Power-Demand Routing in Massive Geo-Distributed Systems

by

Asfandyar Qureshi

S.B. Computer Science and Engineering (2004)
M.Eng. Electrical Engineering and Computer Science (2005)
Massachusetts Institute of Technology

Submitted to the
Department of Electrical Engineering and Computer Science

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2010

Author .....
Department of Electrical Engineering and Computer Science
September 3, 2010

Certified by .....
John V. Guttag
Professor
Thesis Supervisor

Accepted by .....
Professor Terry P. Orlando
Chairman, Department Committee on Graduate Theses

Power-Demand Routing in Massive Geo-Distributed Systems

by

Asfandyar Qureshi

Submitted to the Department of Electrical Engineering and Computer Science
on September 3, 2010, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

There is an increasing trend toward massive, geographically distributed systems. The largest Internet companies operate hundreds of thousands of servers in multiple geographic locations, and are growing at a fast clip. A single system's servers and data centers can consume many megawatts of electricity, as much as tens of thousands of US homes. Two important concerns have arisen: rising electric bills; and growing carbon footprints. Our work develops a new traffic engineering technique that can be used to address both these areas of concern.

We introduce Power-Demand Routing (PDR), a technique that redistributes traffic between replicas with the express purpose of spatially redistributing the system's power consumption, in order to reduce operating costs. Cost can be described in monetary terms or in terms of pollution. Within existing Internet services, each client request requires a meaningful amount of marginal energy at the server. Thus, by rerouting requests from a server at one geographic location to another, we can spatially shift the systems marginal power consumption at Internet speeds.

We show how PDR can be used to reduce electric bills. We describe how to couple request routing policy to real-time price signals from wholesale electricity markets. In response to price-differentials, PDR skews client load across a system's clusters and pushes server power-demand into the least expensive regions. Our analysis quantifies the potential reduction in energy costs. We use simulations driven by empirical data and models: we collected a real-world request traffic workload in collaboration with Akamai; constructed data center energy models; and compiled a database of historical electricity market prices. We conclude that existing systems can use PDR to cut their annual electric bills by millions of dollars.

We also show how PDR can be used to reduce carbon footprints. Not all joules are created equal and in power pools like the grid the environmental impact per joule varies geographically and in time. We show how to construct carbon cost functions that can be used with PDR to dynamically push a system's power-demand toward clean energy.

Thesis Supervisor: Professor John V. Guttag

to my parents,

and to my brother.

Acknowledgments

I'm sure I'm forgetting someone. Acute sleep deprivation and a decade of drinking from the MIT firehose does wonders for one's non-technical memory.

This one's easy: John Guttag. I started working with him as an undergraduate and throughout the years he has patiently put up with my profound unpredictability and accompanied me on all those odd tangents I went on before I finally fell down this rabbit hole.

I don't think I can overstate Hari Balakrishnan's contributions to this work. There were so many times I thought I understood something, until one of Hari's offhand remarks revealed the depths of my ignorance and sent me back to the drawing board.

And Frans Kaashoek has been a great mentor, ever since I was an undergraduate. His advice during the final days of this dissertation has greatly improved the quality of this document.

This work would not have been possible without the involvement of Bruce Maggs, Rick Weber, and others at Akamai.

I'm just going to list names now: Ali Shoeb, Hariharan Rahul, Eugene Wu, Srikanth Kandula, Irene Fan, Al Kharbouch, Jennifer Carlisle, Vladimir Bychkovsky, Yang Zhang, Magdalena Balazinska, Michel Goraczko, Omair Malik, Allen Miu, Godfrey Tan, Yuan Mei, James Cowling, Pouya Kheradpour, Eugene Shih, Stavros Harizopoulos, Philippe Cudre-Mauroux, Dina Katabi, Robert Morris, John Ankcorn, Jamey Hicks, Nick Feamster, John Crowcroft, David Anderson, Dorothy Curtis, and Sheila Marian.

That's everyone. I think. No. Probably not.

Contents

1 Introduction **21**

1.1 Power-Demand Routing (PDR) 23

1.2 Important Factors 26

1.3 Dissertation Structure 31

2 PDR: An Optimization Problem **33**

2.1 Mathematical Model 34

2.2 Reconciling Competing Costs 42

2.2.1 Aggregate Cost Function 43

2.2.2 Sequential Optimization 44

2.3 Implementing Optimization Mechanisms 44

2.3.1 Linear Programming 45

2.3.2 Nonlinear Programming 45

3 Modeling Energy Consumption **47**

3.1 Electricity Consumption in Data Centers 49

3.2 IT Equipment Energy 51

3.2.1 Google Cluster Model 52

3.2.2 Alternative Component-Based Models 53

3.3 Cooling System Energy 61

3.4 Power Distribution System Overhead 65

3.5 Cluster Energy Models 70

3.6 Modular Data Centers 72

3.7 Energy Proportionality Metrics 73
3.8 Increase in Wide-Area Routing Energy 75

4 Modeling Workloads and Performance **77**
4.1 Akamai Traffic Workload 78
4.1.1 Raw Traffic Data 78
4.1.2 Synthetic Traffic Model 80
4.2 Modeling Performance Cost 83
4.2.1 Calculating Client-Server Distances 83
4.2.2 Distance and Latency 84
4.2.3 Monetary Performance Cost Functions 86

5 Cutting Electric Bills **89**
5.1 The Scale of Electricity Expenditures 91
5.2 Wholesale Electricity Markets 93
5.3 Empirical Market Analysis 97
5.3.1 Price Variation 97
5.3.2 Geographic Correlation 100
5.3.3 Price Differentials 102

6 Quantifying Savings **107**
6.1 Simulation Strategy 108
6.1.1 Simulation framework. 108
6.1.2 Routing Algorithms 110
6.2 At the Turn of the Year: 24 Days of Traffic 114
6.3 Synthetic Workload: 39 Months of Prices 118
6.4 Performance and Savings 121
6.5 Different Energy Models 122
6.6 Different Geo-Distributions 126
6.7 Accounting for Complex Network Costs 128
6.8 Imperfect Knowledge 130

7 Other Considerations **133**

7.1 Existing Electricity Contracts 133

7.2 Impact on Market Prices 135

7.3 Actively Engaging with the Grid 136

7.4 Partial Replication 139

7.5 Multihomed Clusters 141

7.6 Complex Network Cost Models 142

7.7 Valuing Resources in Clouds 146

8 Greener Systems **147**

8.1 Generation and Pollution 148

8.2 Carbon Cost Functions 151

8.2.1 Annual Average Intensities 151

8.2.2 Wind Power 153

8.2.3 Real-Time Pollution Data 154

8.2.4 Extrapolating from Grid Load 155

9 Related Work **159**

10 Conclusion **161**

List of Figures

1-1 Estimated annual server electricity consumption, for some large companies. These estimates assume $60/MWh. See §5.1 for derivation details. For comparison, we have included the EPA's 2008 estimate of the annual electricity consumption of US homes; and the 2007 consumption and utility bill reported for the MIT campus. Generation cost can be viewed as the variable portion of the electric bill, the part that depends exclusively on the number of watt-hours consumed, and the electricity market prices for those watt-hours. 22

1-2 Rerouting clients in order to spatially shift a system's power demand. . . . 23

2-1 The PDR problem posed as a min-cost network flow problem. 34

2-2 Cluster and network properties, represented as node and edge labels. . . . 37

2-3 PDR as an optimization problem. 42

3-1 Reported PUE values and the years for which they were reported. The table is sorted by PUE. Where available, we have also listed the fraction of total power used by the cooling (C) and power distribution (PD) sub-systems. Most PUE's were calculated from measurements; an asterisk indicates an estimate. 51

3-2 Server power curves can have a variety of shapes. To keep the focus on the shape, the power for each machine type is given as a fraction of its peak power in this figure. The curves are constructed from SPECpower measurement reports [19], and a disclosure by Google [44]. We linearly interpolate between reported measurements. 55

3-3 Server power measurements. The rate is the normalized transaction rate reported for that machine type. The data is from SPECpower measurement reports [19], from public disclosures by Google [56, 44] and a SeaMicro press release [101]. Asterisks indicate estimates. 56

3-4 IT power curves for simulated 1000-server clusters, comparing different load distribution policies and different machine types. We assume that 5% of the peak IT power is accounted for by network switches and other equipment. 57

3-5 IT power curves for two types of simulated 1000-machine clusters with DSS, using a safety margin of 50 servers that are always on. 59

3-6 IT power curves when we relax server homogeneity assumptions. In (a) different kinds of machines are mixed together to simulate 4800-core clusters; there are twice as many 4-core machines as 8-core machines, etc.; and load is spread evenly. In (b) a homogeneous 1000-machine cluster of Google-class machines experiences uneven load balancing, causing individual servers to deviate from mean utilization. 60

3-7 We model a chiller's power consumption as a quadratic function of the thermal load placed on it. Efficiency can vary dramatically between different kinds of chillers. In these figures, each chiller's power has been normalized to its heat removal capacity. 64

3-8 Power consumption models for several chillers. The equations map thermal load (0.0-1.0) to chiller plant power usage. Usage is defined as a fraction of the chiller's heat removal capacity. 64

3-9 Schematics for two representative power distribution systems. The top design is based on Amazon; the bottom design is based on Google. 65

3-10 Transformer losses increase with load. Loss is given as a fraction of the transformers' output capacity. 67

3-11 Transformer power loss models, from statistical regressions of measurements conducted by the DOE and reported by Powersmiths [82]. The equations map load (0.0-1.0) to transformer power overhead (as a fraction of transformer output capacity). 67

3-12 UPS losses and efficiency increase with increasing load. Loss is given as a fraction of the UPS's output capacity. 68

3-13 UPS power loss models, from statistical regressions of data reported by two UPS manufacturers. The equations map load (0.0-1.0) to UPS power overhead (as a fraction of UPS output capacity). 68

3-14 A collection of cluster energy models. See §3.7 for EPG and EAP. 71

3-15 Power curves for some energy models (1000-server clusters; 150W peak server power). The IT component dominates the overall shape (e.g., compare `E1` with `E2`). 72

3-16 Cluster utilization distribution used to calculate EPG and EAP scores. . . 74

4-1 Aggregate traffic in the Akamai data set. We see a peak hit rate of over 2M hits per second. Of this, about 1.25M hits/s come from clients in the US. Note the traffic dip during the holidays. The traffic in this data set comes from roughly half of the CDN servers Akamai had. In comparison, in total, Akamai saw around 275B hits/day (more than 3M hits/s) during this period. 79

4-2 Traffic in the Akamai data for individual origin states. 80

4-3 As shown by this alternate view of the raw data, traffic volume depends on the hour-of-day, but the nature of the dependency varies from state to state. 81

4-4 Comparing the aggregate traffic volume generated by different models. . . 82

4-5 Difference between observed US traffic and model generated traffic. 82

4-6 Difference between observed traffic volumes and model (median, -10 days) generated volumes for two locations. 83

4-7 The relationship between network latency and client-server geo-distance. . 85

4-8 The relationship between revenue and latency and monetary performance cost functions that capture this relationship. Google-1 and Google-2 are upper and lower bounds respectively. 87

5-1 Estimated annual electricity costs for large companies. These are conservative estimates, meant to be lower bounds @ $60/MWh. See §5.1 for derivation details. For comparison, we have included the EPA's 2008 estimate of the annual energy consumption of US homes; and the 2007 consumption and utility bill for the MIT campus, including dormitories and labs. 90

5-2 The different regions studied in this dissertation. The listed hubs provide a sense of RTO coverage. 94

5-3 Daily averages of day-ahead peak prices at different locations [96]. The elevation in 2008—everywhere but the hydroelectric dominated Northwest—correlates with record high natural gas prices. The Northwest consistently dips near April (this seems to be correlated with seasonal rainfall). Correlated with the global economic downturn, all prices shown here exhibit a downward trend. 98

5-4 Comparing price variation in different wholesale markets, for New York City. The top graph shows a period when prices were similar across all markets; the bottom graph shows a period when there was significantly more volatility in the real-time market. 99

5-5 The real-time market is more variable at short time-scales than the day-ahead market. Standard deviations for Q1 2009 prices for NYC are shown, averaged using different window sizes. 99

5-6 Real-time market statistics, covering hourly prices from January 2006 through March 2009 (*statistics are from the 1% trimmed data). 100

5-7 Histograms of hour-to-hour change in real-time hourly prices for two locations, over the 39-month period. Both distributions are zero-mean, Gaussian-like, with long tails. 100

5-8 The relationship between price correlation, distance, and parent market. Each point represents a pair of locations (29 locs, 406 pairs), and the correlation coefficient of their hourly prices over the specified period. Red represents pairs with sites from different markets; blue points are labeled with the shared market. 101

5-9 The relationship between price MI, distance, and parent market. Each point represents a pair of locations (29 locs, 406 pairs), and the normalized MI of their hourly prices over the specified period. Red represents pairs with sites from different markets; blue points are labeled with the shared market. . . 101

5-10 Variation of price differentials with time. 102

5-11 Hourly price differential histograms for six location pairs (2006-2009). . . 103

5-12 PaloAlto-Virginia price differential distributions for each month. The monthly median prices and inter-quartile range are shown. 104

5-13 Price differential distributions (median and inter-quartile range) for each hour of the day. 105

5-14 For PaloAlto-Virginia, short-lived price differentials account for most of the time. 105

6-1 A collection of cluster energy models. See §3.7 for EPG and EAP. This is a reproduction of figure 3-14. Models with DSS turn off idle servers. 110

6-2 Pseudocode for the *nlp* router. *GeoDist* calculates the great-circle geographic distance, and *ModeledEnergyCost* uses an energy model and the local electricity prices to calculate a cluster's energy cost from a traffic volume. 112

6-3 Pseudocode for the *radial* router. 113

6-4 24-day simulation results with the `G6` energy model, comparing energy costs (per 10K servers) and client-server distances for several routing policies and the Akamai server geo-distribution. 115

6-5 24-day simulation results with the `D1` energy model, comparing energy costs (per 10K servers) and client-server distances for several routing policies and the Akamai server geo-distribution. 116

6-6 Variation in hourly energy costs and savings, with the *nlp* router. Clusters are not evenly sized in the Akamai geo-distribution; the thin slivers at the base of the graph represent the small clusters. 117

6-7 CDF of hourly savings over 24 days, for the *nlp* router. 117

6-8 Cluster utilization under different routing policies. In each region PDR increases the average utilization of clusters with low costs. These clusters have unequal capacity: e.g., the MA cluster is about 2% the size of the NYC cluster; the IL cluster is about half the size of the NYC cluster. . . . 118

6-9 39-month simulation results with the G6 energy model, comparing energy costs (per 10K servers) and client-server distances for several routing policies and two different server geo-distributions. 119

6-10 Monthly energy costs over 39 months, with the unconstrained *nlp* router. This shows cost per 10K servers, for the synthetic-5 distribution and the G6 model. 120

6-11 The average client-server distance required to achieve a certain level of savings: (a) depends on the routing policy (24-day savings, Akamai geo-distribution); and (b) depends on the clusters' geo-distribution (39-month savings, with the radial-gradient router). These curves are for the G6 energy model. 121

6-12 Savings achieved by the unconstrained *nlp* router for different energy models, with the synthetic-5 geo-distribution. The absolute 3-month savings are for 10K servers. 123

6-13 Simulations with different energy models, geo-distributions, and time periods show that the EPG score is a good predictor for absolute savings (per 10K servers). 125

6-14 Simulations with different energy models, geo-distributions, and time periods show that the EAP score is a good predictor for relative savings. . . . 125

6-15 39-month energy costs for several cluster distributions, with the G6 model. The first four distributions place all servers in one cluster. Even with two clusters, PDR can yield a meaningful cost reduction. 126

6-16 39-month energy costs for the least expensive market and several cluster distributions, with the E2 model. When three or more clusters exist, the *nlp* router can lower costs below the least expensive market. 127

6-17 Simulation results for over a hundred different server geo-distributions, with between 2 and 28 clusters. The results are from one year costs, and compare the costs of a radial router (1000km threshold) with the cost unaware variant. All these simulations used the G6 energy model. 128

6-18 The impact on savings when we incorporate 95/5 bandwidth constraints to prevent network costs from rising. These simulations used the G6 model. . 130

6-19 Simulation results when the electricity prices seen by the router deviated from actual prices (Akamai geo-distribution, 24-day raw traffic traces, and G9 model). In some simulations prices were delayed so that the router saw hour-old or day-old prices. In other simulations the router was passed randomly generated prices. We also passed in artificial prices that did not vary in time: the mean price for a location (ave); a partial order (part); and prices that ordered locations inversely (inv). 131

7-1 A PDR graph for a partially replicated system. There are two clusters, three shards, and two ingress points. Shard #1 is the only shard replicated at both clusters. 140

7-2 Modeling multihomed clusters with a modified flow graph. 142

8-1 Annual average 2005 carbon emissions and generation fuel mixes for some different states. These numbers are from the EPA's eGRID data [24]. The emissions are calculated from consumption, not from generation (the states also imported electricity, so the consumption fuel mix may not match the generation fuel mix). 149

8-2 Normalized costs and carbon emissions from simulations with static carbon cost functions, the Akamai geo-distribution, the 24-day traffic trace, and the G6 energy model. In both graphs, the left-most pair of bars represent routing policies unaware of carbon costs and market prices; and the right-most bar shows the *nlp* router set up to optimize for market prices instead of carbon. 152

8-3 Normalized load and pCI values for two regions with different generation profiles. Power in Illinois gets dirtier at higher loads, the opposite of what happens in Texas. The load signals are aggregate loads for the local RTO and are each localized to their RTO's maximum. 156

Chapter 1.

Introduction

With the rise of "Internet-scale" systems and "cloud computing" services, there is an increasing trend toward massive, geographically distributed systems. The largest Internet companies operate hundreds of thousands of servers, sectioned into several clusters in different locations. Some of these clusters are entire data centers, others are smaller units that use shared space in multi-tenant facilities.

A single system can consume tens of megawatts of electricity, an energy footprint that is comparable to tens of thousands of US homes [76]. Our estimates for the energy demands of some Internet companies are shown in figure 1-1. These companies are secretive about their energy consumption, and we calculated these demands by collecting and composing many pieces—leaked information, public disclosures, server energy models, and back-of-the-envelope calculations. Later in this dissertation we describe our estimation methodology in detail (§5.1).

Energy footprints have grown large enough that organizations such as Google, Microsoft, Facebook, and many other operators of large systems cannot ignore their energy costs. Millions of dollars must be spent annually on the electricity needed to power one of these systems. Furthermore, while impressive advances have been made in the areas of server and data center energy efficiency, total energy consumption has continued to rise rapidly.

Efficiency gains have been swamped by server growth. These already enormous systems are increasing in size at a rapid clip. In 2006, it was estimated that Google

Company	Servers	Electricity	Gen. Cost	Utility Bill
eBay	20K	65 GWh	$3.8M	10M
Rackspace	50K	160 GWh	$9.6M	20M
Facebook	60K	190 GWh	$11.5M	25M
Akamai	65K	210 GWh	$12.5M	25M
Microsoft	>200K	>600 GWh	>$36M	>$75M
Google	>800K	>1120 GWh	>$67M	>$135M
USA (2006) [25]	10.9M	61,000 GWh	$4.5B	
10,000 US homes [32]		<160 GWh		<$30M
MIT campus [20]		270 GWh		$62M

Figure 1-1: Estimated annual server electricity consumption, for some large companies. These estimates assume $60/MWh. See §5.1 for derivation details. For comparison, we have included the EPA's 2008 estimate of the annual electricity consumption of US homes; and the 2007 consumption and utility bill reported for the MIT campus. Generation cost can be viewed as the variable portion of the electric bill, the part that depends exclusively on the number of watt-hours consumed, and the electricity market prices for those watt-hours.

had 450K servers [86]. In 2010, Google disclosed that they had grown to over 800K servers, and that they were redesigning their infrastructure to scale to millions of servers [53, 66]. Microsoft added hundreds of thousands of servers over the course of a few years [12]. Similarly, Facebook's infrastructure has been doubling in size every six months or so, growing from 10K servers in April 2008 to 30K servers in November 2009 to 60K servers in June 2010 [67].

