Underprovisioning the Grid Power Infrastructure for Green Datacenters

Xu Zhou†*, Qiang Cao†*$, Hong Jiang‡, Lei Tian‡, Changsheng Xie†*
† School of Computer, Huazhong University of Science and Technology, Wuhan, China
* Wuhan National Laboratory for Optoelectronics, Wuhan, China
‡ Dept. of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE, USA
$ Corresponding Author: caoqiang@hust.edu.cn

ABSTRACT

While there have been prior studies on underprovisioning the power distribution infrastructure for a grid-based datacenter, how to save grid capital investment by means of leveraging renewable energy to underprovision the grid power infrastructure in green datacenters remains largely an unexplored, open issue. Aggressively underprovisioning grid infrastructure can trigger power emergency in which simultaneous peak power draws across the datacenter exceed the tightly budgeted grid power capacity, leading to possible serious consequences including power shutdown. The resulting power emergency mandates a graceful reaction mechanism to sustain the power requirement to avoid power overdraw. While leveraging renewable energy in a green datacenter provides a possibility to prevent power overdraw during such a power emergency, the intermittent nature of renewable energy makes it a very challenging task because of the potentially unpredictable performance impact on individual applications. This paper addresses this issue by designing a novel renewable energy delivery infrastructure and considering performance consequences to individual applications of underprovisioning the grid power infrastructure in the presence of varied renewable power and limited battery's energy capacity in a datacenter. We build an experimental prototype to demonstrate such grid power underprovisioning on a cluster of 10 servers with a simulated solar power generator. Using representative datacenter benchmarks to evaluate the effectiveness of the renewable solution in handling power emergencies, we show that renewable energy by itself can sustain different duration lengths of power emergency when its supply is sufficient. Batteries play an important role in performance boost when the supply of the renewable energy is insufficient. Our theoretical solution provides a seamless bridge across the whole spectrum of duration lengths of power emergency.

Categories and Subject Descriptors

C.0 [**Computer Systems Organization**]: General

Keywords

Green Datacenters; Grid Power Infrastructure; Underprovision; Power Management; Battery; Renewable Energy

ICS'15, June 8–11, 2015, Newport Beach, CA, USA.

http://dx.doi.org/10.1145/2751205.2751222.

1. INTRODUCTION

To tackle the challenges brought by power consumption and carbon emissions, many datacenters have begun to seek renewable energy such as solar or wind energy, along with innovative energy-efficient technologies, to reduce the cost and environmental impact. Several IT giants such as Apple, Google, Microsoft and Yahoo! have proposed plans to build on-site "green" datacenters that will be powered by renewable energy sources. Due to the intermittent and variable nature of the renewable energy, these green datacenters have to deploy grid utilities as a backup to complement and compensate for the intermittency of the green power [1, 2].

Unlike the grid power provision in traditional datacenters that must meet the entire datacenter power demand, the grid infrastructure in green datacenters is called upon only when there is no sufficient green power. Consequently, given the backup characteristics, provisioning a cost-effective grid power infrastructure for green datacenters poses interesting research challenges. *Should we provision the grid power infrastructure to supplement the green power according to the peak power requirement of a datacenter?* In a typical datacenter, the capital cost spent on provisioning the grid utility infrastructure is between $10-$25 for each watt [3], no matter that watt is actually consumed or not. Recent studies [4, 5, 6, 7, 8] also show that provisioning for the theoretical peak (using face-plate ratings of the equipment) has proven very expensive, contributing to over a third of the amortized cost of a large datacenter. A study of power consumption on a Google datacenter [3] shows that the probability of the datacenter's power demand exceeding 90% of the potential peak is less than 1%. Given the increasing incorporation of the renewable energy into green datacenters, the actual power requirement for the grid in such datacenters is likely to gradually reduce.

But *how should the grid power infrastructure be underprovisioned in a green datacenter while ensuring an acceptable performance for individual applications hosted there?* In particular, underprovisioning the grid infrastructure for a green datacenter can result in power emergency when the simultaneous power draw across the datacenter exceeds the much tighter grid power budget leading to serious consequences including power shutdown. Figure 1 shows how three cases of power emergency might be coped with when underprovisioning the grid infrastructure. In Case A, the green power supply complementing the grid power can fully satisfy the surged power requirement. In Case B where the green power supply is insufficient for the power demand surge, the batteries, strategically charged when there is a surplus of power supply from either the renewable or grid source, discharge to make up for the power shortage. In Case C where all available power sources are unable to meet the power demand surge from the current workloads, certain power management knobs (e.g., DVFS, clock throttling and workload/data migration, etc.) can be adopted to relegate the power requirement to match the current power provision, potentially resulting in a performance degradation for some applications. Obviously, underprovisioning the grid infrastructure can help significantly save the capital expense, provided that the power emergencies that arise in such an underprovisioned grid infrastructure can be addressed in a seamless fashion.

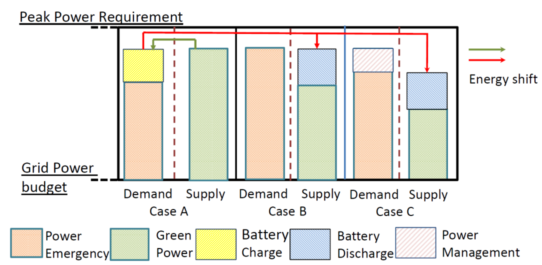

Figure 1: Power Emergency for Underprovisioned Green Datacenter.

To this end, we attempt to explore the interplay among the renewable energy, batteries, grid energy and power management techniques to cost-effectively and gracefully deal with the power emergencies in green datacenters with a dedicated underprovisioned grid power infrastructure. When using renewable energy to underprovision the grid-based power infrastructure, we have several choices for a fixed target of underprovisioning that include: (a) varying the renewable-energy provisioned power capacity; (b) varying the energy capacity of the batteries; and (c) a combination thereof. With reduced grid capacity, application performance may be impacted in case of a power emergency if the supplies of renewable power and battery are insufficient. There are different alternatives to avoiding power overdraw, such as consolidation and power throttling [9, 10, 11, 12, 15]. In this paper, we use the term availability to loosely refer to renewable energy *availability* of the datacenter during a power emergency and we quantify it in our proposed scheduling mechanism.

To the best of our knowledge, while much prior arts have focused on underprovisioing the grid power distribution infrastructure, the issues of underprovisioning the grid capacity in the presence of green energy and its cost-benefit trade-off have not been addressed. In exploring the design space trade-offs of employing renewable energy for underprovisioning grid power infrastructure, this paper makes the following contributions. (1) We propose a novel renewable power distribution architecture that enables the underprovisioning of the grid power infrastructure in green datacenters. (2) We present a theoretical framework that integrates several key power management mechanisms (e.g., renewable energy, batteries, power states, workload migration, etc.) to find a performance-optimal way of handling a power emergency for a given degree of grid power underprovisioning. (3) We develop an experimental prototype consisting of 10 servers, a simulated solar power generator, and a server-level battery provision, coupled with existing software and hardware mechanisms, to demonstrate and evaluate our proposed approach to underprovisioning the grid infrastructure for green datacenters. Using representative datacenter workloads, we evaluate the effectiveness of different mechanisms such as throttling and workload migration on suppressing power emergencies. Our renewable solution is shown to provide a seamless bridge across all duration lengths of power emergency and be able to minimize the performance impact.

2. BACKGROUND AND MOTIVATION

In this section we provide the necessary background to motivate our proposed research.

Datacenter Power Underprovisioning: The high cost of power provisioning and consumption in datacenters draws attentions to underprovision the power infrastructure [13, 9, 14, 15, 3, 10, 6, 5, 7, 16, 17, 18]. Prior studies to peak power reduction primarily rely on: (a) power saving via active processor power states and inactive deep sleep states, (b) workload shaping via migration or distribution; and (c) peak power shaving via energy-storage/battery based mechanisms. While sharing a common goal of reducing power consumption, leveraging *green* power to underprovision and *existing* power underprovisioning without green power are quite different in that: (i) Unlike existing underprovisioning mechanisms that do not leverage renewable energy, we focus on designing a renewable energy delivery architecture to underprovision the grid power capacity, the intermittent nature of the *green* power means that both the intrinsic characteristic of green power and the energy storage capacity must be considered simultaneously. (ii) This characteristic of the green mechanism calls for power management mechanisms to handle the varied green power supply and limited energy storage capacity, as opposed to handling only limited energy capacity in the existing underprovisioning mechanisms. Efficiently utilizing green power becomes as important as capping power peaks. (iii) Green energy as a power source, as opposed to energy storage that is considered as an *energy buffer* in the state-of-art underprovisioning mechanisms, has the potential to provide a long-term power supply. This may lessen the performance impact of and ensure seamless operation for long-duration power emergencies.

