1 INTRODUCTION

Computers are used extensively in a variety of applications, e.g., in hosting and searching the Web, in predicting the weather, in managing online markets and social networks, in analyzing genomes, in processing signals to detect gravitational waves, etc. Increased computational demands over the years have resulted in significant energy and power costs to provision for and operate systems. Today, power and energy are among the most critical constraints for the design and use of computer systems [22, 160].

Datacenters host large numbers of computers that serve the computational needs of its users. A recent report [164] states that U.S. datacenters used around 91 billion KWh of electricity, equivalent to the energy consumption of 34 coal-fired 500 MW power plants, in 2013. This is projected to rise to 140 billion KWh of electricity, equivalent to the energy consumption of 50 coal-fired 500 MW power plants and costing around $13 billion, in 2020. Thus, reducing the energy consumption of datacenters will have significant economic and environmental benefits.

Datacenters use energy not only for running servers, but also for operating power distribution systems, cooling systems, lighting systems, etc. The PUE (Power Usage Effectiveness) metric [16] was developed to quantify these extra energy overheads. PUE is the ratio of the total facility energy to the IT equipment energy. Smaller values of PUE are better since they indicate that a greater portion of the total energy is being used to run the IT equipment to do useful work instead of being used up by the non-IT systems mentioned above.

The PUE metric has been very influential in driving down overheads due to non-IT infrastructure. In the early days of the metric, PUE values of around 3.0 were common. In recent years, the average PUE value is 1.7 [108]. A number of modern datacenters report far lower PUE values of around 1.1 [73, 83, 156, 166, 210]. Sophisticated cooling

technologies allow even lower PUE values [1]. PUE, however, does not track overheads in IT infrastructure. For low PUE values, most of the datacenter energy is used by the IT equipment, particularly, by the servers.

In order to quantify a part of the energy losses in the servers, Barroso and Hölzle proposed the SPUE (Server PUE) metric [104]. This tracks losses in the server power supply units (PSUs) that happen due to inefficiencies in converting A.C. power to low voltage D.C. power required by components within the server, e.g., processors, disks, fans, etc. Some modern PSUs can attain upwards of 95% efficiency over a portion of their operating range.

Barroso and Hölzle [22] also observed that servers lose energy during computation due to under-utilization. They observed that the energy efficiency (work done per unit of energy used) of servers peak at maximum utilization but drop drastically as the utilization decreases. The reason for this is energy losses that persist even when the server is idle. At zero utilization, that is, when a server is not performing any useful work, it still draws power. This energy consumption is due to leakage in processor components, DRAM refresh, powered-on hard disks, fans, other components in the motherboard, and high PSU inefficiency at low loads. At higher utilizations, the server needs more energy to perform the given computations. This reduces the relative overheads due to the other components and improves the energy efficiency.

This dependence of server energy efficiency on server utilization means that, when not fully utilized, less work gets done per unit of energy used, or equivalently, more energy is needed to do the same amount of work. This is problematic since servers in datacenters are typically only 10–50% utilized [22], thus using more energy to perform the computations than they would use if fully utilized.

