Power Management in Data Centers

Mor Harchol-Balter
Associate Department Head, Computer Science Department, Carnegie Mellon University

U.S. data center energy consumption is expected to double every 5 years, as shown in Fig. 1, at a huge cost both financially and to the environment. Microsoft itself launched two new mega-data centers last year, in Dublin and in Chicago. Sadly, much of the power consumed by these data centers is wasted. As documented in a recent Google study, [2], servers are only busy 20-30% of the time on average, yet idle servers are left on, consuming over 60% of the power of a busy server.

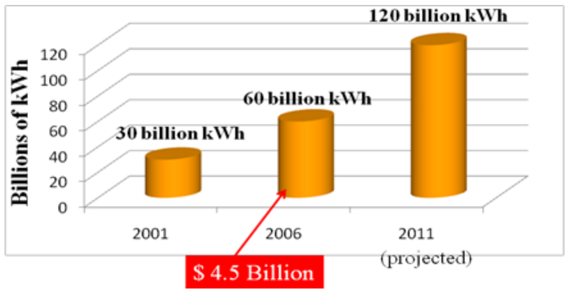


Fig. 1. Projected U.S. data center energy consumption [1].

The main problem is that server farms are provisioned for peak load, and thus are operating at a much higher number of servers than they really need. A secondary problem is that the question of how capacity provisioning scales is poorly understood, resulting in far more servers being used than is necessary, even in the case where load is known a priori. The goal of this project is to solve both these problems. We will use queueing-theoretic analysis and computational thinking to design algorithms for reducing power consumption in data centers while simultaneously minimizing response times. All of our algorithms will be implemented in our server farm at CMU comprising 14 Intel Xeon E5520 servers, with two quad-core 2.27 GHz processors and 16 GB of memory.

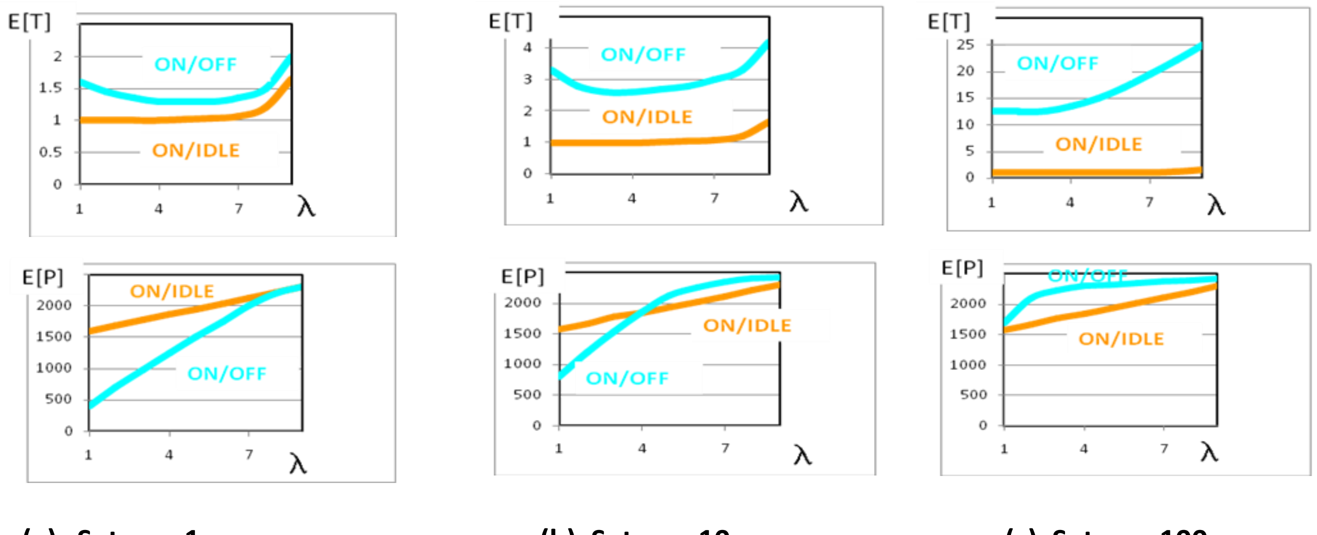
The Counter-intuitiveness of Scale

Power provisioning in server farms does not scale as people might imagine, and this confusion leads to over-provisioning. Consider the following example. Suppose that our goal is to guarantee that fewer than 20% of jobs are delayed. We find that in our lab setting, if servers each serve $\mu = 1$ job/sec, and the arrival rate is $\lambda = 9$ jobs/sec, then we need R = 12 servers to meet our delay guarantee. Observe that the server farm is operating at load $\rho = 9/12 = 75\%$. Based on our lab experiments, we now want to provision a real-world data center, where the arrival rate is $\lambda = 9{,}000{,}000$ jobs/sec, and each server still serves $\mu = 1$ job/sec. It is tempting to believe that we will need R = 12,000,000 servers to meet the same delay guarantee. However, the queueing literature shows that this intuition is *false*, and that in fact only R = 9,003,000 servers are required. Observe that with an arrival rate of $\lambda = 9{,}000{,}000$ jobs/sec and R =9,003,000 servers, we are actually operating at load $\rho = 99.9\%$, and yet still achieving our guarantee of delaying only 20% of jobs. A naïve designer might choose to go with R = 12,000,000 servers, over-provisioning by 3,000,000 servers, resulting in more than 3 Billion dollars of wasted power each year!

