2 ENERGY EFFICIENCY IDEALS AND THE IRON LAW

Energy efficiency is the new fundamental limiter of processor performance, way beyond numbers of processors.

— Shekhar Borkar and Andrew A. Chien [35]

2.1 Overview

Energy efficiency is the work done per unit amount of energy consumed. Maximizing energy efficiency is important as it allows more work to be done for a given energy budget and also allows work to be done faster for a given power budget. This has economic and environmental benefits as it minimizes the energy needed to do a given computation.

While compute capability, in terms of the number of transistors per chip, has steadily increased (Moore's Law [158]), operating voltage has not reduced in proportion (limited Dennard scaling [63]). Borkar and Chien [35] observed that although Pollack's rule [170] predicts a speedup potential (beyond speedups in transistor switching) in proportion to the square root of the number of transistors in a processor, energy-efficiency concerns discourage many microarchitectural techniques that can enable those performance gains. Thus, improving energy efficiency will also improve processor performance.

Energy-proportional computing, that uses energy in proportion to the work done, is an important concept for energy-efficient systems since it seeks to eliminate energy waste by only using as much energy as the work done. However, modern computers are not energy proportional. For example, they use non-trivial power when they are powered on but not used. This is due to processor leakage power, DRAM refresh, and power draw by various components such as fans, hard disks, etc. Energy proportionality has been an useful goal for system designers to make their systems more energy-efficient.

One way to increase proportionality could be to use innovative power-delivery solutions. PowerNap [151] proposed a new power delivery system called RAILS that reduces idle power consumption and proposed to rapidly transition the system to a nap (sleep) state. The nap state retains volatile information, e.g., memory state. With an expected transition time of 10ms or less, the system should be able to save power during idle periods of short durations. The RAILS system consists of multiple power supplies to improve upon the low efficiency of individual power supply units. The RAILS supplies are provisioned such that the power consumption of the idle system is in the efficient operating range of a single supply. As server blades become active, more RAILS supplies get electrically connected so that all the supplies operate in their efficient ranges. However, the PowerNap approach does not work well for some workloads, e.g., OLDI workloads, because full system idleness is relatively rare [152]. Moreover, the napping opportunity decreases further as the number of cores increase [154].

Modern computers also have reconfigurable resources, e.g., processor voltage and frequency levels. We show that intelligent reconfiguration can cause these computers to exceed the efficiency of conventional energy-proportional machines when they are performing work. The original definition of energy-proportional computing, first proposed by Barroso and Hölzle, does not characterize the energy efficiency of recent reconfigurable computers, resulting in non-intuitive "super-proportional" behavior (more work done in proportion to the energy used). This chapter introduces a new definition of "ideal" energy-proportional computing and new metrics to help guide both system architects and operators to configure systems to operate close to this ideal efficiency.

We show that the traditional ideal of energy-proportional may be significantly energy inefficient, and hence, not suited to be an ideal model. Instead, we propose Energy Optimal Proportional (EOP) as the new ideal model for system designers. An ideal EOP system has the maximum efficiency over its entire performance range. Making systems more EOP is a design goal for system architects.

Currently, real systems are not EOP any more than earlier systems were EP. We propose Dynamic EO (Dynamic Energy Optimal), which is the power-performance Pareto frontier and which can be realized on the current system, as the new ideal model for system operators. The Pareto frontier is a set of Pareto-optimal configurations. A system configuration is Pareto optimal if it is not possible to reconfigure the system to improve performance without also increasing power consumption or to reduce power consumption without also degrading performance. Pareto-optimal power-performance system configurations help enforce service-level objectives such as maximizing performance for a given power budget or minimizing power for a given performance target, both leading to energy savings. System operators should aim for Dynamic EO to achieve power-efficient performance for the current system.

This chapter focuses on defining new ideals for energy proportional computing and new metrics to quantify operational energy wastage of computing systems.
The main contributions of this chapter are:

1. We show that the conventional "ideal" model of energy proportionality does not fully describe the energy efficiency potential of modern super-proportional systems.

2. We propose new ideals for both system designers and system operators. EOP is the new design ideal that subsumes conventional "ideal" energy proportionality. Dynamic Energy Optimal (Dynamic EO), that is the power-performance Pareto frontier, is the new operational ideal.

3. We propose a new metric called Computational Power Usage Effectiveness (CPUE) to quantify excess computational energy used with respect to that by EOP.

4. We propose new metrics, Load Usage Effectiveness (LUE) and Resource Usage Effectiveness (RUE), that can help system operators to focus on load management and configuration management to make the system operate efficiently.

5. We develop the "Iron Law of Energy" that quantifies the impact of poor load management and poor configuration management on CPUE.

Section 2.2 defines energy efficiency and describes our experimental setup. Section 2.3 shows why the conventional "ideal" model is inadequate for modern systems. Section 2.4 proposes our new design and operational ideals. Section 2.5 discusses several properties of the power-performance Pareto frontier and their implications on managing for efficient operations. Section 2.6 proposes a new metric for quantifying energy waste and its decomposition into two components, pertaining to load management and configuration management. Section 2.8 describes how Pareto frontiers of individual systems can be composed to determine the Pareto frontier for a collection of systems. Section 2.7 discusses some of the overheads and challenges involved in energy-efficient scheduling.

2.2 Terminology and Infrastructure

Similar to Barroso and Hölzle [22, 104], we define energy efficiency as $\frac{\text{Work}}{\text{Energy}}$, or equivalently, $\frac{\text{Performance}}{\text{Power}}$. The performance of a system is measured as the rate of doing work, e.g., the load serviced, or transactions completed per unit time. Performance normalized to that at peak load levels is the system utilization [22].

