Balance of Power: Energy Management for Server Clusters

Jeffrey S. Chase

Department of Computer Science

Duke University

chase@cs.duke.edu

Ronald P. Doyle*

Application Integration and Middleware

IBM Research Triangle Park

rdoyle@us.ibm.com

Abstract

One unintended effect of the growth of the Internet is increased energy usage by servers and the infrastructure supporting them. This paper promotes a research agenda to improve the energy efficiency of Internet server sites. This goal is timely as the costs of energy usage and our vulnerability to energy supply disruptions become increasingly evident.

In particular, we propose energy-conscious server switching as one approach to improving efficiency of server clusters at low request loads. Our proposal builds on the commercial acceptance of load-balancing server switches and recent industry initiatives to reduce power drain of computer systems running below their peak loads. The potential savings from this technique are compelling given that Internet service loads fluctuate by up to an order of magnitude at all time scales.

1 Introduction

Internet data centers are often cited as a major source of increases in energy consumption, particularly in the US. Scientists at Lawrence Berkeley National Laboratory recently estimated that Web sites and other servers consumed over 7 terawatt-hours (TWh) of electricity in the US in 1999, more than the Internet switching infrastructure itself. Representatives of the oil, coal, and nuclear industries cite much higher estimates: George W. Bush recently stated that "the Internet accounts for 8% of US electricity consumption". Although the actual figure is not yet known, it is clear that Internet services are

significant energy consumers in the US and globally.

Moreover, the Internet service infrastructure continues to grow rapidly, adding to strains on the global power grid. For example, Internet industry analysts project that at least 50 million square feet of data center capacity will come on line for third-party hosting services in the US by 2005. These facilities have typical power densities of 100 watts per square foot for servers, storage, switches, and cooling. Thus these new centers alone could add 5 GW of power demand, about 10% of the current generating capacity for California. New data centers in the Seattle area are forecast to increase the city's power demands by 25% [3].

In this paper we promote a research agenda directed at improving the energy-efficiency of Internet server clusters. There are several reasons why the time is right to take on this challenge. First, energy represents a significant element of the Total Cost of Ownership for servers. The new data center capacity projected for 2005 would require approximately 40 TWh (\$4B at \$100 per MWh) per year to run 24x7 unless they become more efficient. Second, data centers require a significant capital investment in equipment for cooling and backup power generation, and this cost scales with power demand. Third, excessive power demand leaves Internet services unnecessarily vulnerable to energy supply disruptions. For example, wholesale electricity recently spiked to \$800 per MWh on the California spot market.

Finally, electricity production harms the environment, and power pricing in most countries does not yet factor in environmental costs. For example, emissions of CO_2 from US coal-burning power plants alone are equivalent to the total CO_2 out-

^{*}R. Doyle is also a student in the Computer Science department at Duke University.

put of Western Europe [1]. The CO_2 concentration of the atmosphere has increased by 30% since the industrial revolution, an alarming figure given the current scientific consensus that higher levels of CO_2 and other gases are responsible for global temperature increases over the last century, and present a significant risk of further global warming over the next century [2]. Under the 1997 Kyoto accords, industrialized nations must reduce CO_2 emissions by 5% below 1990 levels over the next decade, and yet generating electricity for the new US data centers would release about 25M tons of additional CO_2 each year by 2005. Thus reducing energy demand for Internet servers is not only good business sense, it is a matter of professional and social responsibility.

Our proposal adds a new dimension to power-aware resource management [7], which views power as a first-class resource to be managed by the operating system. That work and other OS research on power management focus primarily on power-constrained systems for mobile computing. This paper applies related energy management goals to servers and clusters, in order to reduce the energy consumed for a given request load. We emphasize energy management in the *network* operating system, to complement and leverage recent industry initiatives on advanced power management for server hardware and server operating systems. Section 2 sets the context for our proposal and outlines principles for energy management in servers and clusters. Section 3 proposes energy-conscious server switching for server clusters, in which network switching elements direct incoming requests in a way that reduces aggregate energy consumption when the cluster is running below capacity. Section 4 outlines future research issues for energy management in server clusters. Section 5 concludes.

2 Server Energy Consumption

Our goal is to reduce the energy consumption needed to deliver a service at a given level of request throughput. In other words, we minimize an energy-per-unit-of-service metric, which we call *joules per operation* (JOPs). A joule is a basic unit of energy, which is power (e.g., watts) consumed over some period of time. Broadly speaking, there are four ways to improve JOPs. These are easily understood by

considering a familiar analogy: reducing the energy consumed to warm the occupants of a building.

