approach, with need to build experts a-priori as part of training requirements.

PowerNap (Wang et al., 2011) scheme handles the dominant server idle time by quickly transitioning in and out of a low power sleep mode. Under *PowerNap*, the server runs at the maximum speed in executing jobs if there are jobs in queue, and is immediately put into the sleep mode once the queue is empty and is immediately waken up once a new job arrives. *PowerNap* mechanism includes a sleep-active state scheduling component and a network interface card (*NIC*) supported by Wake-on-LAN (*WOL*) functionality. The system is put into the sleep mode when there are no workloads and awakens within 1ms upon request packet arrival. However, mode transitions between the active mode and the sleep mode introduce a transition time overhead for the server, which degrades the performance.

With respect to solution weakness, author's state that *PowerNap* is effective if state transition time is below 10 ms, and incurs no overheads below 1 ms. In reality transition time of lesser than 10 ms is not yet practical. When the server utilization is low, the *PowerNap* scheme performs better than pure *DVS* scheme. The higher the server utilization is, the obvious need for mode transition and corresponding overheads, which in-turn impacts system performance. Therefore, it is necessary in the design with the sleep mode to account for mode transition overheads.

DreamWeaver (Meisner et al., 2009) is a framework to facilitate deep sleep for request-parallel applications on multicore servers. *DreamWeaver* comprises two elements: (a) *Weave scheduling*, which is a scheduling policy to coalesce idle and busy periods across cores to create opportunities for system transition to low power sleep states; and (b) *Dream Processor*, a light-weight co-processor that monitors incoming network traffic and suspended work during sleep to determine when the system must wake. *DreamWeaver* is based on two key concepts: (a) stall execution and sleep anytime any core is unoccupied—task preemptive sleep; and (b) maximum request time limit, after which the request may be stalled. *DreamWeaver* keeps the server operational only when all of its cores are utilized. The *Dream Processor* is a simple micro-controller that tracks accumulated stall time for suspended requests and receives, enqueues, and counts incoming network packets during sleep. When all idle cores is assigned with enough packets, or when the allowable stall time for any request is exhausted, the *Dream Processor* wakes the system to resume execution. *DreamWeaver's* approach is workload dependent. CPU cores are switched on and off according to the workload to improve power proportionality of the system.

With respect to solution weakness, this approach is application workload dependent and the workload task should be pre-emptable for it to be stalled and re-started or processed again in select processor cores later on.

We find that most of the works using server sleep state transition approach are at conceptual level considering single server or deal with physical non-virtualized server cluster level. One common weakness with most of the works is the non-consideration of server state transition overheads with respect to transition power consumption and transition latency times.

3.2. Server heterogeneity controls

In a server cluster level, heterogeneity has also been exploited to allocate workload to the most energy-efficient server architecture. Approaches leveraging heterogeneity are complementary to dynamic server provisioning, and typically provide improvements over dynamic server provisioning depending on heterogeneity levels. Job scheduling problem in heterogeneous computing systems without energy consideration has been shown to be NP-complete (Ibarra and Kim, 1977). Hence, the need for heuristics based solutions to select the best server that meets the objective (e.g. performance, cost and energy aspects) and schedule workload requests to this server (Kumar and Raghunathan, 2013).

A server is defined as being power-proportional if it consumes *x%* of its peak power when operating at a utilization of x% (Barroso and Holzle, 2007). But commodity servers are not strictly power-proportional, where-in actual power consumption is not linear (Fig. 5). Datacenters typically contain servers of different configuration due to very many reasons such as specific workload processing requirements (web, high performance computing etc.), technology obsolescence, failures etc. (Delimitrou and Kozyrakis, 2013). This presence and availability of different server configurations drives the level of heterogeneity in a datacenter. On a high level, a server's heterogeneity with respect to energy efficiency is attributed to processor's performance and power consumption for a range of load or utilization. Each server is characterized by the CPU performance defined in Million instructions per second (*MIPS*), amount of RAM, amount of cache and network

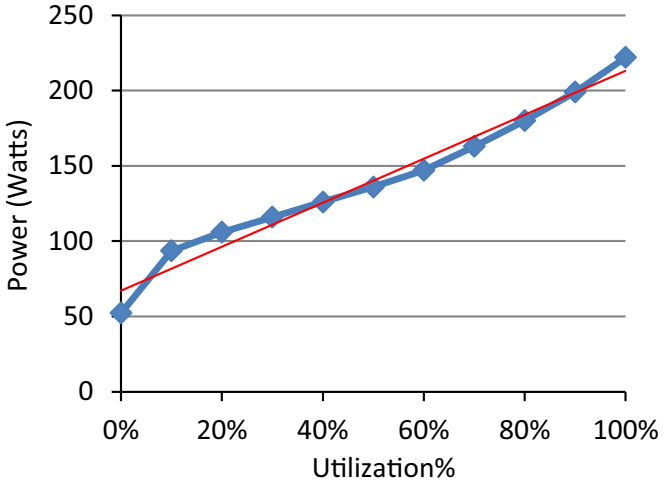

Fig. 5. Power proportionality (HP ProLiant DL380 G7 Intel Xeon X5675) (Standard Performance Evaluation Corporation (SPEC), 2011).

bandwidth (Beloglazov and Buyya, 2012). Normally, servers' in datacenters' do not have direct-attached storage, while the storage is provided by a Network Attached Storage (*NAS*) or Storage Area Network (*SAN*) to enable VM live migration (Beloglazov and Buyya, 2012). With *IaaS* type of environment, knowledge of application workloads and time for which VMs are provisioned is unknown. Our focus is primarily with workload which is CPU bound, and hence server processor characteristics are critical to identify best energy efficient server.

3.2.1. Server heterogeneity controls—classification

A classification of the related research works specifically to server heterogeneity characteristics is shown in Table 3. Also, we improve upon this taxonomy with a detailed study of the contribution and areas for improvements in these works.

Paragon (Delimitrou and Kozyrakis, 2013) is a two staged approach, with a goal to meet application QoS and minimize energy consumption by accounting for resource contention or interference when VMs are hosted concurrently in the same physical server. In the first stage is to find the best server configuration that would run the application fast. In the second stage is to quantify the amount of interference it can tolerate and cause when run with other workloads concurrently in the same physical server without performance loss. The first stage is solved by a classification engine which uses Netflix type recommender solution to identify and recommend the best server configuration for one or more application workloads. The second stage is solved by using pre-constructed micro-benchmarks., comparing the source application workload against the benchmark, and recommend minimally interfering co-runners to the source application workload. Micro benchmark tracks a sensitivity score for the application workload against each of source of interference resources (*SoI*) such as memory (bandwidth and capacity), cache hierarchy (L1/L2/L3 and *Translation Lookaside Buffer—TLBs*) etc. that the workload has contentions with when concurrently processed. To fix any gaps in the benchmark's sensitivity score, the authors have used single value decomposition (*SVD*) and *PQ* reconstruction (*PQ*) techniques. As a composite outcome, for a given workload, Paragon approach recommends the best server configuration and the servers with minimum interference. With respect to solution weakness, this work is a supervised approach requiring training data to be in-place a-priori. This work does not consider server power state transitions times and power overheads in the energy efficiency server provisioning logic.