Two factors driving this growth are: the increasing usage of Internet services; and the shift to cloud computing. By all accounts, we should expect the load on Internet services to grow for some time. Additionally, the shift toward cloud computing is resulting in a consolidation of IT resources into these geo-distributed systems. Smaller companies are outsourcing their IT infrastructure into the cloud, leasing servers on Amazon's EC2 instead of deploying their own, for example, or relying on Google to manage their email instead of deploying their own servers.

Unsurprisingly, the increasing energy demands of these systems have been a cause for concern. Two areas that have received appreciable attention are: rising electric bills and growing carbon footprints. This dissertation develops a new traffic engineering technique that can be used to address both these areas of concern.

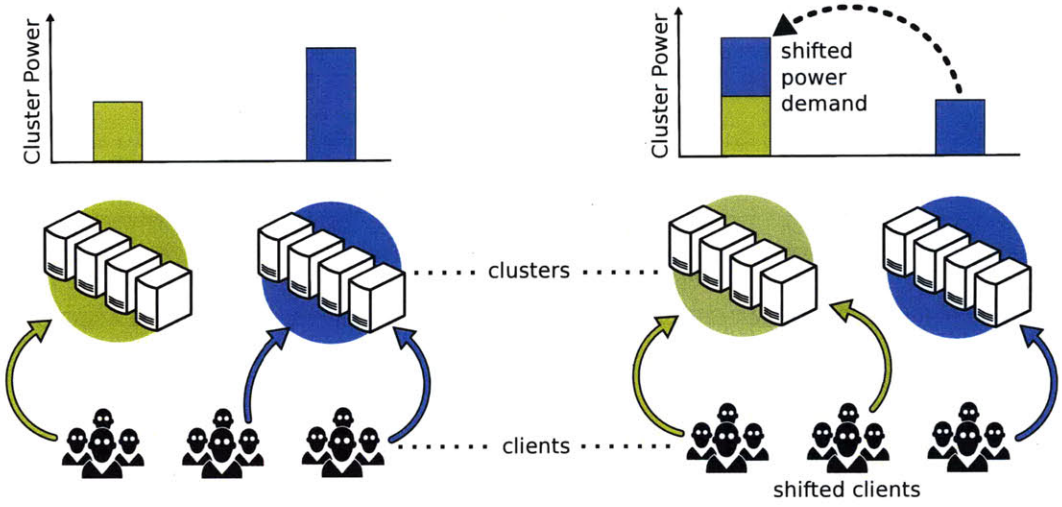


Figure 1-2: Rerouting clients in order to spatially shift a system's power demand.

1.1 Power-Demand Routing (PDR)

Server clusters consume some marginal energy for every client request they process. In systems like Google's search infrastructure, for example, complex energy-intensive operations potentially spanning hundreds of servers are triggered by each search query [71]. Our work builds upon the observation that by rerouting requests from a cluster at one geographic location to another, a system can rapidly move a portion of its electric power demand between those locations. While many consumers on the electric grid can shift their consumption in *time*[1]—delaying consumption to improve the grid's efficiency[2]—only geo-replicated systems can shift their consumption in *space*.

In this dissertation, we introduce Power-Demand Routing, or PDR, a technique that dynamically redistributes traffic between clusters with the express purpose of spatially redistributing the system's power consumption. Figure 1-2 depicts this process. PDR modulates the power-demand distribution with the goal of minimizing total operating costs.

The geo-distributed systems that we focus on already engineer their traffic, rebal-

[1] e.g., electric cars, aluminium smelters, hotel laundry machines, and even residential consumers enrolled in 'demand response' programs

[2] Consumers can be staggered in time to reduce daily peak load; or when demand rises close to the capacity of active power plants, consumption can be delayed instead of activating another plant.

ancing client load across clusters, to optimize for performance or network bandwidth costs, or both. To provide clients good performance and to tolerate faults, these systems implement some form of flexible request routing to map clients to servers, and have mechanisms to replicate the data necessary to process requests at multiple sites. Conventional traffic engineering mechanisms, however, ignore energy consumption.

In contrast, PDR takes the spatial distribution of a system's energy consumption into account and the PDR optimization reasons about trade-offs between several different objectives: performance, network costs, and energy costs. We develop PDR as an extension of conventional traffic engineering approaches. Energy costs can be described in monetary terms or in terms of pollution.

A key idea in our work is to track the geographic and temporal variations in energy cost (e.g., regional differences in electricity prices or differences in how 'clean' the electricity is) and to dynamically adapt request routing policy to take advantage of, possibly transient, differentials. This dynamic adaptation skews client load, increasing the load on less costly regions whenever excess capacity exists, and pushing server idleness into the most costly regions. In this way, we can cut operating costs. We describe in this dissertation how to couple request routing policy to price signals from real-time electricity markets, and estimate that existing Internet-scale systems could use this approach to reduce their annual operating costs by millions of dollars.

In addition to benefiting Internet companies, space-shifting electricity consumption in this manner has the potential to improve the efficiency of the grid. Electricity cannot always be efficiently transported to where it is needed: not all points on the grid are connected, and line losses and congestion give rise to other inefficiencies. Space-shifting would allow demand to be rapidly relocated to where cheap or clean energy is being produced (e.g., where the wind is blowing), or relocated away from a temporarily overloaded region (e.g., preventing new gas turbines from being activated). Requests can be efficiently rerouted between Internet replicas and, by manipulating a small amount of information (routing tables), we can move megawatts of power-demand. This makes request triggered work in replicated systems fundamentally different from other forms of work.

Given this potentially symbiotic relationship, what then should the interface between the electric grid and geo-replicated systems look like? Proposing a new interface is beyond the scope of this dissertation; instead, we focus on existing interfaces. In most of our work we assume that wholesale electricity markets provide the interface: PDR observes prices on the spot electricity market for a cluster's location, and modulates power consumption at that location. We also consider a more interactive interface: demand response, where the grid operator sends a signal to a cluster when it wants power consumption to be quickly decreased at that location.

Potential Obstacles

The ability to dynamically displace a system's power consumption could offer many potential benefits. However, it is not immediately obvious that PDR would be useful in practice. Some concerns are:

- *Does load redistribution appreciably affect a cluster's energy consumption?* If the energy consumption of a cluster is not proportional to the number of requests it services, PDR will be unable to move consumption between clusters. Older data centers[3] have a low degree of proportionality, consuming 85% or more of their peak power when they are idle. One system we studied consumed more than 95% of its peak power when idle.

- *Are there enough tasks that can be shifted in space?* If the average load is close to the system's capacity, PDR will be forced to use resources everywhere. Additionally, if the state needed to service a class of requests (e.g., a user's inbox data) is not replicated at all clusters, PDR will be constrained in its load redistribution choices.

- *What is the impact on latency?* In order to extract meaningful savings, PDR may need to route clients to distant clusters, in search of inexpensive energy. Many Internet services however strive to minimize request processing latencies, and strict network latency requirements could severely restrict PDR's routing choices.

[3] e.g., with a PUE rating of 1.9 (see ch. 3) and idle servers drawing 65% of their peak power.

- *Do meaningful cluster cost differentials exist?* Even when one can route a significant portion of a system's power demand, one cannot always appreciably reduce energy costs by doing so. For PDR to generate tangible benefits, geographic differences in request servicing costs must exist. Furthermore, for the dynamic approach we advocate, service costs at different clusters should ideally vary in time and different locations should not exhibit highly correlated behaviour.

- *Could PDR increase other non-energy costs?* A load redistribution that cuts energy costs could raise network costs, and overcome the energy-related savings.

Determining the applicability of PDR to a given system scenario requires an involved analysis that carefully considers the above issues and more. To facilitate such an analysis, we identify a number of factors that govern the effectiveness PDR.

1.2 Important Factors

We first proposed the use of traffic engineering to modulate the spatial distribution of a system's energy consumption in response to geographic differences in service costs, in 2008 [97, 98]. Rudimentary precursors to PDR had been previously proposed (e.g., follow-the-sun strategies coupled with solar power) and since our initial work, other proposals similar to PDR have emerged [80]. The optimization framework (chapter 2) and the routing algorithms (chapter 6) we describe demonstrate that PDR could be implemented relatively easily.

Our work in this dissertation focuses on identifying factors that dictate the effectiveness of PDR. Some factors constrain how power-demand can be routed (proportionality, capacity, replication, etc.), while others determine whether dynamically relocating power-demand can reduce operating costs. The set of factors that follows is not meant to be an exhaustive list. Using system models derived from empirical data we use simulations to quantitatively explore how these factors affect PDR. Our work offers a template for how to evaluate an actual system, to determine whether PDR can be used to improve that system's operational efficiency.

Energy Proportionality

The effectiveness of PDR hinges on the *energy proportionality* of the individual clusters that constitute the system. Energy proportionality is the degree to which the electrical energy consumed by a cluster depends on the traffic load placed on it. To understand how well PDR will perform, we must therefore first express the energy consumption of a cluster as a function of its load.

Data centers are highly complex and evolving systems. Chapter 3 is devoted to a discussion of the energy consumption characteristics of these facilities. We build a portfolio of cluster energy models that covers a wide range of different data center architectures. We construct models for existing, legacy and proposed future architectures. Our modeling work relies heavily on empirical data and extends and composes existing models.

We also propose two novel energy proportionality metrics. These metrics summarize energy-vs-load curves as single numbers. Simulation results show that these metrics are good predictors for how well PDR will perform for a given energy curve.

Spare Capacity

Another critical factor for PDR is the availability of sufficient spare capacity. PDR works by shifting load away from servers in high cost locations to servers in lower cost locations. However, at high load levels, the system is constrained to utilize most of its resources, wherever they may be located.

Fortunately, existing systems spend most of their time operating below their capacity. System operators provision for peak load, to ensure that performance does not degrade when traffic spikes. This leaves most resources idle much of the time. Google has reported that its average server load levels are below 30% of their capacity [44]. Twitter's peak request rate on the day of the 2009 US presidential inauguration was 5× its normal rate [92].

To better understand the nature of real web workloads, we collected a set of traffic traces in collaboration with Akamai (chapter 4). The traces were from Akamai's content distribution network and represent a meaningful slice of web traffic in the US,

with about a hundred billion requests per day. We used this workload in simulations to investigate how different routing policies affect the spatial distribution of a system's electric power-demand throughout a day.

In the future, systems may operate closer to their capacity. There are strong economic incentives to find useful work for idle servers. One motivation for companies like Amazon and Google to build their cloud computing platforms was so that their idle resources could be used to host paying tenants. However even such multi-tenant systems need to provision for infrequent events that cause usage spikes (e.g., Amazon during the holiday season; Twitter during the inauguration).

Geo-Distribution

Internet-scale systems tend to be geographically distributed, with machines at several—even hundreds of—sites around the world. To provide clients good performance and to tolerate faults, these systems implement some form of dynamic request routing to map clients to clusters, and have mechanisms to replicate the data necessary to process requests at multiple sites.

There is no dominant server geo-distribution pattern. In the US: Akamai spreads its servers across hundreds of locations; Google has tens of clusters; while others, like Microsoft, use a small number of primary sites (on the order of 3). We expect that newer multi-tenant clouds will also be composed of multiple sites, because of reliability concerns and because at least some tenants will be latency sensitive applications.

We simulate a wide range of geo-distributions and show that even systems with two clusters can benefit from PDR. From our analysis, we conclude that five or more *well-placed* clusters are enough to extract the maximum benefit from PDR.

Replication

Throughout most of this dissertation, we assume that the system is fully replicated at all its clusters. This assumption is reasonably accurate for large web services, such as search. In practice, however, systems tend to be only partially replicated, e.g.: user data is replicated at multiple clusters, but not at all clusters; so a request may

be serviced by more than one cluster, but not by any cluster. Partial replication constrains PDR's load redistribution choices and can dampen its usefulness. In chapter 8, we show how to augment the basic PDR optimization framework (chapter 2) to take partial replication into account.

Latency Considerations

Many Internet services strive to minimize request processing latencies. In order to extract meaningful savings, PDR may need to route clients to distant clusters, in search of less costly energy. Strict network latency requirements could severely restrict PDR's routing choices, rendering it ineffective.

Using simulation we explore the relationship between monetary savings from PDR and increases in network latency. We also survey the literature to understand the relationship between lost revenue and increased latency in web search and e-commerce applications. We conclude that PDR can achieve meaningful savings without increasing network latencies to the point where they become problematic.

Network Costs

A reduction in a system's electric bill may be overshadowed by an increase in its network usage costs. By redirecting traffic to regions with low energy costs, PDR may be unknowingly increasing the system's network costs by sending traffic to regions where bandwidth is expensive. There can be large differences between costs charged by different network providers at different locations, and sometimes by the same provider over time. Network costs often overshadow energy costs.

From the outset we describe how the PDR optimization can reason about tradeoffs between network costs and energy costs. We also discuss a common complex network billing model, 95/5 billing, and investigate it in simulation. We conclude that PDR should be able to reduce energy costs without increasing network costs in existing systems.

Cluster Cost Differentials

Even when one can route a system's entire power demand without any constraints, PDR may not yield any benefits. For PDR to be useful the cost of servicing a request must be different for different clusters. Then one can shift load from a higher cost cluster to a lower cost one, thereby reducing energy costs. The nature of the cost differentials between clusters is crucial in determining PDR's utility.

At the very least, costs should vary geographically. A time-invariant ordering of the clusters can be exploited by PDR, but instead of the dynamic optimization approach proposed in this dissertation, a static routing scheme would suffice.

Dynamic PDR yields the most benefit when costs vary in time and the variation is not well correlated for different locations. If the cost differential[4] for a pair of clusters has a zero mean, neither cluster is strictly better than the other. If the differential also has a high variance, there are many hours when one cluster is better than the other. By dynamically shifting power-demand to whichever cluster has the lowest cost, PDR can capture the value represented by the differential's variance.

We show that these sorts of cost differentials exist in practice. We study electricity price volatility in US wholesale electricity markets and show that cluster electric bills exhibit the sort of spatial and temporal variability that PDR can productively exploit. We also provide evidence that carbon emissions cost functions have these features.

Interfacing with Power Producers

Finally, we need to consider the interface between the distributed system and the entities producing its electricity. In this dissertation we focus on one particular interface, but also discuss some others.

For the most part we assume that the electric grid and the associated wholesale electricity markets provide the interface. The system operator observes settled prices in the regional markets for each cluster and modulates its power consumption accordingly. However, system operators often commit to fixed-price contracts (either directly with power plants, or with intermediary utilities). In this case, even when

[4] i.e., service cost at cluster A minus service cost at cluster B.

cost differentials exist, the system operators will not be exposed to them and so would not be able to use PDR to reduce costs.

Alternatively, instead of passively observing market prices, the system operator may be able to more actively participate in these markets. Generally, electricity prices are set using auctions, and a system using PDR could sell its ability to reduce consumption at a location. We also discuss interactive non-market interfaces. In particular, we discuss 'demand response' an existing discipline where grid operators send signals to consumers to request a temporary reductions in consumption.

1.3 Dissertation Structure

The first part of this dissertation, this chapter and the next, introduces power-demand routing. Chapter 2 formulates PDR as an optimization problem. It builds an optimization framework that combines traditional traffic engineering goals with our goal of intelligently modulating the spatial distribution of a system's power consumption.

The second part of this dissertation lays the groundwork for subsequent analysis. Chapter 3 develops energy models for server clusters, building empirical models that relate cluster load levels to cluster energy consumption. The chapter also proposes two new numerical metrics for energy proportionality. Chapter 4 discusses how we modeled user workloads and performance in our analysis. It details the traffic traces we collected at Akamai and the synthetic traffic workload models we built using them. It also discusses the relationship between system revenue and client-server latencies.

The third part of this dissertation addresses how one can use power-demand routing to cut the electric bills of Internet-scale systems. Chapter 5 makes the case that one can use PDR to exploit the uncorrelated price volatility that exists in US wholesale electricity markets. The chapter provides background on these markets, and details our empirical analysis of historical market price data. In chapter 6, we quantify the potential monetary savings achievable using PDR. Combining our cluster energy models and traffic workloads with the market price data, we quantitatively analyze the effectiveness of different PDR algorithms through simulation. In chapter

7, we qualitatively analyze what happens if we relax some earlier assumptions and explore some extensions. For instance, we show how our work is still relevant where wholesale markets do not exist, and extend the basic PDR framework to deal with partially replicated systems.

The fourth part of this dissertation, chapter 8, shows how PDR can be used to shrink carbon footprints. We show how to build environmental cost functions that can be plugged into our optimization framework and speculate on how effective this approach would be.

Finally, chapter 9 discusses related work, and chapter 10 is the conclusion.

Some figures use icons from *Pictoico* by Luka Pensa.

Chapter 2.

PDR: An Optimization Problem

In this chapter, we formalize power-demand routing as an optimization problem. We present the problem in a manner that is reminiscent of conventional traffic engineering formulations. Our contention—and one contribution of this chapter—is that a unified framework can be constructed that subsumes conventional network optimization goals and also understands the benefits of spatially redistributing the system's power consumption.

Given a large system composed of several server clusters, the optimization problem is to map client requests to clusters such that the total operating cost of the system is minimized. Cost may be defined in monetary terms, or in terms of the pollution generated, or some combination of multiple factors. Our framework is flexible, independent of the chosen cost function.

We start this chapter by building a mathematical model for the optimization (§2.1). We then discuss how to reconcile competing optimization objectives (performance, energy cost, and network cost; §2.2). We conclude by briefly exploring implementation strategies (§2.3). In later chapters we will describe concrete PDR algorithms that we have implemented and evaluated in simulation (chapter 6); and also show how to extend our optimization model to deal with partially replicated systems, multihomed clusters and complicated ISP network billing models (chapter 7).