Real or Prototype Implementations of Green Datacenters: The prototype of Blink [19] modulates the motherboards' duty cycles to deal with energy variability without the grid provision. HP Labs aim to build a *net-zero* datacenter with clusters of servers partially provisioned by renewable power system [20]. The grid utility is still used to compensate for the intermittent unavailability of renewable energy. In addition to focusing on workload management, attention is also paid to the cooling power. Researchers from Rutgers University have built a solar-powered micro-datacenter called Parasol [21] whose workloads and energy sources are managed by a system called GreenSwitch. Parasol schedules non-critical and interactive workloads for utilizing renewable energy.

Since our work focuses on reducing grid capacity in the presence of renewable energy, we choose to connect the renewable energy system at the power distribution unit (PDU) level, allowing us to underprovision the grid power capacity on a per-rack basis. Parasol and *net-zero* focus on the datacenter-level power integration, which cannot support grid capacity underprovisioning very well because the underprovision level determine the grid capital expenditure saving. Moreover, we choose not to synchronize the renewable energy with grid power because it provides a convenient way to scale the renewable power capacity provision to be commensurate with the degree of underprovision.

Our work leverages both renewable energy availability and existing power management mechanisms to handle power emergencies and investigates the performance impact of different renewable energy provision schemes. The grid capacity is overprovisioned in the Parasol and *net-zero* designs. They use the grid as backup without forcing power management knobs to shave the power peaks to avoid power overload when renewable energy and battery are insufficient.

Li et al. proposed a real solar-powered server-level prototype called *Oasis* [8] to ensure power capacity scale out economically and sustainably as datacenter reaches the power capacity. In contrast to *Oasis*, the underprovisioned grid power capacity calls for a novel renewable energy delivery infrastructure to exploit the green power capacity provision, so as to supplement the tighter grid capacity to achieve the underprovision goal while considering the varied green power generation and the potential power emergencies. Green energy and battery in our scheduling mechanisms are called upon only for occasional power peaks, unlike the totally green and power scale-out designs that have more frequent green power demands.

Scheduling for Green Energy: While some researchers have studied the scheduling of batch jobs to maximize the use of renewable energy [22, 23], others have proposed to adapt the amount of batch processing dynamically to a mix of interactive and batch workloads in datacenters [24]. SolarCore [12] leverages per-core DVFS on multi-core systems to temporarily lower server power demand when solar power drops. Li et al. [11] proposed an architecture in which two sets of servers draw power respectively from two different sources (e.g., the grid or a wind farm). In their setup, energy source management entails migrating workloads between the two sets. In [25], Li et al. investigated the benefits of the load following mechanism in distributed generation powered datacenters and tailor datacenter power demand for improving renewable energy utilization.

By leveraging the above existing approaches to improve the efficacy of renewable energy, our contributions lie in evaluating the performance trade-offs with underprovisioning grid power infrastructure, the handling techniques used during the power emergencies, different availability of renewable energy and durations of power emergency and diverse application characteristics.

3. DESIGN PRINCIPLES

3.1 Overview of An Underprovisioned Infrastructure

Figure 2 depicts an architectural overview of an on-site green datacenter power hierarchy with an underprovisioned grid power infrastructure. To achieve the underprovisioning goal, we directly connect the on-site renewable power supplies such as photovoltaic (PV) and wind to the power distribution unit (PDU) level to provide the dual-power supply of grid and renewable power rather than integrate the renewable energy into the utility power. The PDU-level renewable energy provision allows us to underprovision the grid power capacity on a per rack basis. Moreover, the choice decreases the impacts of voltage transients, frequency distortions and harmonics on the grid utility. For power emergencies, there are servers powered only by the renewable energy with a separate green power bus while the other servers will depend on the utility power. As we aggressively push the underprovisioning design to the rack level, there is potential for additional cap-ex savings compared to the higher levels. As shown in the figure, grid power capping at the server level is key to achieving capital expenditure saving at different levels of the power hierarchy (including the rack, PDU, diesel-generator and utility levels).

Our design targets the three key objectives: (1) effectively utilize the reduced grid power capacity across the entire dynamic power load range while reducing the cap-ex of the grid power infrastructure; (2) gracefully employ renewable power to sustain the power emergencies; and (3) provide a convenient way to scale the renewable power capacity provision to be commensurate with the degree of underprovision. Since the grid power capacity is underprovisioned, the renewable power can join the PDU via an inverter device, which can combine electricity produced from the provisioned green power and the underprovisioned grid power in the same circuit for power supply. Note that, the PDU connected with green power is provisioned as normal capacity (i.e., theoretical nameplate-based provision) and the capacity of the PDU allows for the infusion of the renewable energy. The DC renewable power is converted to AC to match the output level of the grid power that is routed to the rack-level power distribution strip that further powers a cluster of server nodes. However, the PDU for the utility-dependent racks is capacity underprovisioned. Upon increasing computing demand, more renewable power capacity can be added and underprovisioned PDU capacity of utility-dependent racks will be adjusted as well. When we approach the next upgrade, the probability of power emergencies will be higher. The solutions like this can help mitigate the overload problem during these periods.

Since we connect the green power to the PDU level, we leverage the distributed battery architecture shown in Figure 2, which is widely employed by IT companies such as Google [26] (server-level battery) and Facebook [27] (rack-level) to smooth the supply of the renewable power. The Facebook design uses a cabinet of batteries to provide standby power for a rack triplet to replace the centralized battery. The Google design attaches a battery to every server after the power supply unit (PSU). This design could achieve energy efficiency through bringing the AC distribution (green and grid) even closer to the IT load, before it is converted (we adopt this design in our solution).

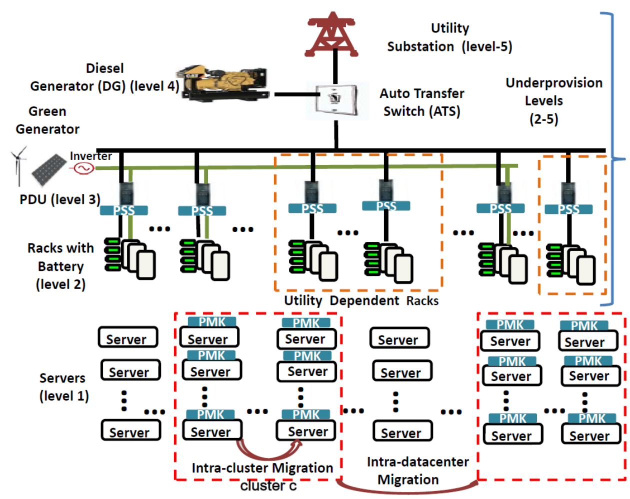

Figure 2: The Underprovisioned Power Infrastructure.

The advantage of distributed battery architecture over the centralized battery architecture are threefold. First, it provides a convenient way to dynamically scale the battery system with the power capacity of the renewable power provision. Second, it allows a fine-grained way for activating a battery to match the incremental power draw, thus, as each server/rack exceeds the underprovision grid capacity, a distributed battery would be brought on line. Third, it improves energy efficiency by avoiding AC-DC-AC double conversion.

To cope with the time-varying, intermittent renewable power, we employ a power-source selector (PSS) to adaptively switch among different forms of power supplies (i.e., green, battery and underprovisioned grid power). PSS performs switch tuning based on the discrepancy between the workload power demand and the grid power provision. PSS is also configured to charge the battery when there is an excess of green power, and discharge it when green power is insufficient or unavailable. Once the battery is fully charged, excess green power is used for the reduction of carbon emission and electricity bills in a conventional way. PSS is able to identify the switching parameters for the inverter and charge controllers of batteries to allow for a full control of every power source. Programmable power electronics circuitry can be used to implement the PSS. However, since our experimental setup does not provide this functionality, our evaluations account for such control capability in the form of managing the underprovision decisions, which is detailed in Section 4.1. As shown in Figure 2, a server-level power management knob (PMK) receives the execution output from the PSS to control the power demand on a per-server basis. When renewable power and battery are not sufficient, PMK leverages the existing hardware/kernel/VM interfaces to keep servers within the grid power budget by considering the application's diverse characteristics. As shown in the figure, we can dynamically change the CPU power states and trigger the intra-cluster and intra-datacenter workload migration to reduce the power demand for each server to be commensurate with the reduced grid power capacity. We dynamically change the settings for these scheduling behaviors according to our power emergency handling mechanisms as elaborated in Section 4.2.

The underprovision design is orthogonal to the design consideration for operational expenses (op-ex) (e.g., cutting electricity bills), and we leave the work on op-ex reduction to our future work.