Moreno et al. (2013) introduced a combined interference score (*CIS*) metric to track performance degradation in processing workloads in a server with concurrent VM usage. VMs hosted in a server share common resources such as memory cache, network interconnects, memory etc. Application workload when hosted in VM in a server, which also has other VMs concurrently in operation exhibits performance degradation due to shared resource contention issue. *CIS* score for a VM is the ratio of performance difference between when the workload is processed in the VM in isolation and the performance when it is processed with another contending VM to the performance when the workload is processed in the VM in isolation. *CIS* score for a server is the net sum of *CIS* scores for all the VMs in it. The authors propose an approach to compute energy efficiency (which signifies the run-time performance to power ratio) of a server and using the empirically pre-derived CIS score for the server to arrive at a weighted energy efficiency (*WEE*) metric value for the server. New workload is assigned to the server with maximum *WEE* metric value.

With respect to solution weakness, this work, considers homogeneous VM configurations. Provisioning and workload classification logic requires knowledge of application. This work does not consider server power state transitions times and power

Table 3
Server heterogeneity level research.

Literature	Target system	System resources	Approach	Evaluation		Results
				Methodology	Workload	
Paragon (Delimitrou and Kozyrakis, 2013)	Server cluster – cloud	CPU RAM Disk Network	Collaborative filtering technique using training data accounting for interference in concurrent VM processing and accounts for server heterogeneity to improve server utilization and to meet QoS guarantees	Prototype Virtualized heterogeneous servers	PARSEC, SPLASH-2, BioParallel, Minebench, SpecWeb,	Utilization increases from 19% to 58% Maintains QoS for 71% of workloads
Samee (Khan et al., 2011)	Server cluster	CPU Memory	Heuristic based approach is to assign the right task to right machine so as to consume minimum energy and maximize task performance using computed score function value on task completion times and machine's processing speed.	Simulation Physical heterogeneous servers	Synthetic workloads	energy efficiency is the best for the consistent instances
Moreno (Moreno et al., 2013)	Server cluster	CPU RAM Disk Network	arrive at energy efficiency factor of each server Arrive at combined interference score based on supervised learning	Simulation Virtualized heterogeneous servers	Google Cluster-Usage Traces	Interference aware approach gives increased energy-efficiency up to 15% when compared with prioritized First-Come, First-served (FCFS) algorithm
Nathuji (Nathuji et al., 2007)	Server cluster	CPU	a prediction component to estimate the performance and power characteristics of various workload to platform mappings, and an allocator which utilizes policies and prediction results to perform decisions	Testbed and simulation Virtualized heterogeneous servers	SPEC CPU2000 suite benchmarks	improves power efficiency by 20% on average

overheads in the energy efficiency server provisioning logic.

Khan et al. (2011) proposed an approach which uses task priority and *computed score function* value on task completion times and machine's processing speed. The objective of this approach is to assign the right task to right machine so as to consume minimum energy and maximize task performance. The authors have assumed that the energy consumption of an idle resource at any given time is set using a minimum voltage based on the processor's architecture, and maximum when the processor is performing work or it is in an active state. Each task has to be processed completely on a single machine. The authors consider an estimated time to complete (*ETC*) model to track the estimation or prediction of the computational load of each task, the computing capacity of each resource, and an estimation of the prior load of the resources. Each position *ETC[task][machine]* in the matrix indicates the expected time to compute task on machine. This model allows representing the heterogeneity among tasks and machines. Energy expended in processing tasks is influenced by tasks *ETC* factor. The authors propose priority and score functions. Priority function deals with task priority, which is based on objective such as maximum or minimum or average task completion times. The score function has two terms, the first term aims to minimize the completion time of the task and the second term aims to assign the task to the fastest machine or the machine on which the task takes the minimum *ETC*. The rationale is to minimize the workload of machines and intrinsically minimize the energy used to carry out the work. The authors propose a low-cost heuristics algorithm which uses score function and the task priority, and finally provides the tasks the machines map detailing on tasks that can be processed on select machines.

With respect to solution weakness, this work, considers physical servers or machines. Application knowledge is required to benchmark task completion-times for a particular application task to be executed in a particular machine. This work does not consider server power state transitions times and power overheads in the energy efficiency server provisioning logic.

Nathuji et al. (2007) leveraged heterogeneity by mapping workloads to the best fitting platform using an analytical layer to predict workload's performance and platform power consumption to process the workload. Goal of this work is to minimize power consumption, while meeting the baseline application performance. They classify heterogeneity into four quadrants with respect to dimensions of microarchitecture heterogeneity and memory subsystem heterogeneity. Other heterogeneity classifications are across-platform heterogeneity, within platform heterogeneity, *DPM*-capability heterogeneity. Performance is observed in terms of rate at which multiple transactions can be sustained or transaction throughput. The authors have proposed an approach which uses the following components viz., *platform* or *workload descriptors*, *power* or *performance* predictor, and *allocator. Platform descriptor* tracks information regarding the hardware and power management capabilities of a server. *Workload descriptor* design allows multiple attribute definitions, where each definition is predicated with component parameter values that correlate back to *platform descriptors*. Role of the *allocator* is to evaluate the power efficiency tradeoffs of assigning a workload to a variety of platforms. The authors use a Blocking Factor (*BF*) model which is used to predict architectural platform properties based on heterogeneity specifications. Energy-efficient allocation framework decides upon *workload-platform* assignments by utilizing right *workload* and *platform descriptors*. After processing the *workload* and *platform descriptors*, and utilizing *BF* model for performance prediction, the final step performs allocation of resources to a set of applications in a datacenter. The authors propose a greedy policy towards allocating workloads. In particular, for each application *i*, they associate a cost metric for executing on each platform type *k*. The *allocator* performs application assignment of applications to the platform, with applications with higher worst-case costs being the priority ones. The platform type chosen for an application is a function of this cost metric across the available platforms as well as the estimated *DPM* benefits.

With respect to solution weakness, this work is application workload dependent. This work does not consider server power state transitions times and power overheads in the energy efficiency server provisioning logic.

Server heterogeneity based power management approaches are still being researched and looked into. Approaches like server profiling based on energy efficiency parameters, use of supervised learning techniques to characterize servers with respect to energy efficiency and workload characteristics are some of the aspects showing progress.

3.3. VM consolidation and deconsolidation controls

Typically systems are provisioned to handle potential peak loads, which very rarely occur; hence all servers in datacenters are unlikely to be fully utilized simultaneously. On an average datacenter servers have an average resource utilization of less than 50%, which represent inefficiency. One solution is consolidating the workloads to fewer physical servers and switching off the idle servers, which improves the server utilization of resources and thereby reduces power or energy consumption. The aim is to improve resource utilization by consolidation of workloads into less number of servers and eliminate server idle or static power consumption. Eliminating static power consumption by servers is only possible by switching servers to off, or to a low-power sleep state. Workload consolidation is a non-trivial problem, since aggressive consolidation may lead to application performance degradation. Dynamic VM consolidation is one area that is finding traction and research problems such as, when should we consolidate VMs into server and when should VM be de-consolidated, when to migrate VMs, which VMs to migrate, and to which target server have to be properly addressed to save energy consumption (Beloglazov and Buyya, 2012). Cloud computing heavily leverages virtualization technology, which has enabled availability of scalable resources and charged for the actual resource usage (Buyya et al., 2009). This flexibility is one of the incentives for organizations to migrate some of their computation activities to a cloud computing environment (Kumar and Raghunathan, 2013). Cloud computing helps improve power efficiency: improves utilization by virtue of server consolidation (switching off servers which are not used), location independence—migration of virtual-machines (VMs) to cheaper energy cost datacenter or server destinations, optimal scaling-in and scaling-out capacities depending on demand, efficient management of resources etc. With the above mentioned many advantages in going with virtualization, there are few areas such as performance degradation due interferences when VM's are co-located to share common server resources (Moreno et al., 2013; Delimitrou and Kozyrakis, 2013; Novaković et al., 2013), that have to be accounted for in solutions architected for virtualization backbone.