We do not argue that our modeling approach is the best approach. Some of the goals of this chapter are to: (a) present an abstract high-level description of

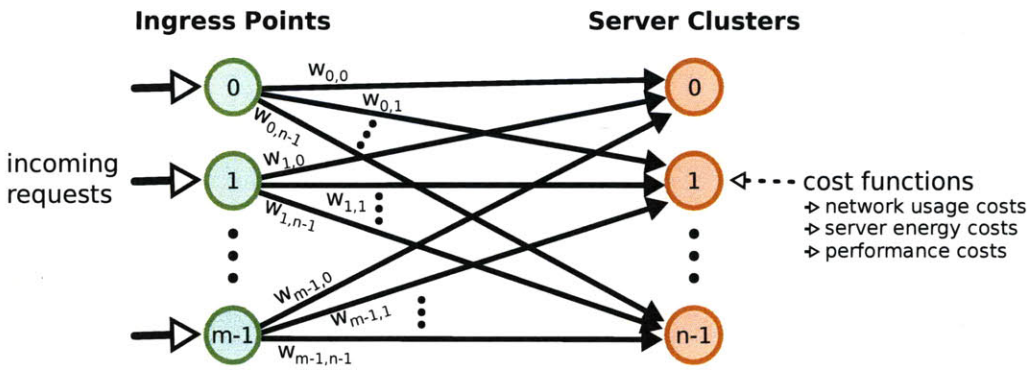


Figure 2-1: The PDR problem posed as a min-cost network flow problem.

the problem; (b) demonstrate by construction that PDR can be unified with other traffic engineering goals; and (c) identify how to deal with trade-offs between different aspects of operating cost (network, energy, and performance). Our approach is similar to the way in which others have recently tackled related problems [61, 80, 115]

2.1 Mathematical Model

We model power-demand routing as a minimum-cost network flow optimization problem on a bipartite graph. See figure 2-1. Client requests arrive at m network ingress points (the sources) and must be distributed among n server clusters (the sinks). The optimization process derives the *traffic splits* $w_{i,j}$ specifying what fraction of requests at ingress i should be sent to cluster j. There are constraints (server capacity, service-level-agreements, etc.) and the optimization must reconcile a number of competing objectives (server electricity costs, network usage costs, client performance, etc.).

Ingress points may be network prefixes, ISP's, or even geographic groupings (states and cities). We use clusters to represent co-located groups of servers and their support infrastructure. A cluster may be an entire data center, or a set of racks in a multi-tenant facility, or some other grouping.

We assume that the optimization takes place in a centralized location. Information about the system is collected at a single point, the optimal $w_{i,j}$ are derived, and then pushed out into the distributed system's request routing framework. The optimization is performed infrequently (e.g., hourly). Thus, even though efficiency is desirable

in implementing the optimization mechanism, complexity is not a strong concern. Hourly routing updates will be slow enough to be compatible with existing routing mechanisms, but fast enough to respond to electricity market fluctuations.

Time. We model PDR as a discrete time process. At the start of each timestep, the optimization is performed and the traffic splits $w_{i,j}$ are instantaneously derived and deployed. Instantaneous traffic optimization is a valid model when the time to derive and deploy the new routes is a small fraction of the timestep. In some of our later simulations we relax this assumption and analyze the impact of routing delays.

We model the optimization as a memoryless process with no foresight. Decisions made in one timestep are independent of decisions made in previous timesteps. The process does not have any knowledge (e.g., electricity prices and ingress traffic volumes) of the future (but may have access to expectations).

The memoryless model is not always an accurate representation. For example, memory is necessary to deal with 95-th percentile bandwidth cost functions (we address this in chapter 7). Memorylessness is, however, a good starting point and simplifies the discussion.

Uncertainty. In our discussion we assume that the optimization process has perfect knowledge of its inputs (e.g., ingress traffic volumes and electricity prices) at the start of the timestep. In practice, there will be some uncertainty, and we will need to use expected values for these inputs. Optimizing with expectations for inputs will minimize the expected total cost. Internet traffic exhibits predictable cyclical patterns (see chapter 4) and some of our simulations demonstrate that costs can be reduced even when very coarse price expectations are used. Therefore, omitting uncertainty from our model is acceptable. In contrast, we could have formulated the problem as a more complicated stochastic optimization problem.

Replication and Request Types. For simplicity, we assume that the service is fully replicated and that there is a single class of requests. Thus, any request can be sent to any cluster, and the network and server resources used at a cluster are

a function of the *number* of requests routed to that cluster[1]. These assumptions significantly simplify the optimization problem; in a later chapter we will consider the implications of relaxing these assumptions (chapter 7). For many services, like web search, these assumptions are true in practice.

Formal Specification

At each timestep, the optimization selects the traffic splits:

$$W = \{w_{i,j}\}$$

Where $w_{i,j}$ specifies what fraction of the request volume at ingress i should be sent to cluster j. We can write down the optimization problem for a timestep as:

$$\begin{aligned} \text{select} \quad & W \\ \text{to minimize} \quad & C \\ \text{given} \quad & (R, N, S) \\ \text{subject to} \quad & X \end{aligned}$$

Where C is a multi-dimensional cost function, R is the incoming request traffic during that timestep, N is the set of network ingress points, S is the set of server clusters, and X is a set of constraints. This is a multi-criteria optimization problem; instead of having a single numerical value, C is a set of real valued cost functions (one each for: energy, network and performance). In the text that follows, we describe each problem component in detail.

[1] This follows from the restriction that there is only one class of requests. In the simplest model, every request uses a fixed number of CPU cycles and a fixed number of packets. In a more realistic model, the number of CPU cycles needed to service a randomly selected request are a random sample from a distribution R. With large traffic volumes—millions to billions of requests per day—we can slice the traffic among different clusters, and the CPU usage distribution for each slice is still R. Consequently, a cluster's resource usage is a function of its traffic volume.

Network (edge labels)	
$\delta_{i,j} > 0$	Mean network latency (round-trip time, in ms) between ingress i and cluster j.
$\Delta_{i,j}(x) > 0$	x-th percentile network latency (round-trip time).
Cluster (node properties)	
$\alpha_j \rightarrow \mathbb{N}$	Cluster capacity: the maximum number of requests the cluster can service in one timestep, before it becomes overloaded.
$\varepsilon_j(r) > 0$	Energy (in kWh) used when r requests are serviced at j.
$\pi_j(r) > 0$	Mean processing delay: the average time, in ms, that would elapse between a request arriving at cluster j and a response exiting the cluster, when cluster load is r requests.
$\Pi_j(x, r) > 0$	x-th percentile processing delay ($\Pi_j(50, r)$ is the median delay).

Figure 2-2: Cluster and network properties, represented as node and edge labels.

Input Traffic. The incoming traffic is defined by the input variables r_i. Each r_i is a non-negative integer specifying the total number of requests arriving at ingress i during the timestep. Thus the r_i are the instantaneous request rates, evaluated at a timestep granularity. Our homogeneous request assumption allows us to simply count requests to characterize traffic.

Cluster and Network Properties. In modeling the distributed system, we restrict ourselves to the system properties needed to specify the constraints and cost functions used in our work.

We use a simple fully-connected network model, with all ingress points being connected to all clusters (figure 2-1). We assume that each of these network links is overprovisioned (common in backbone networks) and so do not model network link capacity constraints. For each ingress-to-cluster link, we model the network round-trip latency with two variables: the mean and x-th percentile latency (see figure 2-2). The homogeneous request assumption allows us to avoid more complicated packet-granularity latency models.

For clusters, we model their capacities, request processing delays, and energy consumption (see figure 2-2). A real system's servers are typically distributed in a non-uniform manner among different clusters. This can result in different clusters having very different properties. Additionally, a cluster's properties can vary in time, due to factors such as failures and dynamic resource scaling (turning off idle servers).

The energy model is the most complex part of our cluster model. A PDR implementation will need an integrated energy model, to evaluate its load distribution choices. In order to optimize, PDR must be able to predict how a cluster's energy consumption would change if the request load on it were increased or decreased. How close to optimal the resulting weights will be, will depend in part on the accuracy of the integrated energy model. We devote the next chapter to a discussion of how to build good energy models.

For complicated services, like web-search, a significant fraction of the overall request-response delay is due to delays within the cluster. As with network latency, we use two parameters to model processing delays (see figure 2-2). Unlike network delays, in our model cluster delays are a function of traffic volume.

Our constraints and cost functions rely on the *total request latency.* Average latency $\ell_{i,j}$ can be calculated by adding the cluster and network means. For percentile end-to-end latencies $L_{i,j}$, an upper-bound is the sum of the two components.

$$\ell_{i,j}(r) = \delta_{i,j} + \pi_j(r)$$

$$L_{i,j}(x, r) \leq \Delta_{i,j}(x) + \Pi_j(x, r)$$

Another useful definition is the request volume assigned to a cluster:

$$ra_j = \sum_{i=0}^{m-1} \left(w_{i,j} \cdot r_i\right)$$

Finally, note that the functions in figure 2-2 may be non-linear and have discontinuities (e.g., the energy models in chapter 3). This will limit what optimization methods we can use to solve the problem.

Constraints. We represent optimization constraints as invariants, specified in terms of the elements of W and R and the system properties introduced above.

Weights are real numbers between 0 and 1:

$$\forall i, j : 0 \leq w_{i,j} \leq 1$$

Traffic is conserved at each ingress:

$$\bigwedge_{i=0}^{m-1} \left(\sum_{j=0}^{n-1} w_{i,j} = 1 \right)$$

The number of requests routed to a cluster should not exceed its capacity:

$$\bigwedge_{j=0}^{n-1} \left(ra_j \leq \alpha_j \right)$$

Finally, there may be constraints on acceptable performance. For example, service-level-agreements (SLA's) may set upper bounds on acceptable request latency. SLA's are often alternatively described in terms of some high percentile of the latency distribution. An SLA specifying that 95-th percentile latency should not exceed D_{max} can be written as:

$$max\big(\{\forall i, j : L_{i,j}(95, ra_j)\}\big) \leq D_{max}$$

We do not discuss SLA's in this dissertation, but this shows that our framework can factor SLA's into the optimization.

Cost functions. The goal of the optimization is to select the weights that yield the minimum system operating *cost* at each timestep. A complication is that PDR optimization is a multi-criteria optimization problem: cost has components defined along multiple possibly independent dimensions. In our model, there are cost components for energy consumption, network usage, and system performance. We do not force the problem into the simpler single-objective domain, although, as we describe later, PDR implementations may be able to adapt single-objective optimization mechanisms. We describe the multiple, possibly competing, cost functions below.

Cost: Energy. The bulk of our work is concerned with characterizing and optimizing energy costs. A system's energy cost is the sum of all the cluster's energy costs[2]. Each cluster's energy cost is a non-decreasing function of the amount of electrical energy consumed by that cluster. If E_j is the energy cost function for cluster j, mapping energy consumption to cost, the system's energy cost C_E is:

$$C_E = \sum_{j=0}^{n-1} E_j\big(\varepsilon_j(ra_j)\big)$$

A system operator may choose to minimize the energy's monetary cost[3]:

$$E_j(x) = x \cdot eprice_j$$

Where $eprice_j$ is the price-per-kWh of electricity at the geographic location of cluster j. Electricity prices vary with location and in time (we discuss price variation in some detail in chapter 5). As a special case, when there is no geographic diversity and all prices are equal, this form of E_j will direct the optimization to attempt to minimize total energy consumption. Furthermore nonlinear variations of E_j are possible, but not discussed in this dissertation.

Alternatively, a system operator may choose to minimize the environmental impact of the energy being consumed (e.g., the dynamic carbon footprint). In this case the E_j can be expressed in pollution units (e.g., $kg\ CO_2$). The simplest approach is to use a linear function, as above, replacing $eprice_j$ with a time-varying pollution-per-kWh cost function (see chapter 8).

Note that even when optimizing for pollution, it is useful to express C_E in monetary terms. Expressing all costs in a common unit allows the optimization to account for trade-offs between energy, networking, and performance.

Cost: Network. Network usage costs are a significant component of the total operating costs of large distributed systems [61, 115]. We know that there can be

[2] As we show in chapter 3, we can safely ignore energy costs related to the network infrastructure.

[3] As defined here, E_j is the electricity *generation cost*, the variable component of the electric bill.

large differences between costs at different network locations, and sometimes on the same location over time. We define total network cost as the sum of the individual cluster network usage costs, and a cluster's network cost as a linear function of its request volume:

$$\begin{aligned} C_B &= \sum_{j=0}^{n-1} N_j(ra_j) \\ N_j(r) &= r \cdot nprice_j \end{aligned}$$

Even though this is a highly simplified model of reality, there is evidence that optimizing using a cost function such as this results in a reasonable approximation to a proper network cost optimization [115]. We discuss more realistic network cost functions later (chapter 7).

Cost: Performance. In our cost model, the third and final cost component is related to system performance. Degraded system performance is known to result in lost revenue (we study this in detail in chapter 4). We use end-to-end request latency as the metric of performance.

One way to factor performance into the optimization is to start by defining ℓ_{ave} the average end-to-end request latency across the entire system as[4]:

$$\ell_{ave} = \frac{1}{\sum_{i=0}^{m-1} r(i)} \sum_{i=0}^{m-1} \left(r(i) \cdot \sum_{j=0}^{n-1} \left(w_{i,j} \cdot \ell_{i,j}(ra_j) \right) \right)$$

The optimization can be directed to simply minimize average latency:

$$C_P = \ell_{ave}$$

A more realistic approach is for the system to have upper and lower bounds on latency. Reducing the latency below the lower bound yields no additional utility to

[4] We can also define $L_{ave}(x)$, the x-th percentile latency analogue, but note that $L_{ave}(95)$ is not the overall 95-th percentile latency. It is an upper-bound—the weighted average of the different clusters' $L_{i,j}(95, ra_j)$ values.

$$
\begin{aligned}
\text{SELECT:} \quad & W = \{w_{i,j} : 0 \leq i < m, 0 \leq j < n\} \\
\text{TO MINIMIZE:} \quad & C = \{C_E, C_N, C_P\} \\
& C_E = \sum_{j=0}^{n-1} \left(\varepsilon_j(ra_j) \cdot eprice_j\right) \\
& C_N = \sum_{j=0}^{n-1} \left(ra_j \cdot nprice_j\right) \\
& C_P = max(0, \ell_{ave} - D_{min}) \\
\text{SUBJECT TO:} & \\
& X1: \quad \forall i, j : 0 \leq w_{i,j} \leq 1 \\
& X2: \quad \bigwedge_{j=0}^{n-1} \left(\sum_{i=0}^{m-1} w_{i,j} = 1 \right) \\
& X3: \quad \bigwedge_{j=0}^{n-1} \left(ra_j \leq \alpha_j \right) \\
& X4: \quad \ell_{ave} \leq D_{max}
\end{aligned}
$$

Figure 2-3: PDR as an optimization problem.

the system operators; exceeding the upper bound is considered unsound.

C_P is expressed in millisecond units above, but it is useful to express C_P in monetary terms. Expressing all costs in a common unit allows the optimization to account for trade-offs. Monetary performance cost functions can be constructed by modeling the relationship between end-to-end latency and revenue loss (we explore this in chapter 4).

Summary. Figure 2-3 shows the various pieces of the model put together. In the interest of brevity, we have omitted the optimization's inputs (the graph description, the node and edge properties, and the r_i inputs).

2.2 Reconciling Competing Costs

In general, optimization problems with multidimensional cost functions are more complicated than problems in which the optimization criteria can be expressed as a function with a single numerical value. A large body of literature addresses how to optimize using multidimensional cost functions.

One complication is that multidimensional optimization problems do not always have dominant solutions—it is not guaranteed that there will be a choice of the $w_{i,j}$ that will simultaneously minimize all our cost components. The result of the optimization will be a *Pareto surface*: a set of possible solutions, such that moving from one Pareto solution to another Pareto solution will reduce cost along some dimension(s), but increase it along some other dimension(s). For example: we may be able to decrease energy cost by routing clients to cheap energy located far away from them, but this will increase the latency cost.

We describe two approaches that can be used to reconcile the competing optimization criteria in PDR. Both use techniques commonly applied to solving multi-criteria optimization problems. Each approach presents different advantages, so one is not strictly better than the other.

2.2.1 Aggregate Cost Function

A common way to map a multidimensional cost function down to a single dimension is to use a weighted sum of all the component costs. This is equivalent to converting all the components into the same units, using constant conversion coefficients.

To facilitate this, we recommend that all three component cost functions should always be expressed in monetary units. Reasoning about trade-offs between network usage costs and the cost of energy then becomes straightforward. Furthermore, the construction of a monetary performance cost function, forces a system operator using PDR to confront and be explicit about what are normally hidden preferences. Performance is not free; it should not be maximized at the expense of everything else. With this approach, operators must put a monetary value on each millisecond reduction in request latency. Similarly, when optimizing for the environmental cost of energy, this requires a monetary valuation of pollution.

2.2.2 Sequential Optimization

Another common approach is to rank the different component costs, and then to optimize for one component after the other. So, for instance, one could optimize first for performance. The minimum performance cost would then be treated as a constraint, as one further optimized for network cost. Finally, taking the minimum performance and network costs as constraints, one would then optimize for energy costs. This process can be relaxed slightly by, for example, weakening the performance constraint (e.g., performance cost should be no more than 10% higher than optimal).

An advantage of this approach is that it allows existing traffic engineering frameworks to be reused to implement PDR. Viewing PDR as a sequential optimization allows us to take existing frameworks and bolt an energy optimization module at the end of the existing optimization pipeline.

Traffic engineering frameworks that optimize for network costs and performance already exist. Companies like Akamai have developed complex proprietary solutions to this part of the optimization problem. Previous research in academia also addresses this area [61, 115].

A disadvantage of this approach is that it does not allow trade-offs to be properly considered. It is hard to specify, for example, that a $50ms$ elevation in average latency is acceptable only when it results in a substantial energy cost reduction.

2.3 Implementing Optimization Mechanisms

Having set up the optimization problem, we now turn to investigating optimization mechanisms that can be used to implement PDR. One reason we formulated PDR as an abstract optimization problem was to map it to a well understood domain, a domain that has been studied rigorously, and for which a large number of solutions have been proposed.

2.3.1 Linear Programming

Linear programming (LP) is an extensively used optimization technique with many known efficient solution methods. In order for PDR to be solved using LP methods, the various component cost functions must be linear functions of W. This limits them in certain ways:

Network: cost per request must be independent of a link's traffic volume.

Energy: cost per kWh must be independent of the total energy consumed by a cluster; and a cluster's consumption ε must be a linear function of the request volume allocated to it (ra_j).

Performance: end-to-end request latency on any network-cluster path must be independent of the request volume on that path.

LP is attractive because it has many computationally efficient solution methods. The downside is that restricting ourselves to LP may force us to approximate realistic costs and constraints. Linear energy models are reasonably accurate first approximations, but nonlinearities often exist (see chapter 3). A linear network cost model is a good starting point [115], but is not realistic in practice (see chapters 6 and 7). Latency independent of load is plausible if clusters and network links are overprovisioned, as they often are in practice.

2.3.2 Nonlinear Programming

In general, cost functions that are nonlinear or piecewise linear functions of W are more accurate representations of reality. Throughout this dissertation we present examples of nonlinear network, performance, and energy costs.

Computationally efficient nonlinear programming (NLP) methods can be used when the cost functions are convex. In fact some LP methods can be used to solve convex non-linear problems. With non-convex functions, far less efficient NLP methods must be used.

However, there are no guarantees that the cost functions used by PDR are convex (e.g., chapter 3 provides evidence that some types of clusters have non-convex ε energy functions). Another possible complication is that the cost functions may have discontinuities, and not be differentiable everywhere.

It is therefore plausible that the least efficient optimization methods may be necessary to derive the optimal PDR solutions. This is still a feasible implementation strategy. With tens of ingresses, tens of clusters, and routing updates sent out once every hour, there is enough time to find a good solution. In fact, the NLP algorithm we evaluate in this dissertation (chapter 6) is efficient enough to be used in practice.

Summary

In this chapter we presented an abstract description of the PDR problem. We formulated it as a traffic engineering optimization problem and outlined implementation strategies. We will extend this optimization model to deal with partial replication and other complications in chapter 7. An important element of our model is the cluster energy model. The following chapter is devoted to this aspect of the problem, and develops a portfolio of energy models.

Chapter 3.