- 1. Turn down the thermostat. Previous work has explored quality-of-service tradeoffs for energy [4], e.g., reducing the cost of displaying an image by scaling down its resolution. This approach is useful for battery-powered systems where energy is severely constrained, but it does not meet our goal of improving energy efficiency. Demand for Internet services and the resource-intensity of those services continues to expand because these services offer value that improves lives and helps people to be more effective. We focus on providing these services more efficiently rather than lowering their quality or using them less.
- 2. Generate more heat with less energy. We can reduce the energy cost to heat a building by installing a high-efficiency furnace. Similarly, hardware advances continue to improve the "heat" computational power delivered per unit of energy, although at a rate slower than Moore's Law. This suggests a cycles per joule (CPJs) metric for hardware efficiency, here using "cycle" loosely and generically to cover storage and transmission resources as well as processing. Our proposal is complementary to ongoing improvements in CPJ.
- 3. Improve the software. Adding insulation improves heating efficiency by wasting less of the generated heat. Similarly, streamlining service software reduces the "heat" needed to deliver the service, thereby reducing its energy consumption. This suggests a cycles per operation (COPs) metric, emphasizing that software energy-efficiency can be viewed as a basic performance measure. While software efficiency is an important factor, it does not offer interesting new research problems.
- 4. Turn down the heat in empty rooms. This paper focuses on a fourth approach: reduce the energy cost of maintaining surplus capacity. Like buildings, servers and clusters are sized for peak load and are "empty" much of the time. Many advances in energy-conscious systems are directed at reducing the capacity cost of idle components by transitioning them to a low-power state, e.g., disk spindown [5] and power-aware page allocation [6]. New technologies such as RDRAM and Intel's QuickStart and SpeedStep processors help by offering a

 $^{^1\,\}rm The~25M$ ton figure assumes (1) a 40TWh annual energy load, (2) 60% of electricity is generated from coal as in the US today, and (3) coal burning produces 1.8 lbs of CO_2 per KWh produced.

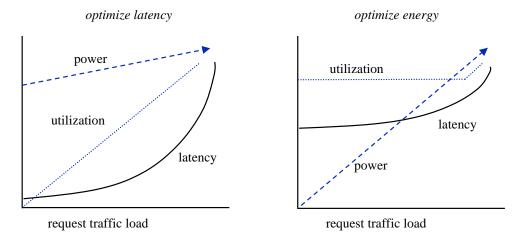


Figure 1: Trading off latency for energy in a server cluster.

wider range of low-power states and reducing the cost of the transitions.

To formalize this as a simple model, power demand for a service is given by the formula:

$$(OPS * COPs)/CPJ + \sum_{i} p_s(i)$$

where component i consumes power $p_s(i)$ in power state s. We can save energy by using the service less (reducing OPS), making the hardware more efficient (increasing CPJ), making the service software more efficient (reducing COPs), or by consuming less power in idle components, reducing capacity cost. We consider the base capacity cost $p_s(i)$ for an active component as fixed independent of load, with CPJ giving a "surcharge" to use that capacity. For example, DRAM in the active state consumes a fixed amount of power for continuous refresh even if it is not active, and disks consume most of their energy just to rotate the spindle. This principle applies for complete servers as well: for example, the IBM xSeries 330 consumes a peak 210 watts of power at full utilization and 150 watts at 70% load, but it consumes 90-100 watts even while it is idle. One effect of this property is that these servers deliver better JOPS ratings in their "sweet spot" at higher utilizations.

A fundamental principle of energy management is that energy consumption should be proportional to load. This is the ultimate goal of any approach that manipulates the power states of idle components to reduce capacity cost. In the Internet server context, request traffic may vary by factors of 3-6 or more through any day or week. When the cluster runs at peak throughput there is no energy cost for idle capacity, and consumption is determined purely by

the hardware (CPJs) and service software (COPs). However, when the cluster serves a lower request load, it is more efficient to concentrate request traffic at a subset of the servers while the remaining servers enter a low-power state, as described in Section 3. This saves the energy expended to maintain surplus capacity on the near-idle servers; in the heating analogy this corresponds to herding the occupants into a smaller number of rooms to turn off the heat in the rest of the building. When the servers are needed again, they may be awakened from the network using the recent industry Wake-On-LAN standard, in which the server's network interface listens for a wake packet while in its low-power state.