3.3.1. VM consolidation and deconsolidation controls—classification

A classification of the related research works specifically to server consolidation is shown in Table 4. Also, we improve upon this taxonomy with a detailed study of the contribution and areas for improvements in these works.

Beloglazov et al. (2012) proposed an architectural framework to minimize power consumption and achieves VM reconfiguration, allocation and reallocation. The authors propose algorithm to dynamically consolidate VMs to reduce global infrastructure power consumption at run-time according to current utilization of

Table 4
Virtualization server consolidation level research.

Literature	Target system	System resources	Approach	Evaluation		
				Methodology	Workload	Results
Beloglazov et al. (2012)	Server cluster *IaaS* Cloud	CPU RAM	Proposed Modified Best Fit Decreasing (MBFD) and VM consolidation approach	Simulation Virtualized heterogeneous servers	Planetlab workloads	MBFD approach gives energy consumption savings in comparison to NPA and DVFS policies—by 77% and 53% respectively with 5.4% of SLA violations.
Srikantaiah et al. (2008)	Server cluster Cloud	CPU RAM, Disk, Network	Heuristic approach is used to maximize the sum of the Euclidean distances of the current allocations to the optimal energy efficient combination of CPU and storage utilization point at each server.	Testbed and simulation Physical heterogeneous servers	Synthetic workload	Best energy depends on servers optimal point.
pMapper (Verma et al., 2008)	Server cluster	CPU	Proposed approach uses minPowerPlacement Algorithm (mPP) which uses first fit decreasing VM placement logic.	Simulation Virtualized heterogeneous servers	LinPack benchmarks	mPP approach gives a reduction of 25% in power consumption when compared to approach using load-balanced and static VM placements
Nathuji and Schwan (2007)	Server cluster	CPU	VirtualPower infrastructure uses complex hierarchical management policies comprised of local policies (PM-L) running on each physical system and driven by local guest VM and platform parameters. PM-L is coordinated by global policies (PM-G) which has the knowledge about rack- or blade-level characteristics and requirements. PM-G policies help with consolidation with migration thus making way for power efficient allocations in heterogeneous environments.	Simulation Virtualized heterogeneous servers	Rubis workload	34% improvements in power consumption
LLC (Kusic et al., 2008)	Server cluster	CPU	Uses Kalman filter to estimate the number of future arrivals, thus helps predict the future state of the system and accordingly performs reallocations in CPU shares and host to VM mappings Considers Switching cost	Testbed and simulation Virtualized heterogeneous servers	Soccer World Cup 1998 Web trace logs	server cluster managed by the controller conserves, on average, 22% of the power required by a system without dynamic control while still maintaining QoS goals
Belaglozov and Buyya (2015)	Server cluster	CPU	VM consolidation approaches simulated in Openstack environment	Simulation Virtualized heterogeneous servers	PlanetLab workloads	Max-ITF approach gives around 33% energy when compared with the baseline approach
Chen et al. (2015)	Server cluster	CPU	Proposes an utilization-based migration algorithm (UMA) to migrate VMs to stable hosts, which efficiently reduces migration time and power consumption	Simulation Virtualized heterogeneous servers	PlanetLab workloads	UMA approach reduces about 77.5–82.4% migrations and saves up to 39.3–42.2% power consumption compared with the MinPower policy
Ma and Zhang (2015)	Server cluster	CPU	Uses a multi-objective optimization approach based on sliding-window and thresholds to decide on when to migrate VM, which VM to migrate is based on VM selection policies, and where to migrate the VM to is based on Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) method.	Simulation Virtualized heterogeneous servers	Unknown workload	Multi-objective approach not only gives lower SLA violations, power consumption, but also has the least number of VM migrations.

resources applying live migration, switching idle nodes to sleep mode. VM placement is done through a modified best fit decreasing (*MBFD*) algorithm in which the VM with highest utilization is placed in the server that provides the least power consumption increase due to this allocation. Due to workload dynamism and factors such as heterogeneous workload types (in *IaaS*), leads to VM usage level changes which in-turn could change server utilization dynamically as-well. VM consolidation process helps optimally place or consolidate VMs into fewer servers thereby improving server utilization, creating an opportunity to switch idle nodes to sleep mode and reducing power consumption. The authors have proposed a set of VM selection algorithms such as minimization of migration time (MMT), minimum utilization (MU) and random choice (RC) to select VMs that need to be migrated, and as the next step the chosen VMs are placed on the hosts using the *MBFD* algorithm.

With respect to solution weakness, this work, uses a different VM migration models, reactive VM reconfiguration actions instead of proactive ones, and they transition times and power consumption to switch on and off servers, and vice-versa.

Srikantaiah et al. (2008) proposed an approach to consolidate applications in cloud computing environment to reduce energy consumption. The authors have studied the inter-relationships between energy consumption, resource utilization, and performance in consolidating workloads. Higher workload consolidation implies an increase in server resource utilization. But, workload consolidation contributes to performance degradation and energy consumption impacts due to high utilization of different resources (CPU, I/O). The authors have found that the energy consumption per transaction results in a “U” shaped curve. When the resource utilization is low, idle power of servers contribute to higher energy per transaction. Also, at high resource utilization, energy consumption is higher due to performance degradation and longer execution time. So, to achieve better energy efficiency, it is required to operate at median resource utilization. The authors empirically show that, there exists an optimal combination of CPU and disk utilizations where the energy per transaction is minimum and within acceptable performance degradation limits. The authors have proposed a heuristic multidimensional bin packing approach to consolidate multi-resource workloads.

With respect to solution weakness, this work is workload type and application dependent and resource requirements of applications are assumed to be known a priori and constant. Node state transition overheads with respect to performance and energy consumption have not been modeled. For each server in the system, server characteristics like optimal or energy efficient resource utilization points have to be empirically determined for it to be used in models.

pMapper (Verma et al., 2008) is an application placement framework in virtualized heterogeneous server cluster environment to minimize power and meet fixed performance requirement. It consists of three *manager* modules and an *arbitrator* module. Manager modules are *performance manager, power manager,* and *migration manager. Performance manager* monitors the applications and resizes the VMs according to the current resource requirements and *SLAs. Power manager* performs hardware power state transition and applies *DVFS. Migration manager* performs VM live migration to consolidate the workload. *Arbitrator* has a global view of the system such as VM placement, VM reallocation to make the placement possible. The authors propose an algorithm called min Power Parity (*mPP*). Goal of this algorithm is to place VMs on set of servers so as to optimally minimize overall power consumed by all the servers. The algorithm takes the following inputs: VM sizes for the current time window that can meet the performance constraints, a previous placement, and the power model for all the available servers. *mPP* has two phases. In the first phase, target utilization is determined for each server based on the power model for the server. The target utilization is computed in a greedy manner, starting from utilization 0 and upwards for each server. A server with the least power increase per unit increase in capacity is picked till all VMs are covered by server capacity allocations. In the second phase, bin-packing algorithm based on First Fit Decreasing (*FFD*) is used to place VMs on the servers, while trying to meet the target utilization on each server. The experimental results showed that with this approach, power savings of about 25% is possible in comparison to the static and load balanced placement algorithms.

With respect to solution weakness, this work requires knowledge of server's power model. Impacts of migration, especially on total power consumption and performance are not dealt with.