The system that we use in this work is a single-socket quad-core Haswell-based Xeon E3-1275 v3 server with 32 GB memory (DDR3-1600), henceforth referred to as HS. HS runs RHEL with kernel version 2.6.32. It has a frequency range of 0.8–3.9 GHz with 3.5^{+}–3.9 GHz being the turbo boost region. The turbo boost plan is 2/3/4/4 meaning that the maximum frequency can be 3.5 + 0.1*2 = 3.7 GHz with all four cores active, 3.5 + 0.1*3 = 3.8 GHz with three cores active, and 3.5 + 0.1*4 = 3.9 GHz with two or one cores active. We run the system with all four cores, hyperthreading (2 hardware threads per core, that is, 8 hardware threads per socket), and cache prefetching enabled by default. All cores run at the same frequency (except perhaps in turbo mode where individual cores may be throttled differently). The socket frequency can changed in steps of 100 MHz by writing to Model Specific Registers. Any value for the turbo region implies a limit on the maximum frequency. HS has a socket TDP of 84W and a remarkably low socket power of ~0.27W when idle. DRAM idle power is ~4.3W.

We use the SPECpower benchmark [205] in this chapter. This Java workload simulates warehouse transaction processing, with (by default) as many warehouses as logical processors on the system under test, that is, the server. Transaction requests to each warehouse arrive in batches of 1000 transactions each. The batches have (negative) exponentially distributed interarrival times. The server load is measured in total transactions per second. The workload first calibrates the maximum, or 100%, load. Next, it does measurement intervals by varying the load offered to the system under test from 100% (max. utilization) to 0% (no utilization) in decrements of 10%. In these intervals, the load served must be within 2% (up to 2.5% shortfall for the 100% and 90% intervals is allowed) of the offered load. We use a Watts Up? (.net) meter [107] for system (wall) power measurements. SPECpower uses its own software utility (daemon) for periodically measuring and reporting system power. SPECpower reports power numbers only for the measurement intervals. This is what is plotted against performance in all the graphs for SPECpower profiles.

We refer to 100% load as the maximum load achieved for the Peak Performance Configuration (all cores at the highest frequency and prefetching enabled). All loads are normalized with respect to that peak load.

2.3 Inadequacy of Conventional Energy Efficiency Ideals

We see that peak energy efficiency occurs at peak utilization and drops quickly as utilization decreases.

— Luiz André Barroso and Urs Hölzle [22]

The average efficiency is always less than the peak efficiency; modern servers are only maximally efficient at 100%.

— David Meisner and Thomas F. Wenisch [153]

Barroso and Hölzle observed that real systems—at that time—attain peak efficiency at peak utilization, but quickly lose efficiency as utilization drops as they are unable to proportionately reduce power consumption. They posit that an "ideal" energy-proportional system should always use energy in proportion to the work done, by maintaining this peak efficiency even at reduced load.

Figures 2.1 and 2.2 illustrate this original model for HS running SPECpower. Figure 2.1 shows the server's power-performance profile at different load levels with the highest processor frequency. We label these points with *Peak Performance Configuration*) since the machine can serve maximum load (peak performance) with this configuration.

The *EP* line represents Barroso and Hölzle's "ideal" energy-proportional profile where performance is linearly proportional to power. We consider this a *design ideal* for future systems, since current systems have unavoidable idle power consumption. The *Dynamic EP* line accounts for idle power [141], and represents an *operational ideal* for the current

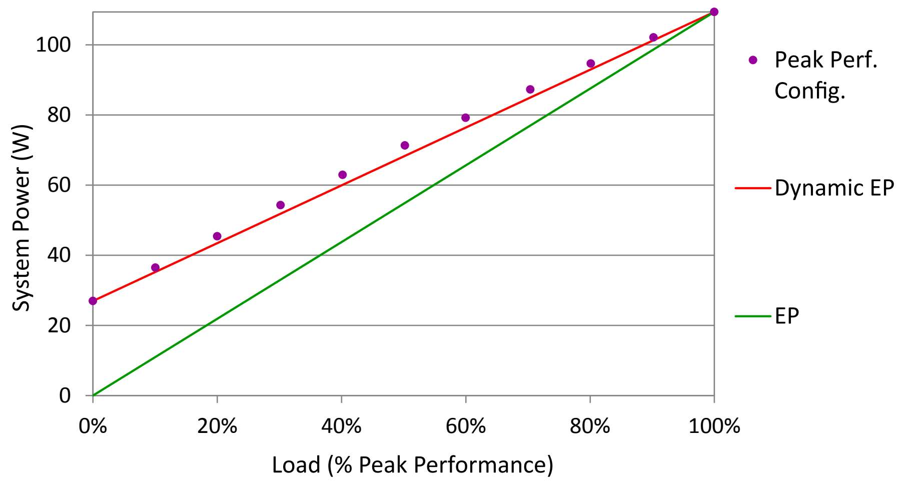

Figure 2.1: Power-Performance profile with conventional server configuration.

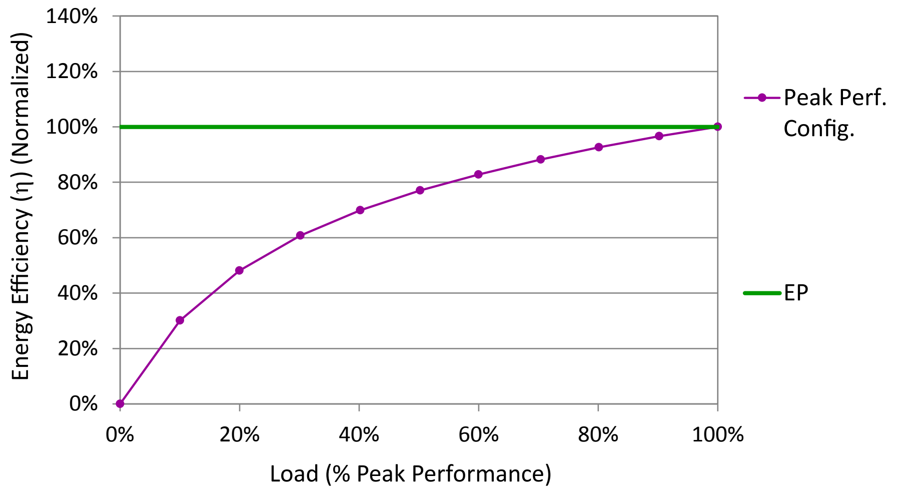

Figure 2.2: Conventional efficiency model of servers.