This approach to energy-conscious server management trades off latency at lower request loads to save energy. The left-hand side of Figure 1 depicts the scenario common today: servers are managed to minimize latency. In this scenario all components remain in an active state even at low load, so power never drops below the minimum to maintain full capacity. At low load, this capacity is underutilitized; basic queuing theory dictates that this yields lower latency. The right-hand side of Figure 1 depicts energy-conscious server management. We propose that a cluster maintain the minimum number of active servers needed to serve the offered request load at some fixed utilization. At low load, most of the servers idle in a low-power state; this reduces energy usage, but latency is higher because the remaining active servers must handle all of the requests. To return to the heating analogy: all occupants stay warm at lower energy cost, but the rooms are more crowded.

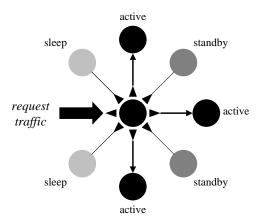


Figure 2: Energy-conscious request switching.

Similar tradeoffs between energy and latency are common in energy-conscious system design, leading some researchers to consider the product $energy \bullet delay$ as the figure of merit. [7]

3 Energy-Conscious Server Switching

We now propose a simple scheme to allow power to scale with load in Internet server clusters. We consider the simple case of a symmetric cluster in which all servers are equally powerful and any server may handle any request. Figure 2 illustrates the architecture. At the center of the cluster is a server switch similar to load-balancing switch products on the market from major Ethernet switch vendors and from other companies, including IBM through its WebSphere product. Large Web sites uses these switches to distribute request traffic across servers. We propose to extend the switch with an energyconscious routing policy that leverages the power management features of the back-end servers. This section outlines the simplest such policy to illustrate the basic principles; the next section explores issues for generalizing this policy.

Servers typically run an operating system module that monitors local load conditions and reports load measures to the load-balancing switch. The switch maintains a list of active, ready servers called the active set, together with load status for these servers. To simplify the discussion, we assume that load is a combined measure reflecting utilization of the CPU, memory, and disks. The switch uses the load status to direct incoming requests to the least loaded servers in the active set, so as to balance incoming request traffic evenly.

We propose that the switch may also add and re-

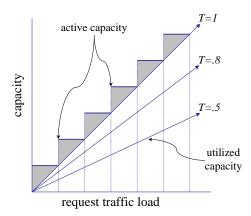


Figure 3: Ramping up capacity with request traffic.

move nodes to or from the active set to hold the average load (utilization) of the active set close to a configured threshold T. If the switch detects that the average utilization has fallen below TN/(N-1), where N is the current size of the active set, then it selects a victim server V to deactivate and shifts V's request traffic to the remaining members of the active set, dropping V to a low-power state. As described in Section 2, this reduces energy usage at lower request traffic levels, but it also increases request latency because the surviving active servers run at higher utilization (bounded above by T) than they would if all servers were active. If request traffic increases again so that active set utilization approaches T, then the switch selects an inactive node to (re)activate and shifts some of the request traffic to it.

In this way, the cluster automatically adjusts its active capacity — and its energy consumption — to adapt to variations in request traffic. This is similar to dynamic resource recruitment for wide-area replication [8]. Figure 3 illustrates this behavior. The y-axis gives the active or utilized capacity as a function of request traffic, which increases along the x axis. The switch increases capacity as a step function by adding servers to the active set as the incoming request rate increases through a sequence of threshold levels, represented by the vertical lines. The diagonal lines qualitatively depict the capacity consumed at each request level for different Tvalues: T gives the proportion of available capacity that is utilized, which the switch holds constant across all threshold points. Note that it may be misleading to directly compare the slopes of the lines in this qualitative graph, which does not fix the units on the x or y axis. In truth they are the same line, but changing T expands or contracts the x-axis

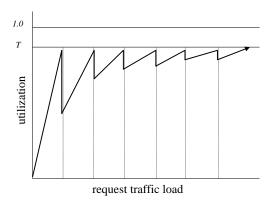


Figure 4: Utilization as request traffic increases.

proportionally, changing the values of the threshold points and thus the active capacity available at each request level. This is why each line has an apparent slope of T.

