Power Routing: Dynamic Power Provisioning in the Data Center

Steven Pelley David Meisner
Pooya Zandevakili Thomas F. Wenisch
Advanced Computer Architecture Lab
University of Michigan
{spelley,meisner,zandv,twenisch}@umich.edu

Jack Underwood
Medical Center Information Technology
Michigan Medical Center
under@med.umich.edu

Abstract

Data center power infrastructure incurs massive capital costs, which typically exceed energy costs over the life of the facility. To squeeze maximum value from the infrastructure, researchers have proposed over-subscribing power circuits, relying on the observation that peak loads are rare. To ensure availability, these proposals employ power capping*, which throttles server performance during utilization spikes to enforce safe power budgets. However, because budgets must be enforced locally—at each power distribution unit (PDU)—local utilization spikes may force throttling even when power delivery capacity is available elsewhere. Moreover, the need to maintain reserve capacity for fault tolerance on power delivery paths magnifies the impact of utilization spikes.*

In this paper, we develop mechanisms to better utilize installed power infrastructure, reducing reserve capacity margins and avoiding performance throttling. Unlike conventional high-availability data centers, where collocated servers share identical primary and secondary power feeds, we reorganize power feeds to create shuffled *power distribution topologies. Shuffled topologies spread secondary power feeds over numerous PDUs, reducing reserve capacity requirements to tolerate a single PDU failure. Second, we propose* Power Routing*, which schedules IT load dynamically across redundant power feeds to: (1) shift slack to servers with growing power demands, and (2) balance power draw across AC phases to reduce heating and improve electrical stability. We describe efficient heuristics for scheduling servers to PDUs (an NP-complete problem). Using data collected from nearly 1000 servers in three production facilities, we demonstrate that these mechanisms can reduce the required power infrastructure capacity relative to conventional high-availability data centers by 32% without performance degradation.*

Categories and Subject Descriptors C.5.5 [*Computer System Implementation*]: Servers

General Terms Design, Measurement

Keywords power infrastructure, data centers

1. Introduction

Data center power provisioning infrastructure incurs massive capital costs—on the order of $10-$25 per Watt of supported IT equipment [18, 26]. Power infrastructure costs can run into the $10's to $100's of millions, and frequently exceed energy costs over the life of the data center [15]. Despite the enormous price tag, overprovisioning remains common at every layer of the power delivery system [8, 13, 15, 17, 18, 23]. Some of this spare capacity arises due to deliberate design. For example, many data centers include redundant power distribution paths for fault tolerance. However, the vast majority arises from the significant challenges of sizing power systems to match unpredictable, time-varying server power demands. Extreme conservatism in nameplate power ratings (to the point where they are typically ignored), variations in system utilization, heterogeneous configurations, and design margins for upgrades all confound data center designers' attempts to squeeze more power from their infrastructure. Furthermore, as the sophistication of power management improves, servers' power demands will become even more variable [19], increasing the data center designers' challenge.

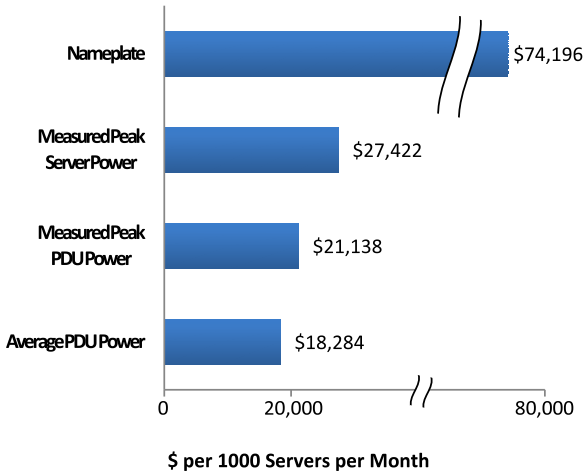


Figure 1: The cost of over-provisioning. Amortized monthly cost of power infrastructure for 1000 servers under varying provisioning schemes.

Although the power demands of individual servers can vary greatly, statistical effects make it unlikely for all servers' demands to peak at the same time [13, 23]. Even in highly-tuned clusters running a single workload, peak utilization is rare, and still falls short of provisioned power capacity [8]. This observation has lead researchers and operators to propose *over-subscribing* power circuits. To avoid overloads that might impact availability, such schemes rely on *power capping* mechanisms that enforce power budgets at individual servers [17, 28] or over ensembles [23, 29]. The most common power-capping approaches rely on throttling server per-

ASPLOS'10, March 13–17, 2010, Pittsburgh, Pennsylvania, USA.

formance to reduce power draw when budgets would otherwise be exceeded [17, 23, 28, 29].

Figure 1 illustrates the cost of conservative provisioning and the potential savings that can be gained by over-subscribing the power infrastructure. The graph shows the amortized monthly capital cost for power infrastructure under varying provisioning schemes. We calculate costs following the methodology of Hamilton [15] assuming high-availability power infrastructure costs $15 per critical-load Watt [26], the power infrastructure has a 15-year lifetime, and the cost of financing is 5% per annum. We derive the distribution of actual server power draws from 24 hours of data collected from 1000 servers in three production facilities (details in Section 5.1). Provisioning power infrastructure based on nameplate ratings results in infrastructure costs over triple the facility's actual need. Hence, operators typically ignore nameplate ratings, instead provisioning infrastructure based on a measured peak power for each class of server hardware. However, even this provisioning method overestimates actual needs—provisioning based on the observed aggregate peak at any power distribution unit (PDU) reduces costs 23%. Provisioning for less-than-peak loads can yield further savings at the cost of some performance degradation (e.g., average power demands are only 87% of peak).

Power capping makes over-subscribing safe. However, power budgets must enforce local (PDU) as well as global (uninterruptible power supply, generator and utility feed) power constraints. Hence, local spikes can lead to sustained performance throttling, even if the data center is lightly utilized and ample power delivery capacity is available elsewhere. Moreover, in high-availability deployments, the need to maintain reserve capacity on redundant power delivery paths to ensure uninterrupted operation in the event of PDU failure magnifies the impact of utilization spikes—not only does the data center's direct demand rise, but also the potential load from failover.

An ideal power delivery system would balance loads across PDUs to ensure asymmetric demand does not arise. Unfortunately, since server power demands vary, it is difficult or impossible to balance PDU loads statically, through clever assignment of servers to PDUs. Such balancing may be achievable dynamically through admission control [4] or virtual machine migration [6], but implies significant complexity, may hurt performance, and may not be applicable to non-virtualized systems. Instead, in this paper, we explore mechanisms to balance load through the *power delivery infrastructure*, by dynamically connecting servers to PDUs.

Our approach, *Power Routing*, builds on widely-used techniques for fault-tolerant power delivery, whereby each server can draw power from either of two redundant feeds. Rather than designating primary and secondary feeds and switching only on failure (or splitting loads evenly across both paths), we instead centrally control the switching of servers to feeds. The soft-switching capability (already present for ease of maintenance in many dual-corded power supplies and rack-level transfer switches) acts as the foundation of a power switching network.

In existing facilities, it is common practice for all servers in a rack or row to share the same pair of redundant power feeds, which makes it impossible to use soft-switching to influence local loading. Our key insight, inspired by the notion of skewed-associative caches [25] and declustering in disk arrays [2]), is to create *shuffled distribution topologies*, where power feed connections are permuted among servers within and across racks. In particular, we seek topologies where servers running the same workload (which are most likely to spike together) connect to distinct pairs of feeds. Such topologies have two implications. First, they spread the responsibility to bear a failing PDU's load over a large number of neighbors, reducing the required reserve capacity at each PDU relative to conventional designs. Second, they create the possibility, through a series of switching actions, to route slack in the power delivery system to a particular server.

Designing such topologies is challenging because similar servers tend to be collocated (e.g., because an organization manages ownership of data center space at the granularity of racks). Shuffled topologies that route power from particular PDUs over myriad paths require wiring that differs markedly from current practice. Moreover, assignments of servers to power feeds must not only meet PDU capacity constraints, they must also: (1) ensure that no overloads occur if any PDU fails (such a failure instantly causes all servers to switch to their alternate power feed); and (2) balance power draws across the three phases of each alternating current (AC) power source to avoid voltage and current fluctuations that increase heating, reduce equipment lifetime, and can precipitate failures [14]. Even given a shuffled topology, power routing remains challenging: we will show that solving the dynamic assignment of servers to PDUs reduces to the partitioning problem [12], and, hence, is NP-complete and infeasible to solve optimally. In this paper, we address each of these challenges, to contribute:

- **Lower reserve capacity margins.** Because more PDUs cooperate to tolerate failures, shuffled topologies reduce per-PDU capacity reserves from 50% of instantaneous load to a $1/N$ fraction, where N is the number of cooperating PDUs.
- **Power routing.** We develop a linear programming-based heuristic algorithm that assigns each server a power feed and budget to minimize power capping, maintain redundancy against a single PDU fault, and balance power draw across phases.
- **Reduced capital expenses.** Using traces from production systems, we demonstrate that our mechanisms reduce power infrastructure capital costs by 32% without performance degradation. With energy-proportional servers, savings reach 47%.