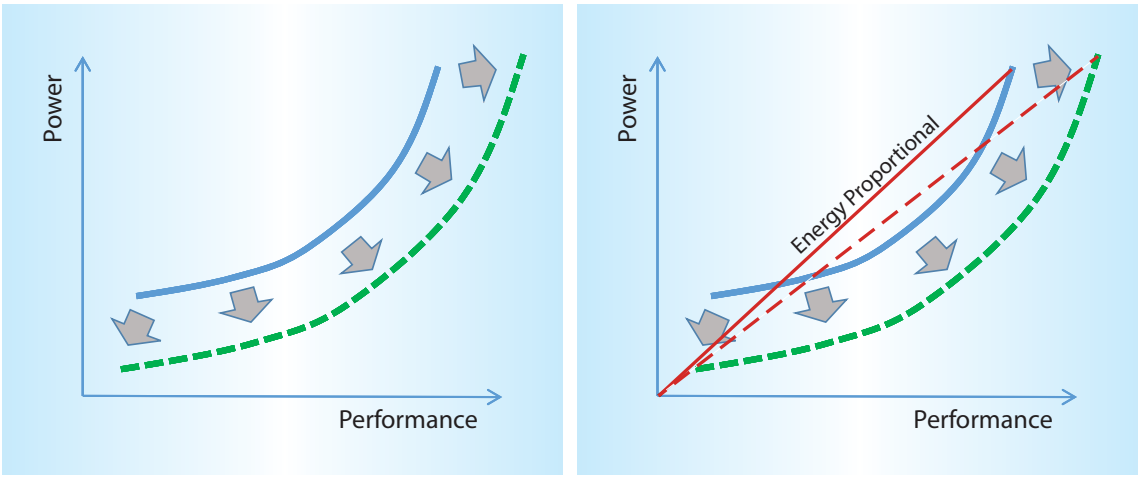
To eliminate utilization-dependent energy losses, Barroso and Hölzle [22] advocated

for "energy-proportional" system designs. Such systems would use energy in proportion to work, or equivalently, power in proportion to utilization. This would ensure that they retain their maximum energy efficiency even at low utilizations. They would significantly save server energy consumption by preventing the drastic drop in energy efficiency at low utilizations.

Power overheads that persist when the system is idle make it non–energy-proportional. Thus, perfect energy-proportional systems must have zero idle power. This model of an ideal system has inspired system designers to build systems that have low idle power (thus, have low power overheads) and a wide dynamic power range (so that the relative power overheads are low at high utilizations).

Conventionally, attaining energy proportionality has been the cherished ideal for minimizing energy waste. However, with recent technological and architectural advances, modern systems may exhibit super-proportional behavior, that is, they exceed the energy efficiency of energy-proportional systems. In these systems, being only energy-proportional is significantly wasteful. Proportionality is a lesser prize to aim for in this changed situation with substantial energy savings remaining to be realized with more ambitious goals.

Figure 1.1 illustrates this point. Figure 1.1a shows a representative power vs performance profile with voltage and frequency scaling for an Intel Haswell processor [98]. Power consumption increases *non-linearly* with performance. Equivalently, energy consumption increases *non-linearly* with the amount of work (computation) done. The dashed line shows how improvements in processor design have lowered the power-performance profile, enabling more energy-efficient operations. Figure 1.1b augments the original graph with "Energy-Proportional" lines. Most of the power-performance curve lies below the energy proportional line. This means that configurations represented by points on the



(a) Per Hammarlund et al. [98] (b) Augmented with Energy-Proportional lines

Figure 1.1: Trends in Processor power-performance profiles.

curve are more efficient, that is, super-proportional. These configurations use less power (and less energy) than a perfect energy-proportional system at the same performance (or, serviced load). Chapter 3 corroborates this observation for our workloads, both batch and interactive ones. Intel's data suggests that this trend is increasing with technological and architectural advances.

Since energy proportionality, which is the conventional ideal model, is no longer sufficient to describe the energy efficiency potential of modern computers, we need to define new ideals for system energy efficiency. These ideals will be useful to system designers for building more efficient future systems and to system operators for operating current systems more efficiently.

Moreover, neither PUE, nor SPUE, nor energy proportionality quantifies waste in computational energy with reference to ideal system operations. We propose a new metric, CPUE (Computational PUE), to address this need.

Super-proportionality happens in conjunction with reconfiguration capabilities found

in modern computers. Many resources may be configurable, e.g., processor frequency (and voltage), cache size, prefetching ability, etc. Reconfigurability and super-proportionality together necessitate the development of new concepts and mechanisms to minimize computational energy waste.

The dissertation focuses on the energy efficiency of reconfigurable computers that may also exhibit super-proportional behavior. We develop new models for ideal system design and operations, new metrics for quantifying computational energy waste and attributing losses to root causes, and mechanisms to efficiently operate such systems.