system. This server's Peak Performance Configuration achieves power-performance very close to Dynamic EP. Figure 2.2 shows that the corresponding energy efficiency (η), normalized to that at peak performance, reduces quickly from 100% as performance drops. In contrast, an EP system is always 100% efficient.

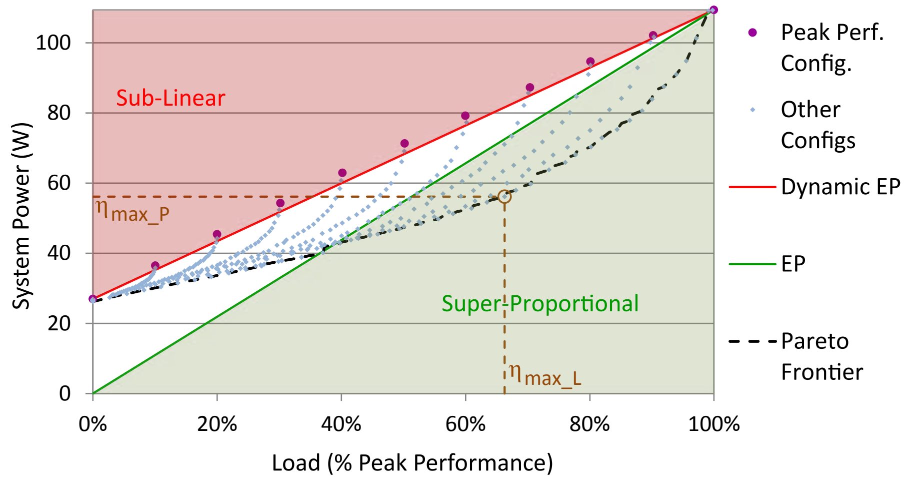

Figure 2.3: Power-Performance profile for super-proportional systems.

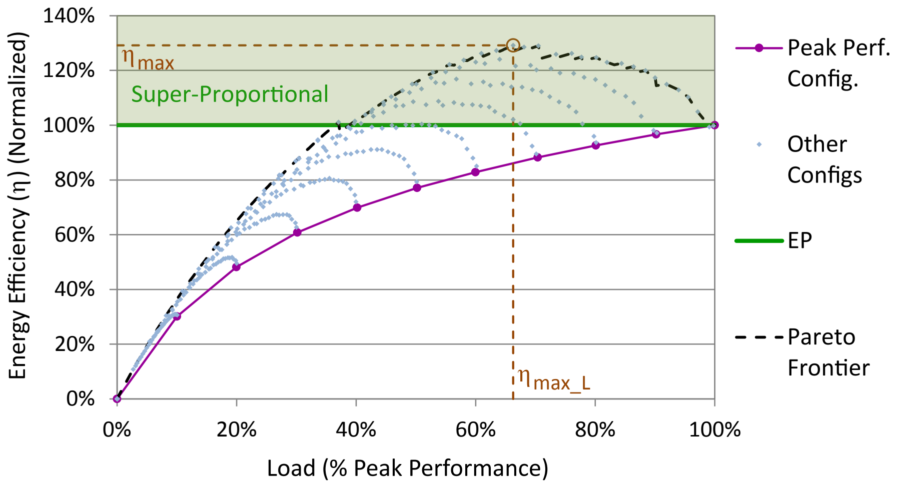

Figure 2.4: Performance (Load) vs Efficiency for super-proportional systems.

Barroso and Hölzle's observation has been instrumental in helping drive recent system designs to have lower idle power and a wide dynamic power range. However, their model describes systems with *fixed resources*, while these modern, more-efficient processors have *reconfigurable resources*—e.g., core frequencies, voltages, number of active cores, threads per core, etc. that can be varied at runtime.

Operating with fixed resources can be inefficient when a server faces variable loads, either due to fluctuating demands, or service consolidation and load balancing among other servers [47, 60, 148].

Servers are usually configured for maximum performance (that is, the Peak Performance Configuration), but other configurations can trade performance for greater energy efficiency. Figure 2.3 shows that changing just the socket frequency (and consequently voltage) results in energy efficiency that exceeds the "ideal" EP profile. Specifically, by varying the frequency from 3.9 to 0.8 GHz, the Haswell server can achieve super-proportional efficiency over almost 60% of the performance range (points in the shaded Super-Proportional region—where performance is super-proportional to power). Figure 2.4 shows that the maximum efficiency (η_{max}, occurring at approximately two-thirds load) is 29% higher relative to the EP energy efficiency, for this server.

Reconfigurable systems create opportunities for increased efficiency even outside the super-proportional region. For example, Figure 2.4 shows that the Peak Performance Configuration attains a relative efficiency of 61% at 30% load, while a different configuration achieves a relative efficiency of 88% at the same load. In other words, the usual server configuration uses 44% more energy than necessary to satisfy the same load, despite being nearly on the Dynamic EP line.

Ideally systems would exhibit energy-proportionality, wherein servers consume power in proportion to their load.

— David Meisner et al. [152]

In an energy-proportional system, explicit power management is unnecessary, as power consumption varies naturally with utilization.