The rest of this paper is organized as follows. In Section 2, we provide background on data center power infrastructure and power capping mechanisms. We describe our mechanisms in Section 3 and detail Power Routing's scheduling algorithm in Section 4. We evaluate our techniques on our production data center traces in Section 5. Finally, in Section 6, we conclude.

2. Background

We begin with a brief overview of data center power provisioning infrastructure and power capping mechanisms. A more extensive introduction to these topics is available in [18].

Conventional power provisioning. Today, most data centers operate according to power provisioning policies that assure sufficient capacity for every server. These policies are enforced by the data center operators at system installation time, by prohibiting deployment of any machine that creates the potential for overload. Operators do their best to estimate systems' peak power draws, either through stress-testing, from vendor-supplied calculators, or through de-rating of nameplate specifications.

In high-availability data centers, power distribution schemes must also provision redundancy for fault tolerance; system deployments are further restricted by these redundancy requirements. The Uptime Institute classifies data centers into tiers based on the nature and objectives of their infrastructure redundancy [27]. Some data centers provide no fault tolerance (Tier-1), or provision redundancy only within major power infrastructure components, such as the UPS system (Tier-2). Such redundancy allows some maintenance of infrastructure components during operation, and protects against

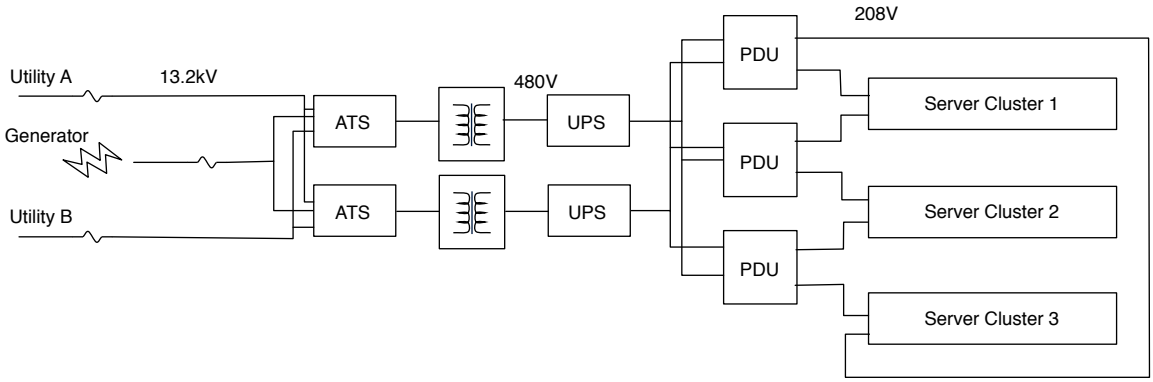


Figure 2: Example power delivery system for a high-availability data center.

certain kinds of faults, but numerous single points-of-failure remain. Higher-tier data centers provide redundant power delivery paths to each server. Power Routing is targeted at these data centers, as it exploits the redundant delivery paths to shift power delivery capacity.

Example: A high-availability power system. Figure 2 illustrates an example of a high-availability power system design and layout for a data center with redundant distribution paths. The design depicted here is based on the power architecture of the Michigan Academic Computer Center (MACC), the largest (10,000 square feet; 288 racks; 4MW peak load including physical infrastructure) of the three facilities providing utilization traces for this study. Utility power from two substations and a backup generator enter the facility at high voltage (13.2 kVAC) and meet at redundant automated transfer switches (ATS) that select among these power feeds. These components are sized for the peak facility load (4MW), including all power infrastructure and cooling system losses. The ATS outputs in turn are transformed to a medium voltage (480 VAC) and feed redundant uninterruptible power supply (UPS) systems, which are also each sized to support the entire facility. These in turn provide redundant feeds to an array of power distribution units (PDUs) which further transform power to 208V 3-phase AC.

PDUs are arranged throughout the data center such that each connects to two neighboring system clusters and each cluster receives redundant power feeds from its two neighboring PDUs. The power assignments wrap from the last cluster to the first. We refer to this PDU arrangement as a *wrapped topology*. The wrapped topology provides redundant delivery paths with minimal wiring and requires each PDU to be sized to support at most 150% of the load of its connected clusters, with only a single excess PDU beyond the minimum required to support the load (called an "N+1" configuration). In the event of any PDU fault, 50% of its supported load fails over to each of its two neighbors. PDUs each support only a fraction of the data center's load, and can range in capacity from under ten to several hundred kilowatts.

Power is provided to individual servers through connectors (called "whips"), that split the three phases of the 208VAC PDU output into the 120VAC single-phase circuits familiar from residential wiring. (Some equipment may operate at higher voltages or according to other international power standards.) Many modern servers include redundant power supplies, and provide two power cords that can be plugged into whips from each PDU. In such systems, the server internally switches or splits its load among its two power feeds. For servers that provide only a single power cord, a rack-level transfer switch can connect the single cord to redundant feeds.

The capital costs of the power delivery infrastructure are concentrated at the large, high-voltage components: PDUs, UPSs, facility-level switches, generators, transformers and the utility feed. The rack-level components cost a few thousand dollars per rack (on the order of $1 per provisioned Watt), while the facility-level components can cost $10-$25 per provisioned Watt [18, 26], especially in facilities with such high levels of redundancy. With Power Routing, we focus on reducing the required provisioning of the facility-scale components while assuring a balanced load over the PDUs. Though circuit breakers typically limit current both at the PDU's breaker panels and on the individual circuits in each whip, it is comparatively inexpensive to provision these statically to avoid overloads. Though Power Routing is applicable to manage current limits on individual circuits, we focus on enforcing limits at the PDU level in this work.

Phase balance. In addition to enforcing current limits and redundancy, it is also desirable for a power provisioning scheme to balance power draw across the three phases of AC power supplied by each PDU. Large phase imbalances can lead to current spikes on the neutral wire of a 3-phase power bus, voltage and current distortions on the individual phases, and generally increase heat dissipation and reduce equipment lifetime [14]. Data center operators typically manually balance power draw across phases by using care in connecting equipment to particular receptacles wired to each phase. Power Routing can automatically enforce phase balance by including it as explicit constraints in its scheduling algorithm.

Power capping. Conservative, worst-case design invariably leads to power infrastructure over-provisioning [8, 13, 23, 29]. Power capping mechanisms allow data center operators to sacrifice some performance in rare utilization spikes in exchange for substantial cost savings in the delivery infrastructure, without the risk of cascading failures due to an overload. In these schemes, some centralized control mechanism establishes a power budget for each server (e.g., based on historical predictions or observed load in the previous time epoch). An actuation mechanism then enforces these budgets.

The most common method of enforcing power budgets is through control loops that sense actual power draw and modulate processor frequency and voltage to remain within budget. Commercial systems from IBM [21] and HP [16] can enforce budgets to sub-watt granularities at milli-second timescales. Researchers have extended these control mechanisms to enforce caps over multi-server chassis, larger ensembles, and entire clusters [9, 17, 23, 28], examine optimal power allocation among heterogeneous servers [10] and identify the control stability challenges when capping at multiple levels of the power distribution hierarchy [22, 29]. Others have examined ex-