Modeling Energy Consumption

The effectiveness of power-demand routing hinges on the *energy proportionality* of the individual server clusters that constitute the system. Energy proportionality is the degree to which the electrical energy consumed by a cluster depends on the traffic load placed on it. Ideally, an idle cluster would use no energy. In the worst case, there would be no difference between a cluster's peak and the idle energy consumption.

It is worth noting that we do not make a strong distinction between energy and power here. When we speak of *power*, we are speaking of *average power* over reasonably long time windows (e.g., 5 minutes), rather than of *instantaneous power*. Modeling instantaneous power is a harder problem, and unnecessary for our work.

In the absence of adequate energy proportionality, one cannot productively route a system's power-demand between different clusters. Recall that PDR works by spatially redistributing a system's load. If sufficient energy proportionality exists, load redistribution will result in a meaningful spatial redistribution of the system's energy consumption. In contrast, a system without proportionality is forced to always consume energy everywhere, even in idle regions.

Therefore, to understand the potential of PDR, we must investigate and model the energy characteristics of real clusters. As we noted in chapter 2, an energy model will also be needed by a PDR implementation, to guide the optimization process. Our models must account both for the energy consumed by servers and the energy consumed by the data center infrastructure supporting those servers.

The bulk of this chapter explores the relationship between cluster power consumption and cluster utilization. As in the rest of the dissertation, our approach is heavily empirical. We outline the way in which electricity is consumed within modern data centers and then build mathematical models that describe how cluster power consumption depends on utilization.

We define cluster *utilization* abstractly, as a real number between 0.0 (idle cluster) and 1.0 (operating at maximum capacity). If a cluster can service a peak request rate of R and is faced with a rate r, then utilization u can be thought of as the normalized rate $\frac{r}{R}$. As in the case of power, u is an average and not the instantaneous utilization.

We concentrate on clusters that are entire data centers, but we also consider smaller clusters in multi-tenant facilities. Real systems are typically built using a mixture of different kinds of clusters in different locations.

The models we develop in this chapter improve upon previous work on modeling data center energy consumption [56, 93]. Previously proposed models do not address the design diversity that exists nor do they account for new techniques like dynamic server scaling [1]. We therefore extend those models and construct some of our own. Rather than developing a single unified model, we approach the problem from a number of different angles, cover several design choices, and synthesize a collection of cluster models. Since data centers are highly complex and evolving systems, we must abstract away many architectural details. We also conduct an extensive survey of industry literature and collect energy efficiency reports for existing facilities and projections for future technologies. From this survey we derive model parameters that we can use to estimate the effectiveness of PDR in the context of current and future systems.

Model diversity also serves to improve the believability of our results. It shows that our final analytical results are not brittle—that our results are not sensitive to an idiosyncrasy of a particular model (e.g., linearity). Diversity also allows us to determine which aspects of the clusters' power-vs-utilization curves most influence the effectiveness of PDR. This will tell us what sorts of real architectures can best exploit power-demand routing.

This chapter introduces two novel energy proportionality metrics. Given a cluster type's energy-vs-utilization curve, we can calculate a numerical score for each metric. We show in chapter 6 that these scores are very good predictors of how well PDR will perform for clusters of that type.

The chapter concludes with a discussion of the impact of PDR on the energy consumption of wide-area networks. The optimization framework proposed in chapter 2 ignores energy consumption within the network. We show that when PDR operates under reasonable constraints, changes in the spatial distribution of wide-area network power will be many orders of magnitude smaller than changes in cluster power. We can, therefore, safely ignore network power.

3.1 Electricity Consumption in Data Centers

Understanding and modeling data center power consumption is a complicated undertaking. Individual data centers are highly complex systems, with a number of interacting mechanical, electrical and computational sub-systems. The power consumed by the cooling system, for example, can depend both on the nature of the airflow in the server room and on the server load balancing algorithms being used. Additionally, the data center design space is evolving rapidly. For instance, monolithic data center designs are giving way to newer highly modular container based designs [44, 84, 5]; and many new cooling designs have emerged in recent years [62, 60, 41, 58].

Our modeling and analysis is focused on the kinds of data centers built to house servers for geo-distributed replicated systems (e.g., the tier II facilities[1] used by Google). Enterprise data centers (e.g., tier IV facilities), such as those used by banks, are more complex, and their behaviour may deviate slightly from our models.

Our approach is to focus on the three data center sub-systems that typically account for almost all of the power consumption, outline how they consume electricity,

[1] Data centers are commonly classified according to a four tier reliability standard [23]. The higher the tier, the higher the data center's availability. Tier II facilities have some redundant components and promise up to 99.74% annual uptime; tier IV facilities have redundant power distribution and cooling systems and promise over 99.99% annual uptime.

and describe the mathematical formulas we can use to model the relationship between each sub-system's electricity consumption and the data center's utilization level.

The three sub-systems on which we focus are:

- IT equipment (servers, storage and networking hardware)
- Cooling (chillers and fans)
- Power distribution (UPS's, PDU's, etc.)

We assume that other sub-systems, such as lighting, draw a small ($\approx$1%) and fixed amount of power, independent of utilization levels. This approach builds on work done by others. In particular, we draw on reports from industry practitioners [56, 68, 64], other academics who have studied data center power consumption [93], and empirical data from data center component manufacturers and the EPA.

PUE. Data center operators commonly disclose Power Usage Effectiveness (PUE) numbers for their facilities. PUE is an industry standard metric for data center energy efficiency. In its simplest form[2]:

$$\text{PUE} = \frac{\text{Total Power Entering Data Center}}{\text{Power Consumed by IT equipment}}$$

Thus a PUE of 2.0 implies that, on average, for every watt drawn by a server, an additional watt is consumed by support systems (cooling, power distribution, etc).

Figure 3-1 tabulates PUE numbers, compiled from a number of different sources[3]. The fraction of total power used by IT is easy to compute, given the PUE. Some of these reports also provide other details that allow us to estimate the fraction of total power spent on cooling and the fraction spent on power distribution overheads.

The general trend shows steadily improving energy efficiency, with decreasing marginal improvements below PUE's of 1.3. The electricity consumed by the IT equipment represents the useful work done in the data center. In legacy data centers,

[2] PUE is not a static value, so it must be averaged in some way, and there will be some measurement uncertainty that should be accounted for (e.g., see [62]).

[3] Since not all reports specify how PUE was averaged, we treat every reported number as a PUE calculated when IT power consumption is at its maximum.

Description	Year	PUE	IT	C	PD
Legacy data center [21]	2008	2.5	40%		
Industry average [85, 25]	2008	2.0	50%	37%	10%
EPA (current-trends target) [25]	2011	1.9*	53%		
Efficient data center [64]	2009	1.7	59%	29%	11%
Advanced data center [68]	2010	1.5	67%	21%	11%
EBay (tier IV, Utah) [91]	2010	1.4	71%		
EPA (best-practices target) [25]	2011	1.3*	77%		
Sun Micro. (Santa Clara) [85]	2008	1.28	78%	16%	4%
Microsoft (Containers/Illinois) [84]	2009*	1.22	82%		
Google (average, 6 locations) [62]	2008	1.21	83%		
Google (average, 6 locations) [62, 83]	2010	1.18	85%	10%	4%
Yahoo (Coop/Lockport) [58]	2010	1.1*	91%		

Figure 3-1: Reported PUE values and the years for which they were reported. The table is sorted by PUE. Where available, we have also listed the fraction of total power used by the cooling (C) and power distribution (PD) sub-systems. Most PUE's were calculated from measurements; an asterisk indicates an estimate.

IT power is less than 50% of the total. However, strong incentives exist to minimize the power consumption of any other sub-systems, and, as figure 3-1 shows, in today's cutting edge data centers, IT power can be more than 90% of the total. Note how advances in cooling efficiency have been the main factor driving down PUE's.

The three sections that follow discuss the IT, cooling and power distribution sub-systems in detail. In summary: we can model IT power as either a roughly linear function of utilization, or as a piecewise linear function; and we can model infrastructure power as either fixed, or, more accurately, as a polynomial function of utilization.

We then combine the sub-system models to build complete cluster models. During the sub-system discussion, consider a cluster to be an entire data center. We will return to the more general view of a cluster when we describe the cluster models.

3.2 IT Equipment Energy

We first describe an empirically verified IT power model proposed by Google. Since Google has a history of building atypical cluster architectures, we also describe an alternative modeling approach that composes individual server power curves provided

by server manufacturers.

3.2.1 Google Cluster Model

Researchers at Google recently described IT power models based on a study of servers in their data centers [56]. The study measured and analyzed the power-vs-utilization characteristics of their clusters—thousands of servers and associated network switches—using realistic workloads and microbenchmarks. Google's study is unique: a number of other studies have used empirical data to characterize the power-vs-utilization characteristics of IT equipment, but few of them have used data from actual data centers.

The study concluded that IT power consumption can be approximated with reasonable accuracy using a simple linear model. Idle clusters consume large amounts of electricity (more than half their peak power in the studied clusters) and electricity consumption rises, roughly linearly, with utilization[4] u. Thus:

$$P_{IT}(u) = P_{other} + P_{idle} + (P_{peak} - P_{idle}) \cdot u$$

Where P_{idle} is the total power consumed by the servers when they are all idle; P_{peak} is the total power when all servers are at peak load; and P_{other} is the power consumed by IT equipment other than servers (e.g., network switches). P_{idle} and P_{peak} depend on the specific server hardware used and the number of servers in the cluster[5], and so can vary from cluster to cluster.

A nonlinear model was also described:

$$P_{IT}(u) = P_{other} + P_{idle} + (P_{peak} - P_{idle}) \cdot (2u - u^{1.4})$$

This model was the result of fitting a curve to their empirical data. For groups of 20-60 racks, this model was able to match the measured power with less than 1% error (using average CPU utilization for u).

[4] We use abstract utilization; Google's paper uses average CPU utilization.

[5] For a cluster with n identical servers: $P_{peak} = n \cdot P_{server\ peak}$.

We model non-server components like networking hardware as having a fixed power draw. Unlike servers, these components tend to operate within narrow dynamic power ranges [56, 48]. Networking hardware in the data center typically accounts for less than 6% of the IT power [68]. If f_{net} is the fraction of IT power used by non-server equipment, then:

$$P_{other} = f_{net} \cdot P_{IT}(1) = \left(\frac{f_{net}}{1 - f_{net}}\right) \cdot P_{peak}$$

Few architectural details were disclosed about the clusters in the study. For instance, it is unclear how cluster load was distributed across different machines. We know from other sources that Google's clusters mix together different classes of server hardware and that idle servers were left powered on. It is not immediately clear whether Google's model will apply to other data centers or not.

3.2.2 Alternative Component-Based Models

We therefore model individual components (servers and network switches) and thereby build up a set of alternative IT power models. These models have not been empirically verified in the same way that Google's has been. However, these models are independent of Google's architectural idiosyncrasies. Furthermore, these models capture the effects of features such as load skewing and dynamic server scaling (turning idle servers off).

We find that linear models are reasonably good approximations for a wide-variety of cluster types. In fact, with aggressive server load skewing and/or fine-grained dynamic server scaling, the power-vs-utilization curves are almost perfectly linear, for large clusters. However, we also find that some cluster implementation choices (e.g., coarse-grained server scaling) lead to significant nonlinearities and discontinuities.

Modeling Preliminaries. We divide the IT equipment into two categories: servers and storage systems; and networking hardware. In the discussion that follows, we assume, for simplicity, that the collection of servers consists of homogeneous hardware and that storage modules are simply servers that have been assigned special roles.

We will briefly consider the implications of relaxing our homogeneity assumption. Heterogeneous hardware deployments are common. For instance, newly deployed servers can coexist with older hardware that has not been phased out. Further, storage modules may be specialized hardware, such as NAS boxes. Industry estimates put storage devices at somewhere between 8% [77] and 27% [85] of total IT power.

We assume that all servers in a cluster are interchangeable, i.e. incoming request can be handled by any server with spare capacity. Interchangeability may not be true in practice, since service data may not be replicated evenly across all servers. Without assuming interchangeability, we would need to develop a storage replication model, adding substantial complexity to our models. For services like web-search, interchangeability is a reasonable assumption.

Finally, as before, we model networking hardware as having constant power consumption, about 5% of total IT power. Networking hardware in the data center typically accounts for less than 6% of the IT power [68], possibly less than 4% [85], and even high estimates do not place it above 10% [77]. Studies have shown that the power consumed by routers is largely independent of utilization; the power consumption of an idle router can be 97% its peak power [48] or higher.

Thus, we model clusters as collections of homogeneous interchangeable servers with some fixed power overhead due to networking equipment. To generate cluster power-vs-utilization curves we spread incoming requests over these servers (we study different load distribution policies) and model the power used by each server.

Individual Servers. Individual servers are known to consume power proportional to their utilization. Although linear approximations are plausible, the exact relationship tends to be be complicated and varies depending on server hardware details and on features of the service (e.g., disk intensive or CPU intensive).

Our server models are based on power measurements of actual server hardware. Figure 3-2 shows power-vs-utilization curves for several different server-class machine types. Idle server power ranges from about 20% of peak, to almost 80% and there is a variety of curve shapes: some are almost linear (e.g., IBM X3250), others are convex

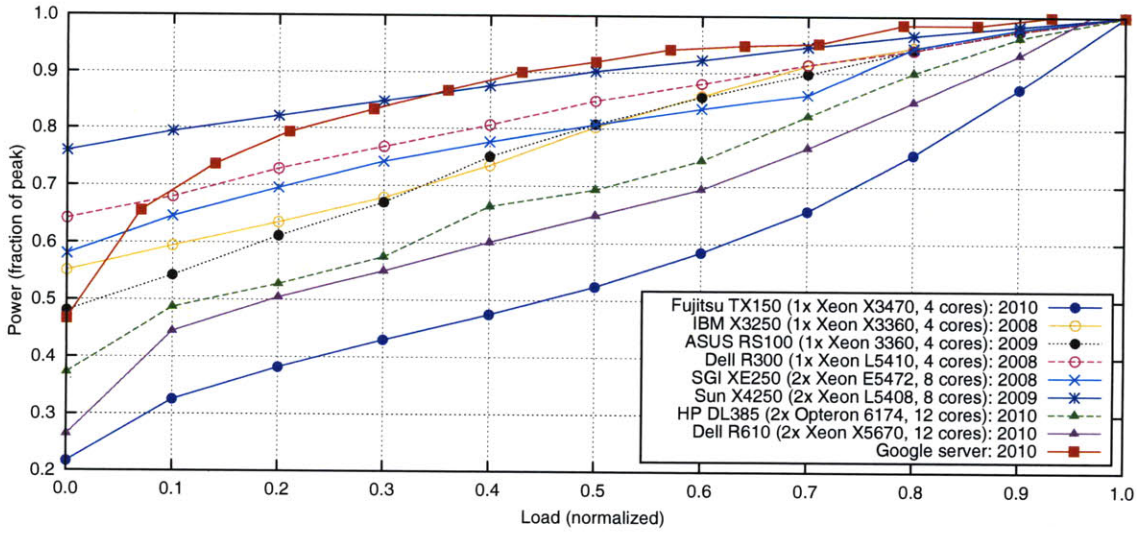


Figure 3-2: Server power curves can have a variety of shapes. To keep the focus on the shape, the power for each machine type is given as a fraction of its peak power in this figure. The curves are constructed from SPECpower measurement reports [19], and a disclosure by Google [44]. We linearly interpolate between reported measurements.

(e.g., Fujitsu TX150), while others are concave (e.g., Google).

The Google power curve is based on disclosures [44, 56] about a 'recent' Google server (no hardware or workload details were provided). The other power curves are constructed using data from published SPECpower benchmarks [19]. The SPECpower benchmark is designed to simulate how efficiently a machine can run a typical business application on an enterprise Java platform[6].

The curves do not illustrate the differences in server processing capacity or absolute power consumption. Figure 3-3 tabulates power and performance characteristics for some machines. In particular, note that even though the IBM and SGI machines had similarly shaped power-vs-utilization curves, the SGI machine has twice as many cores and consumes 2.5× as much power. Furthermore, comparing the Fujitsu and HP machines in the table, we can see an order of magnitude performance gain, with less than an 8× increase in power.

Generally speaking, server power proportionality has been improving, but some server components are still not proportional. Individual processor cores can use fre-

[6] Specifically: we use `SPECpower_ssj2008` numbers. Refer to its measurement methodology to understand the details of what is being measured.

Machine	CPU	Cores	Disk	Rate	Power		
					peak	$\frac{1}{2}$-load	idle
Fujitsu TX150	Xeon	4	HDD	0.08	112W	52%	22%
IBM X3250	Xeon	4	HDD	0.06	127W	80%	55%
SGI XE250	Xeon	8	HDD	0.10	322W	81%	58%
Dell R610	Xeon	12	SSD	0.27	244W	63%	26%
HP DL170h	Xeon	48	SSD	1.00	883W	63%	24%
Google	-	*4	HDD	-	145W	92%	47%
SeaMicro SM10K	Atom	512	-	≈6*	*2kW	*55%	*10%

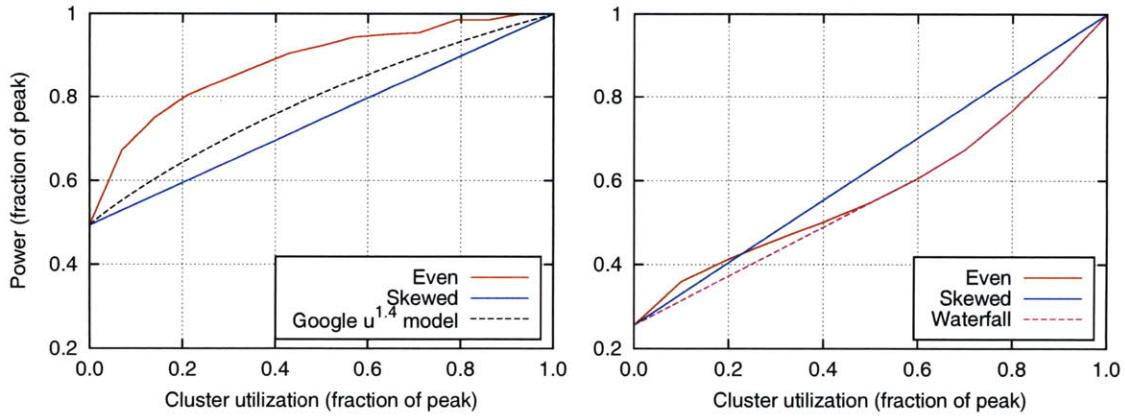
Figure 3-3: Server power measurements. The rate is the normalized transaction rate reported for that machine type. The data is from SPECpower measurement reports [19], from public disclosures by Google [56, 44] and a SeaMicro press release [101]. Asterisks indicate estimates.

quency scaling and shift into low-power states to modulate their power in response to load. The power used by disks tends to be relatively independent of load (although disks can be spun down, disks on Internet servers rarely are, because of reliability and performance concerns). The power consumed by other server components, such as fans and power supplies, is also relatively independent of server load.

The SeaMicro server [101] is a recently announced server architecture for cloud applications, specifically designed to minimize power consumption without sacrificing performance. SeaMicro has developed a novel hardware architecture that can pack a high density of Intel Atom processors into conventional data center racks. These servers have not yet shipped, so the power-performance characteristics in figure 3-3 are estimates extrapolated from SeaMicro disclosures. For our purposes, SeaMicro is important because it demonstrates that newer server architectures can be much more proportional than conventional servers.

It is reasonable to project that future servers will achieve less than 10% idle server power consumption. Conversely, measurements at Akamai, conducted by our collaborators, demonstrated that idle power can be 95% of peak, if hardware power management features are disabled by server operators (as they sometimes are, due to performance concerns).