— David Meisner et al. [151]

energy-proportional computing must be the ultimate goal for both hardware architecture and software-application design.

— Shekhar Borkar and Andrew A. Chien [35]

As we have demonstrated, neither EP nor Dynamic EP (that is, the conventional ideal models) describes the full potential of modern computing systems. While non-linearity with reconfiguration is well-known, e.g., with frequency (and voltage) control, the existing ideal models do not consider its impact on peak efficiency. New models are needed to aid operating system schedulers and system administrators to configure systems to deliver maximum efficiency.

2.4 Redefining EP and Dynamic EP

The EP model assumes that maximum energy efficiency occurs at maximum (100%) load and argues that an ideal system should achieve that efficiency for all loads. Yet Figure 2.4 shows that a reconfigurable server actually attains maximum efficiency (η_{max}) at a lower load ($\eta_{max\_L} < 100\%$). We argue that a better ideal model is one that achieves this optimal efficiency η_{max} for all loads.

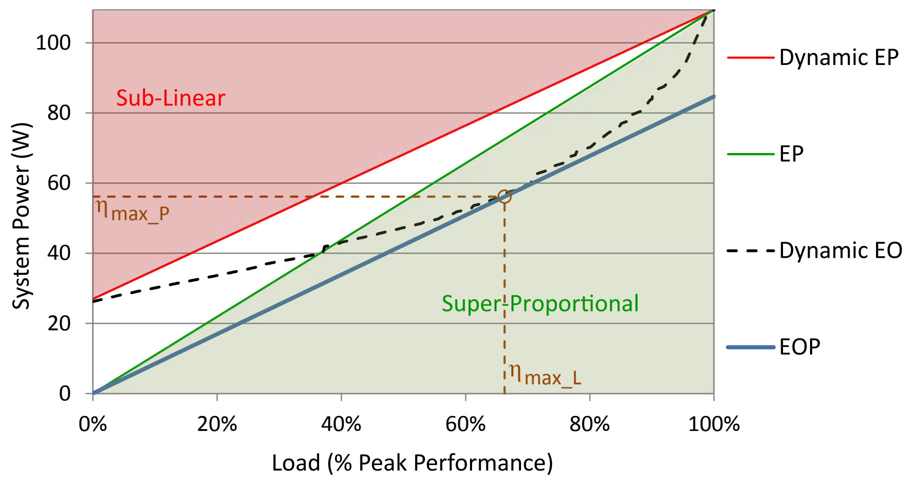

Figure 2.5: EOP and Dynamic EO models.

Similar to the EP model, the ideal system should have maximum efficiency (η_{max}) at every load. This implies that for a given computation, it will use minimum energy (E_{min}) to do it irrespective of the computing rate (performance or load). Figure 2.5 shows its geometric interpretation as a straight line passing through the points (0, 0) and (η_{max_L}, η_{max_P}). This ideal system, that is energy optimal at every load, uses power linearly proportional to load (l/η_{max} power at load l). Energy optimality at every load implies energy proportionality, but the converse is not true, e.g., EP is proportional but not optimal at all loads.

We call this new model *EOP* (Energy Optimal Proportional) since it is both optimal and proportional. EOP is a *design ideal* that gives system designers a way to measure how far the energy efficiency of a target design differs from the best possible design, hopefully leading to more energy-efficient systems. EOP subsumes the EP model for all systems—it improves upon EP for super-proportional systems and is identical to it for all others.

Of course real systems are unlikely to achieve this design ideal, e.g., due to unavoidable idle power, so system software needs an operational model that characterizes the maximum efficiency that can be realized by the current system at different loads. We address this using the well-known power-performance *Pareto frontier* [19, 27, 183], shown as a dashed line in Figures 2.3–2.6. The Pareto frontier represents configurations in the current system that use the lowest power, and hence are the most efficient, among all configurations that can serve a given load. These configurations are Pareto optimal in the sense that, among these configurations, one cannot reduce power without also reducing load or increase load without also increasing power.

We call this model *Dynamic EO*. Like Dynamic EP, it is an operational ideal that seeks to characterize the best energy efficiency that can be achieved for a given system. But it differs from Dynamic EP in two aspects—it characterizes optimality that can already be realized by some among the multitude of configurations in the current system and it does not assume linearity of the power-performance profile.

Figure 2.5 illustrates the different models. These are the

- *design ideals*: conventional (EP), new (EOP), and
- *operational ideals*: conventional (Dynamic EP), new (Dynamic EO).

The EOP line meets (is tangential to) the Dynamic EO line only at points having the maximum efficiency ($\eta_{\max}$).The following energy efficiency relations hold for any system:

$$\text{Dynamic EP} \leqslant \text{EP} \leqslant \text{EOP}$$
$$\text{Dynamic EO} \leqslant \text{EOP}$$

where $\leqslant$ means less than or equal to for values of efficiency. Systems, like our server, that can operate in the non-Sub-Linear region for any portion of their performance range have Dynamic EP $\leqslant$ Dynamic EO for all such loads.

2.5 Power-Performance Pareto Frontier (Dynamic EO)

In this Section we describe some properties of Dynamic EO and their implications for optimal system operations.

Every configuration of the system can be characterized by its performance and power consumption. We call each such (Configuration, Performance, Power) tuple a system state. The Pareto frontier is determined by only those states that use the lowest power among all states having at least that performance. It is a subset of the set of system states. The governors that we develop in Chapters 3 and 5 seek to constrain system operations to Pareto-optimal states.

Let Π denote the set of system states with Π_i representing the i^{th} state having performance $\Pi_i.Perf$ and power consumption $\Pi_i.Power$. Let the highest performing state be Π_0. We apply the well-known concepts of Pareto dominance and Pareto optimality. State Π_i Pareto-dominates state Π_j if $(\Pi_i.Perf \geqslant \Pi_j.Perf) \wedge (\Pi_i.Power \leqslant \Pi_j.Power)$.