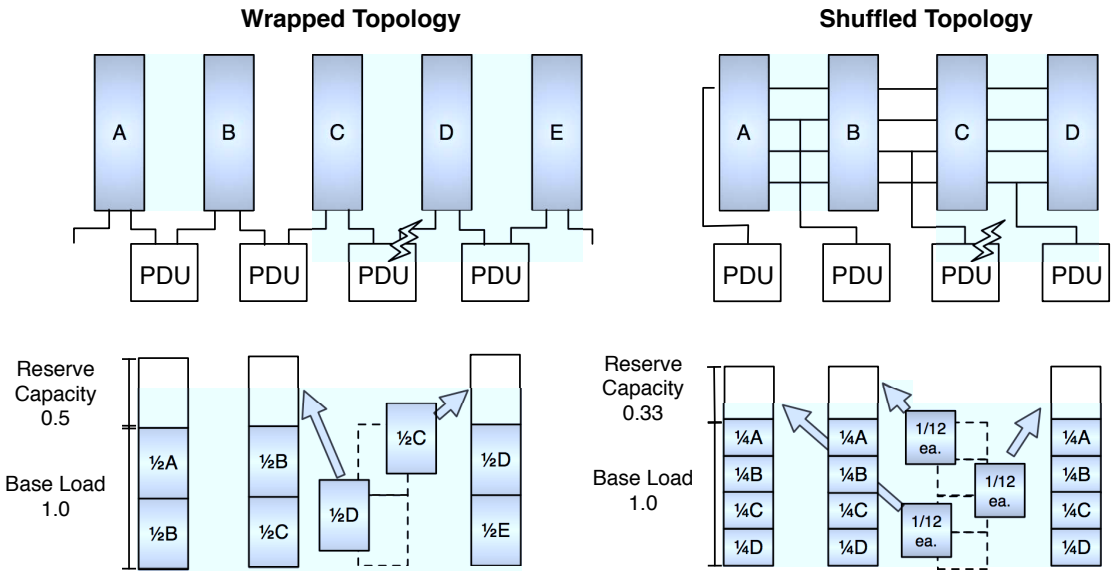


Figure 3: Reduced reserve capacity under shuffled topologies (4 PDUs, fully-connected topology).

tending power management to virtualized environments [20]. Soft fuses [13] apply the notion of power budgets beyond the individual server and enforce sustained power budgets, which allow for transient overloads that the power infrastructure can support. Finally, prior work considers alternative mechanisms for enforcing caps, such as modulating between active and sleep states [11].

Like prior work, Power Routing relies on a power capping mechanism as a safety net to ensure extended overloads can not occur. However, Power Routing is agnostic to how budgets are enforced. For simplicity, we assume capping based on dynamic frequency and voltage scaling, the dominant approach.

Though rare, peak utilization spikes do occur in some facilities. In particular, if a facility runs a single distributed workload balanced over all servers (e.g., as in a web search cluster), then the utilization of all servers will rise and fall together [8]. No scheme that over-subscribes the physical infrastructure can avoid performance throttling for such systems. The business decision of whether throttling is acceptable in these rare circumstances is beyond the scope of this study; however, for any given physical infrastructure budget, Power Routing reduces performance throttling relative to existing capping schemes, by shifting loads among PDUs to locate and exploit spare capacity.

3. Power Routing.

Power Routing relies on two central concepts. First, it exploits *shuffled topologies* for power distribution to increase the connectivity between servers and diverse PDUs. Shuffled topologies spread responsibility to sustain the load on a failing PDU, reducing the required reserve capacity per PDU. Second, Power Routing relies on a *scheduling* algorithm to assign servers' load across redundant distribution paths while balancing loads over PDUs and AC phases. When loads are balanced, the provisioned capacity of major power infrastructure components (PDUs, UPSs, generators, and utility feeds) can be reduced, saving capital costs. We first detail the design and advantages of shuffled topologies, and then discuss Power Routing.

3.1 Shuffled Topologies.

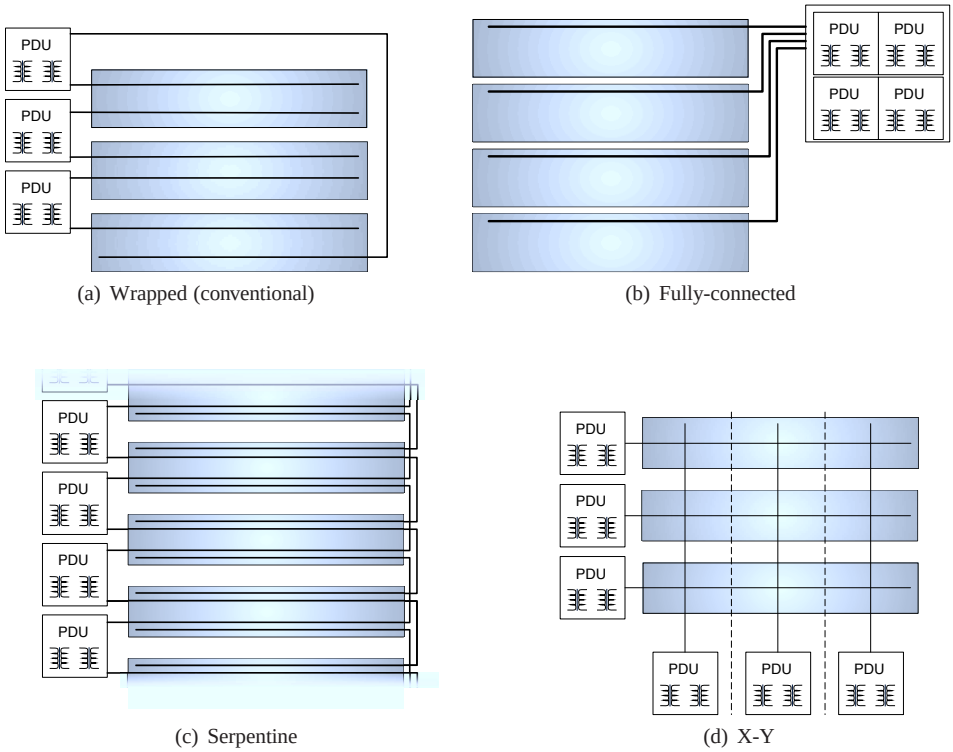
In high-availability data centers, servers are connected to two PDUs to ensure uninterrupted operation in the event of a PDU fault. A naive (but not unusual) connection topology provisions paired PDUs for each cluster of machines. Under this data center design, each PDU must be sized to support the full worst-case load of the entire cluster; hence, the power infrastructure is 50% utilized in the best case. As described in Section 2, the more sophisticated "wrapped" topology shown in Figure 2 splits a failed PDU's load over two neighbors, allowing each PDU to be sized to support only 150% of its nominal primary load.

By spreading the responsibility for failover further, to additional PDUs, the spare capacity required of each PDU can be reduced—the more PDUs that cooperate to cover the load of a failed PDU, the less reserve capacity is required in the data center as a whole. In effect, the reserve capacity in each PDU protects multiple loads (which is acceptable provided there is only a single failure).

Figure 3 illustrates the differing reserve capacity requirements of the wrapped topology and a shuffled topology where responsibility for reserve capacity is spread over three PDUs. The required level of reserve capacity at each PDU is approximately X/N, where X represents the cluster power demand, and N the number of PDUs cooperating to provide reserve capacity. (Actual reserve requirements may vary depending on the instantaneous load on each phase).

The savings from shuffled topologies do not require any intelligent switching capability; rather, they require only increased diversity in the distinct combinations of primary and secondary power feeds for each server (ideally covering all combinations equally).

The layout of PDUs and power busses must be carefully considered to yield feasible shuffled wiring topologies. Our distribution strategies rely on overhead power busses [24] rather than conventional under-floor conduits to each rack. The power busses make it easier (and less costly) to connect many, distant racks to a PDU. Power from each nearby bus is routed to a panel at the top of each rack, and these in turn connect to vertical whips (i.e., outlet strips) that



(a) Wrapped (conventional) (b) Fully-connected

(c) Serpentine (d) X-Y

Figure 4: Shuffled power distribution topologies.

supply power to individual servers. The whips provide outlets in pairs (or a single outlet with an internal transfer switch) to make it easy to connect servers while assuring an appropriate mix of distinct primary and secondary power feed combinations.

Though overhead power busses are expensive, they still account for a small fraction of the cost of large-scale data center power infrastructure. Precise quantification of wiring costs is difficult without detailed facility-specific architecture and engineering. We neglect differences in wiring costs when estimating data center infrastructure costs, and instead examine the (far more significant) impact that topologies have on the capacity requirements of the high-voltage infrastructure. The primary difficulty of complex wiring topologies lies in engineering the facility-specific geometry of the large (and dangerous) high-current overhead power rails; a challenge that we believe is surmountable.

We propose three shuffled power distribution topologies that improve on the wrapped topology of current high-availability data centers. The *fully connected* topology collocates all PDUs in one corner of the room, and routes power from all PDUs throughout the entire facility. This topology is not scalable. However, we study it as it represents an upper bound on the benefits of shuffled topologies. We further propose two practical topologies. The *X-Y* topology divides the data center into a checkerboard pattern of power zones, routing power both north-south and east-west across the zones. The *serpentine* topology extends the concept of the wrapped topology (see Figure 2) to create overlap among neighboring PDUs separated by more than one row.