1.1 Iron Law of Energy

Knowledge about ideal system efficiency and energy waste with respect to the ideal is not sufficient to correct the situation to eliminate energy waste. We also need to quantify the root causes of energy waste.

To do this, we draw inspiration from the popular Iron Law of (processor) Performance [72, 194] that decomposes workload execution time into three components as shown below:

$$\frac{\text{Time}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Time}}{\text{Cycle}}$$

This separation helps compiler writers to focus on improving the first component, computer architects to focus on improving the second component, and circuit designers to focus on improving the third component.

A similar decomposition of workload energy consumption does not exist. We now

propose a new Iron Law of Energy in Chapter 2 as follows:

$$E(c, l) = LUE(l) \times RUE(c, l) \times E_{min}, \qquad l > 0$$

where:

- $E(c, l)$ is the energy consumption by the system to do the work using configuration c at non-zero load (or, processing rate or, performance) l,
- $LUE(l)$ is the relative energy used by the most energy-efficient configuration for load l compared to the energy needed (E_{min}) by the most energy-efficient configuration (the optimal load can be different from l),
- $RUE(c, l)$ is the relative energy used due to operating with configuration c at load l compared to the energy needed by the most energy-efficient configuration for that same load l,
- E_{min} is the minimum energy needed by the system to do this work when operating at the optimal configuration and load.

Similar to the Iron Law of Performance, we expect that the above decomposition will help different actors to focus on particular aspects of energy consumption—system designers to focus on reducing E_{min} and system operators to focus on reducing LUE and RUE through load management and configuration management.

1.1.1 Load Management

One major source of wasted energy is due to suboptimal load levels, particularly low loads, serviced by the machine. This aspect is captured by the LUE term in the Iron Law of Energy. A value greater than 1 for LUE means that energy is wasted due to operating

with non-optimal loads. Chapter 2 shows that non-optimal loads can use in excess of 350% energy compared to that needed at optimal loads. System operators can prevent or mitigate this loss by managing load levels serviced by machines.

In a datacenter, load management is a global policy decision since multiple machines are affected by it, either to absorb load from or relinquish load to other machines so that the total serviced load is unaffected. This may not always be possible, e.g., in the case of stateful services with expensive state migration costs.

We do not focus on mechanisms to perform inter-server load management in this dissertation.

1.1.2 Configuration Management

Modern computers are reconfigurable in multiple ways and not being careful about the operating configuration can be extremely wasteful in terms of energy consumption. This aspect is captured by the RUE term in the Iron Law of Energy. A value greater than 1 for RUE means that energy is wasted due to operating with non-optimal configurations. As Chapter 2 shows, up to 51% more energy can be used by suboptimal configurations even if load is managed to be in the optimal range. For a given load, configuration management is local to the machine in the sense that other machines need not be aware of or affected by the configuration of the target machine.

This dissertation focuses primarily on configuration management of single machines—management of processor frequency (Chapters 3 and 5), cache prefetching (Chapter 3), and cache organization (Chapters 4 and 5).

1.2 Service-Level Agreements (SLA)-aware Governors

We use existing terminology of calling resource management policies as governors. System operators may need to use governors to ensure that runtime Service-Level Agreements (SLAs) are satisfied. There is thus a need for SLA-aware governors. Some examples of SLAs are:

- Maximize energy efficiency, that is, minimize energy used.
- Maximize performance while operating within a given power budget.
- Maximize power savings, that is, minimize power consumption, while meeting a given performance target.

Minimizing system energy consumption reduces the operational cost of machines and datacenters by reducing electricity bills. Limiting/capping power consumption has multiple benefits as follows.

