

Energy Adaptive Computing: A New Paradigm for Sustainable IT

Krishna Kant

George Mason University and Intel Corp.

Abstract

The sustainability concerns of Information Technology (IT) go well beyond energy efficient computing and require techniques for minimizing environmental impact of IT infrastructure over its entire life-cycle. Traditionally, IT infrastructure is overdesigned at all levels from chips to entire data centers and ecosystem; the paradigm explored in this chapter is to replace overdesign with rightsizing coupled with smarter control. Such an approach can substantially reduce the emissions impact but poses several challenges in adapting the infrastructure and the workload for the occasional power and thermal limitations that arise. This chapter explains how the paradigm can make information technology environmentally friendlier, and lays out numerous challenges. It also illustrates the ideas via a simple mechanism for enabling server-level power and thermal adaptation in a data center.

1 Introduction

The rapid proliferation of information technology (IT) continues to increase the number of computing devices, their supporting infrastructure and the overall energy consumption. At the chip level, a dramatic increase in wire resistance, increasing device density, and the emerging 3-D integration is conspiring to make power densities unsustainable and heat removal very difficult [32, 10]. Because of increasing miniaturization and computing power,

similar issues arise at higher levels as well. For example, the tight form factors of blade servers, notebooks and PDAs make heat dissipation very challenging to manage. Cooling limitations essentially limit the amount of power that the device can consume. For mobile devices, the increasingly sophisticated processing stresses the battery life further. At the global level, it is expected that in spite of aggressive efforts at enhancing power efficiency of computing systems, the IT power consumption is likely to continue to go up [20] and so is the electricity cost associated with running large data centers.

Many of these issues have been well recognized and have resulted in substantial improvements in energy efficiency at a variety of levels – from low-power HW design to aggressive use of available power modes to intelligent load and activity management (e.g., see [15] and references therein). Coordinated power and thermal management has been examined at multiple levels [27, 23, 22, 21]. These efforts are expected to continue in the foreseeable future. Yet, the sheer increase in the computing base and its environmental impact have raised a number of sustainability concerns. Although important, energy efficiency is only one of many dimensions of sustainable computing. Sustainability also relates to such diverse subjects as minimization of toxic chemicals used in chip fabrication, design for reuse, and easy end-of-life disposability, operation with renewable energy sources that are often variable, and overall minimization of materials and natural resources used. In this chapter, we address the last two aspect and show that they are related.

The fundamental paradigm that we consider is to replace the traditional overdesign at all levels with rightsizing coupled with smart control in order to address the inevitable lack of capacity that may arise occasionally. In general, such lack of capacity may apply to any resource, however, we only consider its manifestation in terms of energy/power constraints. Note that power constraints could relate to both real constraints in the power availability as well as the inability to consume full power due to cooling/thermal limitations. Power consumption limitation indirectly relates to capacity limitation of other resources as well, particularly the dominant ones such as CPU, memory, and secondary storage devices. We call this as *energy adaptive computing* or EAC.¹ The main point of EAC is to consider

¹Here “energy adaptation” implicitly includes power and thermal adaptation as well.

energy related constraints at all levels and dynamically adapt the computation to it as far as possible.

It is important to note that the operational energy consumption does not matter much if the energy comes from renewable sources, the main issue instead is life-cycle impact of the infrastructure required to generate and distribute the renewable energy [2, 8]. A direct use of locally produced renewable energy could reduce the distribution infrastructure, but must cope with its often variable nature. This again comes back to the need for coping with occasional limitation in available power. Thus, better adaptation mechanisms allow for more direct use of renewable energy.

The outline of the rest of the chapter is as follows. Section 2 discusses how energy adaptive computing can help enhance sustainability of computing infrastructure. Section 3 discusses issues of end to end energy adaptation. Section 4 then discusses the challenges posed by EAC. Section 5 discusses some concrete results on energy and thermal adaptation in data centers. Finally, section 6 concludes the chapter.

2 Sustainability and Energy Adaptive Computing

It is well recognized by now that much of the power consumed by a data center is either wasted or used for purposes other than computing. In particular, up to 50% of the data center power may be used for purposes such as chilling plant operation, compressors, air movement (fans), electrical conversion and distribution, and lighting. Furthermore, the operational energy is not the only energy involved here. Many of these functions are quite materials and infrastructure heavy and a substantial amount of energy goes into the construction and maintenance of the cooling and power conversion/distribution infrastructures. In fact, even the “raw” ingredients such as water, industrial metals, and construction materials involve considerable hidden energy footprint in form of making those ingredients available in usable form.

It follows that from a sustainability perspective, it is not enough to simply minimize operational energy usage or wastage; we need to minimize the energy that goes into the

infrastructure as well [8]. This principle applies not only to the supporting infrastructure but to the IT devices such as clients and servers themselves. Even for servers in data centers, the increased emphasis on reducing operating energy makes the non-operational part of the energy a larger percentage of the life-cycle energy consumption and could almost account for 50% [7]. For the rapidly proliferating small mobile clients such as cell-phones and PDAs, the energy used in their manufacture, distribution and recycling could be a dominant part of the life-time energy consumption.

Towards this end, it is important to consider data centers that can be operated directly via locally produced renewable energy (wind, solar, geothermal, etc.) with minimal dependence on the power grid or large energy storage systems. Such an approach reduces carbon footprint not only via the use of renewable energy but also by reducing the size and capacity of power storage and power-grid related infrastructure. For example, a lower power draw from the grid would require less heavy-duty power conversion infrastructure and reduce its cost and energy footprint. The down-side of the approach is more variable energy supply and more frequent episodes of inadequate available energy to which the data center needs to adapt dynamically. Although this issue can be addressed via large energy storage capacity; however, energy storage is currently very expensive and would increase the energy footprint of the infrastructure.

In large data centers, the cooling system not only consumes a substantial percentage of total power (up to 25%) but also requires significant infrastructure in form of chiller plants, compressors, fans, plumbing, etc. Furthermore, chiller plants use a lot of water, much of which simply evaporates. Much of this resource consumption and infrastructure can be done away with by using ambient (or “free”) cooling, perhaps supplanted with undersized cooling plants that kick in only when ambient temperature becomes too high. Such an approach requires the energy consumption (and hence the computation) to adapt dynamically to the available cooling ability. The energy available from a renewable source (e.g., solar) may be correlated with the temperature (e.g., more solar energy on hotter days), and such interactions need to be considered in the adaptation mechanisms.