VirtualPower (Nathuji and Schwan, 2007) is an online power management approach which performs guest VMs soft level power management state updates to control underlying virtualized hardware. *VirtualPower* supports hardware scaling, soft scaling, and consolidation. Hardware scaling capabilities vary across different platform and device architectures. The virtual power management (*VPM*) mechanisms supporting hardware scaling permits VPM rules to set the hardware states to be used during the execution of a particular guest VM. Soft resource scaling is used when hardware scaling is not always possible, or it may provide only small power benefits. Soft scaling helps in enabling sharing resources between VMs or VM consolidation. *VirtualPower* infrastructure uses complex hierarchical management policies comprised of local policies (*PM-L*) running on each physical system and driven by local guest VM and platform parameters. PM-L is coordinated by global policies (*PM-G*) which has the knowledge about rack- or blade-level characteristics and requirements. *PM-G* policies help with consolidation with migration thus making way for power efficient allocations in heterogeneous environments.

Kusic et al. (2008) proposed an energy cost minimization approach for a virtualized server cluster two-tier application architecture environment under workload uncertainty using sequential optimization limited look-ahead control (*LLC*). Switching cost (transitioning from off to on) incurred while provisioning VMs is considered and also server's not needed during periods of low workload arrivals are powered down and placed in the Sleep cluster to reduce power consumption. Kalman filter is used to estimate the number of future arrivals, thus helps predict the future state of the system and accordingly performs reallocations in CPU shares and host to VM mappings.

Some weakness with this approach is that DVS controls are not considered, this model requires supervised learning for application specific adjustments, number of VMs contributes more to the power consumption rather than resource utilization aspects. Moreover, because of the model's complexity, the execution time is abnormally higher.

Belaglozov and Buyya (2015) have implemented a practical OpenStack framework for dynamic VM consolidation along with DPM controls. DPM sleep mode transitions could not be achieved effectively due to server specific ACPI control deficiency. Using the dynamic VM consolidation approach, the authors have reported a savings of 33% energy consumption over the base approach. This work is the first step OpenStack implementation of dynamic VM consolidation approach to minimize system energy consumption.

This work is the first of its kind to adopt VM consolidation techniques for energy efficiency improvement in OpenStack framework. There are some limitations with respect to using Server sleep state transition overheads. This needs to be accounted for to get the full impact of VM consolidation approach in OpenStack.

Chen et al. (2015) proposed a utilization-based migration algorithm (UMA) to migrate VMs to stable hosts, to reduce both VM migration time and power consumption. The authors have

reported that UMA can reduces about 77.5–82.4% VM migrations and saves up to 39.3–42.2% power consumption compared with the MinPower policy.

With respect to solution weakness, this work does not consider server state transition power and transition latency overheads. Impacts of migration, especially on total power consumption and performance are not dealt with.

VM consolidation techniques are quite understood in virtualization environment. The main factors governing VM consolidation is the impact to performance and energy consumption. There are still some challenges in adopting VM consolidation approach to improve energy efficiency specifically with IaaS service delivery model with no knowledge about the workloads executing in the VMs.

3.4. Datacenter thermal controls

In a datacenter with multitude of IT systems and varying workload processing needs, thermal state of IT components and its surrounding spatial environment tend to change from time to time. Life of a computer system is directly related to its operating temperature. Based on Arrhenius time-to-fail model every 10 °C increase of temperature leads to a doubling of the system failure rate (Hale, 1986). Hence, it is recommended that computer components be kept as cool as possible and within the components permissible upper or critical temperature bounds for maximum reliability, longevity, and return on investment (Deng et al., 2013).

Efficient cooling management process is one which maintains the temperature profile of IT compute systems within the permissible operable range in servicing varying workloads (Emerson, 2007). In effect, cooling system's heat removal capability needs to closely follow the heat-generated by IT compute systems' when processing workloads. Fig. 6 shows temperature points such as $Temperature_{supply}$, $Temperature_{inlet}$, $Temperature_{outlet}$ and $Temperature_{HRC}$ which contribute to the thermal profile in a datacenter. Where $Temperature_{supply}$ is the CRAC's supply temperature, $Temperature_{inlet}$ is the inlet temperature at the rack where the cool air from CRAC blows into, $Temperature_{outlet}$ is outlet temperature at the rack's exhaust from where the hot air blows out after dissipating the heat generated in the server's placed in rack, and $Temperature_{HRC}$ is the thermal heat re-circulation (THR) effect of hot air from rack's outlet/exhaust flows back into rack's inlet side. Rack's inlet temperature $Temperature_{inlet}$ is influenced by CRAC's $Temperature_{supply}$ temperature, self *THR* and *THR* influence from other racks in the datacenter.

Thermal profile in a datacenter is far more difficult to control due to factors such as *THR* effect; fundamental heat flow properties such as conduction, convection and radiation; etc. Hence, any discussion on datacenter cooling capabilities has to be done alongside IT system components and workload processing aspects. Power consumed for cooling is almost on par with that consumed by IT systems to process workload requests. Normally, increase in IT power consumption has a direct increase on cooling power required to cool and maintain ambient temperature of datacenter at a predetermined threshold value. There could be situations, when reduction in IT power consumption (achieved using virtualization—server consolidation) could increase cooling power required to remove thermal hotspots due to generation of lot of localized high-temperature heat (Vijaykumar and Ahmad, 2010). Also, processor leakage power is strongly dependent on temperature—the higher the processor temperature, the more energy lost due to circuit leakage. Use of fans at higher speed is a trade-off that needs to be understood. Operating fans with higher speed consumes more power, which in effect cools better and lowers the chip temperatures and hence lower leakage power (Lee et al., 2012).

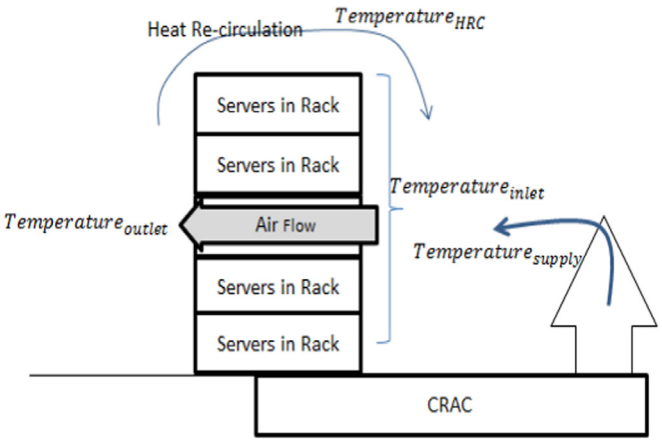

Fig. 6. Datacenter thermal flow.

IT computing power aware server provisioning alone, may not always be effective with modern servers because of the computing vs. cooling power consumption influence trade-off. Given this dependency (both linear and non-linear) between IT and non-IT class and contribution to total datacenter power consumption, any discussion on IT class techniques in isolation without considering non-IT system influence or the vice-versa cannot be considered comprehensive or fully complete. Hence the need for a discussion on Joint IT and non-IT power management.