We can model server power using the curves from figure 3-2 and aggregate to



(a) Google servers **(b) Fujitsu TX150 servers**

Figure 3-4: IT power curves for simulated 1000-server clusters, comparing different load distribution policies and different machine types. We assume that 5% of the peak IT power is accounted for by network switches and other equipment.

construct cluster models. However, we must first account for how load is distributed across servers in the cluster.

Server Load Distribution Policy. It has been noted some time ago that the power consumption characteristics of a cluster are related to the way in which load is spread across its servers [49]. Skewing load to concentrate it on the smallest number of servers, maximizes the number of idle servers, minimizing cluster power if server power curves are almost linear. Spreading load evenly, on the other hand, tends to result in better performance (e.g., lower queuing latencies and improved reliability).

Figure 3-4 shows how different load distribution policies can impact a cluster's power consumption at different utilization levels. To generate these curves: we modeled a collection of 1000 identical machines; assumed that the load on a cluster was composed of infinitesimally small constant-complexity transactions that could be arbitrarily assigned to any of the servers; assumed that transactions arrived at a constant rate; and then simulated different load balancing policies.

Because of our infinitesimal transaction model, evenly spreading the load results in cluster power curves that are scaled versions of the machine power curves. Our version of load skewing, sequentially saturates one machine with transactions, then moves on to the next one. There are a large number of machines, so a load-skewed

cluster's power-utilization curve looks perfectly linear at this scale. Note that the load distribution policy that results in optimal power depends on the shape of the machine power curve. In 3-4a skewing is optimal; in 3-4b even spreading tends to be better than skewing, but neither is optimal[7].

In practice, clusters typically try to spread load evenly, so cluster power curves will have shapes similar to the server curves (figure 3-2)[8]. In contrast, we see that for clusters with aggressive load skewing, the linear approximation is highly accurate, even when the server curves are not linear.

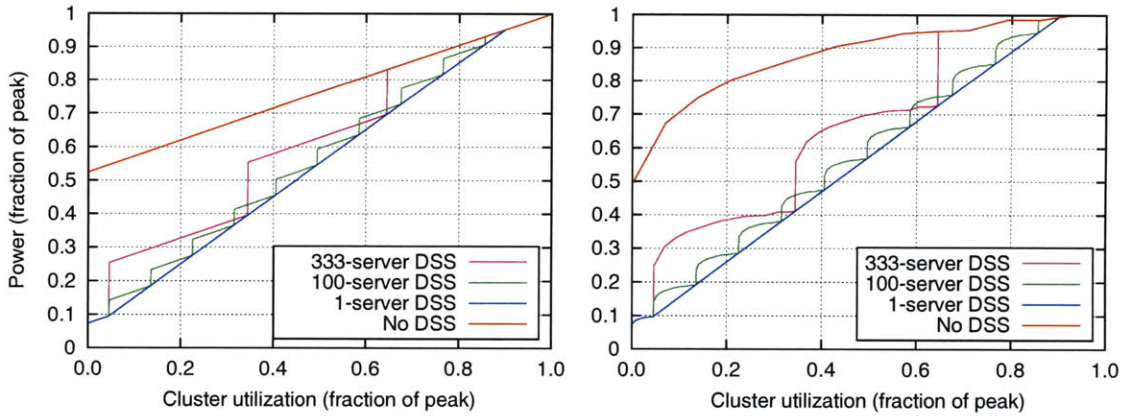
Dynamic Server Scaling (DSS). Contemporary clusters typically leave idle servers powered on. Since idle servers consume a significant amount of energy—in some cases, well over half of their peak power—turning off idle servers is desirable.

On the other hand, turning off servers can adversely impact performance and, possibly, reliability. These potential downsides have long inhibited the widespread adoption of DSS. Servers can take a few minutes to power cycle, so if a flood of client requests arrives after a lull, clients may either experience failures or elevated response times. Furthermore, there is a common belief among system operators that stopping and starting servers and disks will increase hardware failures.

Even though DSS is not widespread today, future clusters are likely to implement some sort of DSS. Most of the time, actual clusters operate at low utilization levels [44]. Because of this, DSS can dramatically reduce the average power consumption of a cluster. Both academics [50] and industry practitioners [1, 15] are developing DSS techniques. Implementing DSS without impacting performance is not trivial. To do so, a cluster must be able to predict its near-future workloads with accuracy. Fortunately, Internet service workloads tend to be cyclical with highly predictable daily minima and base-load levels [18, 85]. Many web applications deployed using

[7] In the figure we have used an approximation to the optimal policy, a greedy *waterfall* load distribution approach: start with $u_{thresh} = 0.6$; assign requests to a machine until its utilization reaches u_{thresh}, then move on to the next machine; when no machines with $u < u_{thresh}$ exist, add 0.05 to u_{thresh} and start over.

[8] When there are several classes of of requests or data is not replicated evenly across the servers, the power curves can deviate significantly from our ideal models. We discuss this later.



(a) Linear servers (50% power at idle) **(b) Google servers**

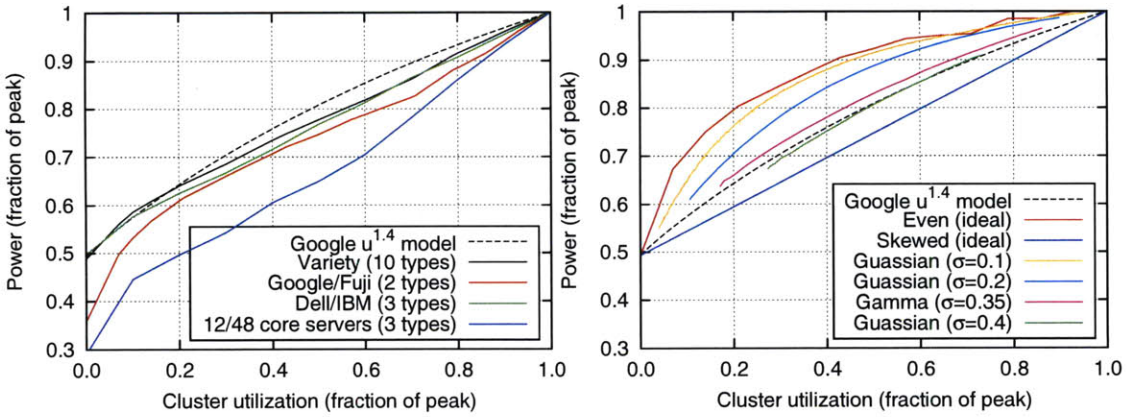
Figure 3-5: IT power curves for two types of simulated 1000-machine clusters with DSS, using a safety margin of 50 servers that are always on.

Amazon's elastic compute cloud have successfully implemented DSS-like logic (for virtual machines rather than for physical machines).

A cluster with DSS will have a lower idle power than a similarly sized cluster without DSS. However, the DSS cluster's power-vs-utilization curve can have significant discontinuities—transition points where blocks of servers are powered on in response to rising load.

We model a basic DSS algorithm. Load is initially spread evenly across servers. When load on the cluster rises to the point that average server utilization exceeds some threshold (e.g., 90%) an additional block of servers is powered on. A block can be a single server, or a larger group (racks, aisles, containers, etc). Load is skewed such that the old servers remain at high utilization and excess requests are spread evenly across the new servers. Lightly loading the new servers makes it easier to turn them off when the load on the cluster drops. The algorithm always maintains a safety buffer—some number of extra servers that are always kept powered on.

Figure 3-5 shows the power-vs-utilization curves for models with this DSS algorithm. The figure shows results for two kinds of simulated clusters: one built from machines with linear power-utilization curves, and another built from machines with the nonlinear Google curves from figure 3-2. We see that when DSS operates on a per-server granularity, the power curves appear almost linear, even with nonlinear



(a) Mixed Hardware Clusters **(b) Noisy Server Utilization**

Figure 3-6: IT power curves when we relax server homogeneity assumptions. In (a) different kinds of machines are mixed together to simulate 4800-core clusters; there are twice as many 4-core machines as 8-core machines, etc.; and load is spread evenly. In (b) a homogeneous 1000-machine cluster of Google-class machines experiences uneven load balancing, causing individual servers to deviate from mean utilization.

machine curves. We also see discontinuous jumps in the power curves with larger DSS block sizes. Finally, we see that even coarse-grained DSS can dramatically improve the energy-efficiency of a cluster at low and moderate loads.

Heterogeneous Server Deployments. Before concluding this IT power modeling section, we briefly consider what happens when we relax our server homogeneity assumption. Recall that we assumed all servers had the same hardware type and were interchangeable so that requests could be sent to any server.

Figure 3-6a shows power-vs-utilization curves for clusters in which different hardware types are mixed. Note in particular that when we simulate a cluster with ten different machine types, the resulting power curve is very similar to the IT power model proposed in the Google study (§3.2.1).

Figure 3-6b illustrates what happens when servers are not interchangeable, but the cluster tries to spread load evenly when it can. When load is spread perfectly evenly, every server's utilization will match the cluster's utilization. We model a lack of interchangeability by adding noise to server utilization, causing individual servers to deviate from the cluster's utilization. The figure shows the results of several Monte Carlo simulations that used different noise functions. Generally speaking, increasing

the variance in the server utilization resulted in a flatter power curve. The simulations with the highest variance resulted in power curves that almost overlap the power model from the Google study (§3.2.1). This overlap will not necessarily occur for other machine types.

Component Model Conclusion. We have seen in this section that there are many plausible architectures whose predicted IT power curves are very different from the earlier IT power models proposed by Google researchers (e.g., some server hardware types, server-rack-granularity DSS, etc.). However we have also seen evidence—independent of the Google study—that nearly linear models are good approximations for IT power.

3.3 Cooling System Energy

The electricity entering a data center is dissipated as a large amount of heat that must be evacuated from the building. Computer Room Air Conditioners (CRAC's) and fans are used to remove hot air from servers on the data center floor and bring in fresh cooler air. Conventional CRAC's transfer heat from the air to a fluid coolant (e.g., water) that is then pumped to large chillers or cooling towers in another part of the facility. The heat is expelled into the external atmosphere and the cooled fluid is circulated back to the CRAC's.

Generally speaking, the cooling system's electricity consumption increases with the amount of heat it needs to evacuate. Modern data centers use variable speed drive chillers and variable speed fans giving their cooling systems a large dynamic range.

At peak load, CRAC's and chillers can account for more than a third of a data center's peak electricity consumption (see figure 3-1). Chillers are the dominant consumers in the cooling system; they can require more than three times as much power as the other cooling components[9].

[9] Some reported $\frac{chiller\ power}{CRAC\ power}$ values: 3.7 [102], 3.2 [55], 2.5 [55], 1.5 [55], 1.1 [39].

Newer data centers try to minimize the time chillers need to run by exploiting low outside temperatures. For instance, *water-side economizers* use evaporative cooling—where the hot water from the CRAC units is cooled by being allowed to evaporate inside a cooling tower [62, 60]. In cold climates, glycol based radiators can be mounted outside the building to dissipate heat [44]. Additionally, if atmospheric air is cool and relatively dry, *air-side economizers* can be used to pump outside air directly into the server rooms [41].

We model the cooling system's energy as a function of the thermal load placed on it, but in reality its energy consumption depends on a number of variables that we are assuming remain fixed (e.g., coolant temperature, external air temperature, server-room airflow and the nature of hotspots). Note that the thermal load can be expressed as a function of the IT power $P_{IT}(u)$ and so is a function of utilization u.

The sections below develop CRAC and chiller energy models. In our modeling we will ignore energy consumed by other components like pumps and humidifiers[10]. For simplicity, we assume that the behaviour of in-rack and on-server fans has already been accounted for (in the IT equipment model). Furthermore, if there are multiple CRAC's and chillers, we assume thermal load is distributed perfectly evenly across them. This allows us to avoid having to model how many units are used in the data center; we can use a single-chiller power model and scale the output by the fraction of cluster power used by all chillers. Similarly, redundant cooling designs (e.g., N+1 chillers) could be modeled by scaling cooling capacity beyond the peak heat generated by the IT equipment.

CRAC units. The energy consumed by the CRAC units tends to be dominated by fan power [93]. Generally speaking, the power consumption of a variable speed fan is a cubic function of its rotation speed [87]. However, as fans speed up, the heat transfer efficiency of the cooling system may improve, somewhat offsetting the effects of this cubic growth [93].

The exact relationship between fan speed and the quantity of heat being evacuated

[10] These represent a relatively small fraction of total energy [102].

from a server room is complicated. Clearly, fans will need to speed up when the heat energy being dissipated by servers increases, and can slow down when servers are cool. Since the CRAC must prevent servers from overheating, fan speed is related to the *hottest* server inlet temperatures in the room, rather than average temperatures. Furthermore, complex air flows can arise, depending on the physical characteristics of the facility. Data center designers typically use techniques such as computational fluid dynamics to arrange server racks and CRAC units to maximize efficiency [87].

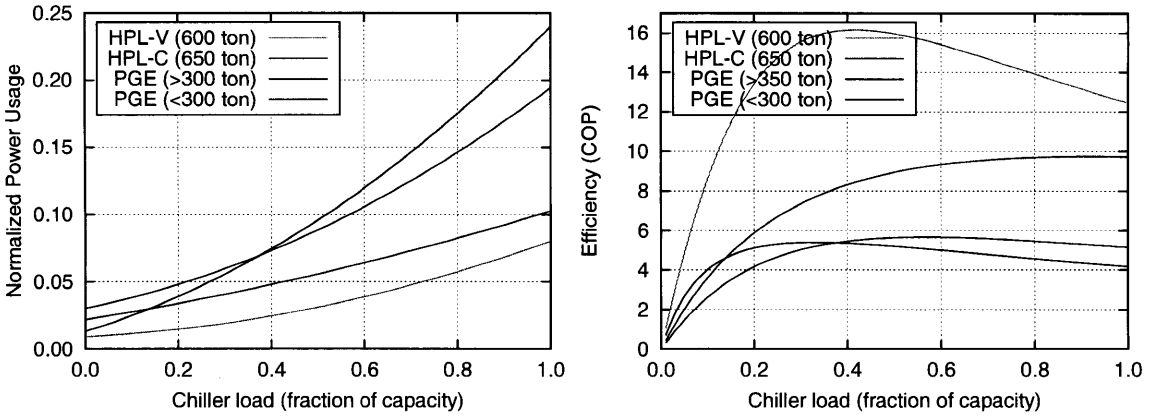
We choose a model that abstracts away most of these details [93]:

$$P_{CRAC}(u_{CRAC}) = \alpha_{AC} + \beta_{AC} \cdot u_{CRAC}^{2.8}$$

Where u_{CRAC} is the utilization of the CRAC (i.e. the amount of heat it is evacuating, relative to its cooling capacity); and α_{AC} and β_{AC} are constants. This model assumes a simple airflow model, that unbalanced hotspots do not arise, and that the CRAC is connected to a constant temperature heat sink. Well designed CRAC's can have very low idle power, relative to their peak power. A realistic set of parameters is $\alpha_{AC} = 0.1kW$ and $\beta_{AC} = 2.9kW$ [93].

Chillers. Chillers extract heat from a coolant fluid and return that fluid to the CRAC units. Among other factors, chiller power consumption depends on the amount of heat extracted and on the selected return temperature for the fluid. A number of different types of chillers exist, with a variety of compressor types, and variable or constant drives. In each of these cases, we can model chiller power consumption as a quadratic function of the thermal load placed on it [93, 29, 45].

The HVAC and data center communities have developed chiller power models that we use directly. We use a set of chilled water plant models specified by the California utility Pacific Gas and Electric (PG&E) [29] as energy efficiency guidelines for data centers. The PG&E models are for entire chilled water plants (chillers, cooling towers, pumps, etc.). Additionally, we use a set of steady-state power models developed by researchers at HP labs [45], using empirical data collected from operational data center chillers and cooling towers. The HP labs models include models for variable



(a) Chiller power consumption **(b) Chiller efficiency**

Figure 3-7: We model a chiller's power consumption as a quadratic function of the thermal load placed on it. Efficiency can vary dramatically between different kinds of chillers. In these figures, each chiller's power has been normalized to its heat removal capacity.

Chiller	Capacity	@ Full load	Power model
HP (variable)	600 ton	0.28 kW/ton	$P(x) = 0.022 + 0.055x + 0.026x^2$
HP (constant)	650 ton	0.36 kW/ton	$P(x) = 0.009 + 0.017x + 0.054x^2$
PG&E (A)	$\geq$300 ton	0.68 kW/ton	$P(x) = 0.030 + 0.069x + 0.094x^2$
PG&E (B)	<300 ton	0.85 kW/ton	$P(x) = 0.013 + 0.104x + 0.122x^2$

Figure 3-8: Power consumption models for several chillers. The equations map thermal load (0.0-1.0) to chiller plant power usage. Usage is defined as a fraction of the chiller's heat removal capacity.

speed chiller plants (HPL-V) and constant speed chiller plants (HPL-C).

Figure 3-7 shows how chiller power consumption varies with load in each of these models. Figure 3-8 shows the associated model equations. The HPL models are more efficient either because of their larger sizes, or because they use a different coolant fluid temperature setting.

Economizers. In this dissertation, we do not model the use of fluid and air economizers. There is a simple way to augment our models to factor in economizers. We can add an instantaneous step function that drives chiller power down to zero (for an air economizer) or cuts it in half (for a fluid-based economizer [11]), when the external atmospheric temperature falls below a certain point. To properly model air and fluid economizers, we would have to factor in atmospheric humidity and the effects of variable atmospheric temperatures on fan and cooling tower efficiency.

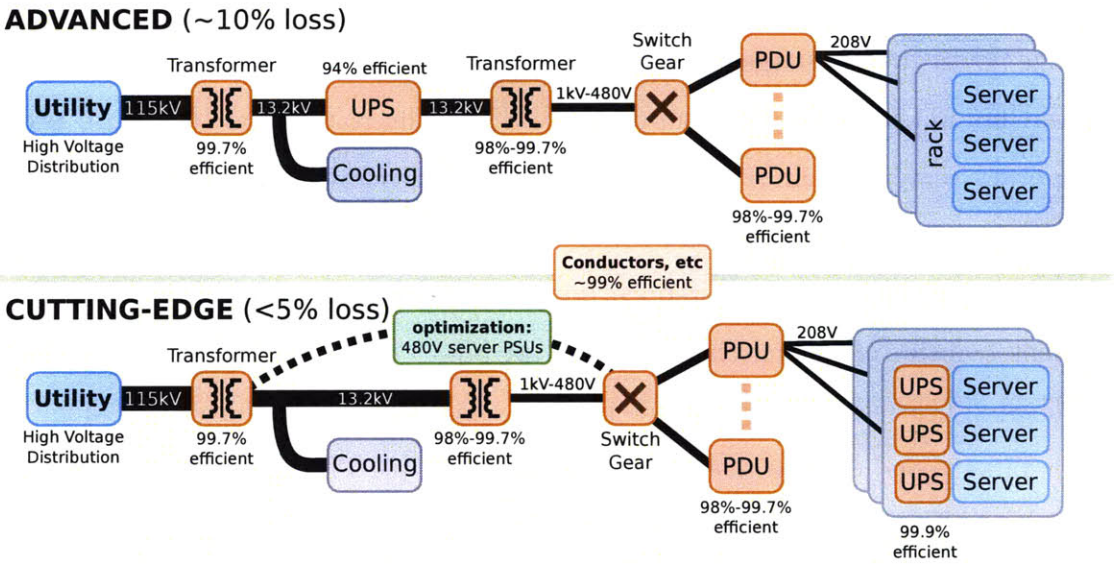


Figure 3-9: Schematics for two representative power distribution systems. The top design is based on Amazon; the bottom design is based on Google.