Does it Pay to Turn Servers Off?

We now move to the case where the load is not known in advance, making over-provisioning more likely. This is problematic because an idle server burns almost as much power as a busy server (160 Watts for an idle server, as compared with 240 Watts for a busy server). In such scenarios it may be desirable to save power by turning servers off. However, turning servers off necessitates a setup cost to later turn them back on. In our lab, this setup cost involves 200 seconds, at full power of 240 Watts. Thus it is not at all obvious whether it pays to turn servers off. This decision is further complicated by the existence of SLEEP states, with shorter setup times, but non-zero power requirements.

Consider two policies: ON/OFF turns servers off immediately when they're not needed, whereas ON/IDLE leaves servers on. The parameter space under which ON/OFF is superior to ON/IDLE is not well-understood, in part because the analysis of ON/OFF is not known (multiserver systems with setup cost have never been analyzed). This year, our group has derived the first closed-form analysis of ON/OFF, see [6]. This analysis allows us to investigate the effectiveness of ON/IDLE versus ON/OFF under a range of loads and setup times. Fig. 2 shows mean response time, E[T], and mean power, E[P], as a function of arrival rate λ, ranging from $\lambda = 1$ (load $\rho = 10\%$) to $\lambda = 9$ (load $\rho = 90\%$), for a server farm with 10 servers, where the mean job size is 1 second, and the setup time ranges from 1 second to 100 seconds. Under a setup cost of 1 second, we see that ON/OFF is clearly superior to ON/IDLE for mean power, but is worse for mean response time. Under a setup cost of 100 seconds, we find that ON/OFF is worse than ON/IDLE for *both* mean power and mean response time. However (not shown), we can also prove that if the number of servers increases from 10 to 1000 (while server utilization is held at 30%), then even for a setup time of 100 seconds, the ON/OFF policy will be superior to the ON/IDLE policy.



(a) Setup = 1 sec **(b) Setup = 10 sec** **(c) Setup = 100 sec**

Fig. 2. Comparison of ON/OFF and ON/IDLE evaluated on mean response time, E[T], and mean power, E[P], as a function of arrival rate (λ), where the setup time is (a) 1 second, (b) 10 seconds, or (c) 100 seconds.

The point is that there are big differences in response time and power usage among even simple policies, like ON/OFF and ON/IDLE. This demonstrates the fact that there is much potential for improving the efficiency of server farms.

Managing Power under Time-Varying Load

When the load varies over time, as shown in Fig. 3, it is particularly difficult to provision for server farms. Companies generally provision for "peak" load, and leave all servers on. This is extremely wasteful of power, since servers are only 20-30% utilized on average [2]. Unfortunately, turning servers off when not needed, as in ON/OFF, is also not very effective in this setting. ON/OFF is too quick to turn off servers when they're not needed, and then suffers a lot of setup time to get them back on. We therefore need an algorithm that turns servers off, but is less aggressive than ON/OFF. We recently developed such an algorithm, which we call DelayedOff/MRB [7]. DelayedOff/MRB has two features. First, it waits a fixed time t_{wait} before shutting down an idle server (we have derived a formula for t_{wait}). Second, to minimize the number of servers in the idle state, we use MRB routing, whereby a new arrival is routed to the idle server which was most recently busy. We can prove that DelayedOff/MRB achieves near-optimal capacity provisioning, under time-varying load [7]. Fig. 3 shows via trace-driven simulation that the number of servers used by MRB (the number of servers that are busy plus the number idle) is very close to the minimum number of servers needed to just handle load at every moment in time.

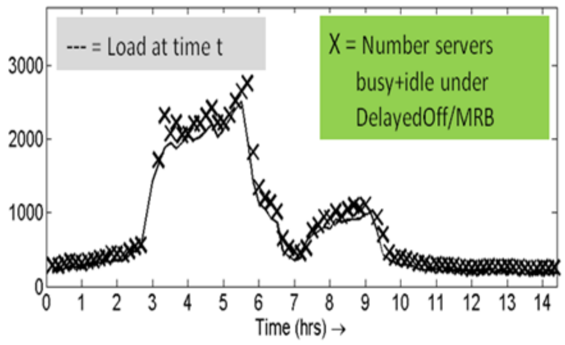


Fig. 3. DelayedOff/MRB shows promise of providing near-optimal provisioning under time-varying load with unknown arrival rate.