Energy-conscious server switching allows only coarse-grained power management; for example, the cluster must activate entire servers even if only a fraction of its resources are needed to handle a load increase. The shaded portions of Figure 3 represent excess capacity beyond what is needed to bound utilization to T. Finer-grained power management such as Intel SpeedStep could help to reduce excess capacity. Note, however, that server-grained power management is sufficient for large clusters: the shaded regions account for a shrinking portion of total capacity as the cluster grows. Large clusters are increasingly common as Web hosting companies seek economies of scale by serving large numbers of Web sites from a common hardware base.

Figure 4 shows this effect in a different way by depicting the impact of power transitions on active server utilization. As utilization approaches the configured threshold T, the cluster activates servers to add capacity. When the active set is small, adding a server yields a significant downward spike in utilization — and latency. For example, adding a server to an active set of one doubles the capacity of the cluster, cutting utilization in half. As the cluster grows, active set changes account for a progressively smaller share of the cluster resources, and utilization converges to closely approximate T.

4 Research Issues

The proposed scheme for energy-conscious server switching raises several issues for future research:

- Server Selection. Several policies are possible for selecting servers to step up (activate) or step down (deactivate). For example, random spreads active time evenly across the servers, smoothing failure rates; less active servers are less likely to experience hardware failure. However, a select by rank policy ("last hired first fired") may yield better performance because it deactivates the active server with the coldest cache, and reactivates the inactive server with the warmest cache.
- Content Switching. Some server switches use more sophisticated "layer 7" routing policies that consider the specific content requested (URL switching) or the source of the request (cookie switching). These constrain the choice of servers to deactivate, since each server handles a specific subset of the request stream. Energy-conscious switching places pressure on content-routing policies to adapt to changes in the active server set by redistributing the content routing function.
- Server Coordination. Inactive servers cannot function as full-fledged cluster members while in a low-power state. Energy-conscious switching induces more frequent cluster membership changes, which may be problematic for cluster applications with strong synchronization or consistency requirements. In particular, servers must release exclusive resources (e.g., cached locks) before deactivating, and revalidate cached copies of cluster data during reactivation. A closely related issue is the question of stability of the active set in the presence of frequent changes in request load.

In addition, it is interesting to consider how the energy-conscious switching applies to other systems that scale by adding functionally identical components, such as decentralized storage systems.

5 Conclusion

This paper promotes a research agenda to improve energy-efficiency in Internet server clusters. In particular, we propose energy-conscious request switching as one approach to reduce energy usage for clusters running below capacity, which is common given that Internet service loads fluctuate by up to an order of magnitude at all time scales.

In our proposal, energy-conscious server switches monitor cluster load and concentrate request traffic on the minimal set of servers that can serve the incoming traffic at a specified utilization and latency. This induces the remaining idle servers to step down to a low-power state. The potential savings from this technique are compelling given the costs of energy usage and the growth in Internet service capacity. Our approach imposes higher service latency at low loads, but allows a site administrator to control tradeoffs of energy and performance by configuring a bound for utilization and latency.

Other energy management techniques, such as server power management, are complementary to our approach and are important elements of an end-to-end approach to energy conservation for servers.

References

- [1] US EPA global warming inventories. http://www.epa.gov/globalwarming/emissions/.
- [2] Hotting up in the Hague. *Economist*, November 2000.
- [3] Robert Bryce. Power struggle. Interactive Week, December 2000. http://www.zdnet.com/intweek/, found under stories/news/0,4164,2666038,00.html.
- [4] Surendar Chandra and Carla Schlatter Ellis. JPEG Compression Metric as a Quality Aware Image Tran scoding. In 2nd Symposium on Internet Technologies and Systems, Boulder, CO, October 1999. USENIX.
- [5] Fred Douglis, P. Krishnan, and Brian Bershad. Adaptive disk spin-down policies for mobile computers. In 2nd USENIX Symposium on Mobile and Location-Independent Computing, April 1995. Monterey CA.
- [6] A. R. Lebeck, X. Fan, H. Zengh, and C. S. Ellis. Power aware page allocation. In Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS IX), 2000.
- [7] A. Vahdat, A. R. Lebeck, and C. S. Ellis. Every joule is precious: The case for revisiting operating system design for energy efficiency. In *Proceedings of the* 9th ACM SIGOPS European Workshop, September 2000.
- [8] Amin Vahdat, Thomas Anderson, Michael Dahlin, Eshwar Belani, David Culler, Paul Eastham, and Chad Yoshikawa. WebOS: Operating System Services for Wide-Area Applications. In Proceedings of the Seventh IEEE Symposium on High Performance Distributed Systems, Chicago, Illinois, July 1998.