The power supply and distribution infrastructure can also be significant energy wasters

at all levels. For example, a large data center fed by a 33KV power requires a number of stages of voltage step-down, conditioning, storage and distribution that together can consume up to 10% of the incoming power, which could be in multiple megawatts. Thus reducing this infrastructure enhances energy efficiency. On the side, the power supplies that go into individual servers and other assets can also be big power wasters. For example, a server consuming 500W and sporting a high efficiency power supply with 85% efficiency still wastes 75W of power. (Since this waste is in form of heat, additional power is wasted in removing the resulting heat.) Often, servers run at rather low utilization levels (5-15%), and the power supply efficiency is typically much poorer at lower utilizations. Even worse, for redundancy, the servers may employ two concurrent load sharing power supplies, which means that each of them will never exceed more than 50% load. Smart *phase shedding* power supplies address this problem by providing a number “phases” [12]. As the server utilization dips more and more phases can be turned off, thereby keeping the power supply utilization and efficiency high. For example, a power supply with 8 phases may have all phases active at 90% server utilization, but at server utilization of 45%, only 4 phases need to be active and will still provide the same power supply efficiency. Similar approaches apply to on-board voltage regulators (VR’s). An intelligent mechanism for phase changes that properly accounts for other energy adaptation mechanisms in effect could significantly increase energy efficiency of the data center assets.

Yet another sustainability issue is the overdesign and overprovisioning that is commonly observed at all levels of computer systems. In particular, the power and cooling infrastructure in servers, chassis, racks, and the entire data center is designed for worst-case scenarios which are either rare or do not even occur in realistic environments. Realistic workloads rarely stress more than one resource at a time – for example, it is easy to saturate the CPU when it executes primarily out of the caches, but if a significant memory access activity is involved, CPU will not be able to get the required data quickly enough and will stall. Therefore, taxing CPU and DRAM simultaneously may not even be feasible. It may be possible to saturate both CPU and NIC simultaneously via one or more steady streams of packets of a suitable size, but this may be possible only when it is possible to deposit incoming packets directly into the cache (without a DMA to the memory).

Although data centers are beginning to “derate” specified power and cooling requirements to address this lack of realism, derating alone is inadequate for two reasons: (a) it still must provide significant safety margin, and (b) derating is used only for sizing up the data center power distribution and cooling capacities, not in server design itself. Instead we argue for much leaner design of all components having to do with power/thermal issues: heat sinks, power supplies, fans, voltage regulators, power supply capacitors, power distribution network, Uninterrupted power supply (UPS), air conditioning equipment, etc. This leanness of the infrastructure could be either static (e.g., lower capacity power supplies and heat sinks, smaller disks, DRAM, etc.), or dynamic (e.g., phase shedding power supplies, hardware resources dynamically shared via virtualization). In either case, it is necessary to adapt computations to the limits imposed by power and thermal considerations. We assume that in all cases the design is such that limits are exceeded only occasionally, not routinely.

3 Distributed Energy Adaptive Computing

It is clear from the above discussion that many advanced techniques for improving energy efficiency of IT infrastructure and making it more sustainable involves the need to dynamically adapt computation to the suitable energy profile. In some cases, this energy profile may be dictated by energy (or power) availability, in other cases the limitation may be a result of thermal/cooling constraints. In many cases, the performance and/or QoS requirements are malleable and can be exploited for energy adaptation. For example, under energy challenged situations, a user may be willing to accept longer response times, lower audio/video quality, less up to date information, and even less accurate results. These aspects have been explored extensively in specific contexts, such as adaptation of mobile clients to intelligently manage battery lifetime [11]. However, complex distributed computing environments provide a variety of opportunities for coordinated adaptation among multiple nodes and at multiple levels. In general, there are three types of distributed energy adaptation scenarios: (a) Cluster computing (or server to server), (b) Client-server, and (c) Peer to Peer (or client to client). These are shown pictorially in Fig. 1 using dashed ovals for the included components and are discussed briefly in the following. Notice that in all cases, the network and the

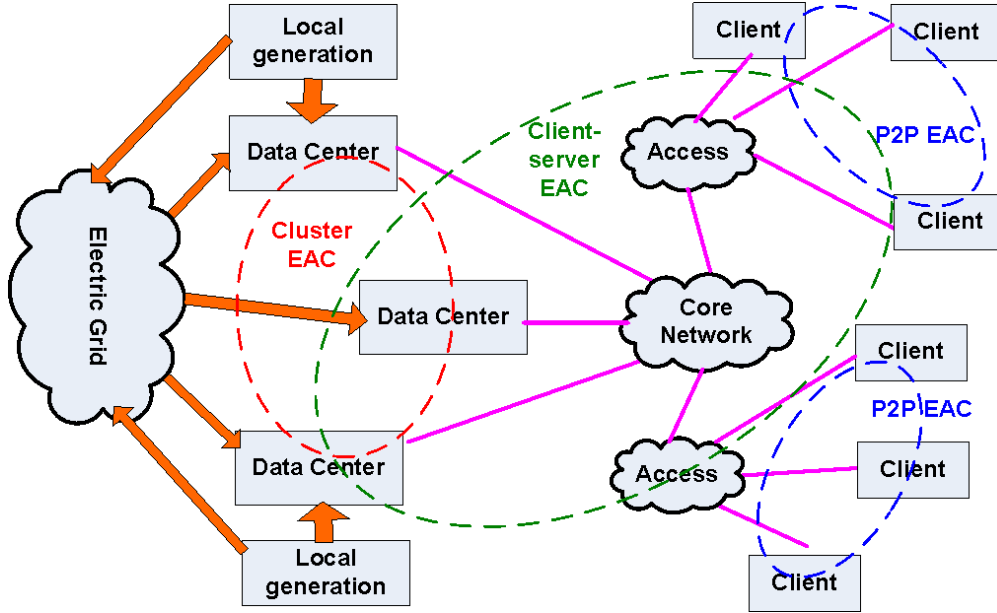


Fig 1: Illustration of energy adaptation loops

storage infrastructure (not shown) are also important components that we need to consider in the adaptation.