Each distribution topology constrains the set of power feed combinations available in each rack in a different manner. These constraints in turn affect the set of choices available to the Power Routing scheduler, thereby impacting its effectiveness.

Wrapped Topology. Figure 4(a) illustrates the wrapped topology, which is our term for the conventional high-availability data center topology (also seen in Figure 2). This topology provides limited connectivity to PDUs, and is insufficient for Power Routing.

Fully-connected Topology. Figure 4(b) illustrates the fully-connected topology. Under this topology, power is routed from every PDU to every rack. As noted above, the fully-connected topology does not scale and is impractical in all but the smallest data centers. However, one scalable alternative is to organize the data center as disconnected islands of fully-connected PDUs and rack clusters. Such a topology drastically limits Power Routing flexibility, but can scale to arbitrary-sized facilities.

Serpentine Topology. Figure 4(c) illustrates the serpentine topology. Under this topology, PDUs are located at one end of the data centers' rows, as in the wrapped topology shown in Figure 2. However, whereas in the wrapped topology a power bus runs between two equipment rows from the PDU to the end of the facility, in the serpentine topology, the power bus then bends back, returning along a second row. This snaking bus pattern is repeated for each PDU, such that two power busses run in each aisle and four busses are adjacent to each equipment row. The pattern scales to larger facilities by adding PDUs and replicating the pattern over additional rows. It scales to higher PDU connectivity by extending the serpentine pattern with an additional turn.

X-Y Topology. Figure 4(d) illustrates the X-Y topology. Under this topology, the data center is divided into square zones in a checkerboard pattern. PDUs are located along the north and west walls of the data center. Power busses from each PDU route either north-south or east-west along the centerline of a row (column) of zones. Hence, two power busses cross in each zone. These two busses are connected to each rack in the zone. This topology scales to larger facilities in a straight-forward manner, by adding zones to the "checkerboard." It scales to greater connectivity by routing power busses over the zones in pairs (or larger tuples).

3.2 Power Routing.

Power Routing leverages shuffled topologies to achieve further capital cost savings by under-provisioning PDUs relative to worst-case demand. The degree of under-provisioning is a business decision made at design time (or when deploying additional systems) based on the probability of utilization spikes and the cost of performance throttling (i.e., the risk of failing to meet a service-level agreement). Power Routing shifts spare capacity to cover local power demand spikes by controlling the assignment of each server to its primary or secondary feed. The less correlation there is among spikes, the more effective Power Routing will be at covering those spikes by shifting loads rather than throttling performance. Power Routing relies on a capping mechanism to prevent overloads when spikes cannot be covered.

Power Routing employs a centralized control mechanism to assign each server to its primary or secondary power feed and set power budgets for each server to assure PDU overloads do not occur. Each time a server's power draw increases to its pre-determined cap (implying that performance throttling will be engaged), the server signals the Power Routing controller to request a higher cap. If no slack is available on the server's currently active power feed, the controller invokes a scheduling algorithm (detailed in Section 4) to determine new power budgets and power feed assignments for all servers to try to locate slack elsewhere in the power distribution system. The controller will reduce budgets for servers whose utilization has decreased and may reassign servers between their primary and secondary feeds to create the necessary slack. If no solution can be found (e.g., because aggregate power demand exceeds the facilities' total provisioning), the existing power cap remains in place and the server's performance is throttled.

In addition to trying to satisfy each server's desired power budget, the Power Routing scheduler also maintains sufficient reserve capacity at each PDU to ensure continued operation (under the currently-established power budgets) even if any single PDU fails. A PDU's required reserve capacity is given by the largest aggregate load served by another PDU for which it acts as the secondary (inactive) feed.

Finally, the Power Routing scheduler seeks to balance load across the three AC phases of each PDU. As noted in Section 2, phase imbalance can lead to numerous electrical problems that impact safety and availability. The scheduler constrains the current on each of the three phases to remain within a 20% margin.

The key novelty of Power Routing lies in the assignment of servers to power feeds; sophisticated budgeting mechanisms (e.g., which assign asymmetric budgets to achieve higher-level QoS goals) have been extensively studied [9, 10, 17, 20, 22, 23, 28, 29]. Hence, in this paper, we focus our design and evaluation on the power feed scheduling mechanism and do not explore QoS-aware capping in detail.

3.3 Implementation.

Power Routing comprises four elements: (1) an actuation mechanism to switch servers between their two redundant power feeds; (2) the centralized controller that executes the power feed scheduling algorithm; (3) a communications mechanism for the controller to direct switching activity and assign budgets; and (4) a power distribution topology that provisions primary and secondary power feeds in varying combinations to the receptacles in each rack.

Switching power feeds. The power feed switching mechanism differs for single- and dual-corded servers. In a single-corded server, an external transfer switch attaches the server to its primary or secondary power feed. In the event of a power interruption on the active feed, the transfer switch seamlessly switches the load to the alternative feed (a local, automatic action). The scheduler assures that all PDUs have sufficient reserve capacity to supply all loads that may switch to them in the event of any single PDU failure. To support phase balancing, the transfer switch must be capable of switching loads across out-of-phase AC sources fast enough to appear uninterrupted to computer power supplies. External transfer switches of this sort are in wide-spread use today, and retail for several hundred dollars. In contrast to existing transfer switches, which typically switch entire circuits (several servers), Power Routing requires switching at the granularity of individual receptacles, implying somewhat higher cost. For dual-corded servers, switching does not require any additional hardware, as the switching can be accomplished through the systems' internal power supplies.

Control unit. The Power Routing control unit is a microprocessor that orchestrates the power provisioning process. Each time scheduling is invoked, the control unit performs four steps: (1) it determines the desired power budget for each server; (2) it schedules each server to its primary or secondary power feed; (3) it assigns a power cap to each server (which may be above the request, allowing headroom for utilization increase, or below, implying performance throttling); and (4) it communicates the power cap and power feed assignments to all devices. The control unit can be physically located within the existing intelligence units in the power delivery infrastructure (most devices already contain sophisticated, network-attached intelligence units). Like other power system components, the control unit must include mechanisms for redundancy and fault tolerance. Details of the control unit's hardware/software fault tolerance are beyond the scope of this study; the challenges here mirror those of the existing intelligence units in the power infrastructure.

The mechanisms used in each of the control unit's four steps are orthogonal. As this study is focused on the novel scheduling aspect of Power Routing (step 2), we explore only relatively simplistic policies for the other steps. We determine each server's desired power budget based in its peak demand in the preceding minute. Our power capping mechanism assigns power budgets that throttle servers to minimize the total throttled power.

Communication. Communication between the control unit and individual servers/transfer switches is best accomplished over the data center's existing network infrastructure, for example, using the Simple Network Management Protocol (SNMP) or BACnet. The vast majority of power provisioning infrastructure already supports these interfaces. Instantaneous server power draws and power budgets can also typically be accessed through SNMP communication with the server's Integrated Lights Out (ILO) interface.

Handling uncontrollable equipment. Data centers contain myriad equipment that draw power, but cannot be controlled by Power Routing (e.g., network switches, monitors). The scheduler must account for the worst-case power draw of such equipment when calculating available capacity on each PDU and phase.

3.4 Operating Principle.

Power Routing relies on the observation that individual PDUs are unlikely to reach peak load simultaneously. The power distribution system as a whole operates in one of three regimes. The first, most common case is that the load on all PDUs is below their capacity. In this case, the power infrastructure is over-provisioned, power capping is unnecessary, and the entire data center operates at full performance. At the opposite extreme, when servers demand more power than is available, the power infrastructure is under-provisioned, all PDUs will be fully loaded, and power capping (e.g., via performance throttling) is necessary. In either of these regimes,

Power Routing has no impact; the power infrastructure is simply under- (over-) provisioned relative to the server demand.

Power Routing is effective in the intermediate regime where some PDUs are overloaded while others have spare capacity. In current data centers, this situation will result in performance throttling that Power Routing can avoid.

To illustrate how Power Routing affects performance throttling, we explore its performance envelope near the operating region where aggregate power infrastructure capacity precisely meets demand. Figure 5 shows the relationship between installed PDU capacity and performance throttling (in terms of the fraction of offered load that is met) with and without Power Routing (6 PDUs, fully-connected topology) and contrast these against an ideal, perfectly-balanced power distribution infrastructure. The ideal infrastructure can route power from any PDU to any server and can split load fractionally over multiple PDUs. (We detail the methodology used to evaluate Power Routing and produce these results in Section 5.1 below.)