3.4 Power Distribution System Overhead

Data centers have intricate power management infrastructures that accept high-voltage AC power from an electrical utility, process that power, and deliver an uninterrupted low-voltage supply to the IT equipment. This infrastructure can impose a significant overhead: even in recently built facilities, distribution losses can account for more than 10% of the total electricity [68].

The efficiency of the power distribution system is related to the electrical load placed on it. Since the loads (IT and cooling) are functions of utilization, we can also model the distribution system's losses as a function of cluster utilization u.

A variety of distribution system designs exist. Figure 3-9 shows schematics for two representative designs. The top schematic sketches how power would be conditioned in a conventional tier II data center (e.g., Amazon), following industry best-practices; the other shows a cutting-edge design (e.g., Google) that is far more efficient. The two diagrams abstract away uncessary details and focus on the major sources of electrical inefficiencies. The schematics are synthesized from a number of sources [56, 44, 68, 8, 83, 103]. We do not model server power supply losses, since such losses are captured by our earlier IT equipment models.

In a conventional data center, power from the utility is first converted to *medium* voltage power (using a large and efficient transformer[11]) and then fed into one or more central uninterruptible power supplies (UPS's). The role of the UPS is to provide temporary power if utility power fails, until local generators can be brought online. Sometimes redundant parallel UPS's are used, so that a UPS failure can be masked. Power from the UPS is then stepped down to a lower voltage (commonly using less efficient transformers) and then delivered to power distribution units (PDU's) that are located near servers. Each PDU performs another voltage conversion, along with other power conditioning tasks, and supplies multiple IT equipment racks (a PDU may supply 20-60 racks, with 10-80 servers per rack, depending on the types of servers and PDU capacities [56]). Some energy is also lost in the wiring and switch gear. In some facilities, cooling may be part of the critical load, and connected to the UPS.

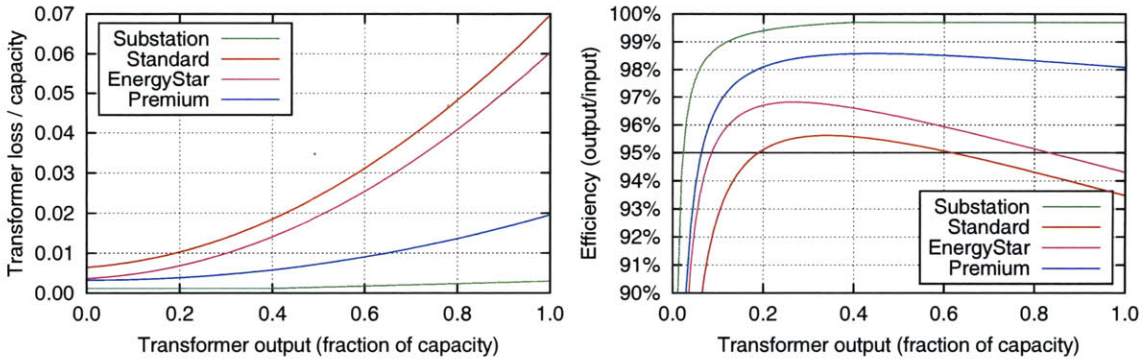
Cutting edge designs can reduce power distribution losses by more than half, as shown in figure 3-9. First, even relatively good central UPS's tend to impose significant losses. Small batteries co-located with servers and networking gear can be used to construct an, almost perfectly efficient, distributed UPS [83]. Additionally, some newer designs use servers that can accept medium-voltage power, eliminating one of the voltage conversions, and thereby improving efficiency [103].

Note that losses deeper in the distribution network impose extra load on earlier components, causing inefficiences to compound. For the conventional design shown in figure 3-9, we can describe the relationship between cluster utilization u and lost watts P_{PD} as:

$$\begin{aligned} P_{total}(u) &= P_{Tr1}\Big(P_{cooling}(u) + P_{UPS}\big(P_{Tr2}\big(n_{PDU} \cdot P_{PDU}(\frac{P_{IT}(u)}{n_{PDU}})\big)\big)\Big) \\ P_{PD}(u) &= P_{total}(u) - \big(P_{IT}(u) + P_{cooling}(u)\big) \end{aligned}$$

Where P_{Tr1}, P_{UPS}, P_{Tr2}, P_{PDU} represent the different components in the distribution network, each function taking electrical load in watts as an input and giving

[11] This transformer is typically located outside the data center, in the utility's substation, and is sometimes ignored in PUE calculations. It's exclusion improves PUE, but we follow conventional wisdom [68] and account for it as part of a data center's power system.



(a) Losses **(b) Efficiency**

Figure 3-10: Transformer losses increase with load. Loss is given as a fraction of the transformers' output capacity.

Transformer	**Efficiency**		**Loss Model**
	40% load	**10% load**	
Standard	96%	93%	$L(x) = 0.0065 + 0.0080x + 0.0553x^2$
EnergyStar	97%	96%	$L(x) = 0.0037 + 0.0055x + 0.0512x^2$
Premium	99%	97%	$L(x) = 0.0033 + 0.0166x^2$

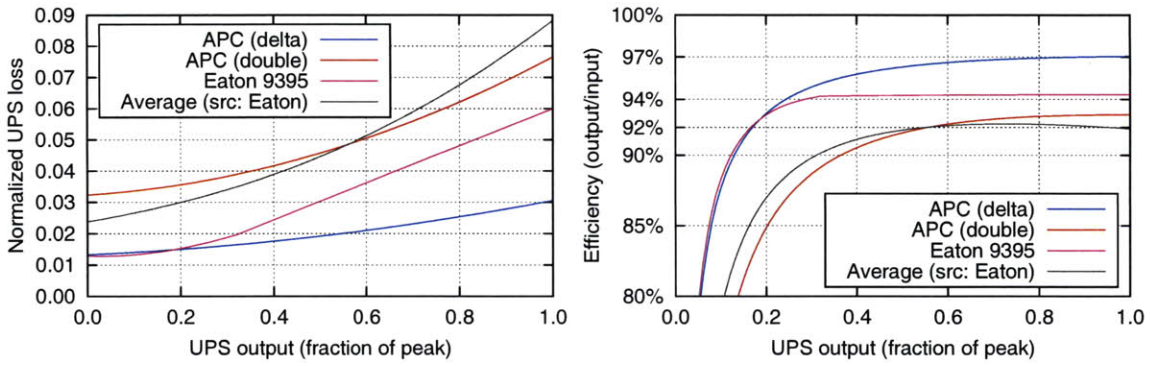
Figure 3-11: Transformer power loss models, from statistical regressions of measurements conducted by the DOE and reported by Powersmiths [82]. The equations map load (0.0-1.0) to transformer power overhead (as a fraction of transformer output capacity).

total power draw in watts as the output. We have assumed, for simplicity, that IT equipment and service load are uniformly distributed among PDU's,

The remainder of this section discusses each of the major inefficiencies in the power distribution system, and presents empirically derived equations to model P_{UPS} etc.

Transformers and PDU's. At multiple points in the power distribution network, transformers and PDU's condition and convert electricity between different voltage levels, a process that is not perfectly efficient. For simplicity, we assume that PDU losses are dominated by voltage conversion, and model PDU's as transformers [103].

In general, transformer losses increase nonlinearly with load. Conventional transformers are constructed using a central core with two sets of coils wound round it. Such transformers experience resistive I^2R losses in the coils; and nonlinear losses due to eddy currents and hysteresis in the core. Transformer efficiency depends on, among other things, core material, cross-section size, the number of windings in the



(b) UPS Losses **(a) UPS Efficiency**

Figure 3-12: UPS losses and efficiency increase with increasing load. Loss is given as a fraction of the UPS's output capacity.

UPS	**Efficiency**		**Loss Model**
	@50%	**@10%**	
APC [105] (delta-c)	96%	88%	$L(x) = 0.013 + 0.006x + 0.011x^2$
APC [105] (double-c)	91%	75%	$L(x) = 0.032 + 0.010x + 0.035x^2$
Eaton [6] (E9395)	94%	88%	$L(x) = \begin{cases} 0.013 - 0.007 \cdot x + 0.074x^{1.843}, & x > 0.32 \\ 0.059x \end{cases}$
Average [6] (src: Eaton)	91%	79%	$L(x) = 0.024 + 0.027x + 0.038x^{2.315}$

Figure 3-13: UPS power loss models, from statistical regressions of data reported by two UPS manufacturers. The equations map load (0.0-1.0) to UPS power overhead (as a fraction of UPS output capacity).

coils, and whether or not nonlinear currents are being drawn by the load[12]. Rather than modeling the physics, we use emprical data to derive transformer loss models.

According to industry reports, the transformers used in data centers tend to have efficiencies in excess of 98%, with substation transformers being close to 99.7% efficient [68, 54, 110][13]. We combine insights from these reports with empirical data collected by the US Department of Energy on behalf of manufacturers [82], to build loss models for several transformer types. Figure 3-10 shows our loss models and figure 3-11 shows the underlying equations[14].

[12] A study of data centers found that electrical loads were close to linear [22].

[13] Transformer efficiency is usually rated at 35%-50% load (EPA NEMA standards).

[14] Lacking empirical data on substation transformers, we model them as having a constant loss until they reach an efficiency of 99.7% at 40% load, and thereafter having a constant efficiency. This is a conservative model of low-load loss.

Central UPS Losses. As figure 3-9 shows, conventional data centers typically have a large central UPS in series with the utility power and the IT equipment. Such UPS's use either batteries or flywheels to store energy when the utility is providing power, and instantaneously switch over to their reserves if the utility supply is disrupted.

A UPS with no load can consume a noticeable amount of power, and its power consumption tends to rise with increasing load (roughly quadratically). UPS's tend to be most efficient when the electric power being drawn from them is close to their capacity. However, even at peak load, advanced central UPS's may loose more than 6% of the power they draw from the utility.

Figure 3-12 shows the relationship between load and efficiency for some large data center UPS's. The curves are constructed using empirical data reported by two UPS manufacturers (APC [105] and Eaton [6]). Note how efficiency plummets at low loads. In the most efficient UPS, with a peak efficiency of 97%, the idle losses are almost half the peak losses. We fit low-degree polynomials to this data and derived the UPS loss models shown in figure 3-13.

Another factor to consider is that some data centers use redundant UPS's (2N or N+1 configurations), to guard against UPS failures. During normal operation, individual UPS's in these data centers may be operating at less than 30% of their capacity, i.e., in their inefficient regions [105, 110]. We assume that no UPS redundancy exists in our models, because of our focus on tier II cloud data centers.

Distributed UPS. Newer data centers eschew central UPS's in favour of distributed UPS's. Small batteries can be integrated directly into the power supplies of servers and network equipment (see figure 3-9). This design can achieve an UPS efficiency of 99.9%, well beyond even state-of-the art central UPS's. Another advantage of this design is that redundant UPS's are not necessary: UPS module failures result in highly localized faults, that can be handled by the mechanisms already in place to deal with other commonplace machine failures. When modeling power systems with distributed UPS's, we can safely ignore UPS losses.

Other Electrical Losses. The wiring and electrical switching gear also introduce losses. The characteristics of these losses are related to the amount of current flowing, the shape of the wiring, etc. These losses are estimated to be 1% at peak electrical load [68, 103]. Like lighting, we model this as a fixed loss. Alternatively, we could model these losses as being linear in the amount of electrical load placed on the wiring system: no loss at 0% load; 0.5% loss at 50% load; and 1% loss at 100% load [103].

3.5 Cluster Energy Models

We construct cluster energy models by composing the sub-system energy models we have described in the past three sections. Rather than offering a single cluster model, we build a collection of models, to cover the design diversity we have noted in the preceding sections. Figure 3-14 lists the 25 cluster models we use in this dissertation. Each model generates a power-vs-utilization curve that is then scaled by the number of servers in the cluster and the peak server power in watts.

There are two sets of models. In the first set (the `L` and `G` models) non-IT power consumption is constant, and equal to: $(PUE - 1.0) \cdot P_{IT}(1)$. In other words, cooling and power distribution losses are assumed to remain fixed at their maximum levels. When utilization is below peak, this approach overestimates non-IT power. In the second set of models (the `X`, `D` and `E` models), non-IT power varies with utilization. These models use the cooling and power distribution equations we presented earlier. By comparing the idle PUE's of the fixed overhead models to the variable overhead ones, we see how the simpler models overestimate overhead (e.g., compare `G6` to `X2`).

We use several different IT energy models. In the `L` models, IT power is a linear function of cluster utilization. The `G` models and some others use the nonlinear model from the Google data center study. We also construct models that use scaled versions of different machine curves (this models even-load balancing in different kinds of homogeneous-hardware clusters). Some models also allow dynamic server scaling (the `D` models and `E2`). Figure 3-15 shows power curves for several models. The IT component is the dominant factor affecting the shape of the cluster power curve.

ID	Idle server power	PUE Peak load	PUE Idle	Comments	EPG	EAP
Fixed Overhead Models						
Linear IT Models ($P_{IT} = m \cdot u + c$)						
L1	65%	1.90	2.35		0.15	0.18
L2	50%	1.70	2.33		0.24	0.28
L3	33%	1.50	2.38		0.42	0.36
L4	25%	1.30	2.04		0.47	0.55
L5	20%	1.10	1.42		0.59	0.69
Google IT Models ($P_{IT} \sim u^{1.4}$)						
G1	65%	1.90	2.35	e.g., EPA 2011 (current-trends).	0.21	0.22
G2	50%	1.70	2.33		0.33	0.36
G3	40%	1.50	2.16		0.45	0.49
G4	33%	1.50	2.38		0.50	0.54
G5	22.5%	1.50	2.90		0.58	0.63
G6	25%	1.30	2.04	e.g., EPA 2011 (best-practices).	0.64	0.70
G7	22.5%	1.20	1.76	e.g., Microsoft containers.	0.72	0.78
G8	20%	1.10	1.42		0.81	0.88
G9	10%	1.10	1.69	Best current tech.	0.91	0.99
Variable Overhead Models						
X1	25%	1.70	1.54	IT $\sim u^{1.4}$; cooling is 52% of total power at peak; inefficient UPS.	0.75	0.84
X2	20%	1.30	1.24	IT $\sim u^{1.4}$; cooling is 17.5%.	0.83	0.92
X3	47%	1.50	1.35	IT $\sim$ Google curve; 36% on cool.	1.17	1.16
X4	22%	1.50	1.37	IT $\sim$ TX150 curve; 36% on cool.	0.63	0.69
X5	26%	1.50	1.35	IT $\sim$ R610 curve; 36% on cool.	0.83	0.88
Dynamic Server Scaling (cooling is 30% at peak)						
D1	25%	1.43	1.91	IT $\sim u^{1.4}$; 50-server blocks.	0.68	0.85
D2	25%	1.43	1.56	IT $\sim u^{1.4}$; 250-server blocks.	0.69	0.83
D3	25%	1.43	1.40	IT $\sim u^{1.4}$; 500-server blocks.	0.67	0.84
D4	26%	1.43	1.78	IT $\sim$ R610; 100-server blocks.	0.69	0.80
Cutting-Edge Clusters (IT $\sim u^{1.4}$; distributed UPS; 17.5% on cooling)						
E1	20%	1.21	1.15	e.g., Google data center.	0.84	0.93
E2	20%	1.21	1.41	with 50-server block DSS.	0.74	0.91

Figure 3-14: A collection of cluster energy models. See §3.7 for EPG and EAP.

We will describe the EPG and EAP metrics soon (§3.7). In short, these numbers summarize the features of the power curve that PDR can exploit. As we will show later, a higher EPG value implies that PDR will achieve higher savings (when we optimize for monetary operating costs).

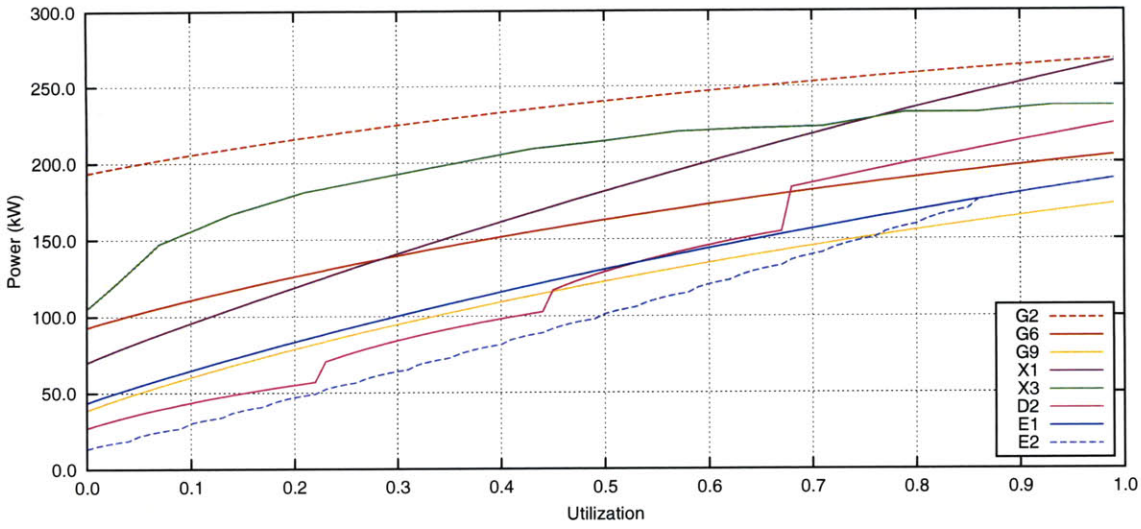


Figure 3-15: Power curves for some energy models (1000-server clusters; 150W peak server power). The IT component dominates the overall shape (e.g., compare `E1` with `E2`).

In the sub-system modeling discussion, we limited ourselves to the case where a cluster was an entire data center. We must be careful about applying these models to smaller clusters (e.g., racks, containers, etc.) in multi-tenant data centers. The complication is the non-IT power component. In a shared facility, the thermal load on a chiller, for example, depends on the utilization levels of all the tenants in that facility. Because of this, a smaller cluster's share of overhead power will deviate from the quadratic chiller models. We can continue to use the fixed overhead models as before—they will still provide overestimates for overhead. We are assuming that multi-tenant facilities account for overhead power by dividing that power among the hosted clusters. Each cluster's share of the overhead is equal to the ratio of its peak IT power to the peak IT power of the entire facility.

3.6 Modular Data Centers

Our models are mostly based on *monolithic* data center designs. An alternative modular design approach is becoming increasingly popular: server racks are placed inside large shipping containers, and power distribution units and CRAC's are inte-

grated into the individual containers. External chillers are necessary to provide chilled coolant to the containers. Additionally, an external UPS may also be necessary. For a sense of scale: a single HP POD container may draw 450-600kW of power and can handle up to 3500 servers in about 20 racks [10].

Container-based designs have demonstrated higher server power densities and more efficient air cooling than legacy designs. A HP POD container has an internal PUE of 1.25 (this doesn't account for the external chillers and UPS's). Cutting-edge facilities therefore commonly use these modular designs. Google [44] and Microsoft [84] have both built container-based facilities. Containers can also be deployed in multi-tenant facilities. Companies like HP, IBM and Oracle/Sun market container-based data center modules [5].