Although we discuss these three scenarios separately, they generally need to be addressed together because of multiple applications and interactions between them. For example, a data center would typically support both client-server and cluster applications simultaneously. Similarly, a client may be simultaneously involved in both peer-to-peer and client-server applications.

3.1 Cluster EAC

Cluster EAC refers to computational models where the request submitted by a client requires significant computation involving multiple servers before the response can be returned. That is, client involvement in the service is rather minimal, and the energy adaptation primarily concerns the data center infrastructure. In particular, a significant portion of the power consumed may go into the storage and data center network and they must be considered in adaptation in addition to the servers themselves.

In cluster EAC, the energy adaptation must happen at multiple levels. For example, the

power capping algorithms may allocate a certain power share to each server in a chassis or rack, and the computation must adapt to this limit. In addition, there may be a higher level limit as well – for example, the limit imposed by the power circuits coming into the rack. At the highest level, energy adaptation is required to conform to the power generation (or supply) profile of the energy infrastructure. The limits placed at the lower level generally need to be more dynamic than at higher levels. Translating higher level limits into lower level limits is a challenging problem and requires a dynamic multi-level coordination [27]. A related issue is that of energy limitation along the software hierarchy (e.g., service, application, software modules, etc.) and corresponding multi-level adaptation.

3.2 Client-Server EAC

The client-server EAC needs to deal with both client-end and server-end adaptation to energy constraints in such a way so that client’s QoS expectations are satisfied. A coordinated client-server energy adaptation could even deliver benefits beyond adaptation per se. As the clients become more mobile and demand richer capabilities, the limited battery capacity gets in the way. The client-server EAC can provide better user satisfaction and service by seamlessly compensating for lack of client resources such as remaining battery, remaining disk space, operating in a hot environment, etc. In fact, such techniques can even help slow down client obsolescence and thus enhance the goal of sustainability. Reference [6] considers middleware for reconfiguring software in pervasive computing scenarios in order to adapt to dynamically changing environment and user requirements. Similar adaptation can be done for energy as well.

Client-server EAC can be supported by defining client energy states and the QoS that the client is willing to tolerate in different states and during state switching. This information could be communicated to the server side in order to effect appropriate adaptation actions as the client energy state changes. For a more comprehensive adaptation, the intervening network should also be communicated the QoS requirements and should be capable of exploiting it. The situation here is similar to but more complex than the contract based adaptation considered in [26]. The major challenge is to decide optimal strategy to ensure the desired

end-to-end QoS without significant overhead or increase in complexity. In a client-server computing, the client adaptation could also be occasionally forced by the server-side energy adaptation. Since server-side adaptation (such as putting the server in deep sleep state and migrating the application to another server) can affect many clients, the server side adaptation decisions become quite complex when interacting with a large number of geographically distributed and heterogeneous clients. Involving the network also in the adaptation further complicates the problem and requires appropriate protocol support.

3.3 Peer to Peer EAC

In a peer to peer setting involving increasingly mobile clients, energy consumption is becoming an important topic. Several recent papers have attempted to characterize energy consumption of P2P content sharing and techniques to improve their energy efficiency [14, 18, 19]. Energy adaptation in P2P environment is quite different from that in a client-server setting. For simple file-exchange between a pair of peers, it is easy to consider the energy state of the requesting and serving peers and that of their network connections; however, a collective adaptation of a large number of peers can be quite complex. Furthermore, it is important to consider the fundamental P2P issue of get-give in this adaptation. In particular, if a peer is in a power constrained mode, it may be allowed to be more selfish temporarily (i.e., allowed to receive the appropriate low-resolution content that it needs without necessarily supplying content to others). In a more general situation such as BitTorrent where portions of file may come from different clients, deciding and coordinating content properties and assembling the file becomes more challenging. In particular, it might be desirable to offload some of these functions to another client (that is not in energy constrained mode). In general, addressing these issues requires defining appropriate energy related metrics relative to the content requester, all potential suppliers (or “servers”), transit nodes and the intervening network. A framework that allows minimization of global energy usage while satisfying other local performance and energy requirements can be quite challenging.

4 Challenges in Distributed EAC

In the above, we made little distinction between “energy” and “power”; however, there are subtle differences with respect to both their minimization and adaptation to limits on them. Energy (or equivalently average power over long periods) can be minimized by deliberately increasing power consumption over short periods. For example, it may be possible to minimize energy for a given set of tasks to be executed by running these tasks at the highest speed and then putting the machine in a low-power mode. This “race-to-halt” policy has traditionally been suboptimal because of the possibility of significant voltage reductions at low speeds in the traditional DVFS (dynamic voltage frequency switching). However, as voltages approach minimum threshold for reliable operation, the voltage differences between various available speeds are becoming rather small. At the same time, the idle power consumption continues to go up due to increased leakage current. In such a situation, race-to-halt increasingly becomes an increasingly attractive scheme. Adaptation to an energy constraint can use race-to-halt strategy effectively by deliberately batching up work so that the utilization during the active period and the length of inactive period are both maximized. Notice that a power limit may still require using operation at lower speeds.

There are many situations where simultaneous energy and power constraints may be required. For example, in a data center, the capacity of power circuits (e.g., chassis or rack circuit capacity, or capacity of individual asset power supply) must be respected while at the same time abiding by energy constraints (or constraint in terms of average power over longer periods)

The real energy or power limitation usually applies only at a rather high level – at lower levels, this limitation must be progressively broken down and applied to subsystems in order to simplify the overall problem. For example, in case of a data center operating in an energy constrained environment, the real limitation may apply only at the level of the entire data center. However, this limitation must be broken down into allocations for the physical hierarchy (e.g., racks, servers, and server components) and also along logical hierarchy (e.g., service, application, and tasks). While such a recursive break-down allows independent management of energy consumption at a finer-grain level, an accurate and stable allocation

is essential for proper operation. We address this issue in detail in section 5.