3.5. Compute and cooling inter dependency

Another problem of datacenter high power consumption and increasing density of server components in datacenter is the heat dissipation. This generated heat has to be dissipated to keep the server within its safe thermal critical limit. In a datacenter, power consumed by non-IT components is around 33–50% and is almost on par with that consumed by servers (Vaid, 2010). Much of the research till now, has focused more towards IT component level power management than non-IT component level power management. The main reason for this focus disparity is due to: (a) higher complexity to model thermal profile and its impacts at both individual server or infrastructure component level and holistic datacenter level; (b) IT level drives the application or system performance service level agreements (SLAs), which normally is the primary or main objective from users and datacenter owners' (DCO) perspective; (c) IT systems' power consumption predominantly influences or drives non-IT level power consumption and related management controls. This focus disparity is even more pronounced around research works at combined IT and non-IT class relationship level. Only recently, has there been a push in research on this combined class.

Datacenter consumes electric power for operating both its servers (IT component) and its cooling systems (non-IT component) which removes the heat dissipated by the servers; each operational IT and non-IT component consumes a specific server and cooling power consumption expend. Active server while waiting for workload requests consumes idle power. Such servers are called idle servers and can be switched off or switched to a low power consumption sleep state to reduce power consumption. Server consolidation is an approach, by which server processes or VMs are optimally migrated between servers with an objective to reduce energy and number of active servers. This consolidation exercise increases server temperature, as these servers operate at higher CPU utilization (upon consolidation) and consumes more IT power and necessitates the need for additional cooling. This consolidation approach in-effect can also create thermal hotspots that

could further increase the temperature of servers in and around the hotspot locale, which additionally increases cooling power required to dissipate the excess heat generated. This brings in the fact that, in a datacenter, processing workload requests by selecting the right server and also the placement location of the servers are both important to minimize overall datacenter power consumption.

3.6. Cooling component layout or placement design

Efficient cooling of a datacenter can be achieved at a high level by adopting (a) proper placement of CRAC units, room dimensions, plenum height, servers, racks etc.; and (b) proper operation to control set-points of CRAC to minimize power consumption. Datacenter layout designs such as traditional raised-floor design, compartmentalization with overhead cooling container, in-row cooling container, circular-in-row cooling container, all-in-box container are possible options available to datacenter owners (Qouneh et al., 2011). With traditional raised-floor design, racks or servers that are nearer to floor (closer to CRAC inlet vent) tend to be in cooler than the racks or servers which are nearer to the ceiling. Placement of layout design that includes CRAC, racks, servers in rack, compartmentalization, inlet, outlet vents etc. are physical in nature and are not likely to change frequently. Proper workload scheduling process which makes use of placement design effectively to reduce maximum temperature of all compute nodes and thermal distribution imbalance; thus control CRAC supply temperature set-points to minimize power consumption.

3.7. Joint thermal and compute aware admission control and scheduling

Datacenter's power consumption is mainly due from IT and non-IT system usage. Combined optimization (minimization) mechanism is required to reduce overall datacenter power consumption and should meet application or system performance SLAs. In a datacenter, the type of compute system used and duration of its usage drives the power consumption of cooling systems. This inter-dependency combined with datacenters' thermal profile implications to be closely understood and handled to minimize datacenter power consumption. We look at few most relevant research works which uses either compute resource usage driven cooling augmentation approach or cooling coverage driven compute workload and resource provisioning approach to minimize the overall datacenter power consumption.

3.7.1. Datacenter thermal controls—classification

A classification of the related research works specifically to joint compute and cooling thermal considerations to achieve power savings is shown in Table 5. Also, we improve upon this taxonomy with a detailed study of the contribution and areas for improvements in these works.

Vijaykumar and Ahmad (2010) proposed a joint optimization technique which focuses on reducing cooling and compute energy consumption wherein it reduces cooling power by spreading load on to many servers in case of hotspot formation (due to server consolidation) and reduces idle power by shutting down unused which translates to fewer servers with concentrated workload. It is a typical case of optimally performing consolidation and de-consolidation exercise with the goal to reduce energy consumption. They proposed a *PowerTrade* method to lower the total energy consumption of active servers, standby servers, and cooling facilities. They also developed a *SurgeGuard* method to maintain an extra number of servers at two time granularities to absorb flash crowd. *PowerTrade* technique has three schemes *PowerTrade-s*, *PowerTrade-s++* and *PowerTrade-d. PowerTrade-s* is a static scheme in which the expected workload and thermal zones (cool, warm and hot) is known a-priori.workload is processed mainly in cool zone first and then the warm zone. Cool zone is less prone to hotspots. Warm zone allows uniform workload distribution. Hot zone is on standby and is used if and only if there are no servers in cool and warm zones to process the workload. Some limitations with *PowerTrade-s* scheme is that it assumes linear relationship between load and temperature and ignores heat exchange between zones due to airflow, gaps with this scheme are that there is chance that cool zone could have hotspots and warm zones could have idling servers, hence this could lead to a suboptimal solution with respect to energy consumption minimization. *powerTrade-s++* scheme solves the first limitation of hotspots formation by allowing multiple reference loadings fixing the zone classification in accordance to temperature change due to workload processing. *powerTrade-d* scheme uses an online approach to dynamically adjust load distribution both across and within zones based on observed temperatures, cooling power, and the observed idle power dissipation. It compares the potential idle power reduction achieved by activating fewer servers against the potential cooling power savings achieved by activating more servers and chooses the option which gives better power savings to reduce the net power. When server utilization increases, application performance response degrades. *SurgeGaurd* scheme overcomes this response time degradation by over-provisioning a limited number of active servers so as to absorb increases in loading.

With respect to solution weakness, this work considers physical servers and requires knowledge of application workload.

Parolini et al. (2011) proposed a set of approaches considering a datacenter to be a cyber-physical system (*CPS*) to minimize energy consumption. A datacenter is considered to consist a IT network level consisting of workload and server capacities etc. referred to as cyber layer and the thermal network level consisting of thermal heat flows, temperature profiles and reference values etc. referred to as physical layer. The thermal and IT networks provide the respective statistics to the workload scheduler, such as air temperature, workload arrival, and execution rate. Apart from job scheduling, it also considers other factors such as job migration between thermal zones and servers, heat flow between zones and servers. Power consumption is dependent on IT network variables such as the desired workload execution rate. The power consumed by servers in the Computing technology (*CT*) network is converted into heat, but the power consumed by the *CRAC* unit is used to supply cool air at a certain temperature. There can be a co-ordination between the performance-oriented IT network and inlet temperature oriented thermal network of devices when making workload placement. Authors have compared the efficacy of the proposed approach using a (a) baseline or worst case approach, (b) independent approach which optimally minimizes thermal and IT power consumption independently, and (c) co-ordinated approach which optimally minimizes both thermal and IT power consumption accounting for thermal and computational coupling aspects. They show that co-ordinated approach gives better savings.

With respect to solution weakness, this work considers physical homogeneous servers. It requires knowledge about the application workload. Workload migration is based on thermal factors more than computational factor.

VMap (Lee et al., 2012) approach for High Performance Computing (*HPC*) cloud datacenters allocates the virtual machines to physical machines using linear programming with a goal to minimize computation power considering the recommended maximum operating temperature of servers. First, the VMs are sorted in decreasing order of their deadlines (or running time). Longer duration VMs are packed together so that server that host smaller duration VMs can be switched off at the completion of

Table 5
Joint compute and cooling controls level research.