3.2 Design Space

Rather than relying on additional renewable energy and battery provision, we propose to use renewable energy and battery that already exists in the green datacenters. The performance of applications is impacted by how much the renewable energy and batteries are provisioned. In particular, we consider a conservative renewable-energy provision mechanism that powers part of the datacenter using renewable energy sources. We assume that the number of renewable-energy powered servers is fewer than 40% of the overall deployed machines since a datacenter typically consumes about 60% of its actual peak power [3]. The rationale behind this assumption is fourfold. First, it allows the datacenters to sufficiently use the grid power capacity. For instance, when we underprovision 30% of the peak grid capacity (name-plate based provision), the overall deployed servers can be supplied by 70% of the peak grid capacity that comprises the majority of power requirement in a datacenter. Second, power emergencies can be reasonably handled since part of the servers is powered by renewable energy. For instance, the power emergency, along with the 30% grid underprovision, can be handled by the green provision since 30% of the deployed servers can draw power from renewable energy. Third, even if the green power is extremely low, there still exists a large portion of the peak grid capacity (e.g., 70% of peak grid capacity) that can be made available for all servers upon a power emergency. Finally, the conservative renewable energy provision could allow us to achieve further grid cap-ex reduction (PDU capacity) since more utility dependent racks will exist (as shown in Figure 2). We conduct a preliminary investigation on the impact of allowing additional renewable energy provision and the resulting cap-ex savings achieved by grid power infrastructure underprovisioning (Section 5.5). A detailed study of such green provisioning cost-performance tradeoffs is considered as the future work.

We vary the renewable energy provision and battery energy capacity to investigate the performance impact of a given underprovision degree (e.g., 30% of peak grid capacity). Table 1 shows the various green configurations for the underprovisioning of the grid power capacity. Current green datacenters employ the provisions of renewable energy and batteries [20] merely as a mechanism to reduce the op-ex. They still rely on the overprovisioned peak grid capacity as a backup and offer seamless performance during power emergencies. We use this setting as a performance baseline (*MaxPerf*) for comparison.

Renewable power capacity: We denote $Peak_{RE}$ as the peak renewable power provision for which a single photovoltaic module can sustain the associated underprovision needs. We use the value from [28] to estimate the DC-AC derate factor α (0.77) of the solar power. For example, when we provision k servers with renewable energy, theoretical maximum renewable power AC output is $k * Peak_{RE} * \alpha$. The actual available solar power output, denoted by $REPower$, is equal to the percentage of the maximum theoretical solar power production, which is actually determined by the solar irradiation, ambient temperature, and the server load connected to it.

Battery behavior: We model a server-level 12V value-regulated lead-acid battery (VRLA), similar to that used by Li et al. [25]. Battery is characterized by their supply time as approximated by Peuker's Law (Peukert's exponent is 1.15 for LA battery [7]), which shows the time taken to drain a certain capacity for different power demands. We exploit this property to force a lower power demand for a longer battery supply time. Depth of discharge (DoD) is a critical parameter determining a battery's lifetime. We assume DoD=40% in our setup, which translates to a lifetime of 1300 recharge cycles [7]. If the energy used to charge a battery is higher than what can be drawn out subsequently, it implies power loss or an energy efficiency η (75% for LA [6]) of less than 100%. We choose two battery capacities to smooth the renewable energy variance as shown in Table 1, where 3.2Ah is the existing server-level battery capacity deployed in the Google datacenter as Uninterrupted Power Supply (UPS) [7].

Lifetime concerns: We need to ensure that we do not compromise on datacenter's power reliability with renewable energy provision. Recent works point out that solar panels can last for 25 years (lifetime) with a degradation rate of less than 1% loss per year [29]. Moreover, reports in different areas [30, 4, 25] reveal that batteries can significantly improve the power availability of datacenters. In order to guarantee reliability, one requirement of our approach is to ensure that the replacement of batteries must be not earlier than their expected lifetime. From our study, the proposed combination of renewable energy and battery provisions is able to handle the relatively frequent power emergencies within the expected lifetime of renewable energy and battery systems.

Performance implications during power emergencies: Underprovisioned grid power infrastructure has multiple performance and power availability ramifications. We use the term *availability* to loosely indicate the available renewable energy. Further, the performance of services or applications are greatly affected by: (i) the variance of the weather and environment dependent renewable energy can greatly affect the performance of service or application due to the varying *renewable peak power*, (ii) the performance of services or applications can be impacted by the *insufficient energy capacity* in batteries and the limitation of their lifetime during power emergencies, (iii) impact to both performance and availability of power sources due to the occurrence time of a power emergency.

The occurrence time of a power emergency during the execution of a workload can determine both the power source selection and the impact to the application performance for the following cases. (a) When a power emergency occurs at a time when there is no green power, i.e., the *availability* is minimum, the underprovisioning mechanism must depend solely on the batteries. The batteries have capacity limitation, which can result in performance degradation. (b) When a power emergency happens at a time when the *availability* is maximum, the renewable power can sustain the power emergency by directly supplying green power to

Configurations	RE	Batt. (Server level)
RE-Batt	30% servers	10Ah
REOnly	30% servers	0
RE-SBatt	30% servers	3.2Ah
SRE-SBatt	20% servers	3.2Ah

Table 1: Options for green provision. "RE" represents renewable energy provision. "S" stands for *small* renewable energy and battery energy capacity provisions

servers without using batteries to boost the performance while simultaneously charging batteries with excess green power. (c) Given the large variance of the renewable power, servers associated with green power cannot draw sufficient power from renewable energy (i.e., the *availability* is medium), where performance of services or applications can be impacted by both the time-varying renewable power supply and batteries energy capacity.

3.3 Problem Statement

We endeavor to answer the following question, that is, given a grid power budget of P_{budget} with underprovision at the cluster level where the cluster c is composed of the racks shown in Figure 2 and accommodates n servers, *how can we cap the grid power under the given power budget while meeting the applications' SLA?* We denote $GridLoad^{A}_{c,t}$ as the aggregate grid power demand of cluster c in epoch t for application A, then the underprovisioning of the grid power capacity exploits property $GridLoad^{A}_{c,t} \leq P_{budget}$, $\forall t$, ($GridLoad^{A}_{c,t} = \sum_{j=1}^{n} P_{j,t}$, $P_{j,t}$ represents the grid power draw of each server j in epoch t). The underprovisioning mechanism with renewable energy can strike a tradeoff between the performance of applications and the cap-ex savings. Since renewable power is varied and the power supply of battery is constrained by the energy capacity, it is possible that the aggregate grid power of the n servers will not be safely accommodated by the design. As a result, we use techniques provided by PMK to deal with the power overdraw to meet the reduced power budget P_{budget}.

4. HANDLING POWER EMERGENCIES WITH RENEWABLE SOLUTIONS

In this section, we describe a theoretical framework for managing power sources and scheduling workloads. The framework is used to assist the power source selector (PSS) and power management knobs (PMK) in their decision-making process.

4.1 Power Source Selector

Goal: The power source selector enables the execution of applications with the renewable energy and/or battery to stay within the tighter grid power budget. PSS relies on the information of the power demand, renewable energy *availability*, and the amount of energy stored in the battery to produce a power-source schedule to handle the power emergency.

Figure 3 illustrates how PSS operates in general. In the figure, the period of a power emergency is divided into discrete scheduling epochs $(t_1, t_2, ..., T)$ and at each epoch the tighter grid power provision is achieved by powering the cluster from renewable energy or battery. The figure shows the four distinct states of the cluster at each epoch t, namely, (i) renewable power is abundant and can be independently used for power emergency handling and the excess power can be used to charge the battery; (ii) renewable power is not sufficient but battery can be activated to supplement the green power to sustain the power emergency over a period of time; (iii) battery can be activated independently to sustain the power emergency when the renewable power is unavailable; and (iv) renewable power is not available while the grid power can be used to charge the battery for handling power emergency at a later time. However, utility dependent rack is only allowed to charge its battery by grid power as shown in Figure 2. Next, we propose a general framework to help PSS make the right decision in each epoch t. Our PSS scheduler is inspired by an online energy sources switching mechanism for op-ex reduction [21].

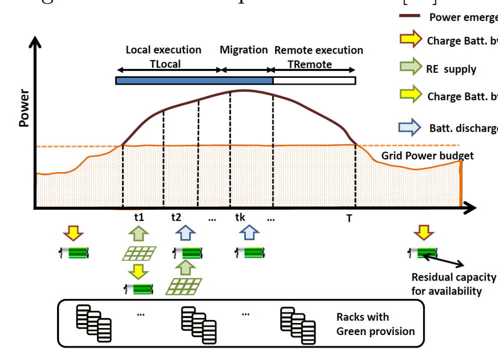

Figure 3: Renewable energy and battery provide power to the cluster to meet the cluster-level power budget requirement for each epoch t during the period of a power emergency. The shaded region indicates the grid power used. Migration happens at t_k instantaneously in this illustration as power sources do not suffice while throttling could be combined with power sources scheduling.