Good energy allocation or partitioning requires an accurate estimation of energy requirements at various layers. Although a direct measurement of energy consumption is straightforward and adequate for making allocations, it only allows reactive (or after the fact) estimation. For example, if additional workload is to be placed on a server, it is necessary to know how much power it will consume *before* the placement decision is made. This is often quite difficult since the energy consumption not only depends on workload and hardware configuration but also on complex interactions between various hardware and software components and power management actions. For example, energy consumed by the CPU depends on the misses in the cache hierarchy, type of instructions executed, and many other micro-architectural details and how they relate to the workload being executed.

A fairly standard method for estimating power is to compute power based on a variety of low-level performance monitoring counters that are available on-chip and becoming increasingly sophisticated. The trick usually is to find a small set of counters that can provide a good estimate of power [4, 5]. While quite accurate, such a scheme does not have much predictive power since the relationship between high level workload characteristics and performance counters is nontrivial. If multiple tasks are run on the same machine, they can interact in complex ways (e.g., cache working set of one task affected by presence of another task). Consequently, neither the performance nor the power consumption adds up linearly, e.g., the active power for two VMs running together on a server does not equal the sum of active powers of individual VMs on the same server. It may be possible to come up with an estimation method that can account for such interference. The situation is similar to that for estimating total bandwidth consumption of multiple applications communicating over a single link. The notion used in this context is “effective bandwidth” [3, 9]. A natural question is whether it is possible to develop a similar notion for power so that it is possible to use simple arithmetic in deciding migration of task and VMs.

As available energy (or power) dips significantly below that required for normal (unconstrained) operation, good energy allocations become progressively more difficult to achieve. These complications arise from the fact that the optimal operating point depends on a variety

of factors including the hardware configurations, nature and importance of the workload, and how frequently the workload characteristics change and interactions between various hardware and software components. The interdependence between various hardware and software components may make their relative energy consumption to change quite substantially as the energy budgets shrink. For example, if CPUs and memory are allocated only 1/2 of their normal power for a workload, the changed workload behavior could make this proportion significantly suboptimal. Thus a continuous monitoring and adjustment to energy needs is required in order to keep energy allocation close to optimal.

When energy availability is restricted, certain applications – particularly those involved in background activities – don't even need to run. Others may run less frequently, with fewer resources, or even change their outputs, and still provide acceptable results. For applications that are driven by client requests and must run, the treatment depends on a variety of factors such as SLA requirements, level of variability in the workload characteristics, latency tolerance, etc. For example, if the workload can tolerate significant latencies and has rather stable characteristics, the optimal mechanism at the server level is to migrate the entire workload to a smaller set of servers so they can operate without power limitations and shut-down the rest. In this case, a tradeoff is necessary with respect to additional energy savings, SLA requirements, and migration overheads.

A comprehensive tradeoff requires accounting for not just the servers but also for the storage and networking infrastructure. As within a single platform, the relative energy consumption behavior between servers, storage and network could change significantly under severe energy constraints and needs to be considered carefully. As the workload becomes more latency sensitive, the latency impact of reconfiguration and power management actions must be taken into account. In particular, if firing up a shut-down server would violate latency and response time related SLA, it is no longer possible to completely shut-down the servers and instead one of the lower latency sleep modes must be used. A less stable workload may also require use of less severe power management actions.

Power management techniques typically take advantage of the low utilization of resources so that idle energy consumption can be minimized. This is done either by putting the devices

into inactive low-power mode when idle (including complete shut-off), or running them at lower frequencies and voltages so as raise the device utilization (i.e., the traditional dynamic voltage-frequency scaling or DVFS controls) [30, 15]. Traffic batching can help reduce the overhead of entering and exiting sleep states [25] at the expense of adding additional latencies. In the past, much of the work has focused mostly on a rather narrow application of these techniques, such as DVFS control of CPUs or nap states for DRAM, but more complex scenarios involving coordinated control of multiple subsystems are beginning to be analyzed [24].

It is important to note that in case of EAC, often the problem is not inadequate work, but rather inadequate energy to process the incoming work. Obviously, in order to reduce the average power consumption, we need to slow down processing, except that this slowdown is not triggered by idling. The basic techniques for slowing down the computation still remain the same and may involve either forcing the device into low-power sleep modes or lower DVFS states. Reference [16] compares the effectiveness of the two methods. However, unlike the situation where the goal is to minimize wasted energy, an energy constrained environment requires careful simultaneous management of multiple subsystems in order to make the best use of the available energy. For example, it is necessary to simultaneously power manage CPU, memory and IO adapters of a server in order to ensure that the energy can be delivered where most required.

In addition to power management, the inability to process all of the incoming workload may require some additional load management actions to avoid build up of long queues. In a high-performance computing type of environment driven by long running jobs, delaying completion of running jobs or startup of new jobs usually has no further consequences. In a transactional system driven by user requests, further actions in form of dropping requests, redirecting them to another facility, migrating away entire applications, or reducing processing requirements at the cost of degraded output quality may be necessary. All of these actions require an accurate mechanism for evaluating “before” and “after” energy requirements for making intelligent decisions. The difficulty here is that because of interference between workloads, power consumptions don’t necessarily add up, as already mentioned

above.

The admission control, migration and power management of a large number of resources at multiple levels raises a lot of interesting issues in terms of the stability and optimality of the control in addition to the issues of the overhead and lag associated with information exchange. A comprehensive control theoretic framework is required in order to address these issues [23, 31]. When the control extends over multiple physical facilities, perhaps each with differing energy costs, the problem becomes even more complex.

Although much of the above discussion concerns servers, similar issues apply to clients and their subsystems. For example, the partitioning and control of power between CPU, memory, storage and other portions of a client involves the same set of issues as servers. However, the peer-to-peer interaction between clients involves some unique issues as already stated above.

Although much of our discussion has focused on servers and clients, storage and network also need serious consideration in energy adaptive computing because of increasing data intensiveness of most applications. Energy management of rotating magnetic media often involves long latencies (in spinning down or spinning up the drives) and reliability issues resulting from RPM changes or repeated starts and stops. The emerging solid state storage (SSD) can be helpful in this regard. The energy management of network devices such as switches and routers is inherently difficult because of its nonlocal impact. For example, if a router/switch port is placed in a low power mode, every application and endpoint whose traffic goes through this port will be affected. Reference [1] discusses the notion of “shadow ports” to allow queuing of packets coming into ports that are in low power mode, but lower level hardware techniques can also ensure that packets arriving into an inactive port are not lost.