1. It generates less heat and thereby reduces cooling needs and operational costs.
2. It requires provisioning for a smaller amount of total power to the system and datacenter. This reduces capital costs.
3. It allows for better utilization of datacenter capacity. Otherwise, the datacenter has to be provisioned for worst-case/nameplate power consumption of all servers whereas most servers are poorly utilized leading to stranded capacity. Improving datacenter capacity utilization can significantly reduce the Total Cost of Ownership (TCO) [21].

Operating under power constraints is important for both servers and mobile systems [98]. However, this may slow down computation. So the user may want to specify SLAs that include performance targets that should be reached.

Linux distributions include governors that manage processor frequency. Some of the well-known governors are Performance (highest frequency), Powersave (lowest frequency) and OnDemand (dynamically change frequency according to utilization). The OnDemand governor is the default for many distributions. However, these governors are not sufficient for managing modern servers. There are three main reasons why the existing Linux governors are inadequate:

1. The existing governors only control processor frequency. However, other aspects of the system, e.g., prefetching, number of active cores, etc. are also controllable at run time. System designers are increasingly making reconfigurable interfaces public and new governors should exploit those to increase energy savings.

2. There is no way for the user to specify SLAs/high-level management goals, e.g., minimize energy while meeting a performance target. The assumption made by the existing governors is that the user always wants peak performance or lowest power, but there could be other management goals. New governors need to be more expressive in order to meet the needs of the users.

3. Modern processors automatically transition to low-power states when idle. This significantly reduces the utility of the OnDemand governor. We will show in Chapter 3 that the OnDemand governor performs almost identically to the Performance governor for all of our workloads.

Chapter 3 shows that, for the SPECpower benchmark, the existing governors are significantly energy-inefficient for most load levels other than peak and idle. We see

significant inefficiencies for batch workloads as well. We develop new governors that address the shortcomings of the current governors by considering prefetch control and cache resizing in addition to frequency control, by being SLA-aware, and by aiming to constrain system operations to the Pareto frontier (Dynamic EO).

1.3 Contributions

The main contributions of this dissertation are:

1. **Definition of new ideals and metrics for reasoning about energy efficiency.**

   Instead of Energy Proportional (EP), we propose Energy Optimal Proportional (EOP) as the new *design ideal* for energy efficiency. A system that is EOP will always use minimum energy, E_{min}, (or equivalently, always have maximum efficiency) to do a given amount of work irrespective of the load. EOP thus characterizes a lower bound on the energy consumption, or equivalently, an upper bound on the energy efficiency of the given system. EOP will be helpful to system designers as they improve the system's maximum energy efficiency (reduces E_{min}) and make a greater portion of the operating range closer to EOP (reduces LUE over a greater range of loads).

   For system operators, we propose a new *operational ideal* called Dynamic Energy Optimal (Dynamic EO). This is determined by the set of system configurations that have the lowest power among all configurations that can serve the same load. System operators should strive to operate their system close to Dynamic EO. This will ensure that RUE is close to 1.

   We also propose a new metric, CPUE, for quantifying computational energy waste. Our new Iron Law of Energy in Chapter 2 helps quantify two operational con-

tributors to wasted energy—non-optimal loads and non-optimal configurations for serving the loads—that will help system operators to focus better on load management and configuration management.

Chapter 2 discusses these new ideals and energy consumption metrics in more detail.

2. **Development of new SLA-aware governors that control both processor frequency and cache prefetching.**

   In contrast to the existing Linux governors, our governors

   a) take user-specified SLAs into account while managing resources, and

   b) aim to constrain system operations to Dynamic EO.

   Chapter 3 shows that our new governors save significant energy compared to the existing Linux governors. Two of our workloads, `md` and SPECpower, show significant performance improvements for the same power budget with dynamic control of prefetch settings.