The graph provides two insights into the impact of Power Routing. First, we can use it to determine how much more performance Power Routing achieves for a given infrastructure investment relative to conventional and ideal designs. This result can be obtained by comparing vertically across the three lines for a selected PDU capacity. As can be seen, Power Routing closely tracks the performance of the ideal power delivery infrastructure, recovering several percent of lost performance relative to a fully-connected topology without power routing.

The graph can also be used to determine the capital infrastructure savings that Power Routing enables while avoiding performance throttling altogether. Performance throttling becomes necessary at the PDU capacity where each of the three power distributions dips below 1.0. The horizontal distance between these intercepts is the capacity savings, and is labeled "Power Routing Capacity Reduction" in the figure. In the case shown here, Power Routing avoids throttling at a capacity of 255 kW, while 294 kW of capacity are needed without Power Routing. Power Routing avoids throttling, allowing maximum performance with less investment in power infrastructure.

4. Scheduling

Power Routing relies on a centralized scheduling algorithm to assign power to servers. Each time a server requests additional power (as a result of exhausting its power cap) the scheduler checks if the server's current active power feed has any remaining capacity, granting it if possible. If no slack exists, the scheduler attempts to create a new allocation schedule for the entire facility that will eliminate or minimize the need for capping. In addition to considering the actual desired power budget of each server, the scheduler must also provision sufficient reserve capacity on each feed such that the feed can sustain its share of load if any PDU fails. Finally, we constrain the scheduler to allow only phase-balanced assignments where the load on the three phases of any PDU differ by no more than 20% of the per-phase capacity.

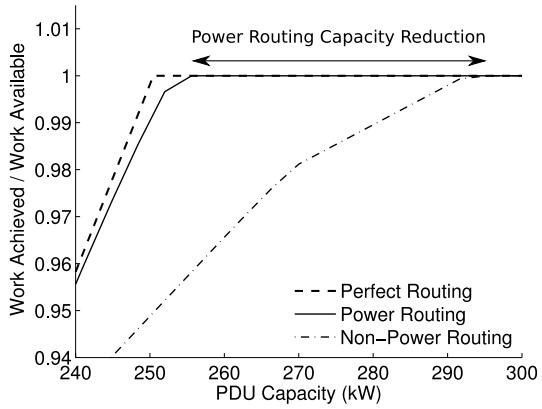


Figure 5: Shuffled Topologies: 6 PDUs, fully-connected

The scheduling process comprises three steps: gathering the desired budget for each server, solving for an assignment of servers to their primary or secondary feeds, and then, if necessary, reducing server's budgets to meet the capacity constraints on each feed.

Whereas sophisticated methods for predicting power budgets are possible [5], we use a simple policy of assigning each server a budget based on its average power demand in the preceding minute. More sophisticated mechanisms are orthogonal to the scheduling problem itself.

Solving the power feed assignment problem optimally, even without redundancy, is an NP-Complete problem. It is easy to see that power scheduling $\in$ NP; a nondeterministic algorithm can enumerate a set of assignments from servers to PDUs and then check in polynomial time that each PDU is within its power bounds. To show that power scheduling is NP-Complete we transform PARTITION to it [12]. For a given instance of PARTITION of finite set A and a size $s(a) \in \mathbb{Z}+$ for each $a \in A$: we would like to determine if there is a subset $A' \in A$ such that the $\sum_{a \in A'} s(a) = \sum_{a \in A-A'} s(a)$. Consider A as the set of servers, with $s(a)$ corresponding to server power draw. Additionally consider two PDUs each of power capacity $\sum_a s(a)/2$. These two problems are equivalent. Thus, a polynomial time solution to power scheduling will yield a polynomial time solution to PARTITION (implying power scheduling is NP-Complete).

In data centers of even modest size, brute force search for an optimal power feed assignment is infeasible. Hence, we resort to a heuristic approach to generate an approximate solution.

We first optimally solve a power feed assignment problem allowing servers to be assigned fractionally across feeds using linear programming. This linear program can be solved in polynomial time using standard methods [7]. From the exact fractional solution, we then construct an approximate solution to the original problem (where entire servers must be assigned a power feed). Finally, we check if the resulting assignments are below the capacity of each power feed. If any feed's capacity is violated, we invoke a second optimization step to choose power caps for all servers.

Determining optimal caps is non-trivial because of the interaction between a server's power allocation on its primary feed, and the reserve capacity that allocation implies on its secondary feed. We employ a second linear programming step to determine a capping strategy that maximizes the amount of power allocated to servers (as opposed to reserve capacity).

Problem formulation. We formulate the linear program based on the power distribution topology (i.e., the static assignment of primary and secondary feeds to each server), the desired server power budgets, and the power feed capacities. For each pair of power feeds we calculate $Power_{i,j}$, the sum of power draws for all servers connected to feeds i and j. (Our algorithm operates at the granularity of individual phases of AC power from each PDU, as each phase has limited ampacity). $Power_{i,j}$ is 0 if no server shares feeds i and j (e.g., if the two feeds are different phases from the same PDU or no server shares those PDUs). Next, for each pair of

feeds, we define variables $Feed_{i,j}i$ and $Feed_{i,j}j$ to account for the server power from $Power_{i,j}$ routed to feeds i and j, respectively. Finally, a single global variable, $Slack$, represents the maximum unallocated power on any phase after all assignments are made. With these definitions, the linear program maximizes $Slack$ subject to the following constraints:

$\forall i, j \neq i$, i and j are any phases on different PDUs:

$$Feed_{i,j}i + Feed_{i,j}j = Power_{i,j} \quad (1)$$

$$\sum_{k \neq i} Feed_{i,k}i + \sum_{lin\,j's\,PDU} Feed_{i,l}l + Slack \leq Capacity(i) \quad (2)$$

And constraints for distinct phases i and j within a single PDU:

$$|\sum_{k \neq i} Feed_{i,k}i - \sum_{k \neq j} Feed_{j,k}j| \leq .2 \times Capacity(i,j) \quad (3)$$

With the following bounds:

$$-\infty \leq Slack \leq \infty \quad (4)$$

$$\forall i, j \neq i : Feed_{i,j}i, Feed_{i,j}j \geq 0 \quad (5)$$

Equation 1 ensures that power from servers connected to feeds i and j is assigned to one of those two feeds. Equation 2 restricts the sum of all power assigned to a particular feed i, plus the reserve capacity required on i should feeds on j's PDU fail, plus the excess slack to be less than the capacity of feed i. Finally, equation 3 ensures that phases are balanced across each PDU. A negative $Slack$ indicates that more power is requested by servers than is available (implying that there is no solution to the original, discrete scheduling problem without power capping).

We use the fractional power assignments from the linear program to schedule servers to feeds. For a given set of servers, s, connected to both feed i and feed j, the fractional solution will indicate that $Feed_{i,j}i$ watts be assigned to i and $Feed_{i,j}j$ to j. The scheduler must create a discrete assignment of servers to feeds to approximate the desired fractional assignments as closely as possible, which is itself a bin packing problem. To solve this sub-problem efficiently, the scheduler sorts the set s descending by power and repeatedly assign the largest unassigned server to i or j, whichever has had less power assigned to it thus far (or whichever has had less power relative to its capacity if the capacities differ).

If a server cannot be assigned to either feed without violating the feed's capacity constraint, then throttling may be necessary to achieve a valid schedule. The server is marked as "pending" and left temporarily unassigned. By the nature of the fractional solution, at most one server in the set can remain pending. This server must eventually be assigned to one of the two feeds; the difference between this discrete assignment and the optimal fractional assignment is the source of error in our heuristic. By assigning the largest servers first we attempt to minimize this error. Pending servers will be assigned to the feed with the most remaining capacity once all other servers have been assigned.

The above optimization algorithm assumes that each pair of power feeds shares several servers in common, and that the power drawn by each server is much less than the capacity of the feed. We believe that plausible power distribution topologies fit this restriction.

Following server assignment, if no feed capacity constraints have been violated, the solution is complete and all servers are assigned caps at their requested budgets. If any slack remains on a feed, it can be granted upon a future request without re-invoking the scheduling mechanism, avoiding unnecessary switching.

If any capacity constraints have been violated, a new linear programming problem is formulated to select power caps that maximize the amount of power allocated to servers (as opposed to reserve capacity for fail-over). We scale back each feed such that no PDU supplies more power than its capacity, even in the event that another PDU fails. The objective function maximizes the sum of the server budgets. We assume that servers can be throttled to any frequency from idle to peak utilization and that the relationship and limits of frequency and power scaling are known a priori. Note, however, that this formulation ignores heterogeneity in power efficiency, performance, or priority across servers; it considers only the redundancy and topology constraints of the power distribution network. An analysis of more sophisticated mechanisms for choosing how to cap servers that factors in these considerations is outside the scope of this paper.