Power sources: Upon an emergency, three power sources can be used to meet the aggregate power demand $PowerD_{c,t}$ of cluster c for each epoch t , namely, the renewable power ($RELoad_{c,t}$), the battery power ($BattLoad_{c,t} = \sum_{j=1}^{n} BattLoad_{j,t}$, $BattLoad_j$ represents the power supply of the battery associated with server j), and the grid power ($GridLoad_{c,t}$). Thus,

$$\forall t \in T:$$
$$RELoad_{c,t} + BattLoad_{c,t} + GridLoad_{c,t} = PowerD_{c,t}$$

Recall that in order to meet our underprovision goal P_{budget}, we must ensure that the grid power demand of cluster c is within P_{budget}:

$$\forall t \in T : GridLoad_{c,t} \leq P_{budget}$$

The available renewable power ($REPower_c$) may be used for multiple purposes, including directly sustaining for handling a power emergency ($RELoad_c$) and charging the battery associated with green provision ($Batt\_RE_{c,t}$). Thus,

$$\forall t \in T : RELoad_{c,t} + Batt\_RE_{c,t} \leq REPower_{c,t}$$

where $Batt\_RE_{c,t} = \sum_{gi=1}^{k} Batt\_RE_{gj,t}$, gj expresses the server with green provision and k is the number of servers with green provision ($k \leq n$).

Similarly, the grid ($GridLoad_c$) can be used to charge each battery j in cluster c ($Batt\_Grid_j$) during a non-emergency period.

Since we choose not to synchronize the renewable power to the grid power line, we cannot use the grid power and renewable power simultaneously for the server gj provisioned with renewable energy:

$$\forall t \in \{T | RELoad_{gj,t} > 0\} : GridLoad_{gj,t} = 0$$

Battery behaviors: Battery charging itself with green power during an emergency can help deliver a good performance and we also account for the battery's energy efficiency (η), i.e., only a fraction (η) of the energy we store in the battery will be available. We define the current available capacity of the battery ($BattCap_j$) at epoch t as:

$$\forall t \in T : BattCap_{j,t} = BattCap_{j,t-1} + Duration \times \eta$$
$$\times Batt\_RE_{j,t-1} - Duration \times BattLoad_{j,t}$$

where *Duration* is the length of each epoch t. As discussed in Section 3.2, leveraging battery as a power source should not harm their lifetime. The depth of discharge (DoD) is an important measurement for battery's lifetime,

$$\forall t \in T: \quad (1 - DoD) \times BattMaxCap_j \le BattCap_{j,t} \le BattMaxCap_j$$

where $BattMaxCap$ denotes the battery maximum capacity.

Obviously, one cannot use more energy than what is actually stored in the battery, thus,

$$\forall t \in T : Duration \times BattLoad_{j,t} \le BattCap_{j,t}$$

Since battery charging itself with grid power is considered for adhering to P_{budget}, we restrict $Batt\_Grid_{c,t}$ to non-emergency periods,

$$\forall t \in \{T | RELoad_{c,t} > 0 \text{ or } BattLoad_{j,t} > 0\} : Batt\_Grid_{j,t} = 0$$

4.2 Power Management Knob

Goal: Upon any power emergency, the power management knobs (PMK) allow us to continue executing the applications when the combined power and energy of the renewable energy and battery sources are insufficient to meet the grid power budget, with a lower power demand to sustain within the underprovisioned grid power infrastructure. A PMK consists of the *Throttling* (local knob) and *Migration* (remote knob). We summarize the demand required by these techniques in Table 2.

Throttling: Throttling techniques, like dynamic voltage and frequency scaling (DVFS) and Clock throttling, are used to enforce an application to operate in a low-performance mode using active power states. Throttling is very effective to reduce the power demand during a power emergency because it is almost instantaneous (within tens of μsecs) to transition to throttling states. However, throttling techniques can significantly degrade the application performance, especially when the application does not provide sufficient slack in their offered load.

Each server $j = 1, ..., n$ in cluster c can operate during an interval within a power emergency in a particular power state S_j (e.g., save-state, DVFS and Clock throttling states), ordered as S_0 where the server enters in the *save-state* (Sleep/Hibernation), to S_r which is the highest power consuming state and highest in performance. We also denote the intensity of a workload during this emergency as L_j, which can be of any of the w levels between the minimum and maximum intensity levels for a given application, such as $L_1, ..., L_w$. The power reduction $P_{TH,t} = \sum_{j=1}^{n}(P_j(L_{j,t}, S_{r,t}) - P_j(L_{j,t}, S_{j,t}))$ offered by *Throttling* for each power emergency epoch t not only depends on the workload intensity level $L_{j,t}$ being served on server j during epoch t but also on the power state S_j. We calculate power demand for each server $P_j(L_{j,t}, S_{j,t})$ *a priori* and obtain $S_j \in (S_0, ..., S_r)$ for different workload intensity levels $L_j \in (L_1, ..., L_w)$. Hence, an optimal *Throttling* is achievable that corresponds to the power supply of the power sources with the least performance slowdown $Perf^A_{TH}$ due to power throttling (where $Perf^A_{TH} \in [0, 1]$).

Many workloads, such as MapReduce and scientific computation, can tolerate some delays due to natural deadlines as stated in [23, 22]. Such loose deadlines provide the flexibility that allows compute workloads to be close to the *availability* of renewable energy and thus completely avoid any power emergency. In this scenario, the jobs can be suspended instead of being cancelled, where the *Sleep* state is ideally the power state for the delayed operation and $Perf^A_{TH}$ will be set to zero.

PMK techniques	Time overhead	Power state
Throttling	Tens of μ seconds	Throttled state
Migration	A few minutes	Consolidated state
Sleep	A few seconds	2-4W per DIMM
Hibernation	A few minutes	0W

Table 2: Impact of PMK techniques on the handling of power emergency.

Migration: As an alternative, applications can be migrated to the designated regions of the datacenter with sufficient headroom immediately upon a power emergency. By consolidating applications on fewer servers, it enables the sharing of CPU cycles, memory, and other resources among applications for the purposes of better energy efficiency, as a result of the lack of energy proportionality in today's datacenter.

In this paper, we consider two migration strategies, *intra-datacenter* migration and *intra-cluster* migration. intra-datacenter migration allows an application to be migrated from one cluster to another with sufficient headroom within the datacenter. For simplicity, we assume that a cluster with sufficient headroom elsewhere in the datacenter can be found, and thus do not consider possible performance interference on the clusters without sufficient headroom as discussed in [4]. Intra-cluster migration is used when intra-datacenter migration is not possible, e.g., unavailability of outside-cluster servers with sufficient headroom, or administrative reasons to force the application to be tied within the current cluster, or the application needs (communication requirement, locality-favoriable workload, etc.).

Each server j in cluster c maintains the application state D_j during epoch t within the power emergency, and the application state D_j of server j in cluster c can also be ordered from low to high. If migration is necessary, since the application's state determines the migration execution time, we start from the server with the lowest D_j and migrate applications one by one. The peak power demand can be decreased by $P_{Mig,t} = \sum_{j=1}^{m}(P_j(L_{j,t}, S_{j,t}, D_{j,t}) - P_j(S_{0,t}, D_{j,t}))$ for epoch t, where m ($m \le n$) denotes the number of servers involved in migration. After migration, the servers can be put in the *save-state*. We still require *a priori* knowledge for application's memory state of each server to achieve an optimal migration operation given the power supply of power sources and the throttling techniques. After being migrated to another server or cluster, the application can also suffer from a performance slowdown $Perf^A_{Mig}$ (where $Perf^A_{Mig} \in [0, 1]$) if the destination server is highly utilized or there is severe interference from native applications.

4.3 Integration of PSS and PMK

Table 3 summarizes how the individual PSS and PMK techniques behave in handling power emergency in each of the three phases. Obviously, it is preferred if PSS alone could handle the emergency since the techniques provided in PMK may result in performance degradation. A power supply-demand equation can be derived as follows,

$$\forall t \in T : RELoad_{c,t} + BattLoad_{c,t} + P_{TH,t} + P_{Mig,t} + GridLoad_{c,t} = PowerD_{c,t}$$

Recall that we use the term *availability* to loosely indicate the available renewable energy. PMK helps control the power demand on the server with the renewable energy provision to effectively utilize the varying renewable power. Note that, even after migration, the power demand is still determined by *Throttling* technique. Hence, the available renewable power ($REPower_c$) has a lower bound determined by the power demand of a green provisioned server gj:

$$\forall t \in T : P_{gj}(L_{gj,t}, S_{gj,t}) \le REPower_{c,t}$$

Technique	Start of power emergency	During power emergency	After power emergency
Renewable energy	Direct supply or Charge Battery	Direct supply or Charge Battery	Charge Battery
Battery with renewable energy	Discharge/Charge by renewable energy	Discharge/Charge by renewable energy	Charge by renewable energy
Battery with Grid	Discharge	Discharge	Charge by Grid power
Throttling	Throttled Perf.	Throttled Perf.	Restore full service
Migration	Migrate to remote memory	Consolidated service	Migrate back
Sleep	Suspend to local memory	No service	Resume from memory
Hibernation	Suspend to local disk	No service	Resume from disk

Table 3: Operational and Performance implications of the individual PSS and PMK mechanisms

$P_{gj}(L_{gj}, S_{gj})$ is a *start point* for a single server gj to switch to the green power bus while determining the renewable energy *availability*. $P_{gj}(L_{gj}, S_{gj})$ is determined by the server type, application characteristics and the workload's intensity level. The renewable energy *availability* reaches its minimum when $REPower_{c,t} < P_{gj}(L_{gj,t}, S_{gj,t})$. In this scenario, the renewable energy can not sustain the power emergencies independently and the power of batteries ($BattLoad_{c,t}$) is required to help utilize the small amount of green power ($RELoad_{c,t}$) to improve the renewable energy utilization. Similarly, renewable energy *availability* achieves its maximum when $REPower_{c,t} \geq \sum_{gj=1}^{k} P_{gj}(L_{gj,t}, S_{r,t})$, where k is the maximum number of green-powered servers in cluster c and S_r is the highest power state for server gj. In the latter case the renewable energy can self-sufficiently handle the power emergency without the support of PMK and batteries associated with green provision, and also possibly charge the batteries ($Batt\_RE_{c,t} \geq 0$).