Literature	Target system	System resources	Approach	Evaluation		
				Methodology	Workload	Results
PowerTrade (Vijaykumar and Ahmad, 2010)	Datacenter	CPU Cooling	online approach to dynamically adjust load distribution both across and within zones based on observed temperatures, cooling power, and the observed idle power dissipation	Testbed and simulation Homogeneous servers	Fifa WorldCup 1998, Berkeley's home IP server, ClarkNet WWW server, NASA Space Center Web, Saskatchewan's WWW server	combining Power-Trade and SurgeGuard reduces total power by 30%
Parolini et al. (2011)	Datacenter	CPU Cooling	Linear programming optimization	Simulation Homogeneous servers	Unknown workload	Shows savings in comparison graphs.
VMap (Lee et al., 2012)	Datacenter	CPU RAM Disk Network Cooling	Uses linear programming optimization methodology	Testbed and simulation using homogeneous servers	HPC workload RIKEN Integrated Cluster of Clusters (RICC)	VMAP is 9%, and 35% more energy efficient than best-fit and “cool job
Tang et al. (2008)	Datacenter	CPU Cooling	Linear programing based on Genetic Algorithm and Sequential Quadratic Programming	Simulation Using homogeneous servers	Unknown workload	This approach saves 24–35% power compared to Uniform Task (UT) and Uniform Outlet Profile (UOP) approaches
Chen et al. (2010)	Datacenter	CPU Others Cooling	dynamic workload placement (pod controller) and cooling management approach using Local Workload Placement Index (LWPI) based on thermal zone	Prototype Using two type of server configurations	modified version of RUBiS	integrated datacenter management solution can reduce energy consumption of servers by 35% and cooling by 15%
Yeo et al. (2014)	Datacenter	Cooling system	uses ambient temperature aware capping (ATAC) approach. ATAC senses the inlet temperature to reduce the core temperature, ATAC can dynamically cap the performance of the server using DVFS	Simulation using SimWare framework Homogeneous Servers placed in blade chasis	Google production trace	ATAC gives 6%, savings in terms of the total datacenter power.
TAPO (Huang et al., 2011)	Datacenter, Server	Cooling system Server	Heuristics based reactive control theoretic approach at datacenter level (TAPO-dc) and at server level (TAPO-server)	Prototype Homogeneous server configuration	SPECpower workload	TAPO-DC approach can reduce cooling power by up to 12.4–17% and TAPO-server approach can reduce total server power by up to 5.4% with no performance penalty
Piątek et al. (2015)	Datacenter	CPU, RAM Network Cooling	Uses Load balancing, Load balancing with fan management, and Load balancing with fan management and power capping	DCworms simulation framework.	HPL, EP, openssl workloads Harward stress workload	Using fan management gives 10% savings in energy consumption. With cooling and power capping, an additional 5% savings in energy consumption is possible

workload tasks so to save energy. Once the VMs are sorted according to their deadline, VMs are allocated to server whose residual volume (of the hypercuboid) is the lowest of all servers' after assignment. Server heterogeneity is tracked by resources (such as CPU, RAM, Disk, network bandwidth) and heat threshold as dimensions. A heat imbalance model to predict allows us to predict future temperature maps of the datacenter and take proactive management decisions in workload placement. A proactive thermal-aware VM consolidation solution is done, which reduces the number of servers, minimizes energy consumption for computation, increases resource utilization, and improves efficiency of cooling. Another aspect to further reduce energy consumption is to switch off servers which are not in use.

With respect to solution weakness, this work does not consider server power proportionality. It requires prior knowledge about workload deadlines or running times. This work, does not consider transition times and power consumption overhead to switch on and off PMs, and vice-versa.

Tang et al. (2008) investigated the mechanism to distribute incoming tasks among the servers in order to maximize cooling efficiency while still operating within safe temperature regions. The authors have used an approach to reduce cooling energy by using a linear, low-complexity process model to predict the server inlet temperatures in a datacenter based on a server utilization vector. In the next step, using these predicted inlet temperatures, they formalize and solve a linear program or genetic algorithmic model problem of minimizing the maximal (peak) inlet temperature that allocates the workload among servers to reduce datacenter cooling cost.

With respect to solution weakness, this work considers only homogeneous servers.

Chen et al. (2010) proposed a holistic workload, power and cooling management framework in virtualized datacenters based on location-dependent cooling efficiency. It uses various controllers such as Application controller, node controller, pod controller, cooling controller. Application controller, based on multi-tiered application performance models such as end-to-end response time and workload request rate or intensity, arrives at the VM resource demand requirement. Node controller controls each node. It does the allocation sharing for the VM resource in terms of utilization targets required on nodes. Allocation is termed success if the node can satisfy the combined utilization targets of all the VMs. If not, a service level differentiation based on workload priorities is adopted to either delay if the workload is of lower priority. Pod controller manages a pod. Pod comprises of homogeneous set of servers, and there could be many such pods. A pod controller dynamically arranges VM workloads within its pod based on resource requests of the node controllers associated with the server in the pod. Candidate VMs and hosts for workload consolidation are identified by a genetic algorithm process. Consolidation is done via live-migration to reduce the number of servers, and enabling switching off idle servers to save power. Cooling controller dynamically controls the cooling air flow rates and CRAC supply temperatures, in response to the "hot-spots" of the datacenter. Local workload placement index (*LWPI*) value drives the response actions on the cooling power. LWPI indicates how much cooling capacity is available in a certain location of the datacenter, and how efficiently a server in the location can be cooled. Using *LWPI*, the cooling controller through the pod controller migrates workload into more efficiently cooled servers than others.

With respect to solution weakness, heterogeneity in terms of server power proportionality is not considered. It does not consider sleep state transitions and the corresponding transition times and power overheads.

Yeo et al. (2014) proposed ambient temperature-aware capping (ATAC) scheme. ATAC dynamically controls inlet air supply to furnish less cooling air to save cooling energy. ATAC enables each server to use DVFS to control performance so as to maintain inlet air temperature to be lesser than emergency temperature. In terms of cooling power, savings for ATAC, Dynamic thermal management, power capping, and PowerNap are 39%, 28%, 40%, and 1%, respectively. These savings are translated to about 6%, 10%, 7%, and 1% savings in terms of the total datacenter power, including all the components such as computing, fan, and cooling power.

With respect to solution weakness, heterogeneity in terms of server power proportionality is not considered. It does not consider sleep state transitions and the corresponding transition times and power overheads.

Huang et al. (2011) proposed two approaches: (a) Thermal Aware Power Optimization for data centers (TAPO-dc), which switches between two distinct HVAC chiller setpoints (high and low) for a cooling zone based its utilization level and optimizes aggregated HVAC and server fan power; (b) TAPO-server uses runtime measured power to adjust server thermal setpoint and optimize aggregated server fan and leakage power. TAPO-dc can achieve up to a 12.4–17% reduction in total data center power with no performance penalty. TAPO-server approach can reduce total server power by up to 5.4% for a server processor heavy workload with no performance penalty

Weakness with this approach is that the combined technique using both server and datacenter cooling management has not been dealt with.

Piątek et al. (2015) presented models to deduce power usage, temperatures in air-cooled servers, and models to derive have proposed three approaches in analyzing energy consumption usage: (a) Load balancing-Tasks are assigned to nodes in order to balance the load and fans are working at full speed; (b) Load balancing with fan management- same policy as in the case of Load Balancing policy but the speed of fans is adjusted; and (c) Load balancing with fan management and power capping-policy adds to Load Balancing with Fans Management the power capping mechanism. It is implemented as additional frequency downgrading in order to keep the power usage below the given threshold. The authors have experimentally evaluated the effect of fan management on energy consumption which reduced the total energy consumption by 10%. Also, with cooling and maximum power capping, an additional savings in energy consumption of 5% is possible.

This weakness of this work is that load balancing is primarily done based on number of requests and not based on energy efficiency. Also, server DPM control aspect has not been explicitly discussed.