5. Evaluation

Our evaluation demonstrates the effectiveness of shuffled topologies and Power Routing at reducing the required capital investment in power infrastructure to meet a high-availability data center's reliability and power needs. First, we demonstrate how shuffled topologies reduce the reserve capacity required to provide single-PDU-fault tolerance. Then, we examine the effectiveness of Power Routing at further reducing provisioning requirements as a function of topology, number of PDUs, and workload. Finally, we show how Power Routing will increase in effectiveness as server power management becomes more sophisticated and the gap between servers' idle and peak power demands grows.

5.1 Methodology

We evaluate Power Routing through analysis of utilization traces from a large collection of production systems. We simulate Power Routing's scheduling algorithm and impact on performance throttling and capital cost.

Traces. We collect utilization traces from three production facilities: (1) *EECS servers*, a small cluster of departmental servers (web, email, login, etc.) operated by the Michigan EECS IT staff; (2) *Arbor Lakes Data Center*, a 1.5MW facility supporting the clinical operations of the University of Michigan Medical Center; and (3) *Michigan Academic Computer Center (MACC)*, a 4MW high-performance computing facility operated jointly by the University of Michigan, Internet2, and Merit that runs primarily batch processing jobs. These sources provide a diverse mix of real-world utilization behavior. Each of the traces ranges in length from three to forty days sampling server utilization once per minute. We use these traces to construct a hypothetical high-availability hosting facility comprising 400 medical center servers, 300 high performance computing nodes, and a 300-node web search cluster. The simulated medical center and HPC cluster nodes each replay a trace from a specific machine in the corresponding real-world facility. The medical center systems tend to be lightly loaded, with one daily utilization spike (which we believe to be daily backup processing). The HPC systems are heavily loaded. As we do not have access to an actual 300-node web search cluster, we construct a cluster by replicating the utilization trace of a single production web server over 300 machines. The key property of this synthetic search cluster is that the utilization on individual machines rises and falls together in response to user traffic, mimicking the behavior reported for actual search clusters [8]. We analyze traces for a 24-hour period. Our synthetic cluster sees a time-average power draw of 180.5 kW, with a maximum of 208.7 kW and standard deviation of 9 kW.

Power. We convert utilization traces to power budget requests using published SPECPower results [1]. Most of our traces have been collected from systems where no SPECPower result has been published; for these, we attempt to find the closest match based on vendor descriptions and the number and model of CPUs and installed memory. As SPECPower only provides power at intervals of 10%

utilization, we use linear interpolation to approximate power draw in between these points.

Prior work [8, 23] has established that minute-grained CPU utilization traces can predict server-grain power draw to within a few percent. Because of the scope of our data collection efforts, finer-grained data collection is impractical. Our estimates of savings from Power Routing are conservative; finer-grained scheduling might allow tighter tracking of instantaneous demand.

To test our simulation approach, we have validated simulation-derived power values against measurements of individual servers in our lab. Unfortunately, the utilization and power traces available from our production facilities are not exhaustive, which precludes a validation experiment where we compare simulation-derived results to measurements for an entire data center.

Generating data center topologies. For each power distribution topology described in Section 3.1, we design a layout of our hypothetical facility to mimic the typical practices seen in the actual facilities. We design layouts according to the policies the Michigan Medical Center IT staff use to manage their *Arbor Lakes* facility. Each layout determines an assignment of physical connections from PDUs to servers. Servers that execute similar applications are collocated in the same rack, and, hence, in conventional power delivery topologies, are connected to the same PDU. Where available, we use information about the actual placement of servers in racks to guide our placement. Within a rack, servers are assigned across PDU phases in a round-robin fashion. We attempt to balance racks across PDUs and servers within racks across AC phases based on the corresponding system's power draw at 100% utilization. No server is connected to two phases of the same PDU, as this arrangement does not protect against PDU failure. We use six PDUs in all topologies unless otherwise noted.

Metrics. We evaluate Power Routing based on its impact on server throttling activity and data center capital costs. As the effect of voltage and frequency scaling on performance varies by application, we instead use the fraction of requested server power budget that was not satisfied as a measure of the performance of capping techniques. Under this metric, the "cost" of failing to supply a watt of requested power is uniform over all servers, obviating the need to evaluate complex performance-aware throttling mechanisms (which are orthogonal to Power Routing). Our primary evaluation metric is the minimum total power delivery capacity required to assure zero performance throttling, as this best illustrates the advantage of Power Routing over conventional worst-case provisioning.

5.2 Impact of Shuffled Topologies

We first compare the impact of shuffled topologies on required power infrastructure capacity. Shuffled topologies reduce the reserve capacity that each PDU must sustain to provide fault tolerance against single-PDU failure. We examine the advantage of several topologies relative to the baseline high-availability "wrapped" data center topology, which requires each PDU to be over-provisioned by 50% of its nominal load. We report the total power capacity required to prevent throttling for our traces. We assume that each PDU must maintain sufficient reserve capacity at all times to precisely support the time-varying load that might fail over to it.

Differences in the connectivity of the various topologies result in differing reserve capacity requirements. For an ideal power distribution infrastructure (one in which load is perfectly balanced across all PDUs), each PDU must reserve $\frac{1}{c+1}$ to support its share of a failing PDU's load, where c is the *fail-over connectivity* of the PDU. Fail-over connectivity counts the number of distinct neighbors to which a PDU's servers will switch in the event of failure. It is two for the wrapped topology, four for serpentine, and varies as a function of the number of PDUs for X-Y and fully-connected topologies. As the connectivity increases, reserve requirements decrease, but with diminishing returns.

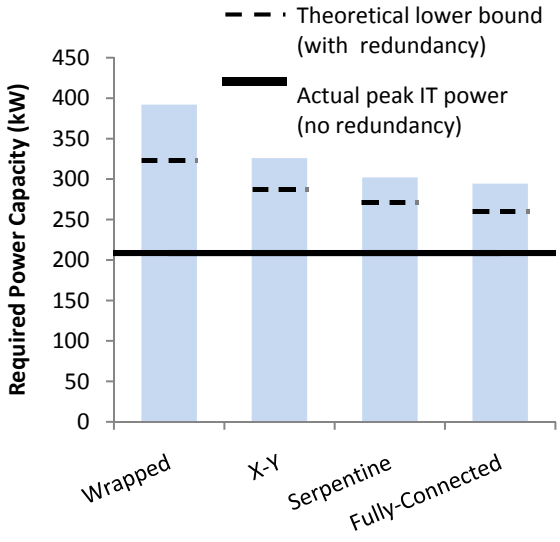


Figure 6: Minimum capacity for redundant operation under shuffled topologies (no Power Routing).

To quantify the impact of shuffled topologies, we design an experiment where we statically assign each server the best possible primary and secondary power feed under the constraints of the topology. We balance the average power draw on each PDU using each server's average power requirement over the course of the trace. (We assume this average to be known a priori for each server.)

In Figure 6 each bar indicates the required power capacity for each topology to meet its load and reserve requirements in all time epochs (i.e., no performance throttling or loss of redundancy) for a 6 PDU data center. For 6 PDUs, the fail-over connectivities are 2, 3, 4, and 5 for the wrapped, X-Y, serpentine, and fully-connected topologies, respectively. The dashed line on each bar indicates the topology's theoretical lower-bound capacity requirement to maintain redundancy if server power draw could be split dynamically and fractionally across primary and secondary PDUs (which Power Routing approximates). The gap between the top of each bar and the dashed line arises because of the time-varying load on each server, which creates imbalance across PDUs and forces over-provisioning. The solid line crossing all bars indicates the data center's peak power draw, ignoring redundancy requirements (i.e., the actual peak power supplied to IT equipment).

Topologies with higher connectivity require less reserve capacity, though the savings taper off rapidly. The X-Y and serpentine topologies yield impressive savings and are viable and scalable from an implementation perspective. Nevertheless, there is a significant gap between the theoretical (dashed) and practical (bar) effectiveness of shuffled topologies. As we show next, Power Routing closes this gap.