When the network power management is triggered by shortage of available power (as opposed to idle or low traffic conditions), the impact is much more severe, since the energy management will result in accumulation of packets and significantly increase flow latencies. The end-to-end admission control required to manage the traffic needs to carefully manage these latencies, performance impact on various applications with varying latency sensitivity,

and application timeouts. A significant amount of work remains to be done to address these issues adequately.

While the topic of applications changing their behavior in the face of energy limitations has been explored in several specific contexts such as audio/video streaming, rendering a web page, P2P content sharing [11, 19], and mechanisms to specify and manage the adaptation have been proposed [26, 29], there is scope for considerable further work on how and when to apply various kinds of adaptation mechanisms (e.g., lower resolution, higher latency, control over staleness and/or accuracy, etc.) under various kinds of power/thermal limitation scenarios.

The main theme in EAC has been to cut down “fat” at all levels and thereby lower not only the direct energy consumption but also the entire life-cycle energy costs that are essential to examine from a sustainability perspective. This leanness has a down-side: the increased fragility in the system which can be exploited by attackers. In particular, just as current systems can be victimized by denial of service (DoS) attacks, the systems proposed here can be further victimized by denial of energy (DoE) attacks. For example, it is possible to craft “power viruses” whose aim is to consume as much power as possible. A carefully planned attack using such viruses can significantly disrupt a distributed EAC scheme and lead to instabilities and poor performance. Protection mechanisms against such energy attacks are essential to realize the EAC vision.

5 Energy Adaptation in Data Centers

In this section we briefly describe results for client-server EAC considered above. The details of the scheme, called *Willow*, are discussed in [17] and covered here only briefly. Willow provides a hierarchical energy adaptation within data centers in response to both demand and supply side variations. Although a comprehensive energy adaptation scheme could include many facets, the current design of Willow is geared towards load consolidation and migration. These mechanisms may need to be augmented with workload alteration such as shutting down of low-priority tasks, change of codecs, or batching of event handling;

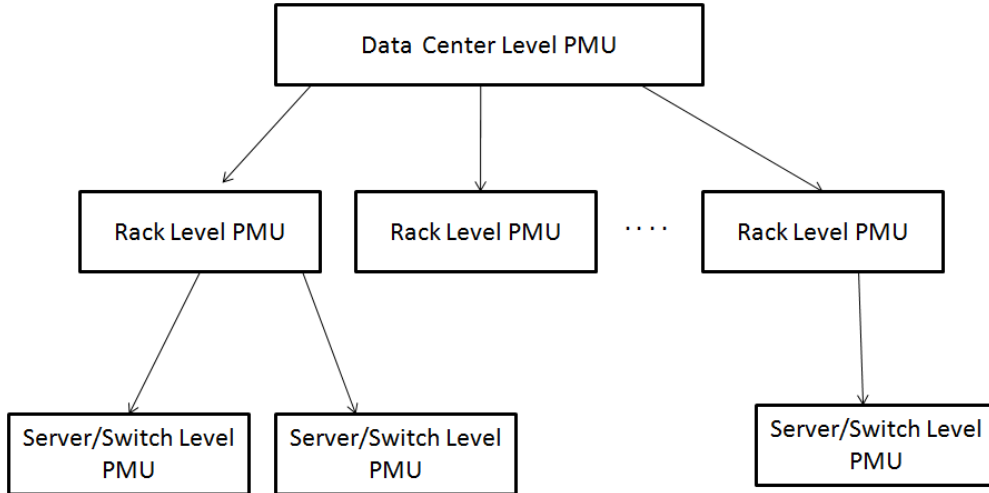


Fig 2: A simple example of multi-level power control in a datacenter

however, since these aspects are very much workload dependent, they are not considered explicitly. Willow does consider adaptation to thermal constraints by considering them in energy-equivalent terms.

5.1 Multi-level Power and Thermal Control

As stated in section 3.1, power/thermal management is required at multiple levels from individual devices within an asset up to the entire data center. For example, it is possible to “shift” power within a server between, say, CPU, memory, and NIC, depending on the low-level workload phase [28]. This can be accomplished by a coordinated control of available operational states of these devices so that the power efficiency can be maximized. The higher levels in this hierarchy include individual assets (servers, switches, routers, etc.), chassis and/or racks, clusters, and the entire data center. In a power limited situation, each level will need to operate under a suitably determined and dynamically varying “power budget”. Figure 2 shows shows such a structure along with a power management unit (PMU) at each level.

In the hierarchical power control model that we have assumed, the power budget in every level gets distributed to its children nodes in proportion to their demands. The component in each level $l+1$ has configuration information about the children nodes in level l . For example

the rack level power manager requires knowledge of the power and thermal characteristics of the individual assets in the rack. The components at level l continuously monitor the demands and utilization levels and report them to level $l + 1$. This helps level $l + 1$ to continuously adjust the power budgets. Level $l + 1$ then directs the components in level l as to what control action needs to be taken.

Because of the fluctuating demand and workload characteristics, the power consumption will also fluctuate and any demand estimation and corresponding adaptations need to be discretized over a suitable time granularity, henceforth denoted as Δ_{Dl} . It is assumed that this time granularity is sufficiently coarse to accommodate accurate power measurement and its presentation, which can be quite slow. Henceforth, we denote consumed power as CP_l at level l . Typically, appropriate time granularity at the level of individual servers are of order of 100s of ms. It may be desirable to use an even coarser granularity at higher levels such as rack or cluster. The consumed power estimate can be further smoothed by using a simple exponential smoothing over successive “slots” of duration Δ_{Dl} .

Since the supplied power could also fluctuate in general, a similar discretization applies to the supply side power estimation as well. The time-constant for supply changes, and hence the discretization period, Δ_{Sl} is typically much larger than Δ_{Dl} . This is typically a result of temporary energy storage in UPS batteries or capacitors in the supply path. For convenience, we assume $\Delta_{Sl} = \eta_1 \Delta_{Dl}$, where η_1 is suitably chosen integer > 1 .