Property 1: *The Pareto frontier is the set of non-dominated states.*

In Figures 2.3 and 2.4, the Pareto frontier is the set of states represented by the dashed line. The states that lie on the EP line in the Super-Proportional region are dominated by the states on the frontier.

Implication: Constraining system operation to the Pareto frontier is important since dominated states are less efficient than dominating states (also see Figure 2.4). The state with the maximum efficiency (η_{max}) lies on the Pareto frontier.

Property 2: *States on the Pareto frontier have the same total order in both power and performance.*

Let Π_i, Π_j be states on the Pareto frontier. Then $(\Pi_i.Perf > \Pi_j.Perf) \iff (\Pi_i.Power > \Pi_j.Power)$. We number the states in decreasing order of performance. The ordering relation for states on the frontier is thus: $i < j \iff (\Pi_i.Perf > \Pi_j.Perf) \wedge (\Pi_i.Power >$

$\Pi_j.\mathsf{Power}$).

Implication: While the state space is inherently two-dimensional, the Pareto frontier is more constrained allowing system operators to qualitatively reason about the other dimension from looking at one dimension alone. For example, increasing the power budget *will* improve performance at the Pareto frontier if the power is used. This is not true for the whole state space where states with less performance can use more power. This positive correlation between the two dimensions exists at the Pareto frontier.

Property 3: *System states that optimize power-performance metrics are located at the Pareto frontier*.

Consider a state Π_i that is not on the frontier. So there exists at least one other state Π_j such that $\Pi_i.\mathsf{Perf} \geqslant \Pi_j.\mathsf{Perf}$ and $\Pi_i.\mathsf{Power} \leqslant \Pi_j.\mathsf{Power}$ with at least one of the inequalities being strict. This implies that the highest performing state with/without a (maximum) power cap and the lowest power state with/without a (minimum) performance bound lie on the Pareto frontier.

In this work we assume that $\mathsf{performance} \propto \mathsf{delay}^{-1}$. Since energy is power multiplied by time (delay), it implies that the lowest energy point with/without a delay cap must lie on the Pareto frontier. Since the state corresponding to the highest performance-per-watt is the same as the state with the lowest energy, that state will also be on the Pareto frontier. Moreover, according to the above condition, states corresponding to the minimum energy-delay (ED) product or ED^2 product or, in fact, any ED$^n, n \geqslant 0$ must also lie on the Pareto frontier.

Since states on the Pareto frontier are more efficient than other states, the highest performing state with/without a maximum power cap, the lowest power state with/without a minimum performance bound, the highest performance-per-watt state, the lowest energy state, the lowest energy-delay state, etc. will lie on the Pareto frontier.

Implication: Optimizing system operations for commonly used power-performance or energy efficiency metric necessitates operating it at the Pareto frontier.

Property 4: *The points of contact between the frontier,* $\mathsf{Power} = f(\mathsf{Perf})$*, and the tangent curve* $\mathsf{Power} = c_n(\mathsf{Perf})^{n+1}, n+1 \geqslant 0$ *and some constant* c_n*, represent configurations that optimize (minimize) metric* ED^n*. (*$n = 0$ *means energy E.)*

Let Π_i be a state that optimizes (minimizes) metric ED^n. By Property 3, Π_i must be on the frontier. Since $\mathsf{E} = \mathsf{Power}(\mathsf{Perf})^{-1}$ and $\mathsf{ED}^n = \mathsf{Power}(\mathsf{Perf})^{-n-1}$, Π_i will be on the curve for the power function $\mathsf{Power} = c_n(\mathsf{Perf})^{n+1}$ if we choose $c_n = \Pi_i.\mathsf{Power}(\Pi_i.\mathsf{Perf})^{-n-1}$. c_n is thus the optimum value for ED^n. Moreover, every point on this power function curve will have the same value for ED^n, which is c_n. No part of the frontier can be below this curve, as then states on this part of the frontier will have lower power for the same performance compared to points on the power function curve directly above them and thus have a smaller value for ED^n than c_n which is a contradiction.

Note that all points on the curve above the linear tangent are suboptimal with respect to E, all points above the quadratic tangent are suboptimal with respect to ED, all points above the cubic tangent are suboptimal with respect to ED^2, and so on.

Implication: This forms the basis for the geometric interpretation of the Pareto Proportional line described in Section 2.4. Every point on the linear tangent has the same slope, which is equal to $\frac{\text{Power}}{\text{Performance}}$, that is, performance-per-watt^{-1} value of the most energy-efficient point.

Property 5: *The Pareto frontier is not necessarily convex (or concave).*

Let Π_i, Π_j, Π_k be states on the frontier with $i < j < k$. The ordering relations only imply $\Pi_i.\mathsf{Perf} > \Pi_j.\mathsf{Perf} > \Pi_k.\mathsf{Perf}$ and $\Pi_i.\mathsf{Power} > \Pi_j.\mathsf{Power} > \Pi_k.\mathsf{Power}$, not $\Pi_j.\mathsf{Power} \leqslant \Pi_k.\mathsf{Power} + \left(\frac{\Pi_j.\mathsf{Perf} - \Pi_k.\mathsf{Perf}}{\Pi_i.\mathsf{Perf} - \Pi_k.\mathsf{Perf}}\right)(\Pi_i.\mathsf{Power} - \Pi_k.\mathsf{Power})$.

Implication: Convex optimization approaches cannot be directly applied while composing multiple Pareto frontiers. Moreover, hill-climbing based search techniques at the frontier can get stuck in local optima instead of reaching global optima. However, as we show in Section 2.8, convex (polynomial) approximations to the Pareto frontier may work well enabling applications of efficient optimization techniques.