5.3 Impact of Power Routing

Power Routing effectiveness. To fully explore Power Routing effectiveness, we repeated the analysis above for all four topologies (wrapped, X-Y, serpentine, and fully-connected) and contrast the capacity required to avoid throttling for each. For comparison, we

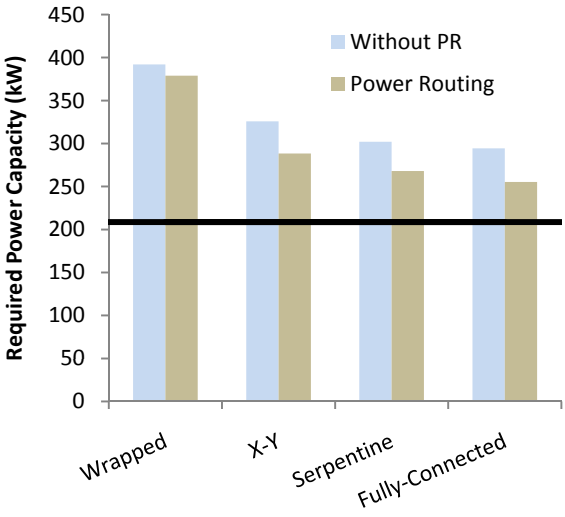


Figure 7: Power Routing infrastructure savings as a function of topology.

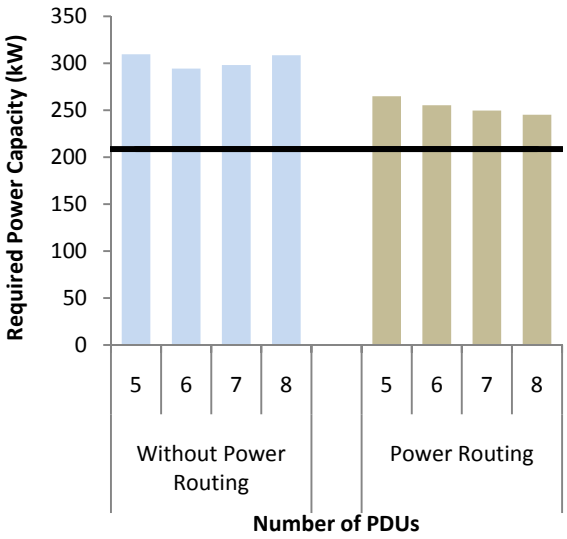


Figure 8: Sensitivity of the fully-connected topology to number of PDUs.

also reproduce the capacity requirements without Power Routing (from Figure 6). We show results in Figure 7. Again, a dashed line represents the theoretical minimum capacity necessary to maintain single-PDU fault redundancy for our workload and the given topology; the solid line marks the actual peak IT power draw. Because the overall load variation in our facilities is relatively small (HPC workloads remain pegged at near-peak utilization; the medical facility is over-provisioned to avoid overloading), we expect a limited opportunity for Power Routing. Nonetheless, we reduce required power delivery capacity for all topologies (except wrapped) by an average of 12%.

From the figure, we see that the sparsely-connected wrapped topology is too constrained for Power Routing to be effective; Power Routing requires 20% more than the theoretical lower bound infrastructure under this topology. The three shuffled topologies, however, nearly reach their theoretical potential, even with a heuristic scheduling algorithm. Under the fully-connected topology, Power Routing comes within 2% of the bound, reducing power infrastructure requirements by over 39kW (13%) relative to the same topology without Power Routing and more than 35% relative to the baseline wrapped topology without Power Routing. Our result indicates that more-connected topologies offer an advantage to Power Routing by providing more freedom to route power. However, the the more-practical topologies yield similar infrastructure savings; the serpentine topology achieves 32% savings relative to the baseline.

Sensitivity to number of PDUs. The number of PDUs affects Power Routing effectiveness, particularly for the fully-connected topology. Figure 8 shows this sensitivity for four to eight PDUs. For a fixed total power demand, as the number of PDUs increases, each individual PDU powers fewer servers and requires less capacity. With fewer servers, the variance in power demands seen by each PDU grows (i.e., statistical averaging over the servers is lessened), and it becomes more likely that an individual PDU will overload. Without Power Routing, this effect dominates, and we see an increase in required infrastructure capacity as the number of PDUs increases beyond 6. At the same time, increasing the number of PDUs offers greater connectivity for certain topologies, which in turn lowers the required slack that PDUs must reserve and offers Power Routing more choices as to where to route power. Hence, Power Routing is better able to track the theoretical bound and the required power capacity decreases with more PDUs.

5.4 Power Routing For Low Variance Workloads

The mixed data center trace we study is representative of the diversity typical in most data centers. Nevertheless, some data centers run only a single workload on a homogeneous cluster. Power Routing exploits diversity in utilization patterns to shift power delivery slack; hence, its effectiveness is lower in homogeneous clusters.

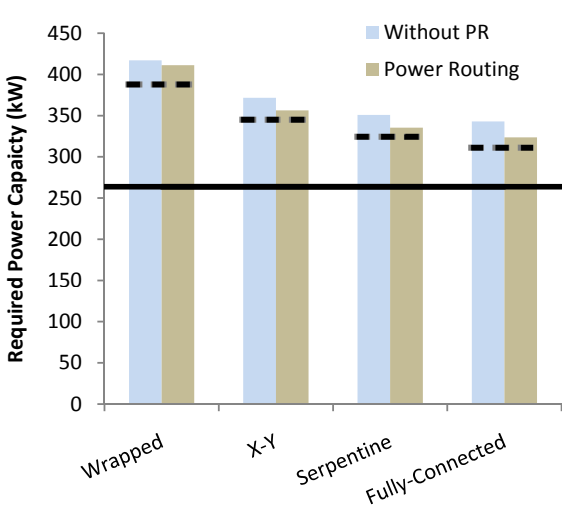
To explore these effects, we construct Power Routing test cases for 1000-server synthetic clusters where each server runs the same application. We do not study the web search application in isolation; in this application, the utilization on all servers rise and fall together, hence, the load on all PDUs is inherently balanced and there is no opportunity (nor need) for Power Routing. Instead, we evaluate Power Routing using the medical center traces and high performance computing traces, shown in Figures 9(a) and 9(b), respectively.

The high performance computing cluster consumes a time-average power of 114.9 kW, a maximum of 116.4 kW, and a standard deviation of 0.8 kW while the medical center computing traces consume a time-average power of 254.6 kW, with maximum 263.6 kW and standard deviation 2.4 kW. In both cases, the variability is substantially lower than in the heterogeneous data center test case.

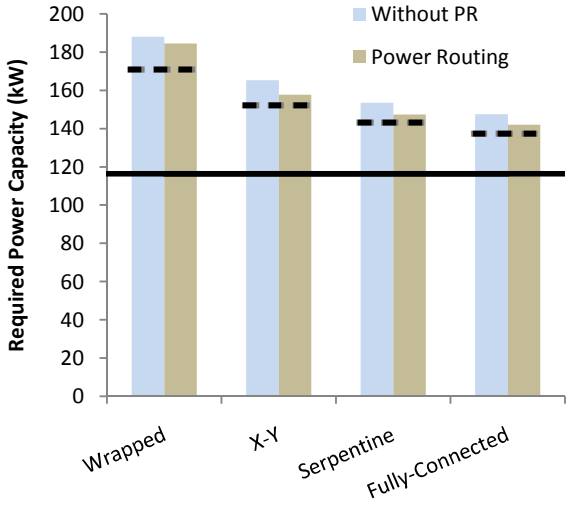
Although Power Routing comes close to achieving the theoretical lower bound infrastructure requirement in each case, we see that there is only limited room to improve upon the non-Power Routing case. Even the baseline wrapped topology requires infrastructure that exceeds the theoretical bound by only 7.5% for the high performance computing cluster and 5% for the medical data center. We conclude that Power Routing offers substantial improvement only in heterogeneous clusters and applications that see power imbalance, a common case in many facilities.

5.5 Power Routing With Energy-Proportional Servers

As the gap between servers' peak and idle power demands grows (e.g., with the advent of energy-proportional computers [3]), we expect the potential for Power Routing to grow. The increase in power variance leads to a greater imbalance in power across PDUs,



(a) Arbor Lakes (clinical operations)



(b) MACC (high-performance computing)

Figure 9: Power Routing effectiveness in homogeneous data centers.

increasing the importance of correcting this imbalance with Power Routing.

To evaluate this future opportunity, we perform an experiment where we assume all servers are energy-proportional—that is, servers whose power draw varies linearly with utilization—with an idle power of just 10% of peak. This experiment models servers equipped with PowerNap [19], which allows servers to sleep during the millisecond-scale idle periods between task arrivals. We repeat the experiment shown in Figure 7 under this revised server power model. The results are shown in Figure 10. Under these assumptions, our traces exhibit a time-average power of 99.8 kW, maximum of 153.9 kW, and standard deviation of 18.9 kW.