Energy efficient operation is an essential aspect of coping with energy deficient scenarios, and they are also valuable during energy surplus periods. In view of this, our Willow system also performs workload consolidation when the demand in a server is low. This allows some servers to be put in a deep sleep state such as S3 (suspend to memory) or even S4 (suspend to disk). Since the activation/deactivation latency for these sleep modes can be quite high, the corresponding time constant, denoted Δ_{Al} , is even higher. In particular, we assume that $\Delta_{Al} = \eta_2 \Delta_{Dl}$, where $\eta_2 > \eta_1$ is an integer constant.

5.2 Supply and Demand Side Adaptations

Let BP_{l+1}^{old} be the overall budgeted power at level $l + 1$ during the last period and BP_{l+1} the overall power budget at the end of current period. If the difference between two is too small to have much of a performance impact, we can either leave the level l budgeted powers unchanged or simply alter the budget of the node with highest allocation. In other cases, we need to reallocate the power budgets of nodes in level l . In doing so we consider both *hard* and *soft* constraints.

1. Hard Constraints are imposed by the thermal and power circuit limitations of the individual components. In our control scheme the thermal limits play an important role in deciding the maximum power that can be supported in the component.
2. Soft Constraints are imposed by the division of power budget among the other components in the same level.

The reallocation of budgets at level l is somewhat different depending upon whether we have a decrease or increase at level $l + 1$. In case of a decrease, the following actions are initiated:

1. If there are any overprovisioned nodes their surplus power is taken away to the extent of satisfying the deficit. If the overall surplus exceeds the deficit, no further action is necessary.
2. In case of unsatisfied deficit, any potential workload modification (e.g., shutting down of low priority tasks) is attempted.
3. If the deficit still persists, a reallocation is attempted following the method described below.

In case of budget increase, the following actions are initiated.

1. If there are any underprovisioned nodes they are allocated just enough power budget to satisfy their demand. (Note that if the an admission control is currently in effect to reduce the demand, it needs to be removed or loosened.)

2. If the surplus power still exists, any actions such as freezing of background tasks or workload modifications in effect (e.g., lower resolution) are undone.
3. If surplus is still available at a node then the surplus budget is allocated to its children nodes proportional to their demand.

The demand side adaptation to thermal and energy profiles is done systematically via migrations of the demands. We assume that the fine grained power control in individual nodes is already being done so that any available idle power savings can be harvested. Our focus in this paper is on workload migration strategies to correct any imbalances in energy supply and demand. For specificity we consider only those type of applications in which the demand is driven by user queries and there is minimum or no interaction between servers, (e.g., transactional workloads). The applications are hosted by one or more virtual machines (VMs) and the demand is migrated by migrating these virtual machines. A finer grain control is possible by controlling the query fraction directed towards multiple virtual machines serving the same application; however, such an approach is not always feasible and not considered here.

There are a few considerations in designing a control strategy for migration of demands.

1. *Error Accumulation*: Because of the top-down subdivision of power budgets, any errors and uncertainties get more pronounced at lower levels.
2. *Ping-Pong Control*: A migration scheme that migrates demand from server A to B and then immediately from B to A due to erroneous estimations leads to a ping-pong control. A ping-pong wastes energy and introduces latencies.
3. *Imbalance*: The inaccuracies in estimation could leave some servers power deficient while others have surplus power.

We avoid these pitfalls by a variety of means including *unidirectional control*, explained in the following, and allowing sufficient margins both at the source and the destination to accommodate fluctuations after the migrations are done.

The *unidirectional* control initiates migrations only when power constraints are tightened. Furthermore, the migrations are initiated in a bottom up manner. If the power budget $BP_{l,i}$ of any component i is too small then some of the workload is migrated to one of its sibling nodes. We call this a local migration. If the sibling nodes of component i do not have sufficient surplus to accommodate its excess demand then the workload is migrated to one of the children of another $l + 1$ component. We call this a non-local migration. Local migrations are always preferred to non-local migrations for two reasons:

1. The overheads involving networking resources are reduced when the migrations are local rather than non-local.
2. The VMs might have some local affinity with common resources like hard disks or SANs and a non-local migration might affect this affinity.

The migration decisions are made in a distributed manner at each level in the hierarchy starting from the lowermost level. The local demands are first satisfied with the local surpluses and then those demands that are not satisfied locally are passed up the hierarchy to be satisfied non-locally. The final rule in the unidirectional control scheme is that the migrations are destined to a node only if the power availability of the node is not reduced by the event that caused the migration. For instance if the power availability of a rack has gone down, no migrations are allowed into that rack. Similarly if the power availability of the entire data center has reduced no migrations can happen immediately. (Migrations may still happen if as a result of budget reductions we decide to shut down certain tasks and consequently increase the power availability).

In order to mitigate against random short-term fluctuations in supply and demand, a migration is done if and only if the source and target nodes can have a surplus of at least P_{min} . Also, a migration does not have the secondary effect of splitting the demand between multiple nodes. Finally Willow also does some consolidation. When the utilization in a node is really small, the demand from that node is migrated away from it and the node is deactivated.

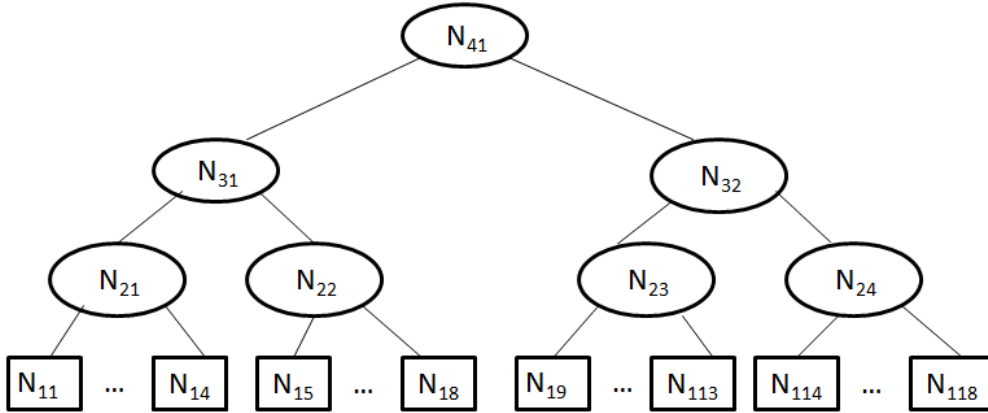


Fig 3: Configuration of the Simulated Data Center

The problem of migration coupled with load consolidation can be formulated as a bin packing problem with variable sized bins. Here the bin size represents the current The variable sized bin packing problem is a NP-hard problem as such and numerous approximation schemes are available in literature [13]. We choose one such simple scheme called FFDLR [13]. The FFDLR successively packs the bins in the order of largest bin first. The net result is packing in minimum number of bins.