2.6 Computational PUE

Datacenters can satisfy a given load by distributing it to machines in different ways. Each machine can also be configured in a large number of ways. These modes for servicing the load differ in the amount of energy consumed, since some modes are more inefficient than others.

A hypothetical ideal system, that is, one that meets the design ideal EOP, achieves maximal energy efficiency (η_{max}) and thus minimizes the energy (E_{min}) needed for a given computation regardless of load. We would like a metric to quantify the excess energy used by a real system, compared to this ideal system.

Our new metric, *Computational Power Usage Effectiveness* (or, CPUE), measures how much energy a server uses with configuration c at load l compared to the energy used by EOP. We define

$$\mathrm{CPUE}(c,l) = \frac{\text{Actual server energy with } c \text{ at } l}{\text{EOP energy at } l}, \qquad l > 0 \tag{2.1}$$

$$= \frac{E(c,l)}{E_{min}}, \qquad l > 0 \tag{2.2}$$

$$\text{Thus}, E(c,l) = \mathrm{CPUE}(c,l) \times E_{min}, \qquad l > 0 \tag{2.3}$$

CPUE(c, l) is inspired by the well-known PUE metric [16] that tracks energy waste for datacenters by taking the ratio of facility energy consumption to energy consumption by IT equipment. PUE > 1 quantifies excess relative energy used by the datacenter due to the non-IT infrastructure. Similarly, CPUE$(c, l) > 1$ quantifies excess relative computational energy used whenever efficiency drops below $\eta_{\max}$.

We have seen that there are two major factors that lead to energy inefficiencies: i) running the system at a non-optimal load and ii) for a given load, running the system with a non-optimal configuration. We can decompose CPUE(c, l) to isolate these two factors.

We defined CPUE(c, l) as $E(c, l)/E_{\min}$. For a given amount of work, energy consumed is inversely proportional to efficiency. Thus,

$$\begin{aligned} \text{CPUE}(c,l) &= \frac{\eta_{\max}}{\eta(c,l)}, && l > 0 && (2.4)\\ &= \left(\frac{\eta_{\max}}{\eta_{\text{Pareto}}(l)}\right) \times \left(\frac{\eta_{\text{Pareto}}(l)}{\eta(c,l)}\right), && l > 0 && (2.5)\\ &= \text{LUE}(l) \times \text{RUE}(c,l), && l > 0 && (2.6)\\ \text{Thus, } E(c,l) &= \text{LUE}(l) \times \text{RUE}(c,l) \times E_{\min}, && l > 0 && (2.7) \end{aligned}$$

where LUE(l) denotes *Load Usage Effectiveness* at load l and RUE(c, l) denotes *Resource Usage Effectiveness* of configuration c and load l.

LUE(l) is the efficiency of EOP$(\eta_{\max})$ relative to that of of Dynamic EO at load l. LUE$(l) \geqslant 1$ with LUE$(l) = 1 \iff l$ can be served at maximum efficiency ($\eta_{\max}$). Since energy consumed is inversely proportional to efficiency, LUE$(l) > 1$ quantifies excess energy used, relative to $E_{\min}$, due to non-optimal loads assuming that the Pareto-optimal configuration has been chosen to serve load l.

RUE(c, l) is the efficiency of Dynamic EO relative to that of configuration c, both at load l. $\text{RUE}(c, l) \geqslant 1$ with $\text{RUE}(c, l) = 1 \iff c$ is a Pareto-optimal configuration. $\text{RUE}(c, l) > 1$ quantifies excess energy used, relative to Dynamic EO at load l, due to using non-optimal (Pareto-dominated) configuration c for serving load l.

Inspired by the "Iron Law of Performance", we call Equation 2.7 the "Iron Law of Energy". System designers will focus on minimizing E_{min} whereas system operators will focus on minimizing LUE and RUE.

Both $\text{LUE}(l)$ and $\text{RUE}(c, l)$ can be expressed in terms of $\text{CPUE}(c, l)$. Since $\text{RUE}_{\text{Pareto}}(l) = 1$ for every l, $\text{LUE}(l) = \text{CPUE}_{\text{Pareto}}(l)$ and $\text{RUE}(c, l) = \text{CPUE}(c, l)/\text{CPUE}_{\text{Pareto}}(l)$.

Our proposed RUE and LUE metrics can help system operators isolate the sources of energy inefficiency and guide new policies to reduce it. LUE is important for load management of Pareto-optimal configurations. RUE is important for configuration management for Pareto-dominated configurations. While LUE is applicable to all systems, both old and new, it only partially quantifies energy waste in reconfigurable systems that can be configured in a plurality of ways. RUE completes the quantification.

2.7 Load and Configuration Management

Most data centers are provisioned to meet peak load, but normally operate at much lower load levels. The LUE metric can help operators quantify the potential benefit of deploying load management policies [47, 60, 148], e.g., concentrating load on some servers and shutting down others. Of course, any such policy must also ensure that service-level agreements are still satisfied [171].