Moreover, the migration operation for green powered server gj will not start without first fully utilizing the available renewable source:

$$\forall t \in \{T | REPower_{c,t} \geq P_{gj}(L_{gj,t}, S_{gj,t})\} : Perf_{gj,Mig} = 1$$

To reduce the migration overhead, workload migration for each server j in cluster c will not be triggered if renewable energy, batteries or their combination can offer a better performance than migration:

$$\forall t \in \{T | Perf_{j,TH} \geq Perf_{j,Mig}\} : Perf_{j,Mig} = 1$$

Performance Impact: We denote the original execution time ($MaxPerf$) for traditional grid overprovision as T_E. To continue processing during power emergencies in the underprovision design, the execution time, denoted by T_S, may be extended. Let T_{Local} denote the application's execution time with the power supply of renewable energy (*RELoad*) and battery (*BattLoad*) in conjunction with *Throttling*. When the remote knob (*Migration*) is triggered, the execution time will be adjusted to T_{Remote}. T_{Save} is the resumed time from the *save-state*. Hence, we can obtain the relationship between the execution time after employing the techniques of handling power emergency and the original execution time, as follows.

$$T_S - (T_{Local} + T_{Remote} * (1 - Perf_{Mig})) * (1 - Perf_{TH}) - T_{Save} = T_E$$

Note that the execution time is increased due to $Perf_{TH}$ and $Perf_{Mig}$ and the save states. The value of $Perf_{TH}$ is mainly governed by the renewable energy *availability* and available battery capacity. As in our assumption the destination always has sufficient headroom (intra-datacenter migration) and the migrated workload will not be impacted by any power-related constraints, $Perf_{Mig}$ is determined by its locality properties.

From Figure 3, it is clear that when power sources are not able to meet the power demand while throttling severely degrades the performance, we must migrate the application to meet the stringent SLA requirement. On the other hand, the application would be migrated back when the power sources become sufficient. In addition, the save-state of server also impacts the performance due to the resume overhead (Table 2). We refer to the solution offered by our framework as *Green*, the solution allowing intra-cluster migration as *Green-C*, and intra-datacenter migration as *Green-D* and the solution it offers solely using local PMK knobs without migration as *Green-L*.

Scheduling Goal: We use $R^{sus}_{j,t}(L_{j,t}, S_{j,t})$, whose minimization corresponds to meeting the application's SLA, to denote a generic performance metric based on the sustain mechanisms used for handling epoch t of a power emergency. We can use this metric to express the problem of minimizing performance impact for a given power budget ($\forall t \in T : GridLoad_{c,t} \leq P_{budget}$):

$$min \sum^{t \in T} \sum^{j \in c} R^{sus}_{j,t}(L_{j,t}, S_{j,t})$$

5. PROTOTYPE EVALUATION

Experimental Setup and Methodology: Our scale-down experimental prototype uses a cluster c of $N = 10$ homogeneous servers with 6-core 2.0GHz Intel Xeon E5-2620 processors (i.e., 12 cores per server), 48GB RAM and 1Gbps Ethernet interface and run our applications hosted on the Unbantu Linux OS. Our cluster has a shared NAS box that is mounted as a NFS storage volume by all the servers. We use another cluster of 10 servers that is not underprovisioned as the destination for migrating workloads to simulate an intra-datacenter migration *Green-D*. The power consumption of each server is monitored using an external power meter [31]. The face-plate rating of these servers is 450W. Their idle power is around 90W. The dynamic power consumption can be modulated with 9 DVFS states and 15 clock throttling states.

To simulate a datacenter with renewable energy provision, we randomly choose one of the renewable power production traces with one-week duration from NREL [32], including irradiation every minute, and replay the chosen trace on our prototype. We scale the solar power production to correspond the power source configuration (Table 1) to simulate the available renewable power output ($REPower_c$ in Section 4.1). In our setup, we consider a solar panel provisioned for a server i with 275W DC output (theoretical peak power), which is in line with the existing capacities in Grapesolar [33]. Hence, we can obtain the peak renewable power AC supply for a single solar panel that corresponds $Peak_{RE} * \alpha = 275 * 0.77 = 211.75W$. Hence, as shown in Figure 4, for the *RE-Batt* configuration, we assume that 3 servers in our prototype are provided with renewable energy system that is capable of supplying the maximum green power of 635.25 W ($REPower_c$). For the configuration with *SRE* that provides 2 servers with renewable energy, the maximum green power obtainable is 423.5W ($REPower_c$). We assume that each server in the cluster is equipped with a battery unit and the battery energy capacity is shown in Table 1.

We use *cpufreq* to implement *Throttling* technique, and leverage the Xen live migration [34] implementation for *Migration*. Moreover, we use the standard OS commands in Linux to implement the *save-state*.

In order to evaluate the efficacy of our design and theoretical framework, we experiment and evaluate four relevant

Workloads	Memory Usage	Performance Metric
Specjbb	10GB	Latency-constrained, ops/sec
Web-search	20GB	Latency-constrained, queries/sec
Memcached	20GB	Queries/sec
SpecCPU2006	24GB	Completion time

Table 4: Workload Description

scenarios, including (i) power emergencies lasting a duration ranging from 15 minutes to 120 minutes, (ii) underprovision degree set to 30% of the potential peak power demand. (we measure the peak power demand for each application based on our setup), (iii) power emergencies occurring given different levels of renewable energy *availability* (i.e., minimum, medium, maximum) for a duration ranging from15 minutes to 120 minutes, and (iv) a comparison among techniques, *Throttling*(*Green-L*) and *Migration* (*Green-C*,*Green-D*), of their effectiveness in handling the power emergency. Moreover, the *start point* is determined as the 70% of the potential peak power corresponding to the underprovision degree for a single green powered server and the *start point* is also determined as the power demand for *Sleep* state when *Migration* is triggered. Note that we are only concerned with performance of cluster c during a power emergency in this work. For each application, we normalize its performance to that in *MaxPerf* (recall Section 3.2). We consider *MaxPerf* in our setup as the maximum performance without allowing any power cap techniques.

We believe that the smaller setup can be used to harvest as many insights (performance impact and power sources scheduling) as that from a larger-scale platform using the following methodology. We subject each application or server to the power sources and workload schedulers described earlier and profile the power consumption per server of each workload *a priori*. We also profile the application performance per server *a priori*, when the application is subject to the mechanisms for handling power emergency provided in our theoretical framework. Power and performance data are collected at fine temporal resolutions and allow us to scale the scheduling mechanism to different system sizes and renewable power and battery energy capacities.

Workloads: We consider the following representative datacenter workloads (Table 4) that have different performance characteristics that result in different peak power demands on renewable energy and batteries. Interactive applications include the Specjbb [35], an in-memory key-value store Memcached benchmark [36], Web-search from Cloudsuite [37]. SpecCPU benchmarks [38] represent High Performance Computing (HPC) applications. We use *mcf* from SpecCPU2006 in our experiments as a representative for memory intensive scientific computation workloads. Since each mcf instance only consumes 2 GB memory, we instantiate 12 mcf instances to increase its memory usage to emulate power emergencies of HPC applications.

5.1 Effectiveness of Power Emergency Handling Mechanisms

We now compare the performance impact offered by different emergency-handling techniques provided in PMK (Section 4.2). We first show representative results for Specjbb, and then show the specific results for the other applications. Applications like Specjbb are known to experience significant load variations. We add replicas of the workload in the cluster until all 10 servers are fully utilized to cause a power emergency since the aggregate draw of these servers can exceed the underprovisioned power budget. For instance, when the workload saturates all 10 servers, the aggregate power consumption hits 1590W. If the grid power infrastructure is underprovisioned by 30%, then the corresponding P_{budget} limit is 1113W. From the renewable energy side, if renewable energy can supply 3 servers in the cluster (i.e., the RE-Batt configuration), the 30% underprovision will be satisfied for the green provision. As specified above, we inject the workloads to deliberately induce power emergency durations of 15, 30, 60 and 120 minutes. We use the average operations/second (ops) of cluster c as our performance metric for Specjbb.