While research efforts adopting integrated compute and cooling power management control approaches show good promise in reducing energy consumption, a lot of these works are conceptual in nature and would need to be evaluated in real datacenter environment to ascertain their efficacy.

4. Conclusion and future directions

With growth in internet usage, increase in number of application services, requirement of data-storage and processing, the size and complexity of operations of modern datacenters' has greatly increased. This increase has led to increase in the power consumption levels of the data centers contributing to significant fraction of modern datacenters' operational expense costs. In a datacenter, cooling and computation infrastructures consumes the major portion of power consumed, where-in compute power consumption has a causal dependency effect and influences cooling power consumption. This dependency has to be factored

properly to reduce total datacenter power consumption. As the complexity of operation of datacenter increases, use of right power management techniques which ensures performance, and cost optimality objectives becomes crucial in modern datacenters.

In this paper, we have highlighted the need for power management in data centers at joint compute (IT) and cooling (non-IT) component levels. Also, we have presented a classification of existing energy efficiency solutions and related research works of a datacenter at IT (compute) and non-IT (cooling) component levels. While many research directions have been studied to save energy, several key problems remain open or partly addressed at a joint IT and non-IT component (i.e., at holistic datacenter) level like: how to schedule workloads or process workloads taking into account of node heterogeneity, workload phase dynamism, thermal profile dynamism, thermal heat re-entry profile, energy efficient placement, rack or container or CRAC thermostat level budget adherence. Not limiting, some of which when solved could give further datacenter energy efficiency gains.

Apart from power management aspects surveyed in this paper, there are areas that could be looked further. First, there is a need to understand application performance degradation and power consumption characteristics due to resource interference caused by workloads processed in concurrent VMs sharing resources like memory cache, network interconnects etc., especially in virtualized environment. Some initial progress has been made in this direction in Delimitrou and Kozyrakis (2013) and Moreno et al. (2013). Second, in case of physical server with multi-way socket cores, understanding of hyperthread-aware power model to attribute power consumption of a server to individual co-running applications is required. Initial work to derive hyperthread specific power models has been proposed in Zhai et al. (2014). Third, in case of multi-data center, to realize the VM migration across the different data centers requires high involvement of network level interaction flows between different entities. Software defined network (SDN) is a new network paradigm that can fulfill the requirements raised by providing flow-level virtualization and network address virtualization aspects, and which could be looked into further to optimize migration needs both from performance and power consumption perspectives.

References

ACPI, 2015. ⟨http://www.acpi.info/DOWNLOADS/ACPIspec50.pdf⟩. [Online] (accessed 25.05.15).

Anderson, D., Dykes, J., Riedel, E., 2003. More than an interface-scsi vs. ata. In: Proceedings of the FAST USENIX Conference on File and Storage Technologies, vol. 2, pp. 245–257.

Annavaram, M., Wong, D., 2012. KnightShift: scaling the energy proportionality wall through server-level heterogeneity. In: Proceedings of the Annual IEEE/ACM International Symposium on Microarchitecture, pp. 119–130.

Barroso, L.A., Holzle, U., 2007. The case for energy-proportional computing. IEEE Comput. 40 (12), 33–37.

Barroso, L.A., Holzle, U., 2009. The datacenter as a computer: an introduction to the design of warehouse-scale machines. Synth. Lect. Comput. Archit. 4 (1), 1–108.

Belaglozov, A., Buyya, R., 2015. OpenStack Neat: A Framework for Dynamic and Energy-Efficient Consolidation of Virtual Machines in OpenStack Clouds. In: Proceedings of the Concurrency and Computation: Practice and Experience (CCPE), vol. 27 (5), pp. 1310–1333.

Beloglazov, A., 2013. Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing (Ph.D. Thesis). University of Melbourne.

Beloglazov, A., Buyya, R., 2012. Optimal online deterministic algorithms and adaptive heuristic for energy and performance efficient dynamic consolidation of virtual machines in cloud datacenters. Concurr. Comput.: Pract. Exp 24 (3), 1397–1420.

Beloglazov, A., Abawajy, J., Buyya, R., 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Futur. Gener. Comput. Syst., 755–768.

Bertran, R., Becerra, Y., Carrera, D., Beltran, V., Tallada, M.G., Martorell, X., Torres, J., Ayguade, E., 2010. accurate energy accounting for shared virtualized environments using PMC-based power modeling techniques. In: Proceedings of the ACM/IEEE International Conference on Grid Computing, pp. 1–8.

Bhatti, K., Belleudy, C., Auguin, M., 2011. Hybrid power management in real time embedded systems: an interplay of DVFS and DPM techniques. Real-Time Syst., 143–162.

Bianchini, R., Rajamony, R., 2004. Power and energy management for server systems. IEEE Comput. 37 (11), 68–74.

Bohra, A.E.H., Chaudhary, V., 2010. VMeter: power modelling for virtualized clouds. In: Proceedings of the IEEE International Parallel & Distributed Processing Symposium (IPDPS), pp. 1–8.

Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I., 2009. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Futur. Gener. Comput. Syst. 25 (6), 599–616.

Chen, Qi, Chen, Jianxin, Zheng, BaoYu, Cui, Jingwu, Qian, Yi, 2015. Utilization-based VM consolidation scheme for power efficiency in cloud data centers. In: Proceedings of the IEEE International Conference on Communication Workshop (ICCW), pp. 1928–1933.

Chen, Y., Gmach, D., Hyser, C., Wang, Z., Bash, C., Hoover, C., Singhal, S., 2010. Integrated management of application performance, power and cooling in data centers. In: Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS), pp. 615–622.

Delimitrou, C., Kozyrakis, C., 2013. Paragon: QoS-aware scheduling for heterogeneous datacenters. In: Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS).

Deng, W., Liu, F., Jin, H., Liao, X., Liu, H., 2013. Reliability-aware server consolidation for balancing energy-lifetime tradeoff in virtualized cloud datacenters. Int. J. Commun. Syst., 1–19.

Dhiman, G., Rosing, T.S., 2009. System-level power management using online learning. IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst., 676–689.

Elnozahy, E.N., Kistler, M., Rajamony, R., 2002. Energy-efficient server clusters. In: Workshop on Power Aware Computing Systems.

Emerson, 2007. Energy Efficient Cooling Solutions for Data Centers. Emerson: Network Power-White Paper.

Enhanced Intel SpeedStep Technology for the Intel Pentium M Processor, 2015. [Online] ⟨ftp://download.intel.com/design/network/papers/30117401.pdf⟩ (accessed 25.05.15).

Fan, X., Weber, W.D., Barroso, L.A., 2007. Power provisioning for a warehouse-sized computer. In: Proceedings of the International Symposium on Computer Architecture (ISCA), pp. 13–23.

Gandhi, A., Harchol-Balter, M., Das, R., Lefurgy, C., 2009. Optimal power allocation in server farms. ACM SIGMETRICS Perform. Eval. Rev. 37 (1), 157–168.

Gandhi, A., Harchol-Balter, M., Raghunathan, R., Kozuch, M., 2012. AutoScale: dynamic, robust capacity management for multi-tier data centers. Trans. Comput. Syst. 30 (4).

Gandhi, A., Kozuch, M., Harchol-Balter, M., 2012. Are sleep states effective in data centers? In: Proceedings of the International Green Computing Conference and Workshops, pp. 1–10.

Gupta, V., Brett, P., Koufaty, D., Reddy, D., Hahn, S., Schwan, K., Srinivasa, G., 2012. HeteroMates: providing high dynamic power range on client devices using heterogeneous core groups. In: Proceedings of the International Green Computing Conference, pp. 1–10.