Figure 2.6 shows that CPUE for the Peak Performance Configuration is always > 1 (wastes energy) and increases as load decreases. The best CPUE for this configuration is 1.29, occurs at peak load, and implies 29% excess energy used relative to E_{min}. LUE (that is, CPUE for Dynamic EO), on the other hand, first decreases to 1, then increases, revealing

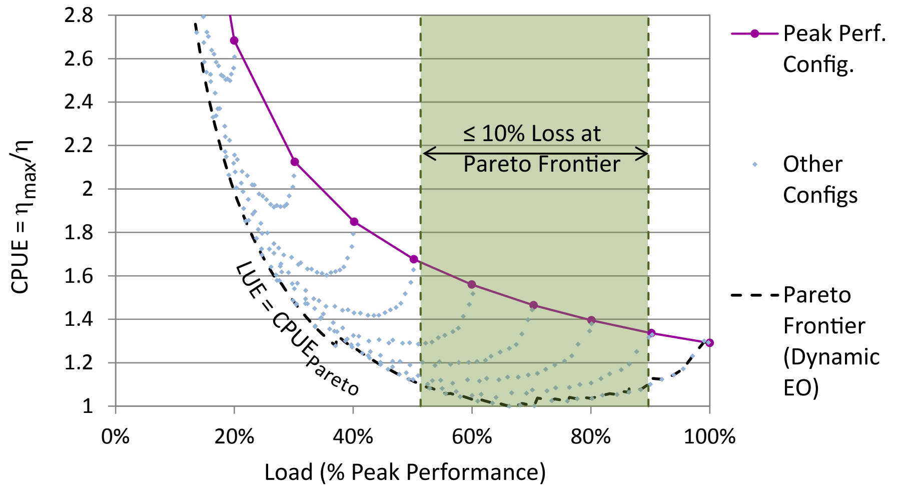

Figure 2.6: CPUE(c, l) and LUE(l). These $\rightarrow \infty$ as load $l \rightarrow 0$ due to non-zero idle power. For any configuration c and load l, CPUE$(c, l) \geqslant$ CPUE$_{\text{Pareto}}(l) =$ LUE$(l) \geqslant 1$.

a sweet spot of $\leqslant$ 10% excess energy used at around 51%–90% of peak performance.

Barroso and Hölzle [22] observed that servers typically operate at 10%–50% load. The LUE curve for SPECpower (Figure 2.6), shows excess energy used due to suboptimal load of approximately 10% at the higher end of this range, to over 250% (not shown) at the lower end. The steep slope of the LUE curve at low loads makes even modest load management very attractive. For example, increasing load from 10% to 20% of peak reduces LUE from 3.55 (255% excess) to 1.99 (99% excess) and a further increase to 25% peak load reduces LUE to 1.68 (68% excess).

Even in a data center with perfect load balancing, reconfigurable servers may be misconfigured, wasting significant energy even at optimal load. Figure 2.7 shows RUE for SPECpower for all system configurations and loads. Operating with the Peak Performance Configuration is significantly wasteful even at low loads, e.g., 21% excess energy used at 10% load compared to operating at Dynamic EO. The excess increases to 51% before decreasing to zero at peak load. Not all Pareto-dominated configurations are as wasteful—

the shaded band identifies configurations that have an RUE of $\leqslant 1.1$ and hence limit the extra energy used to 10%.

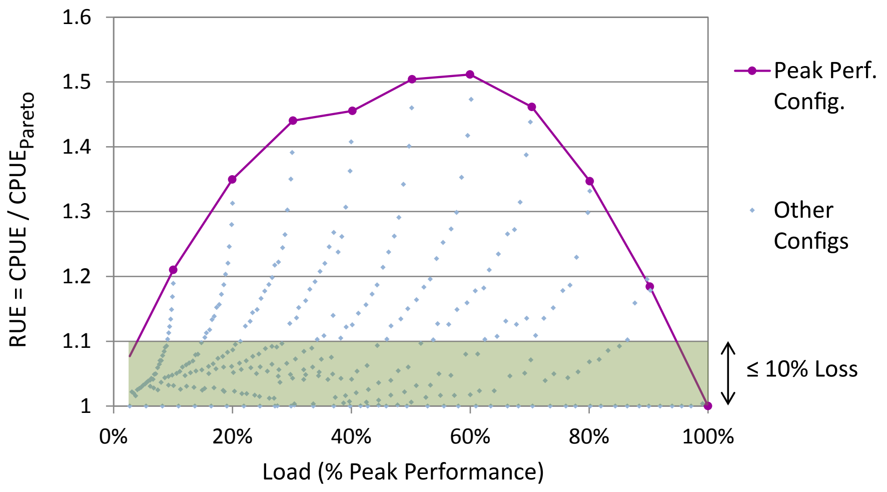

Figure 2.7: Resource Usage Effectiveness.

Configuration management (to reduce RUE) may incur costs, e.g., due to transition times while changing configurations. System designers are making great strides in reducing these costs. For example, processor frequency transitions complete within a few hundred microseconds today.

Calculating LUE and RUE (as well as determining EOP and Dynamic EO) requires knowledge of the Pareto frontier. In this chapter, we determine the frontier offline by running the workload multiple times with the server configured to different frequencies. Offline characterization is also used in prior work [19, 183], but may not be feasible in an online setting with unknown workloads. In Chapter 3 we introduce an online policy that closely approximates the frontier by controlling processor frequency and cache prefetching.

Workload characterization incurs overheads, but researchers have demonstrated [60, 148] its feasibility and utility in large-scale computing environments, e.g., at Google datacenters and Amazon EC2. With modern systems showing trends of increasingly making components reconfigurable, we expect further applications of such techniques to infer characteristics that are relevant to these components.

2.8 The Π-dashboard

With one or more reconfiguration knobs in the system, the user is faced with the daunting task of choosing the right configuration that meets a desired power-performance or energy-efficiency criteria. The Π-dashboard attempts to bridge the gap between high-level power-performance goals and system resource configurations—a mapping capability that is largely missing in today's systems. Π-dashboards enable selection of a variety of power-performance profiles for the system. The user or operating system can select a desired power-performance profile from the Π-dashboard resulting in a "one-shot" transition of the system to the corresponding configuration.

As discussed in Section 2.5, we denote the collection of system states as Π-states. Each state is characterized by the performance and power consumption of the system when operating with that configuration. Pareto-optimal Π-states can be totally ordered (Section 2.5, Property 2). The Π-dashboard is a tabular representation of this totally ordered list of Pareto-optimal Π-states.