Figure 4 shows the evolution of the aggregated peak power of the 3 green-provisioned servers running Specjbb given different levels of renewable energy *availability*. We see high variation of the renewable power production over time. We have evaluated such performance consequences for all the cases of medium availability over different power emergency durations. In this paper, we only present the results of the $B1$ scenario for the applications we specified above for medium renewable energy *availability* to show the performance impact of the renewable energy variation. Moreover, we consider the minimal availability as $RELoad_c = 0$ (recall Section 4.1) for comparison where the underprovisioned goal can only be achieved by the batteries. We determine the *start point* for Specjbb to be the time point when a server with green provision starts to use renewable power as it obtains 111.3W from the green power bus.

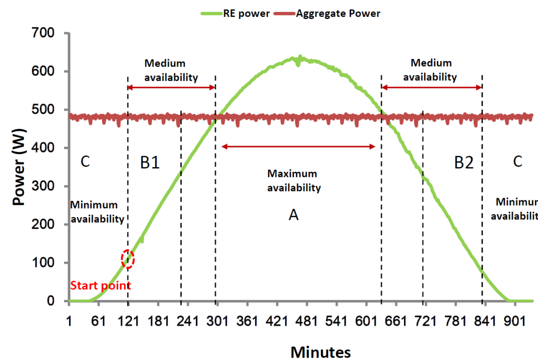

Figure 4: The Specjbb power profile as a function of the renewable energy availability over time.

Impact of renewable energy availability: Figure 5 presents the performance degradation in ops of Specjbb under the different power-emergency handling techniques given the minimum, medium, and maximum renewable energy *availability* for the emergency durations we specify above.

Green-L: For short and moderate power emergencies (15-minute and 30-minute durations), even when the renewable energy is unavailable, battery alone is able to completely handle the power emergency without degrading the performance. In fact, in such power emergencies *Green-C* and *Green-D* are never activated by our theoretical framework. However, battery incurs high performance overhead on long power emergencies since *Green-L* uses a lower power state on servers, stretching the battery's runtime for the minimum renewable energy *availability*. The performance degrades to 88% of *MaxPerf* for the 60-minute duration, 73% of *MaxPerf* for the 120-minute duration with *Green-L*.

For the medium *availability* of renewable energy, 96% and 90% of *MaxPerf* can be achieved for the 60-minute and 120-minute power emergencies respectively. It is interesting to observe that the *start point* of when to utilize renewable energy determines the length of the medium *availability* of renewable energy and the battery's behavior. This shows the importance of dynamically adjusting the *start point* of switching to the green power bus to match the power emergency durations. This, however, is beyond the scope of this paper, which we plan to investigate in our future work.

As expected, in the case of the maximum *availability* of renewable energy (i.e., case A shown in Figure 4), three servers in the cluster can be directly powered by renewable power without any performance degradation. Further, the excess green power can be used to charge the battery for later use.

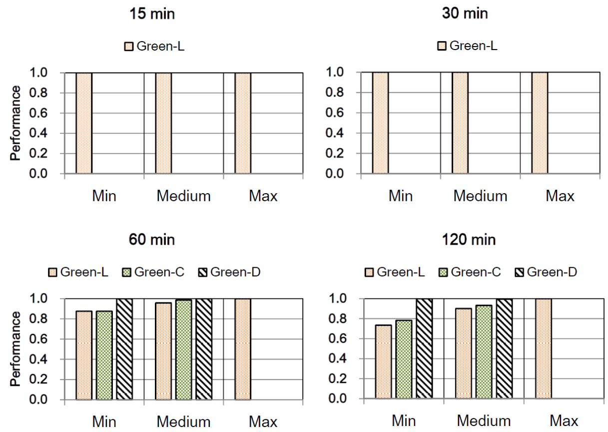

Figure 5: Performance with underprovision, normalized to *MaxPerf* (with overprovision baseline), as a function of the duration length of power emergency and renewable energy availability, for Specjbb with the *RE-Batt* configuration.

Green-D: Specifically, workload migration is only allowed when *Green-L* alone is not sufficient. Upon the minimum *availability* of renewable energy, batteries are activated first to postpone the migration action and then we migrate 3 replicas (3 servers) of Specjbb to another cluster with sufficient headroom to achieve 30% underprovisioning. As analyzed in [4, 16], *Migration* in itself does introduce a spike in power consumption (of 10%) for this workload, and batteries are required to keep the grid power demand under the tighter grid power budget during the process of migration. Live migration can be carried out in about 5 minutes for Specjbb with 10GB memory size. The loss of data locality suffered by *Green-D* is negligible (i.e., small impact of $Perf_{Mig}$), implying very little performance impact after migrating to another cluster. The only situations where *Migration* is an unwise choice is when the power emergency lasts for 5 minutes or less, in which case the migration overhead outweighs the gain.

In the case of the medium *availability* of renewable energy, the *Green-D* mechanism only allows the migration of the servers that are not currently powered by the green source and migrates such servers back from the remote nodes when renewable power production reaches the *start point.* Moreover, unlike the case of minimum *availability*, the power spikes required of carrying out the reverse migration operation can be accommodated by the green power and the energy of batteries will be reserved for boosting performance after the servers migrate back.

The results show that *Green-D* can achieve 99% of *MaxPerf* for cluster *c* upon the 120-minute power emergency. The performance impact (compared to *Green-D* of minimum *availability*) is due to *Throttling* in the green powered servers to control the power demand to match renewable power production while the associated batteries are insufficient to cover such a long duration. *Green-D* achieves *MaxPerf* for the 60-minute power emergency. Consequently, intra-datacenter migration combined with *Throttling* turns out to be the most effective mechanism and is able to handle the entire range of power emergencies with high performance regardless of the renewable energy *availability.*

Green-C: Finally, we assess the intra-cluster migration, *Green-C*. Since the 10 servers running Specjbb are operating at their peak computation-resource requirement, *Green-C* co-locates the workload in a same cluster that is experiencing significant resource shortage, causing performance degradation. In fact, *Green-C* fares worse than *Green-D* in that the former cannot be triggered where *Green-D* is allowed. For the minimum renewable energy *availability*, performance impact is determined by the battery energy capacity. *Green-C* chooses to drain the battery fully to postpone the migration and results in undesirable performance for longer power emergencies since the battery is not able to provide power beyond a certain period of time, thus forcing it to switch to in-cluster migration and degrade the Specjbb performance.

For the medium renewable energy *availability*, renewable energy can trigger the reverse migration when the servers can be directly powered by the green source while alleviating the resource contention in the cluster to reduce the performance impact. We find that even for long power emergencies (120 minutes) with the medium renewable energy *availability*, *Green-C* still achieves 93% of *MaxPerf*, in contrast to the 78% of *MaxPerf* achieved with the minimum renewable energy *availability.*

Save-state: After migration, we put the server into *Sleep* state where preserving application's state in memory reduces the additional performance impact after the emergency. The *Sleep* state dissipates only around 6W per server and resumes back to the normal operation for 8 seconds. The battery can satisfy the power draw of the *Sleep* state for short and moderate power emergencies (15 min - 30 min) in the case of the minimum renewable energy *availability.* However for longer power emergencies (120 min and beyond), suspending in memory may not buy much since the emergency duration far outweighs the overheads of losing state, sleep state could draw relative large energy from battery (DoD consideration, Section 4.1). *Hibernation* suffers 255 seconds for resuming service, which is negligible compared to the 120 minutes of a long power emergency while saving energy. However, for the medium *availability*, despite of the large variance of the renewable power, it still can make room for the power demand of the *Sleep* state.

We find that the most effective techniques for handling power emergencies are different for different levels of renewable energy availability. For the minimum *availability*, the battery is the only power source for handling the power emergencies. This means that the most effective techniques are determined by the energy capacity of battery. For instance, for short power emergencies of 15 or 30 minutes, *Throttling* is preferred to achieve a good performance since the battery can typically supply sufficient power for such durations. Though *Migration* (Green-D) is also able to achieve *MaxPerf* (not shown in the figure), it requires a relatively long migration time of up to 5 minutes. For longer power emergencies lasting 1 hour or more, given the minimum *availability*, *Throttling* becomes infeasible while *Migration* shows the performance advantage, since the limited battery energy capacity can hardly sustain such a long time to provide an ideal performance. For the medium *availability*, *Throttling* is able to handle long power emergencies (60 minutes - 120 minutes) with only a slight performance impact (performance degradation $\leq$ 10%). This is primarily due to renewable energy that directly powers the servers to alleviate the pressure of underprovision on the battery. For power emergencies longer than 2 hours, *Migration* (both intra-cluster and intra-datacenter migration) is able to achieve a better performance with a relatively short migration overhead (5 minutes). On the other hand, *Throttling* and *Migration* are not required irrespective of the length of a power emergency if given a larger battery energy capacity configuration with the medium *availability.*

5.2 Performance impact of different green configurations

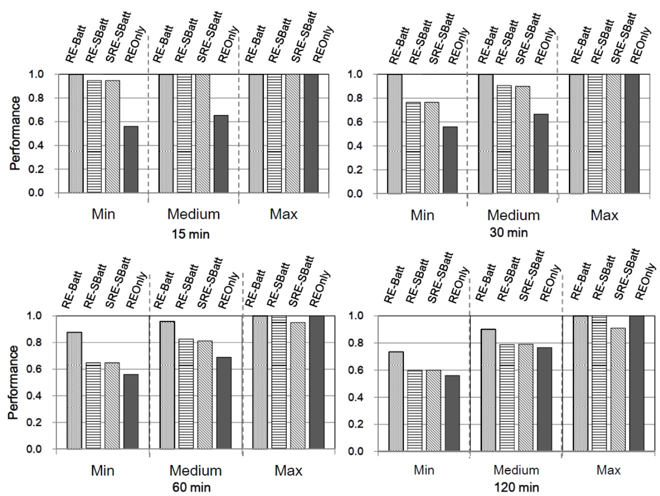

Figure 6: Performance tradeoffs among the 4 green configurations - *RE-Batt*, *RE-SBatt*, *SRE-SBatt* and *RE-Only* - under the Specjbb workload.