Power Routing is substantially more effective when applied to energy-proportional servers. However, the limitations of the wrapped topology are even more pronounced in this case, and Power Routing provides little improvement. Under the more-connected topologies, Power Routing is highly effective, yielding reductions of 22%, 29%, and 28% for the X-Y, serpentine, and fully-connected topologies, respectively, relative to their counterparts without Power Routing. As before, the more-connected topologies track their theoretical lower bounds more tightly. Relative to the baseline wrapped topology, a serpentine topology with Power Routing yields a 47% reduction in required physical infrastructure capacity. It is likely that as computers become more energy-proportional, power infrastructure utilization will continue to decline due to power imbalances. Power Routing reclaims much of this wasted capacity.

5.6 Limitations

Our evaluation considers workloads in which any server may be throttled, and our mechanisms make no effort to select servers for throttling based on any factors except maximizing the utilization of the power delivery infrastructure. In some data centers, it may be unacceptable to throttle performance. These data centers cannot gain a capital cost savings from under-provisioning; their power infrastructure must be provisioned for worst case load. Nonetheless, these facilities can benefit from intermixed topologies (to reduce reserve capacity for fault tolerance) and from the phase-balancing possible with Power Routing.

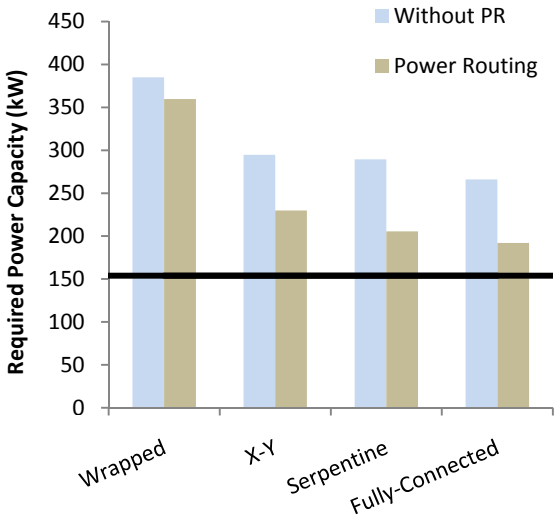


Figure 10: Impact with energy-proportional servers.

6. Conclusion

The capital cost of power delivery infrastructure is one of the largest components of data center cost, rivaling energy costs over the life of the facility. In many data centers, expansion is limited because available power capacity is exhausted. To extract the most value out of their infrastructure, data center operators oversubscribe the power delivery system. As long as individual servers connected to the same PDU do not reach peak utilization simultaneously, over-subscribing is effective in improving power infrastructure utilization. However, coordinated utilization spikes do occur, particularly among collocated machines, which can lead to substantial throttling even when the data center as a whole has spare capacity.

In this paper, we introduced a pair of complementary mechanisms, shuffled power distribution topologies and Power Routing, that reduce performance throttling and allow cheaper capital infrastructure to achieve the same performance levels as current data center designs. Shuffled topologies permute power feeds to create strongly-connected topologies that reduce reserve capacity re-

quirements by spreading responsibility for fault tolerance. Power Routing schedules loads across redundant power delivery paths to shift power delivery slack to satisfy localized utilization spikes. Together, these mechanisms reduce capital costs by 32% relative to a baseline high-availability design when provisioning for zero performance throttling. Furthermore, with energy-proportional servers, the power capacity reduction increases to 47%.

Acknowledgements

The authors would like to thank Joseph Kryza and the University of Michigan Medical Center IT staff for facilitating access to the Arbor Lakes data center, Andrew Caird and the staff at the Michigan Academic Computer Center for assistance in collecting the high performance computing cluster data, Laura Fink for assistance in collecting the departmental server utilization traces, and Vikram Adve and the anonymous reviewers for their feedback. This work was supported by an equipment grant from Intel, financial support from the Michigan Medical Center IT department, and NSF grant CCF-0811320.

References

[1] SPECpower Benchmark Results. [Online]. Available: http://www.spec.org/power_ssj2008/results

[2] G. Alvarez, W. Burkhard, L. Stockmeyer, and F. Cristian, "Declustered disk array architectures with optimal and near-optimal parallelism," in *Proceedings of the 33rd Annual International Symposium on Computer Architecture (ISCA)*, 1998.

[3] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, 2007.

[4] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle, "Managing energy and server resources in hosting centers," *SIGOPS Oper. Syst. Rev.*, vol. 35, no. 5, 2001.

[5] J. Choi, S. Govindan, B. Urgaonkar, and A. Sivasubramanium, "Profiling, prediction, and capping of power consumption in consolidated environments," in *MASCOTS*, September 2008.

[6] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation (NSDI)*, 2005.

[7] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press, 2001.

[8] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA)*, 2007.

[9] M. E. Femal and V. W. Freeh, "Boosting data center performance through non-uniform power allocation," in *Proceedings of Second International Conference on Autonomic Computing (ICAC)*, 2005.

[10] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, "Optimal power allocation in server farms," in *Proceedings of ACM SIGMETRICS 2009 Conference on Measurement and Modeling of Computer Systems*, 2009.

[11] A. Gandhi, M. Harchol-Balter, R. Das, C. Lefurgy, and J. Kephart, "Power capping via forced idleness," in *Workshop on Energy-Efficient Design*, 2009.

[12] M. Garey, D. Johnson, R. Backhouse, G. von Bochmann, D. Harel, C. van Rijsbergen, J. Hopcroft, J. Ullman, A. Marshall, I. Olkin *et al.*, *A Guide to the Theory of Computers and Intractability*. Springer.

[13] S. Govindan, J. Choi, B. Urgaonkar, A. Sivasubramaniam, and A. Baldini, "Statistical profiling-based techniques for effective power provisioning in data centers," in *Proceedings of the 4th ACM European Conference on Computer systems (EuroSys)*, 2009.

[14] T. Gruzs, "A survey of neutral currents in three-phase computer power systems," *IEEE Transactions on Industry Applications*, vol. 26, no. 4, Jul/Aug 1990.

[15] J. Hamilton, "Internet-scale service infrastructure efficiency," Keynote at the International Symposium on Computer Architecture (ISCA), 2009.

[16] HP Staff, "HP power capping and dynamic power capping for ProLiant servers," HP, Tech. Rep. TC090303TB, 2009.

[17] C. Lefurgy, X. Wang, and M. Ware, "Power capping: A prelude to power shifting," *Cluster Computing*, vol. 11, no. 2, 2008.

[18] Luiz André Barroso and Urs Hölzle, *The Datacenter as a Computer*. Morgan Claypool, 2009.

[19] D. Meisner, B. T. Gold, and T. F. Wenisch, "Powernap: eliminating server idle power," in *Proceeding of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, March 2009.

[20] R. Nathuji and K. Schwan, "Virtualpower: coordinated power management in virtualized enterprise systems," in *Proceedings of twenty-first ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*, 2007.

[21] P. Popa, "Managing server energy consumption using IBM PowerExecutive," IBM, Tech. Rep., 2006.

[22] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, "No "power" struggles: coordinated multi-level power management for the data center," in *Proceeding of the 13th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2008.

[23] P. Ranganathan, P. Leech, D. Irwin, and J. Chase, "Ensemble-level power management for dense blade servers," in *Proceedings of the 33rd Annual International Symposium on Computer Architecture (ISCA)*, 2006.

[24] N. Rasmussen, "A scalable, reconfigurable, and efficient data center power distribution architecture," APC by Schneider Electric, Tech. Rep. #129, 2009.

[25] A. Seznec, "A case for two-way skewed-associative caches," in *Proceedings of the 20th Annual International Symposium on Computer Architecture (ISCA)*, 1993.

[26] W. Turner and J. Seader, "Dollars per kW plus dollars per square foot are a better datacenter cost model than dollars per square foot alone," Uptime Institute, Tech. Rep., 2006.

[27] W. Turner, J. Seader, and K. Brill, "Industry standard tier classifications define site infrastructure performance," Uptime Institute, Tech. Rep., 2005.

[28] X. Wang and M. Chen, "Cluster-level feedback power control for performance optimization," in *Proceedings of the 14th IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 2008.

[29] X. Wang, M. Chen, C. Lefurgy, and T. W. Keller, "SHIP: Scalable hierarchical power control for large-scale data centers," in *Proceedings of the 18th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, 2009.