Chapter 3 shows how we use power-performance predictors, using hardware counters, to characterize the expected impact of different configurations and subsequently identify Pareto-optimal configurations. A controller/coordinator creates the dashboard from the predictions and interfaces with the user or operating system. It updates the dashboard periodically as execution profiles change over time. The coordinator may be implemented as a software routine (ISR) that runs on one or more cores, or as a specialized unit such as a PCU in modern systems [181].

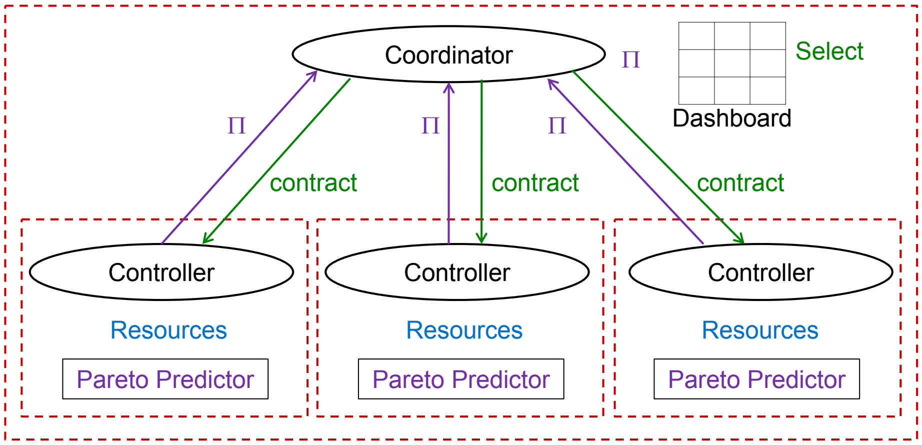

Figure 2.8: Coordination architecture.

Figure 2.8 shows a schematic overview of hierarchical coordination across multiple systems. The Pareto Predictors of individual systems predict power-performance profiles of the reconfigurable Resources of their systems. The local controller/coordinator communicates the Pareto-optimal Π-states to the upper-level coordinator. This coordinator composes the Pareto frontiers to get the overall Pareto frontier and exposes the Π-dashboard to the user. The configurations for the selected profile are then communicated back to the local controllers/coordinators as a "contract" that should be honored by the individual systems for subsequent execution.

The coordinator can compose Pareto frontiers using an optimization program. Let there be n machines, numbered $1..n$. Let $x_{i,j}$ and $y_{i,j}$ respectively denote the performance and power consumption in the j^{th} Pareto-optimal state, Π_j, in machine i. The overall performance range that can be supported is $[\min_{i,j}(x_{i,j}), \sum_{i=1}^{n} x_{i,0}]$. The optimal power consumption, p, for any performance l in this range can be determined by solving the following program:

Model	$\mathbf{R^2}$
Linear	0.9138
Quadratic	0.9813
Cubic	0.9959
Quartic	0.9988
Quintic	0.999

Table 2.1: R^2 values for polynomial fits to SPECpower Pareto frontier.

$$\text{minimize } p = \sum_{i=1}^{n} \sum_{j} I_{i,j} * y_{i,j} \tag{2.8}$$

$$\text{such that } l \leqslant \sum_{i=1}^{n} \sum_{j} I_{i,j} * x_{i,j} \tag{2.9}$$

$$I_{i,j} \in \{0, 1\} \quad \forall i, j \tag{2.10}$$

$$\sum_{j} I_{i,j} = 1 \quad \forall i \tag{2.11}$$

In the above, condition 2.9 requires that the desired performance be met, condition 2.10 allows any state to be either fully selected or not selected, and condition 2.11 requires exactly one state to be selected per system.

In general, the Pareto frontier is not convex (Section 2.5, Property 5). So, a local optima in an optimization program dealing with Pareto frontiers is not necessarily a global optima. But for many real systems, convex (e.g., polynomial) approximations may work well. Table 2.1 shows the coefficient of determination values (R^2 values, best fit value=1) for several polynomial fits to the SPECpower Pareto frontier on HS. A quadratic or higher order approximation works quite well. The approximation errors may be higher if the models are required to include specific points. Convex approximations may reduce the computational effort required to solve the optimization program.

2.9 Conclusion

In this chapter, we explored the relation between two well-known but dissimilar concepts, power-performance Pareto optimality and energy proportionality, both of which share the end goal of making computing more energy efficient. We demonstrated that the conventional model of energy proportionality is inadequate for reconfigurable systems since it does not guarantee energy optimality. We defined a new model, EOP, that guarantees both optimality and proportionality and established its relation to the Pareto frontier.

Real systems are not ideal and hence use more energy than that used by the ideal EOP system (E_{min}). We proposed a new metric, Computational PUE (CPUE), that quantifies how much excess computational energy is used by the system relative to that by EOP. This depends on both the load served and the system configuration used to serve that load.

Our new Iron Law of Energy shows that CPUE can be decomposed into three terms—LUE, RUE, and E_{min}. The LUE and RUE metrics separate the load and configuration aspects of suboptimality. LUE answers the question: how suboptimal is a given load? RUE answers the question: how suboptimal is a given configuration with respect to the most efficient configuration that can also serve that load? LUE is affected by demand fluctuations and inter-server load management whereas RUE is affected by intra-server configuration management.

While system components are increasingly being designed to be reconfigurable, identifying the Pareto frontier is challenging, particularly with multiple reconfigurable resources and dynamically changing runtime environments. Scheduling frameworks that carefully choose configurations and operating ranges will unlock the full potential of current and future reconfigurable systems. This will be our focus in Chapter 3.