In this subsection, we evaluate the performance tradeoffs among different green configurations (i.e., different combinations of renewable power capacities and battery energy capacities) from Table 1. For each green configuration, we use the *Green-L* technique to show the performance impact, since *Green-C* and *Green-D* show similarity performance impacts. Figure 6 presents the performance of Specjbb with various green configurations.

Impact of renewable energy: As renewable energy could power the servers, long emergencies can be transformed into short emergencies from the perspective of performance.

In the REOnly configuration (in Table 1), the performance with the minimum renewable energy *availability* is same as that with only using *Throttling* for power capping. Renewable energy significantly improves performance, from 56% of *MaxPerf* without it (minimum *availability*) to 77% of *MaxPerf* with it for 120-minute long power emergencies with the medium renewable energy *availability*. Configurations with a small battery capacity, RE-SBatt and SRE-SBatt, can ride-out the minimum *availability* but with a significant performance penalty for 30 minutes duration (77% of *MaxPerf*). However, even with small renewable energy provision, *SRE-SBatt* achieves *MaxPerf* for up to 30 minutes with the maximum renewable energy *availability*. With the medium renewable energy *availability*, SRE-SBatt is still able to achieve 81% of *MaxPerf* for a 60-minute power emergency and 79% of *MaxPerf* for a 120-minute power emergency. RE-SBatt shows a similar performance to the SRE-SBatt configuration but achieves *MaxPerf* upon the maximum renewable energy *availability* regardless of the emergency duration.

Impact of battery: Given the minimum renewable energy *availability*, in configurations with the battery (RE-SBatt and RE-Batt) and with the minimum renewable energy *availability*, performance impact for REOnly configuration (56% of *MaxPerf*) can be reduced since the battery can supply power. The RE-SBatt configuration with limited battery energy (3.2Ah) capacity suffers a performance degradation (95% of *MaxPerf*) even during a 15-minute power emergency with the minimum renewable energy *availability* while RE-Batt can achieve *MaxPerf* up to 30 minutes.

With the medium renewable energy *availability*, battery plays an important role in boosting the performance. Even with a small battery energy capacity (3.2Ah), RE-SBatt is able to achieve *MaxPerf* for a 15-minute power emergency and sustains 83% of the (degraded) performance for a 60-minute emergency, in contrast to the REOnly configuration that achieves only 69% of *MaxPerf*. However, for power emergencies lasting 120 minutes or beyond, configurations with a small battery capacity (RE-SBatt and SRE-SBatt) show less benefit than the REOnly configuration, since the battery is not able to sustain such long-time operation for the entire emergency duration. And finally, performance can benefit more from larger battery energy capacity. For instance, RE-Batt (10Ah) achieves 96% of *MaxPerf* for a power emergency of 1 hour and maintains a 90% of (degraded) performance for a power emergency of 120 minutes. This implies that one can significantly improve the performance irrespective of renewable energy *availability* by purchasing more battery energy capacity.

5.3 Impact of application characteristics

For the rest of the evaluation, we only consider *SRE-SBatt* configuration (Table 1) and compare the effect of under-provisioning for applications with diverse performance and power peaks characteristics for the variations in the small renewable power and battery energy capacity. We mainly focus on the differences.

Memcached and Web-search (Figure 7 and Figure 8): Because Memcached has high memory-related CPU stalls, we find that the performance offered by *Throttling* and intra-cluster migration *Green-C* to be much better than that for Specjbb with a 20GB memory size (not shown in the figure for clarity). The performance with the medium and maximum *availability* shows a significant improvement over that with the minimum *availability* as the green power source participates in the handling of the power emergency. Intra-cluster migration (*Green-C*), combined with *Throttling*, achieves better performance than the *Throttling* mechanism (*Green-L*) alone. This implies that Memcached may benefit more from the migration mechanism as renewable energy does not suffice.

Interestingly, *Throttling* for Web-search shows a better performance than that with Memcached and higher power state can outperform the *Green-C* technique for maximum *availability*. This implies the fine-grained power states of *Throttling* for Web-search may cover a larger spectrum of renewable energy availability and offered performance than the other workloads, which we plan to investigate in the future work. Moreover, we find that the *Hibernation* state for Web-search requires 4.8 minutes to resume from disk, resulting in degraded query performance even the workload is available - 47% reduction in throughput - during the 3-4 minutes warm-up duration. Hence, renewable power can help achieve a good performance result and avoid prolonging the emergency duration for Web-search since it can power *Sleep* state.

SpecCPU (Figure 9): We observe that the performance of *SpecCPU* offered by *Throttling* becomes unacceptable because of the high memory contention from the 12 mcf instances. Furthermore, *Throttling* heightens the contention impact while causing a huge slowdown when all cores execute such memory-bound tasks. We choose not to migrate the HPC workload, since it can be delayed as discussed in [23] (recall Section 4.2). From the figure, we can see that the performance is improved as long as the workload can be powered by the green power. For instance, the performance reaches 94% of *MaxPerf* with the maximum *availability*, in contrast to 49% of *MaxPerf* with the minimum *availability* upon a 120-minute power emergency. Even with the medium *availability* for 2 hours duration, renewable energy

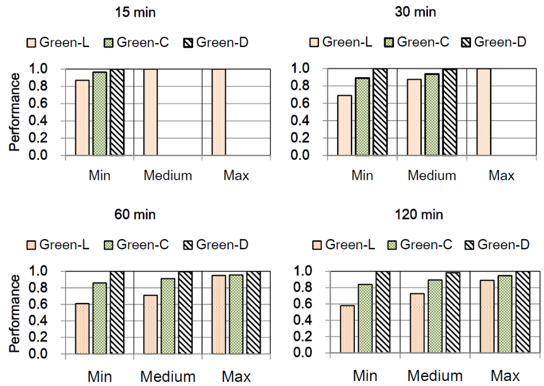

Figure 7: Performance of Memcached with the SRE-SBatt configuration.

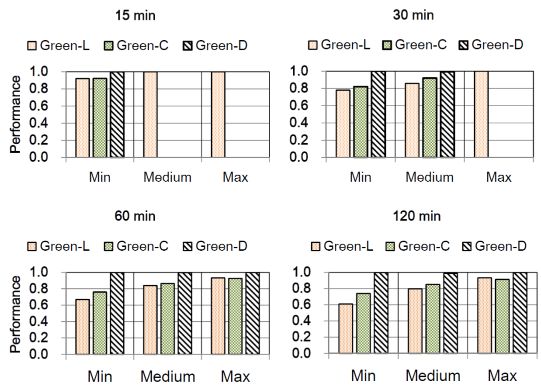

Figure 8: Performance of Web-search with the SRE-SBatt configuration.

helps achieve 30% performance improvement over that with the minimum *availability*.

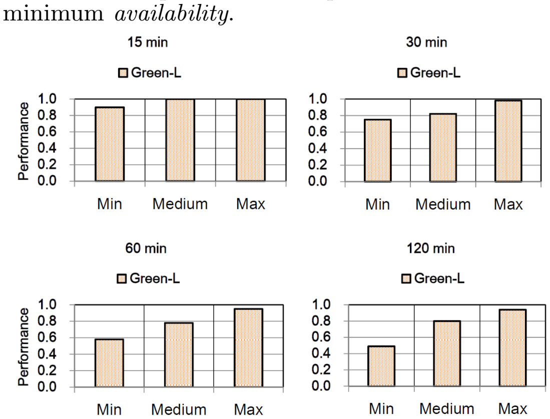

Figure 9: Performance of SpecCPU with the SRE-SBatt configuration.