FFDLR solves a bin packing problem of size n in time $O(n \log n)$. The optimality bound guaranteed for the solution is $1.5 \cdot \text{OPT} + 1$ where OPT is the solution given by an optimal bin packing strategy. We chose this algorithm for two reasons. First, it is simple to implement with guaranteed bounds. Second, the repacking into the fewest (and largest) bins means that the workload is migrated to the fewest servers so that the others can be shut down.

5.3 Experimental Results

In this section we report some experimental results for Willow. The results are obtained primarily via simulation but with parameters determined via actual experiments. Table 4 shows the measured server power consumption as a function of CPU utilization for a CPU bound workload. Not surprisingly, the idle power consumption is quite substantial and increases monotonically with utilization.

To illustrate the impact of the proposed scheme, a tiny data center consisting of 18 servers

Utilization%	Avg. Power (W)
0	160
20	175
40	190
60	215
80	230
100	245

Fig 4: Measured server power consumption vs. utilization

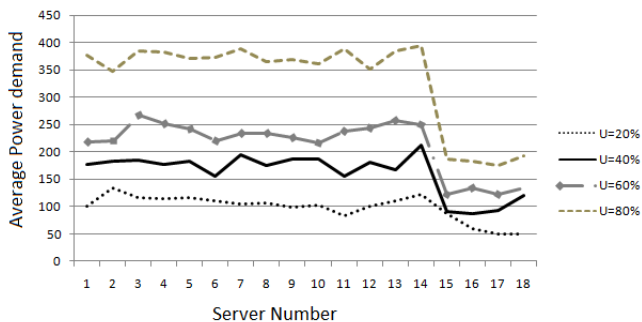


Fig 5: Power Consumption at Various Utilization Levels

was simulated in MATLAB. The servers are arranged in 4 levels as shown in Fig. 3. Each server runs a random mix of 4 different application types with relative processing requirements of 1,2,5 and 9 respectively. We assume that all the applications are transactional, driven by user queries and the query arrival process is Poisson. We assume the multiplicative constants $\eta_1 = 4$ and $\eta_2 = 7$. We assume an average server power consumption of 450 Watts, typical ambient temperature to be 25C and thermal limit of 70C. We assume, for simplicity, that the slot size Δ_{DI} is large enough (typically 100’s of ms) to accommodate CPU temperature changes associated with demand changes. Fig. 5 shows adaptation to situation where servers 1-14 are in a “cool” zone of 25C ambient temperature whereas the rest are in “hot” zone of 40C ambient temperature. The figure shows the average power consumption under various utilization settings as a function of server number. As expected, the control effected to keep the server temperatures below the acceptable limit results in much lower allowable power consumption for server 15-18.

Since Willow attempts to both consolidate the workload as well as abide by power constraints, the migrations are driven by both factors. Fig 6 shows that the migrations are driven by consolidation at lower utilizations and by power constraints at higher utilizations. Fig. 7 shows the proportion of migration trafec as a fraction of normal job arrival rate. This can be interpreted as the probability that a job would need to be migrated. It is seen that the migration probability increase with utilization at first due to more activity in the system and reaches a maximum around 50% utilization level. Beyond this, the opportunities

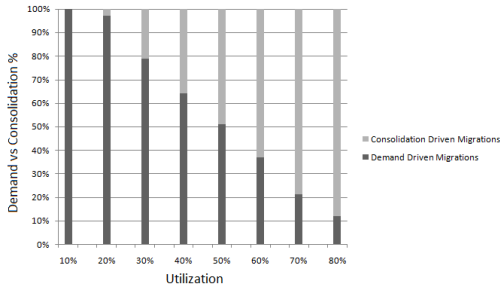


Fig 6: Relative Fractions of Demand vs. Consolidation Migrations

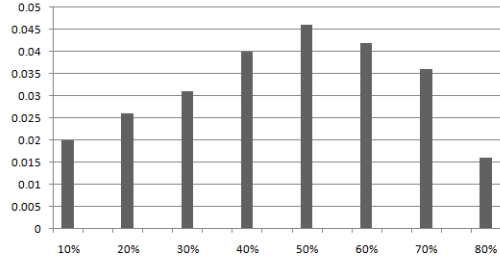


Fig 7: Migration Probability vs. Server Utilization

for migration begin to dry up due to lack of servers with adequate surplus to accept the migrated jobs. At very high utilization levels, it is no longer possible to cope with low power availability periods via migration, and load shedding (not shown) becomes essential.

6 Conclusions

In this article, we discussed the notion of energy adaptive computing that attempts to go beyond achieving energy savings based on the minimization of device idling and energy wastage. One of the goals of energy adaptive computing is to make IT more sustainable by minimizing overdesign and waste in the way IT equipment is designed, built and operated. As pointed out in this article, such an approach brings multiple new challenges in the energy management of IT systems that need to be explored more fully.

As an illustration, we considered energy adaptation in a data center and devised a simple algorithm that is stable, low-overhead and can handle significant variations in energy availability. The stability in the decision making directly translates into reduced networking impact since there are no ip-ops in migrations.

The area of energy adaptive computing can be explored in many different directions including the issues pointed out among the challenges. The data center adaptation reported here can be refined further to consider applications with different QoS requirements running together. Since migrations are often necessary because of normal resource availability issues

such as computing power or memory, it is necessary to coordinate such migrations along with those that are done as a result of power/energy limitations so that the overall QoS objectives can be satisfied.

Acknowledgements: The author wishes to acknowledge the contributions of Mr. Muthukumar Murugan in the data center energy adaptation work reported in section 5.