Hale, P.W., 1986. Acceleration and time to fail. Qual. Reliab. Eng. Int. 2, 259–262.

Huang, W., Allen-Ware, M., Carter, J.B., Elnozahy, E., Hamann, H., Keller, T., Lefurgy, C., Li, J., Rajamani, K., Rubio, J., 2011. TAPO: thermal-aware power optimization techniques for servers and data centers. In: Proceedings of the IGCC.

Ibarra, O.H., Kim, C.E., 1977. Heuristic algorithms for scheduling independent tasks on non identical processors. J. ACM, 280–289.

Intel and Core i7 (Nehalem) Dynamic Power Management, 2015. ⟨https://impact.asu.edu/cse591sp11/Nahelempm.pdf⟩. [Online] (accessed 25.05.15).

Kamga, C.M., 2013. CPU frequency emulation based on DVFS. ACM SIGOPS Oper. Syst. Rev. 47 (3), 34–41.

Kansal, A., Zhao, F., Liu, J., Kothari, N., Bhattacharya, A.A., 2010. Virtual machine power metering and provisioning. In: Proceedings of the ACM Symposium on Cloud Computing (SOCC), pp. 39–50.

Khan, S.U., Diaz, C.O., Guzek, M., Pecero, J.E., Danoy, G., Bouvry, P., 2011. Energy-aware fast scheduling heuristics in heterogeneous computing systems. In: Proceedings of the International Conference on High Performance Computing and Simulation (HPCS), pp. 478–484.

Krioukov, A., Mohan, P., Alspaugh, S., Keys, L., Culler, D., Katz, R., 2010. NapSAC: design and implementation of a power-proportional web cluster. Green Netw., 15–22.

Kumar, M.R.V., Raghunathan, S., 2013. Heterogeneity aware energy minimization workload scheduling approach in cloud computing. Int. J. Appl. Eng. Res. 8, 1789–1806.

Kusic, D., Kephart, J.O., Hanson, J.E., Kandasamy, N., Jiang, G., 2008. Power and performance management of virtualized computing environments via lookahead control. Cluster Comput. – CLUSTER 12 (1), 1–15.

Lee, E., Viswanathan, H., Pompili, D., 2012. VMAP: proactive thermal-aware virtual machine allocation in hpc cloud datacenters. In: Proceedings of the International Conference on High Performance Computing (HiPC), pp. 1–10.

Ma, Fei, Zhang, Lei, 2015. Multi-objective optimization for dynamic virtual machine management in cloud data center. In: Proceedings of the IEEE International Conference on Engineering and Service Science (ICSESS), pp.170–174.

Meisner, D., Gold, B.T., Wenisch, T.F., 2009. PowerNap: eliminating server idle power. In: Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems ASPLOS, pp. 205–216.

Meisner, D., Wenisch, T.F., 2012. DreamWeaver: architectural support for deep

sleep. In: Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems ASPLOS, pp. 313–324.
Mittal, S., 2014. A survey of techniques for improving energy efficiency in embedded computing systems. Int. J. Comput. Aided Eng. Technol.
Mittal, S. Zhang, Z., 2012. EnCache: improving cache energy efficiency using a software-controlled profiling cache. In: Proceedings of the IEEE International Conference on Electro/Information Technology.
Moreno, I.S., Yang, R., Xu, J., Wo, T., 2013. Improved energy-efficiency in cloud datacenters with interference-aware virtual machine placement. In: Proceedings of the IEEE Symposium of Autonomous Decentralized Systems (ISADS), pp. 1–8.
Nathuji, R., Schwan, K., 2007. VirtualPower: coordinated power management in virtualized enterprise systems. ACM SIGOPS Oper. Syst. Rev. 41 (6), 265–278.
Nathuji, R., Isci, C., Gorbatov, E., 2007. Exploiting platform heterogeneity for power efficient data centers. In: Proceedings of the International Conference on Autonomic Computing (ICAC).
Novaković, D., Vasić, N., Novaković, S., Kostić, D., Bianchini, R., 2013. DeepDive: transparently identifying and managing performance interference in virtualized environments. In: Proceedings of the USENIX Annual Technical Conference, pp. 219–230.
Parolini, L., Sinopoli, B., Krogh, B.H., Wang., Z., 2011. A cyber–physical systems approach to data center modeling and control for energy efficiency. Proc. IEEE 100 (1), 254–268.
Patel, C.D., Bash, C.E., Sharma, R. Beitelmal, M. Friedrich, R., 2003. Smart cooling of data centers. In: Proceedings of the Pacific RIM/ASME International Electronics Packaging Technical Conference and Exhibition.
Pedram, M., 2012. Energy-efficient datacenters. IEEE Trans. Comput. Aided Des. 31 (10), 1465–1484.
Piątek, W., Oleksiak, A., Costa, G.D., 2015. Energy and thermal models for simulation of workload and resource management computing systems. Elsevier Simulation Modelling Practice and Theory.
Qouneh, A., Li, C., Li, T., 2011. A quantitative analysis of cooling power in container-based data centers. In: Proceedings of the IEEE International Symposium on Workload Characterization, pp. 61–71.
Ranganathan, P., Leech, P., Irwin, D.E., Chase, J.S., 2006. Ensemble-level power management for dense blade servers. ACM SIGARCH Comput. Archit. News 34 (2), 66–77.
Srikantaiah, S., Kansal, A., Zhao, F., 2012. Energy aware consolidation for cloud computing. In: Proceedings of the USENIX HotPower: Workshop on Power Aware Computing and Systems at OSDI, pp. 10–14.
Standard Performance Evaluation Corporation (SPEC), 2011. SPECpower_ssj2008 Result Benchmarks.
Sueur, E.L., Heiser, G., 2010. Dynamic voltage and frequency scaling: The laws of diminishing returns. HotPower, 1–8.
Tang, Q., Gupta, S.K.S, Varsamopoulos, G., 2008. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: a cyber-physical approach. In: Proceedings of the IEEE Transactions on Parallel Distributed Systems, vol. 19(11), pp. 1458–1472.
Vaid, K., 2010. Datacenter power efficiency: separating fact from fiction. Hotpower.
Verma, A., Ahuja, P., Neogi, A., 2008. pMapper: power and migration cost aware application placement in virtualized systems. In: Proceedings of the Springer ACM/IFIP/USENIX International Conference on Middleware, pp. 243–264.
Vijaykumar, T.N., Ahmad, F., 2010. Joint Optimization of idle and cooling power in data centers while maintaining response time. In: Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), pp. 243–256.
Wang, S., Liu, J., Chen, J.J., Liu, X., 2011. PowerSleep: a smart power-saving scheme with sleep for servers under response time constraint. IEEE J. Emerg. Sel. Top. Circ. Syst., 289–298.
Wang, Y., Deaver, R., Wang, X., 2010. Virtual batching: request batching for energy conservation in virtualized servers. In: Proceedings of the IEEE International Workshop on Quality of Service, pp. 1–9.
Yeo, S., Hossain, M.H., Huang, J.C., Lee. H.S., 2014. ATAC: ambient temperature-aware capping for power efficient datacenters. In: Proceedings of the ACM Symposium on Cloud Computing, pp. 1–14.
Zhai, Y., Zhang, X., Eranian, S., Tang, L., Mars, J., 2014. HaPPy: hyperthread-aware power profiling dynamically. In: Proceedings of the USENIX Annual Technical Conference.