5.4 Summary of Observations

In summary, we find that: (i) Though renewable energy shows large variance for power production, it plays as an important power source for emergency handling from the perspective of offered long time power supply. (ii) Renewable energy can be self sufficient regardless of the durations of emergencies for datacenters willing to delay computer loads (e.g., scientific computations) during emergencies. (iii) Renewable energy, conjunction with batteries can achieve the

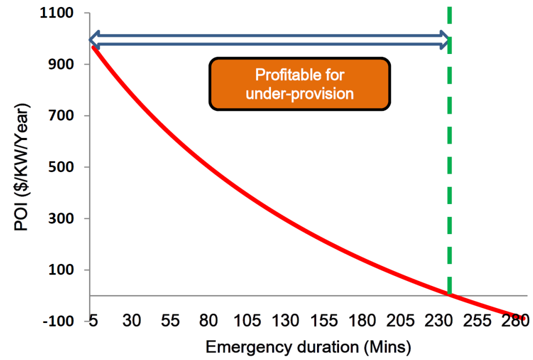

Figure 10: POI with additional renewable energy and battery investment.

same performance as with today's overprovision approach for short and moderate emergencies. (iv) Renewable energy can supplement battery to reduce performance impact. (v) Throttling is infeasible at lower power supply of renewable energy and battery, though it can cover a large spectrum of the renewable energy availability for short to long emergencies for keeping within the underprovison grid power budget. (vi) Migration is preferred for longer emergency to offer a better performance compared to throttling for the power shortage of renewable energy and battery.

5.5 TCO Consideration

Although we have considered a conservative provision of renewable energy and battery that already exist in green datacenters in our experiments, we will illustrate whether the cost of additional green provision can be justified by the potential reduction in the cost of datacenter's underprovisioned grid power infrastructure that we can obtain. We first denote the cost of procuring the additional PV module capacity as $REPowerCost$ \$/watt, the additional battery capacity to sustain T hours of power emergency as $T * BattCost$ \$/kWh and the cap-ex cost reduction of the grid power infrastructure due to underprovision as $GridCost$. We denote $REInCost$ \$/watt as the inverter cost.

We estimate our green-power capacity cost to be \$4.74/W, based on a recent report for moderate-scale PV panel (several hundred KW) [39]. In addition, we ensure that PV panels are amortized over 25 years (lifetime). The cost of central inverters at the multi-megawatt (MWs) level is around \$0.18/W and the inverter lifetime is around 10 years using the estimates of recent study [40]. The cost of the lead-acid battery is \$0.21/Wh as used in [21] while the battery's lifetime is assumed to be 4 years. We assume a grid utility cap-ex of \$5/W based on the estimate of the capital cost of the grid infrastructure for a typical 12-year datacenter from [4].

We use the measure of Profit-On-Investment (POI) to estimate the cost effectiveness of additional green infrastructure. POI is plotted in Figure 10 as a function of the length of a power emergency ($POI = \frac{GridCost}{T*BattCost+REPowerCost+REInCost} - 1$). In this calculation, we ensure that all components are amortized over their lifetime. Any length of a power emergency to the left of the crossover point (which turns out to be around 4 hours in this case) indicates a profitable investment. This suggests that investing in additional green provision may be worthwhile for most power emergencies. In fact, the result is conservative in that we do not consider the operation expense reduction (e.g., electricity cost reduction) through the green provision, which is a significant cost in datacenter.

6. CONCLUSION

We propose a novel renewable power distribution architecture that enables the underprovisioning of the grid power infrastructure in green datacenters. We conduct a framework for dealing with power emergencies arising from underprovisioning of the grid power infrastructure in green datacenters. To avoid power overdraw, power management knobs, such as power state modulation and workload migration, can be supplemented with our new proposal of leveraging the already existing renewable energy and battery in green datacenters to gracefully deal with power emergencies. We demonstrate that, using an experimental prototype and assuming a 30% underprovision of the grid infrastructure, sufficient renewable energy by itself can sustain entire power emergencies (around 15-120 minutes on our prototype) even without the use of battery. Medium renewable energy availability, combined with battery, sustain short and moderate (15-30 minutes) power emergencies without any performance impact. Minimum renewable energy availability requires workload migration since sustained local throttling hurts performance if the limited battery energy capacity is not able to sustain the entire power emergency. Our renewable-energy solution provides a graceful solution to handling power emergencies of all lengths arising from aggressive underprovisionsing of the grid power infrastructure with minimum performance impact.

7. ACKNOWLEDGMENTS

We would like to thank anonymous reviewers for their insightful comments. This research is sponsored by the National Basic Research 973 Program of China under Grant No. 2011CB302303, the National Natural Science Foundation of China under Grant No. 61300046, and the National High Technology Research and Development Program of China (863 Program) under Grant No.2013AA013203, and the Fundamental Research Funds for Central Universities, HUST, (Grant No. 2013KXYQ003).

8. REFERENCES

[1] `http://www.aiso.net/technology-network-sun.html`.
[2] `http://www.greenhousedata.com/`.
[3] X. Fan et al. Power provisioning for a warehouse-sized computer. In *ISCA*, 2007.
[4] S. Govindan et al. Leveraging Stored Energy for Handling Power Emergencies in Aggressively Provisioned Datacenters. In *ASPLOS*, 2012.
[5] D. Wang et al. Virtual Power Distribution in Datacenters. In *ISCA*, 2013.
[6] D. Wang et al. Energy Storage in Datacenters: What, Where, and How much? In *SIGMETRICS*, 2012.
[7] V. Kontorinis et al. Managing Distributed UPS Energy for Effective Power Capping in Data Centers. In *ISCA*, 2012.
[8] C. Li et al. Enabling datacenter servers to scale out economically and sustainably. In *MICRO*, 2013.
[9] C. Lefurgy et al. Server-level power control. In *ICAC*, 2007.
[10] X. Fu et al. How Much Power Oversubscription Is Safe and Allowed in Data Centers. In *ICAC*, 2011.
[11] C. Li et al. iSwitch: Coordinating and Optimizing Renewable Energy Powered Server Clusters. In *ISCA*, 2012.
[12] C. Li et al. Solarcore: Solar energy driven multi-core architecture power management. In *HPCA*, 2011.
[13] Luiz André Barroso and Urs Hölzle. The case of energy-proportional computing. *Computer*, 40(12):22–40, 2007.
[14] S. Govindan et al. Benefits and Limitations of Tapping into Stored Energy for Datacenters. In *ISCA*, 2011.
[15] P. Bohrer et al. Energy Conservation for Servers. In *Workshop on Power Management for Real-Time and Embedded Systems*, 2001.
[16] D. Wang et al. Underprovisioning Backup Power Infrastructure for Datacenters. In *ASPLOS*, 2014.
[17] S. Govindan et al. Statistical profiling-based techniques for effective power provisioning in data centers. In *EuroSys*, 2009.
[18] W. Zheng et al. Exploiting Thermal Energy Storage to Reduce Data Center Capital and Operating Expenses. In *HPCA*, 2014.
[19] N. Sharma et al. Blink: Managing server clusters on intermittent power. In *ASPLOS*, 2011.
[20] Z. Liu et al. Renewable and Cooling Aware Workload Management for Sustainable Data Centers. In *SIGMETRICS*, 2012.
[21] I. Goiri et al. Parasol and Greenswitch: Managing Datacenters Powered by Renewable Energy. In *ASPLOS*, 2013.
[22] I. Goiri et al. GreenHadoop:Leveraging Green Energy in Data Processing Frameworks. In *EuroSys*, 2012.
[23] I. Goiri et al. GreenSlot: Scheduling Energy Consumption in Green Datacenters. In *SC*, 2011.
[24] B. Aksanli et al. Utilizing Green Energy Prediction to Schedule Mixed Batch and Service Jobs in Data Centers. In *HotPower*, 2011.
[25] C. Li et al. Enabling distributed generation powered sustainable high performance data center. In *HPCA*, 2013.
[26] Google Summit. `http://www.google.com/corporate/datacenter/events/dcsummit2009.html`.
[27] Facebook. Hacking conventional computing infrastructure. `http://opencompute.org/,2011`.
[28] `http://rredc.nrel.gov/solar/calculators/pvwatts/`.
[29] D. C. Jordan et al. Photovoltaic Degradation Rates - An Analytical Review. *Progress in Photovoltaics: Research and Applications*, 21:12–29, 2013.
[30] D.Manz et al. Enhanced reliability of photovoltaic systems with energy storage and controls. In *National Renewable Energy Laboratory, Report*, 2008.
[31] Zh-101 portable electric power fault recorder and analyzer, 2009.
[32] Measurement and instrumentation data center. `http://www.nrel.gov/midc/`.
[33] Grapesolar. `www.grapesolar.com/`.
[34] C. Clark et al. Live Migration of Virtual Machines. In *NSDI*, 2005.
[35] SPECJBB 2013:Java Business Benchmark. `http://www.spec.org/jbb2013/`.
[36] MEMCACHED:Memory-object Caching System. `http://www.memcached.org`.
[37] M. Ferdman et al. Clearing the clouds: A study of emerging scale-out workloads on modern hardware. In *ASPLOS*, 2012.
[38] SPECCPU. `http://www.spec.org/cpu2006/`.
[39] D. Feldman et al. Photovoltaic (PV) Pricing Trends: Historical, Recent, and Near-Term Projections. *Joint Technical Report*, 2012.
[40] R. Simpson. Levelized cost of energy from residential to large scale pv. In *The Applied Power Electronics Conference and Exposition*, 2012.