References

- [1] G. Ananthanarayanan and R.H. Katz, “Greening the switch”, Proc. of HotPower, 2008.
- [2] S. Boyd, A. Horvath, et al., “Life-cycle energy demand and global warming potential of computational logic”, Env. Sci. Tech., 2009.
- [3] C-S Chang, J.A. Thomas, “Effective bandwidth in high-speed digital networks”, Vol 12, No 6, pp1091-1100, Aug 2002.
- [4] Y. Cho, Y. Kim, S. Park and N. Chang, “System-Level Power Estimation using an On-Chip Bus Performance Monitoring Unit”, Proc. of ICCAD 2008.
- [5] G. Contreras and M. Martonosi, “Power prediction for Intel XScale processors using performance monitoring unit events”, Proc. of ISLPED 2005, pp. 221226, 2005.
- [6] A. Corradi, E. Lodolo, S. Monti and S. Pasini, “Dynamic Reconfiguration of Middleware for Ubiquitous Computing”, Proc. of 3rd Intl. workshop on adaptive and dependable mobile ubiquitous systems, London 2009.
- [7] J. Chang, J. Meza, P. Ranganathan, et.al., “Green Server Design: Beyond Operational Energy to Sustainability”, Proc. of HotPower 2010.
- [8] [32] A.J. Shah, C.D. Patel and V.P. Carey, “Exergy-based metrics for sustainable design”, IEEEES-4, 2009.

- [9] A.I. Elwalid and D. Mitra, “Effective bandwidth of general Markovian traffic sources and admission control of high speed networks”, *IEEE/ACM trans. on networking*, Vol 1, No 3, pp329-343, Aug 2002.
- [10] P. Emma, E. Kursun, “Opportunities and Challenges for 3D Systems and Their Design”, *IEEE Design & Test of Computers*, Vol 26, No 5, 2009, pp6-14
- [11] J. Flinn and M. Satyanarayanan, “Managing battery lifetime with energy-aware adaptation”, *ACM trans. on computer systems*, Vol 22, No 2, May 2004, pp 137-179
- [12] D. Freeman, “Digital Power Control Improves Multiphase Performance”, *Power Electronics Technology*, Dec 2007 (www.powerelectronics.com).
- [13] D.K. Friesen, M.A. Langston, “Variable sized bin packing”, *SIAM J. of Computing*, 15, 1, 1986, pp 222–230.
- [14] S. Gurun, P. Nagpurkar, B.Y. Zhao, “Energy Consumption and Conservation in Mobile Peer-to-Peer Systems”, *Proc. of Intl. conf. on mobile computing and networking*, Sept 2006.
- [15] K. Kant, “Data Center Evolution: A Tutorial on State of the Art, Issues, and Challenges”, *Elsevier Computer Networks Journal*, Dec 2009.
- [16] K. Kant, “Distributed Energy Adaptive Computing”, *Proc. of ACM Hotmetrics*, 2009, *ACM Performance Evaluation Review*, Vol 37, No 3, Dec 2009
- [17] K. Kant, M. Murugan and D. Du, “Willow: A Control System For Energy And Thermal Adaptive Computing”, *Proc. of IPDPS 2011*.
- [18] I. Kelenyi and J.K. Nurminen, “Energy Aspects of Peer Cooperation Measurements with a Mobile DHT System”, *Proc. of ICC 2008*.
- [19] I. Kelenyi and J.K. Nurminen, “Bursty content sharing mechanism for energy-limited mobile devices”, *Proc. of 4th ACM workshop on Perf. monitoring and measurement of heterogeneous wireless and wired networks*, 2009, pp 216-223.

- [20] J.G. Koomey, “Worldwide electricity used in data centers”, Environmental Research Letters, Vol 3, 2008.
- [21] J. Moore, J. Chase, P. Ranganathan & R. Sharma, “Making scheduling cool: temperature-aware workload placement in data centers”, Proc. of USENIX Annual Technical Conference (Berkeley, CA, USA, 2005),
- [22] R. Nathuji, K. Schwan, “Virtualpower: coordinated power management in virtualized enterprise systems”, Proc. of SOSP 07, pp. 265-278.
- [23] X. Wang and Y. Wang, “Coordinating power control and performance management for virtualized server clusters”, IEEE Trans on Parallel and Distributed Systems, (2010).
- [24] S. Mohapatra and N. Venkatasubramanian, “A game theoretic approach for power aware middleware”, Proc. of 5th ACM/IFIP/USENIX Intl. conf. on Middleware, Oct. 2004
- [25] A. Papathanasiou and M. Scott, “Energy Efficiency through Burstiness”, Proc of the 5th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA’03), pp. 44-53, Oct 2003.
- [26] V. Petrucci, O. Loques, D. Moss, “A framework for dynamic adaptation of power-aware server clusters”, Proc. of 2009 ACM Symposium on Applied Computing (Honolulu, Hawaii). SAC ’09. pp 1034-1039.
- [27] R. Raghavendra, P. Ranganathan, et.al., “No Power Struggles: Coordinated Multi-level Power Management for the Data Center”, Proc. of 13th ASPLOS, Mar 2008.
- [28] P. Ranganathan, P. Leech, D. Irwin and J. Chase, “Ensemble-level Power Management for Dense Blade Servers”, Proc. of ISCA 2006, pp 66-77.
- [29] J.P. Sousa, V. Poladian, D. Garlan, et al., “Task based adaptation for ubiquitous computing”, IEEE trans. on systems, man & cybernetics, 2006.
- [30] V. Venkatachalam and M. Franz, “Power Reduction Techniques for Microprocessors”, ACM computing surveys, Vol 37, NO 3, Sept 2005, pp 195-237. (<http://www.ics.uci.edu/~vvenkata/finalpaper.pdf>)

- [31] Y. Wang, K. Ma, and X. Wang, “Temperature-constrained power control for chip multi-processors with online model estimation”, SIGARCH Comput. Archit. News 37, 3 (2009), 314324.
- [32] B.P. Wong, A. Mittal, Y. Cao and G. Starr, *Nano-CMOS Circuit and Physical Design*, John Wiley, 2005, chapter 1.