Our next step is to try to demonstrate that DelayedOff/MRB is feasible in practice. Our plan is to experiment with the IPMITool [8] to remotely turn servers on and off as specified by the DelayedOff/MRB algorithm and see if the algorithm performs as well in practice as is predicted by theory. If DelayedOff/MRB works well in practice, the impact will be huge, because the algorithm is simple, doesn't require any knowledge or prediction of load, and yet provides performance (with respect to both power and response time) comparable to the best achievable performance when load is known a priori.

Another important area that we still need to research is the potential benefits of SLEEP states. The advantage of the SLEEP state is that the setup costs are low (typically 10-20 seconds) although power is still non-trivial (about 50 Watts). Our eventual goal is to develop a simple load-oblivious algorithm that achieves the optimal number of busy/idle/sleeping/off servers at every moment of time.

Power-Aware Load Manager

Thus far, we have only discussed power-aware capacity provisioning for the compute-bound servers. For applications that are data-intensive, such as Facebook, the compute-bound servers typically occupy the "application layer" of the server farm, as shown in Fig. 4. Our goal is to design a Power-Aware Load Manager that optimally provisions not only the application layer, but also the Memcache layer, where most data is cached, so that the number of servers that are on in the Memcache layer also varies with time [1]. We also plan to deploy optimal frequency provisioning at the servers [4] as well as utilizing power-capping technologies like IdleCap [5].

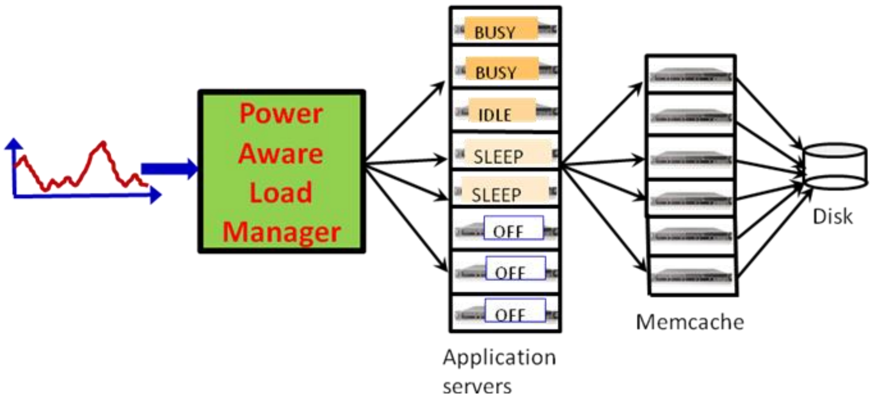


Fig. 4. Power-Aware Load Manager optimally configures not just the Application servers, but also the Memcache.

Grand Challenge/Impact: This research fits under the Grand Challenge of Sustainability and Energy. Given the clear need for power-aware load managers for data centers, it is likely that this work will turn into a product at some point.

Keywords: power management; data center; capacity provisioning, scalability, queueing theory.

MSR collaborators: I am open to collaborating with anyone interested in power. However, two people with whom I already have close ties are: Dushyanth Narayanan and Eno Thereska, both at Microsoft Cambridge.

Funding: I seek 1 year of funding for my student, Varun Gupta, who is on E&GO, and 2 months of summer funding for myself. This totals about $100K.

References

[1] Hrishikesh Amur, Jim Cipar, Varun Gupta, Mike Kozuch, Greg Ganger, Karsten Schwan, "Robust and Flexible Power-Proportional Storage.", *ACM Symposium on Cloud Computing,* Indianapolis, IN, June 2010. pdf.
[2] Luiz Barroso and Urs Hölzle. "The case for energy-proportional computing." *IEEE Computer*, 40(12): 33-37, 2007.
[3] "EPA: Power usage in data centers could double by 2011" URL
[4] Anshul Gandhi, Mor Harchol-Balter, Rajarshi Das, and Charles Lefurgy. "Optimal Power Allocation in Server Farms", *Proceedings of ACM SIGMETRICS 2009.* Seattle, WA, June 2009. pdf .
[5] Anshul Gandhi, Mor Harchol-Balter, Rajarshi Das, Jeff Kephart, and Charles Lefurgy. "Power Capping Via Forced Idleness", *Workshop on Energy-Efficient Design (WEED 09)* Austin, Texas, June 2009. pdf
[6] Anshul Gandhi, Mor Harchol-Balter, Ivo Adan. "Server farms with setup costs." *Performance Evaluation,* 67(11), 2010. pdf.
[7] Anshul Gandhi, Varun Gupta, Mor Harchol-Balter, and Michael Kozuch. "Optimality Analysis of Energy-Performance Trade-off for Server Farm Management." *Performance Evaluation,* 67(11), 2010. pdf
[8] IPMItool. URL