3. **Development of new cache performance models based on reuse distance properties to determine optimal cache size and associativity at runtime.**

   Since workloads differ in their cache utilization properties, being able to efficiently and accurately predict cache performance for different cache sizes and organizations is important for saving cache leakage energy, with controlled performance impact, by dynamically resizing the cache. Determining the optimal cache configuration without needing to try out all possible configurations requires low-cost models that can be used online to predict the performance of potential target cache configurations. Our analytical models are based on the reuse distance distributions of accesses to

the cache. The reuse distance of an element in an address stream is the number of unique elements accessed between two successive accesses to the same element. Chapter 4 develops online methods that monitor cache access streams to determine reuse distance distributions and use that to predict cache miss rates for any cache size and associativity.

In contrast to earlier work on way counters [175, 214] that can predict cache miss rates only for different associativities, our models can predict cache performance for different cache sizes and associativity not only for LRU, but also for other replacement policies such as PLRU, RANDOM, and NMRU.

4. **Development of a new governor that controls for cache organization (size, number of sets) and processor frequency.**

   Depending on the workload characteristics, the last-level cache can be resized and the processor frequency increased to get more performance for the same power budget. Chapter 5 develops a governor that controls these knobs to achieve 0.5–15% performance improvement for a given power budget.

5. **Development of a new classification system for system reconfiguration capabilities.**

   Computer architecture has been greatly enriched by classifications/taxonomies of various aspects of system design and operation, such as Flynn's classification of machine organizations [78], Hill's 3C classification of cache misses [103], Wang-Baer-Levy's classification of virtual-real cache hierarchies [227], etc. We propose a new classification system for reconfigurability, based on the semantics of reconfiguration knobs, in Chapter 6. We hope that this classification is more insightful than a component-based or mechanism-based classification of reconfiguration knobs.

1.4 Implications

The implications of our work are three-fold.

Firstly, the notion of conventional energy proportionality (EP) being a model of ideal system efficiency is no longer true. This has resulted from modern systems being reconfigurable and super-proportional. The energy efficiency of the system at its peak performing point can be significantly less than the best that it can achieve. Consequently, aiming to attain EP may result in significant lost opportunities in improving efficiency. Instead, system designers should aim to attain EOP.

This also means that comparisons between systems based on the energy consumptions at their peak performing points can be misleading since that may not be the best energy efficiency realizable on either system. EOP can form a basis for such comparisons.

Further, the policy of race-to-halt (running the system at its highest speed and then shutting it down to save power) can be suboptimal in terms of energy consumption since race-to-halt aims for attaining the EP power-performance profile, not the EOP profile. Race-to-halt is thus more of a performance-optimal policy rather than an energy-optimal one. As we will demonstrate with our new reactive governors for SPECpower [205] in Chapter 3, it may be more energy efficient to control processing speed so that the system is never under-utilized while still serving the offered load. This strategy minimizes idle time whereas the race-to-halt policy maximizes idle time. Our new governors employ a "jog-to-halt" policy that subsumes race-to-halt by selecting processing speeds that minimize energy consumption while meeting performance targets or power caps.

Secondly, while the intense focus on datacenter PUE has led to significant reductions in datacenter cooling overheads, it has also resulted in IT equipment inefficiencies being one of the largest contributors to energy waste in modern datacenters. With increasing computational demands and electricity costs and reducing PUE numbers, optimizing

computational energy through smart server reconfigurations and load balancing will result in more overall savings than it has in the past. It is important to have useful metrics that can guide such operating decisions. Our new metrics—CPUE, LUE, and RUE—can help to analyze and guide those decisions.

Thirdly, existing governors in Linux as well as RAPL capabilities in modern processors are inadequate for achieving maximum energy efficiency. Currently, Linux governors do not consider the full range of reconfiguration capabilities present in the system. Currently, RAPL also has the same limitation and additionally only guarantees a maximum power cap, but ignores performance considerations. Thus, any performance, including suboptimal ones, is possible within that cap. Decoupling power management from performance management risks missing performance goals or energy goals or both.

Finally, we hope and believe that the concepts and models presented in this dissertation will help improve the energy efficiency of future data centers, that in turn will lower energy needs and associated environmental impacts, increase computational capacity within existing budgets, and promote job growth through improved profit margins.