We choose not to build special models for container-based clusters. The earlier models can be adapted. Inside containers, the IT equipment behaves as before. One of the power distribution system designs we modeled was based on a container facility. We can model the efficient container fans, by reducing the fraction of cooling power used by CRAC's. Finally, we can reuse the chiller models without modification. In fact, model `E1` can be interpreted as a model for a container-based facility.

If a container-based facility were to power-off idle clusters—dynamic container scaling—this would change the nature of CRAC and PDU losses, and the facility's power curve would begin to deviate from our models. However, this sort of dynamic scaling is not used in practice.

3.7 Energy Proportionality Metrics

PDR works better with some kinds of power curves ($P(u)$) than it does with others. Figure 3-15 illustrates a variety of different shapes. The effectiveness of PDR hinges on the degree to which a cluster's energy consumption changes when we change its utilization level by shifting load. To complicate matters, $\frac{dP}{du}$ is not always constant.

We have developed two numerical metrics that summarize a power curve's shape as a number. As we show in our later simulations, a cluster architecture's score for

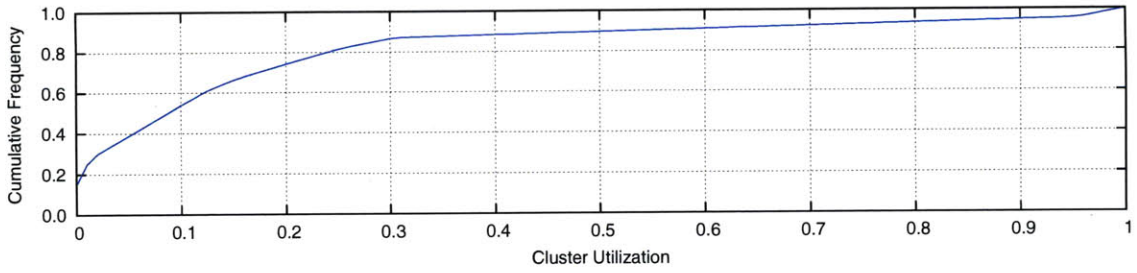


Figure 3-16: Cluster utilization distribution used to calculate EPG and EAP scores.

one of these metrics is a good predictor for how well PDR will perform on a system composed of clusters of that type.

Expected Power Gradient (EPG). The EPG score is a normalized form of the expected value of $\frac{dP}{du}$. Formally it is defined as:

$$EPG = \int_0^1 \left(\frac{P'(u)}{P(1)} \right) \cdot Prob(U = u)\ du$$

Where *Prob* is a probability distribution for the cluster's utilization. The utilization distribution we use is derived from our analysis of cluster utilizations in a data set from Akamai, server utilization distributions reported by Google, and the results of our PDR simulations. The distribution is shown in figure 3-16. If we instead use a uniform distribution, we find that the EPG score is still predictive, but less accurate.

To see why EPG works, consider this: at a random time, PDR attempts to shift power consumption away from a cluster by shifting load away from it. The EPG score measures how far we should expect the cluster's power to fall as the load is shifted away. In our simulations, the monetary savings achieved by PDR for each model were approximately a quadratic function of that model's EPG.

In practice, we use a summation, a histogram for *Prob* and step through the values of u using a step of 0.01:

$$EPG \approx \sum_{u=0.01}^{1} \left(\frac{P(u) - P(u - 0.01)}{P(1)} \right) \cdot Prob(U = u)$$

Expected Active Power (EAP). The *active power* of a cluster is its power consumption minus its idle power. We define EAP formally as:

$$EAP = \frac{\int_0^1 \left(\frac{P(u)-P(0)}{P(1)}\right) \cdot Prob(U = u)du}{\int_0^1 u \cdot Prob(U = u) \cdot du}$$

The denominator normalizes EAP so that, regardless of the probability distribution used, a linear power model with zero idle power has an EAP score of 1.0. As with EPG, we use an approximate summation and the earlier utilization distribution.

EAP is a little harder to understand intuitively, but it is useful in practice. EPG is a good predictor for *absolute* savings, while EAP is a good predictor of *relative* savings (%). We will discuss this distinction in some detail later.

3.8 Increase in Wide-Area Routing Energy

With power-demand routing, clients may be routed to distant servers in search of cheap energy. From an energy perspective, this network path expansion represents additional work that must be performed by something. If this increase in energy were significant, network providers might attempt to pass the additional cost on to the server operators. Given what we know about the economics of network pricing (§7.6), a small increase in routing energy should not impact network prices. Alternatively, server operators may bear all the increased energy costs (suppose they run the intermediate routers).

A simple analysis suggests that the increased path lengths will not significantly alter energy consumption. Routers are not designed to be energy proportional and the energy used by a packet to transit a router is many orders of magnitude below the energy expended at the endpoints. Recall that our focus is on complex services in which a small number of request packets trigger energy intensive processes at the cluster (e.g., web search).

To demonstrate that increased path lengths will not be problematic, we use empirical data for a core network router. We use power consumption measurements of the

Cisco GSR 12008 router under different loads [48]. At a throughput of 540k mid-sized packets/sec, the router consumed 770 watts. At this load, the *average* energy needed for a packet to pass through a core router is less than 1.5 mJ. This is almost six orders of magnitude below Google's endpoint energy of 1 kJ/query [71]. Therefore, even if path lengths are being increased by 10 hops or so, the increase in routing energy will not be significant.

Furthermore, because of the lack of proportionality, the *incremental* energy dissipated by each packet is even smaller. In the reported measurements [48], the power consumption of the idle router was 97% its peak power. This implies that the incremental energy would be as low as a 50 μJ per medium-sized packet. In the future, power-aware hardware may reduce this disparity between the marginal and average energy. Very recently, proposals for energy-proportional data center networks have emerged [35], but we expect core routers to lack proportionality for some time.

Finally, we must also consider what happens if the new routes overload existing routers. If we use enough additional bandwidth through a router it may have to be upgraded to higher capacity hardware, increasing the energy significantly. However, we could prevent this by incorporating constraints, like the 95/5 bandwidth constraints we use in our simulations.

Summary

This chapter focused on understanding and modeling energy consumption in data centers. This work was necessary because comprehensive data center energy models do not exist. The models we propose here are compositions of several empirical component models. Even though we have not been able to verify the accuracy of our composite models, the component models are known to be accurate.

The 25 cluster models described here are used extensively in the power-demand routing simulations in chapter 6. Independent of our simulations, the modeling strategy we have developed is important, because PDR implementations will need to integrate energy models into their optimization mechanisms.

Chapter 4.

Modeling Workloads and Performance

In chapter 2, we sketched a basic system model: a fully replicated system, composed of geographically distributed clusters, services traffic that consists of similar requests. The energy consumption of each cluster depends on its size and on the traffic load placed on it. The cost of that energy depends on the cluster's geographic location (e.g., electricity market prices).

To model the total operating cost we must therefore model how different routing policies spread load geographically. For instance, when routing optimizes for performance the load on a cluster depends on the volume of requests that originate in nearby regions. Thus, in order to spatially model power-demand we need a realistic traffic workload, one that provides sufficient detail about how traffic varies in time, and about how traffic sources are distributed geographically. In this dissertation, we use request traces we collected from Akamai's content distribution network (CDN) and a synthetic workload we derived from these traces (§4.1).

Additionally, we need a way to measure the performance costs associated with different routing policies, to determine whether PDR can be implemented without violating existing performance goals (§4.2). One complication is that the Akamai traces have no network latency information, only geographical information. We must therefore find a way to relate geographic distance to network latency. Furthermore,

in order to construct the sort of joint energy-performance optimization described in chapter 2, we need a performance cost function specified in terms of lost revenue. We use reports from Google, Microsoft and Amazon to propose some monetary performance cost functions.

4.1 Akamai Traffic Workload

In order to understand the behaviour of real workloads in time and space, we acquired a data set detailing 24 days of traffic on Akamai's CDN infrastructure. From this data we also derived a synthetic workload model, based on the notion that traffic exhibits hour-of-day and day-of-week periodicity.

4.1.1 Raw Traffic Data

For a period of 24-days, we collected request trace summaries from 16K servers in Akamai's CDN infrastructure. This is a large subset of Akamai's servers. Our data covers about a hundred billion requests per day, incident on servers in 25 different US cities. Akamai has over 2000 content provider customers hosted on its CDN, so the traffic we observed represents a broad user base, and a meaningful portion of total web traffic.

Figure 4-1 shows how aggregate traffic varied in time. Daily periodicity is easy to see. The day-of-week pattern is less obvious.

The data collection period stretched from the 19th of December 2008 to the 11th of January 2009. A complication is that this period overlapped with the end-of-year holidays (Christmas and New Year's). As the figure illustrates, some holidays exhibited a significant reduction in traffic. We had to account for this when building our synthetic traffic model.

Traffic data was collected at 5-minute intervals on servers housed in Akamai's *public* clusters. Akamai has two types of clusters: public, and private. Private clusters are typically located inside of universities, large companies, small ISP's, and ISP's outside the US. These clusters are dedicated to serving a specific user base, e.g., the

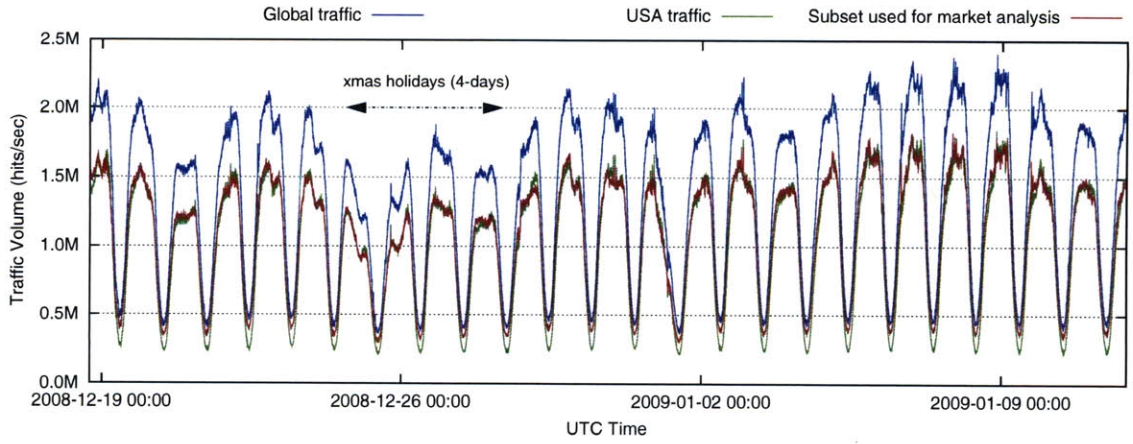


Figure 4-1: Aggregate traffic in the Akamai data set. We see a peak hit rate of over 2M hits per second. Of this, about 1.25M hits/s come from clients in the US. Note the traffic dip during the holidays. The traffic in this data set comes from roughly half of the CDN servers Akamai had. In comparison, in total, Akamai saw around 275B hits/day (more than 3M hits/s) during this period.

members of a university community, and no others. Public clusters are generally located in commercial multi-tenant facilities and can serve any users world-wide. For users not served by private clusters, Akamai has the freedom to choose which of its public clusters serve those users. Requests that end up at public clusters tend to see longer network paths than requests that can be served at private clusters.

The 5-minute data contains, for each public cluster: the number of hits and bytes served to clients during that interval; and a rough geography of where those clients originated. We could therefore tell where requests originated and where Akamai's CDN logic routed them.

Figure 4-2 shows how traffic originating from some individual US states varied. Note that not all states contributed evenly to the total traffic, but behaviour was reasonably well correlated. There is a visible phase-difference between traffic signals from the East and West coasts, because of time-zone differences.

In the data we collected, the geographic localization of clients is coarse: they were mapped to origin states—for requests from the US—or countries. If multiple clusters existed in a city, we aggregated them together and treated them as a single cluster.

Lacking any network level data on clients, we use geographic distance as a rough

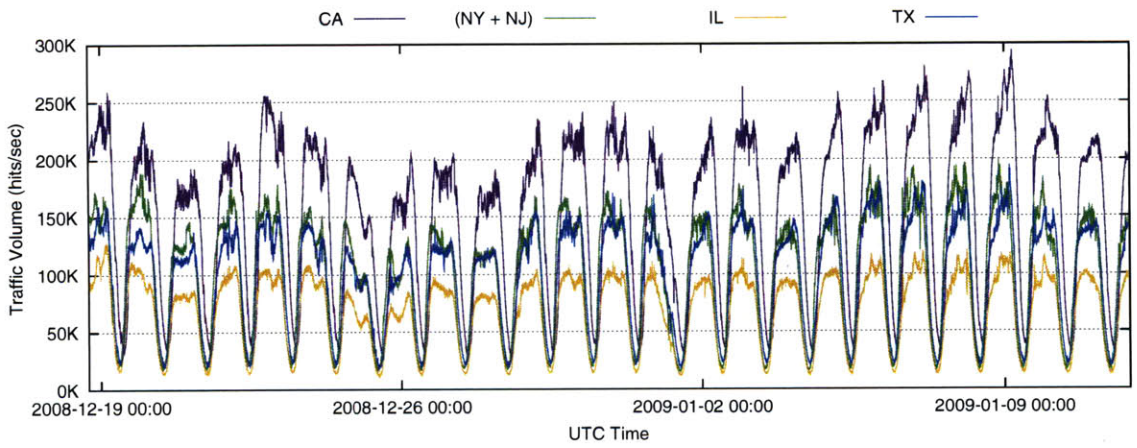


Figure 4-2: Traffic in the Akamai data for individual origin states.

proxy for network performance in our analysis (we discuss the relationship between latency and distance in §4.2). In our traffic data, we see some evidence of geo-locality in Akamai's routing decisions, but there are many cases where clients are not routed to the geographically nearest clusters. One reason is that geographical distance does not always correspond to optimal network performance. Another possibility is that the system is trying to keep those clients on the same network, even if Akamai's servers on that network are geographically far away. Yet another possibility is that clients are being moved to distant clusters because of network costs.

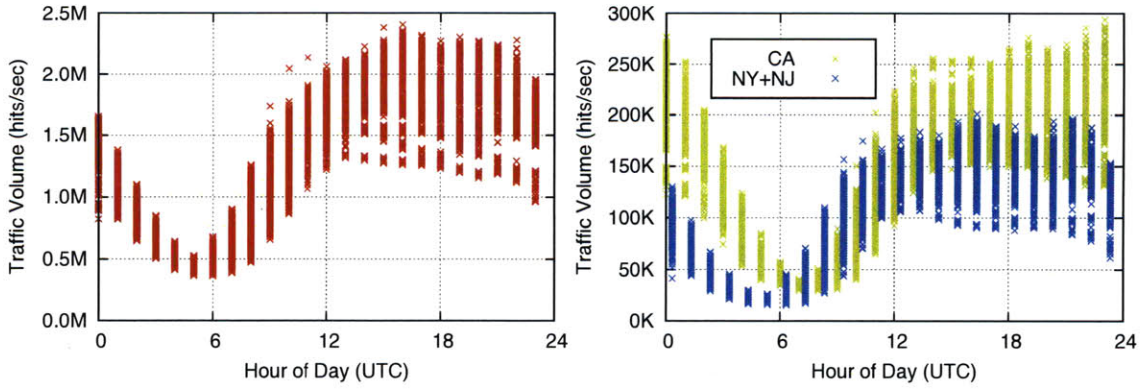
We also collected data on many other aspects of Akamai's system. The 5-minute data included the utilization level of each cluster[1]. We surveyed the hardware used in the different clusters and collected values for measured[2] peak server power. We also looked at the top-level mapping system to better understand how name-servers were mapped to clusters. This additional data informed modeling decisions discussed throughout this dissertation, ultimately leading to a more realistic analysis of PDR.

4.1.2 Synthetic Traffic Model

From the raw traffic data we derived a synthetic workload model. Given a timestamp, the model generates a traffic matrix (hits per second, broken down by origin states).

[1] Utilization was calculated using a proprietary method by Akamai that factored in the different cluster resources (CPU, disk, network, etc.).

[2] As opposed to nameplate power ratings.



(a) Global Traffic **(b) Individual States**

Figure 4-3: As shown by this alternate view of the raw data, traffic volume depends on the hour-of-day, but the nature of the dependency varies from state to state.

The model is based on the subset of the raw traffic that originated in the US.

If we want to simulate how different routing strategies respond to electricity market price fluctuations, using the raw traffic data limits us to the 24 day period those traces cover. With a synthetic model we can simulate other and much longer periods of time. In our final analysis, we simulate periods as long as 39 months, so that we can incorporate all the market price data we have.

Our modeling approach is predicated on the fact that web traffic exhibits diurnal periodicity. This is not a new observation (e.g., [18]). The periodicity is apparent in figures 4-1 and 4-2. Figure 4-3 shows an alternate view of this traffic data, as a scatter-plot of traffic volume versus hour-of-day. Clearly, traffic volume depends on hour-of-day, both for aggregate traffic and for individual states.

The traffic matrix is generated by independently modeling each state of origin. To build a traffic model for a single origin state, we mapped each raw traffic volume sample for that state to a bucket, based on its day-of-week and hour-of-day[3]; then summarized all the samples in each bucket to a single value (e.g., the mean, median, etc.). When a model is asked to generate traffic for a timestamp, it computes the timestamp's bucket and returns that bucket's summarized value.

The final model we used was constructed using the median bucket traffic. To discount abnormally depressed holiday traffic, we excluded 10 days of the raw data.

[3] $bucket = weekday \times 24 + hour\text{-}of\text{-}day$

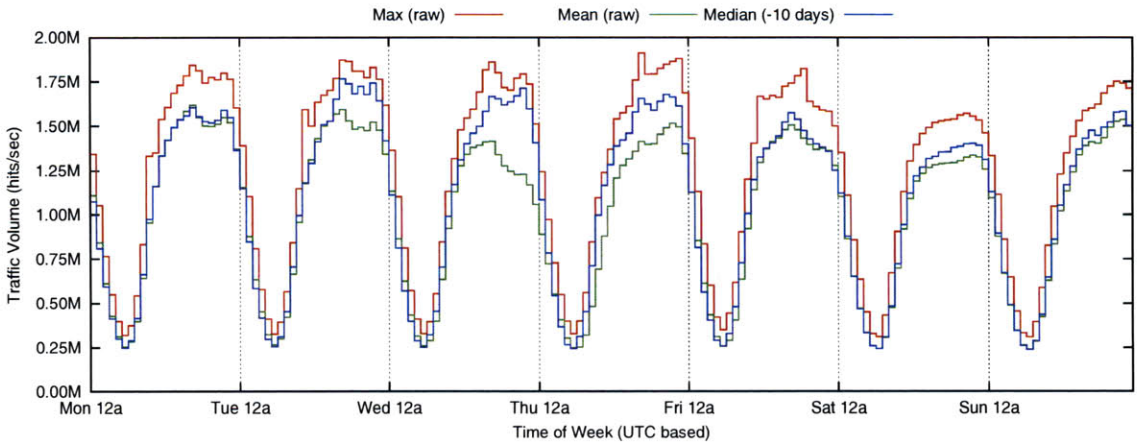


Figure 4-4: Comparing the aggregate traffic volume generated by different models.

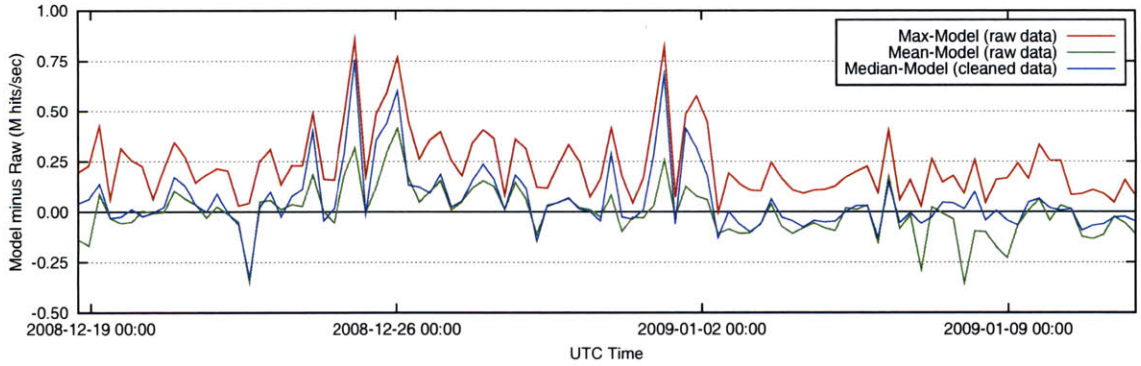


Figure 4-5: Difference between observed US traffic and model generated traffic.

We experimented with models based on mean bucket traffic, maximum bucket traffic, 95-th percentile bucket traffic, etc.; and with models that excluded only the Christmas holidays, and those that used the entire raw data set.

Figure 4-4 compares some different models. We did not use the obvious approach of taking the mean over the entire data, because it results in abnormally low traffic for Wednesdays and Thursdays, due to Christmas. We did not rely on the maximum and 95-th percentile models because these are consistent overestimates, as shown in figure 4-5.

Figure 4-5 shows how model generated traffic volumes deviate from the actual traffic volumes in the raw data (for US traffic). The major deviations for all models occur near Christmas and New Year's, which are both abnormal periods.

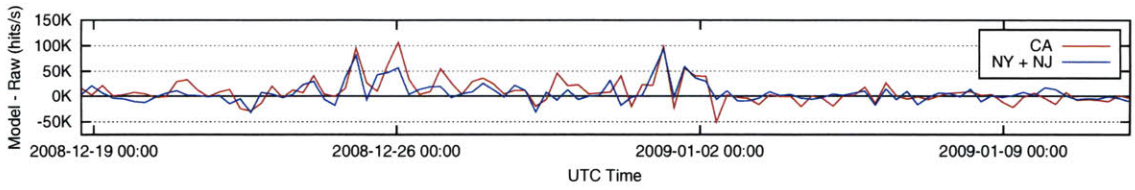


Figure 4-6: Difference between observed traffic volumes and model (median, -10 days) generated volumes for two locations.

4.2 Modeling Performance Cost

Incoming request traffic is distributed among the available clusters according to some routing policy. We need to be able to understand how different policies impact client performance, to understand if PDR is usable in practice.

The optimization model we presented earlier (chapter 2) uses end-to-end latency as the performance metric. Unfortunately, the traffic workloads we use in our analysis do not provide any client-server latency information. In this section we show how to calculate client-server geographic distances for our workloads, and why this can be used as a measure of network latency.

Furthermore, one of our goals is to implement a routing policy that deals with the trade-offs between energy costs, network costs and performance. To facilitate this, we present some performance-cost functions (C_P) that define a relationship between client-server latency and lost revenue.

4.2.1 Calculating Client-Server Distances

The traffic workloads we use in our analysis (the raw Akamai data, and the synthetic model derived from it) contain no network information. In addition, servers are grouped by city and clients are grouped by state. Therefore, the granularity of this workload data does not provide enough information for us to estimate network latency between clients and servers, or even to accurately calculate geographic distances between the two.

We constructed a geo-distance function that calculates a population-density weighted geographic distance. This takes into account the fact that requests are being gen-

erated by a population that is spread out non-uniformly. This approach is more accurate than, for example, using the the geographic center of a state as a client's location. We used data from the US census to derive basic population density functions for each US state. Given a client's origin (state s) and the server's location (city c), we define:

$$distance(c, s) = \sum_{city \in s} \left(\left(\frac{Pop_{city}}{Pop_s} \right) \cdot gdist(c, city) \right)$$

Where Pop_{city} is the city's population (from census), Pop_s is the state's population, and $gdist$ is the great-circle distance (in km) between two city centers. When calculating average distance across the system, we exclude clients from Alaska and Hawaii, and also exclude clients outside the US.

4.2.2 Distance and Latency

We use geographic distance as a rough measure of network latency. We rewrite the earlier optimization (chapter 2) in terms of geo-distance, expressing network latency as a linear function of the distance. This section justifies why this approach is acceptable.

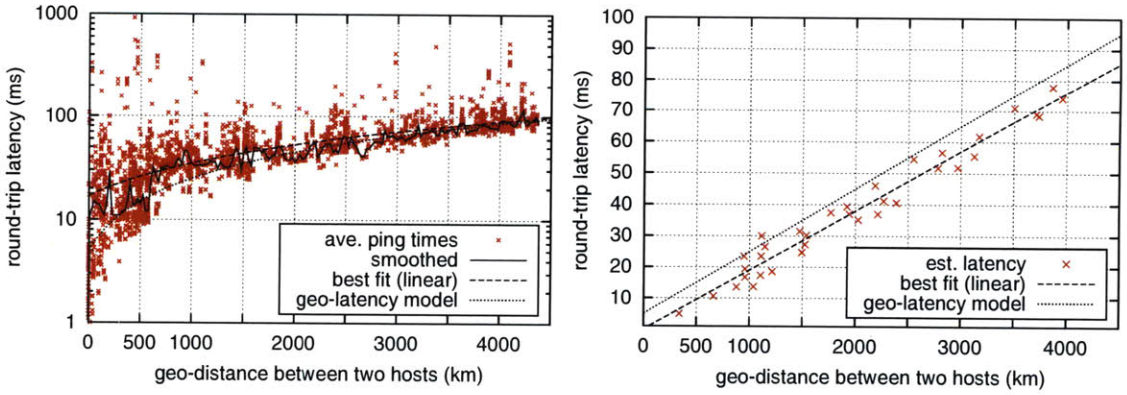
We look at three network latency data sets in which latency can be approximated as a linear function of great-circle geo-distance. In all three data sets, the linear approximations have similar gradients:

$$\frac{\delta(\text{round-trip-time})}{\delta(\text{distance})} \approx 0.02 \; ms/km$$

Note that this gradient is about $\frac{1}{3}$ what we would expect if network messages were moving at the speed of light[4]. The distance-latency relationship tends to be strong for backbone networks (see below), weaker for machines on edge networks, e.g. home PCs, and is more accurate at predicting lower bounds than averages [59, 38].

We start with two latency data sets available to us: pings between about 260 PlanetLab hosts, located at US universities [109]; and one-way latency measurements

[4] $0.02ms/km \rightarrow 50km/ms$ round trip $\rightarrow 100km/ms$ one way $= 100,000,000m/s \approx c/3$



(a) PlanetLab (6000+ pairs) **(b) Internet2 backbone (35 pairs)**

Figure 4-7: The relationship between network latency and client-server geo-distance.

between Internet2 backbone nodes [13]. The PlanetLab data is from December 2005; the Internet2 measurements were conducted in July 2010. In both data sets, we find that a linear relationship exists. Figure 4-7 shows the correlation between latency and geo-distance in each data set. The backbone nodes exhibit a strong linear relationship; the relation is much weaker in the PlanetLab data. In both cases, the following is reasonably accurate[5]:

$$\textit{round-trip-time} = 0.02 \times \textit{distance} + 5$$

Other researchers analyzed a data set of around 50 million latency measurements between 3.5 million home machines [38][6] and derived the following relationship:

$$\textit{round-trip-time} = 0.017 \times \textit{distance} + 60$$

The larger constant represents the fact that these were mostly home machines; our two data sets are for machines on university and backbone networks.

In using distance to evaluate the performance impact of PDR, what we really care about is not how well distance predicts client-server latency. What we care about is how well the change in average client-server distance—PDR relative to another request routing policy—predicts the change in average latency. In other words, if

[5] Linear regression best fits: $\textit{rtt-plab} = 0.18 \times \textit{dist} + 17$; $\textit{rtt-backbone} = 0.019 \times \textit{dist}$.

[6] The distance multiplier in the paper is $0.0269ms/mile$, which is $0.0167ms/km$.

PDR results in geographically longer routes between clients and servers, we want to calculate the resulting elevation in end-to-end latency. If routes expand only in the core, we will see the sort of strongly linear relationship exhibited by the Internet2 data. Due to the vagaries of the Internet's topology, we cannot tell how common this will be.

There is a special case worth noting: the distributed system operator that owns its own network fiber and operates a private backbone between their data centers. In this case distance will be a very strong predictor. Reports indicate that Google operates its own private backbone [78].

4.2.3 Monetary Performance Cost Functions

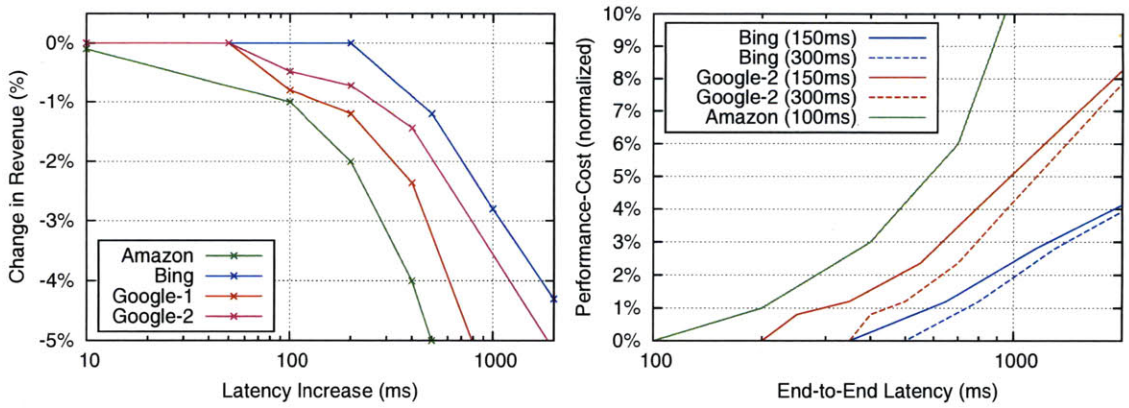
In any revenue generating web service, there is a relationship between revenue and client performance. Often this can be expressed as a piece-wise linear function, quantitatively relating loss in revenue to average client-server latency[7]. As we noted earlier (§2.2), directing the PDR optimization to use a monetary performance cost function may be a better approach than directing the optimization to, for instance, minimize average latency. In this section we demonstrate how monetary performance cost functions can be constructed. We conclude however that these monetary functions may have limited utility. In practice, strict latency constraints are good enough.

Some services are highly sensitive to latency. CDN's such as Akamai's are particularly sensitive, but we do not have a quantitative model for revenue impact. Electronic stock trading is an extreme case. It has been reported that a $5ms$ increase in latency can reduce a broker's revenue by 1%; and a $10ms$ increase in latency can result in a 10% drop in revenue [28].

There is evidence that increased delays in e-commerce services result in substantial lost sales. User experience tests conducted by Amazon concluded that every $100ms$ increase in delay results in a 1% drop in sales [81].

The sensitivity of web search revenue to latency has been studied by both Microsoft (Bing) and Google. Both companies have reported results from user experience

[7] Or to a measure of tail-end client performance, e.g., 95-th percentile latency.



(a) Revenue vs. latency **(b) Monetary C_P functions**

Figure 4-8: The relationship between revenue and latency and monetary performance cost functions that capture this relationship. Google-1 and Google-2 are upper and lower bounds respectively.

tests exploring the revenue-latency relationship [106]. In these tests, random subsets of search users were selected and exposed to artificial page-load delays, ranging from $50ms$ to more than a second. The behaviour of these users was then monitored to see how the added delay affected revenue collected from them.

Figure 4-8a plots the reported revenue-latency relationships (we linearly interpolate between reported data points). Microsoft disclosed change in revenue, but Google only disclosed change in searches-per-user. We assume that conversion from searches to revenue is similar for both Google and Microsoft. We then generate upper and lower bounds for Google's revenue-latency curves.

An important feature in figure 4-8a is how revenue remains unchanged for the first $50ms$ increase in latency (except for Amazon[8]). Looking at the earlier relationship between distance and latency, a flexibility of $50ms$ allows us to use servers that may be more than $2000km$ from their clients (about half the distance between the East and West coasts of the US).

Constructing monetary performance cost functions (C_P) from the curves in figure 4-8a is straightforward. We have to select a base request latency (the optimal network

[8] The reports for Amazon are less specific than those for Microsoft and Google. It seems that the study at Amazon increased delays using a step size of $100ms$. It is therefore possible that Amazon too is unaffected for the first $50ms$. We interpret the reports in a way that is less favourable to our work, assuming that revenue drops at a constant rate for every millisecond increase below $100ms$.

latency plus the time it takes to process a request within a cluster). Figure 4-8b shows some derived cost functions, and their base latencies. The cost functions are specified as fractions of total revenue.

On closer inspection, we conclude that the use of performance cost functions adds unnecessary complexity in practice. For the revenue-latency curves in figure 4-8a, the PDR optimization can use strict latency constraints instead of cost functions, without any loss in accuracy. A 0.1% loss in revenue for a company like Google is more than $250M annually. Thus, allowing latency to rise by more than $50ms$ to reduce energy costs makes no sense. Similarly, PDR can use constraints for Amazon ($\approx 10ms$) and Microsoft ($200ms$), as long as revenues are many orders of magnitude more than energy costs.

Summary

This chapter detailed how we model request workloads and client-server performance in our later analysis of power-demand routing. We described the traffic data we collected in collaboration with Akamai and how we derived a synthetic workload model from it. We also discussed the relationship between client-server latency and geo-distance, and argued that geo-distance can be used as a proxy for network latency. Finally, we explored how revenue in existing Internet services decreases as end-to-end request latencies increase.

Chapter 5.

Cutting Electric Bills

A single Internet-scale system can consume many megawatts of electricity and may run up an annual energy bill that exceeds a hundred million dollars. Furthermore, these energy bills are likely to increase in the future: these already large systems are increasing in size at a rapid clip, outpacing data center energy efficiency gains [47], and electricity prices are expected to rise.

Consequently, organizations such as Google, Microsoft, Amazon, Facebook, and many other operators of large networked systems cannot ignore their energy costs. A back-of-the-envelope calculation for Google suggests its annual electric utility bill is close to $140M (figure 5-1). A modest 1% reduction would therefore exceed a million dollars every year. We project that even a smaller system such as Facebook's consumes an estimated $25M worth of electricity annually.

The conventional approach to reducing energy costs has been to reduce the amount of energy consumed [25, 62]. New cooling technologies [58, 41, 60], virtualization, low-power server hardware [88, 101], dynamic server scaling [1, 15], container-based designs [44, 84, 10] and many other techniques have all been proposed as ways to reduce the power demands of data centers. That work is complementary to ours.

Our work studies how power-demand routing can be used to reduce electricity expenses. We start with the following key observation:

Company	Servers	Electricity	Gen. Cost	Utility Bill
eBay	20K	65 GWh	$3.8M	10M
Rackspace	50K	160 GWh	$9.6M	20M
Facebook	60K	190 GWh	$11.5M	25M
Akamai	65K	210 GWh	$12.5M	25M
Microsoft	>200K	>600 GWh	>$36M	>$75M
Google	>800K	>1120 GWh	>$67M	>$135M
USA (2006) [25]	10.9M	61,000 GWh	$4.5B	
10,000 US homes [32]		<160 GWh		<$30M
MIT campus [20]		270 GWh		$62M

Figure 5-1: Estimated annual electricity costs for large companies. These are conservative estimates, meant to be lower bounds @ $60/MWh. See §5.1 for derivation details. For comparison, we have included the EPA's 2008 estimate of the annual energy consumption of US homes; and the 2007 consumption and utility bill for the MIT campus, including dormitories and labs.

> **Electricity prices vary.** The energy sector is moving toward locationally differentiated real-time electricity pricing. In those parts of the U.S. with wholesale electricity markets, for example, prices vary on an hourly basis and are often not well correlated at different locations. Moreover, hourly prices may vary by as much as a factor of 10 from one hour to the next.

The existence of uncorrelated price volatility can be leveraged in a straightforward manner using the traffic engineering framework described in chapter 2. We construct cluster cost functions based on local electricity prices and plug them into the optimization framework. PDR will then dynamically channel clients to clusters with low spot prices—after accounting for performance and other constraints—relocating power-demand to less expensive energy whenever possible. Since electricity market prices vary hourly, route changes will be infrequent enough to be compatible with existing routing mechanisms.

We hypothesize that this sort of traffic engineering can save large systems' operators a significant amount of money. To establish the validity of this hypothesis, we use simulation-based analysis to quantify the degree of achievable savings. Our analysis projects millions of dollars in savings for large systems like Google's, and we find that PDR will get progressively more relevant, given current trends in technology and in the energy sector. The following chapter details our analysis and results.

The remainder of this chapter provides a context for the subsequent analysis, covering background material on electricity expenses and wholesale electricity markets. Section 5.1 presents evidence that electricity is becoming an increasingly important economic consideration, detailing the calculations that produced figure 5-1; section 5.2 describes salient features of the wholesale electricity markets in the U.S; and section 5.3 details an empirical analysis of over three years of historical wholesale market price data.

One important contribution of this dissertation is to identify the relevance of electricity market volatility to large distributed systems. In section 5.3, we explore price volatility, without the complication of a distributed system or energy model, to get a feel for the underlying market behaviours. We identify several economic opportunities, not all of which can be exploited by real systems.

5.1 The Scale of Electricity Expenditures

In absolute terms, servers consume a substantial amount of electricity. In 2006, servers and data centers accounted for an estimated 61 million MWh, 1.5% of US electricity consumption, an amount that would have cost about 4.5 billion dollars to generate [25]. An EPA study estimated that by 2011 data center energy use could double; at best, by replacing everything with state-of-the-art equipment, we may be able to reduce usage in 2011 to half the current level [25].

Most companies operating Internet-scale systems are secretive about their server deployments, total power consumption and utility bills. Figure 5-1 shows our estimates for several such companies, based on information scavenged from a number of sources and back-of-the-envelope calculations. Private discussions with several people in the industry lead us to believe that our estimates are reasonably accurate.

Our strategy is to start by estimating the total energy consumed by a system, and multiplying that by a reasonable average cost per kWh to calculate the electricity *generation cost*. This is the component of the electric utility bill that depends on market prices and total consumption.

Utility bills have a number of other components, including charges for grid transmission, local-utility distribution, and various others (e.g., low-income assistance program charges, nuclear decommissioning charges, etc.). In order to estimate the overall utility bill from the generation cost, we assume that the non-generation charges account for 50% of the utility bill[1] and round up, to the nearest $5M. We assume that the non-generation part of the utility bill remains constant, unaffected by our modulation of power consumption. This assumption is unfavourable to us. In practice, reducing consumption will likely cause the non-generation components to also fall.

We detail how we derived each company's estimate below.

Facebook, Akamai, etc. Using a linear energy model we can calculate a system's energy consumption in Wh using:

$$E \approx n \cdot \left(P_{idle} + (P_{peak} - P_{idle}) \cdot U + (PUE - 1) \cdot P_{peak}\right) \cdot 365 \cdot 24$$

where: n is the server count, P_{peak} is server peak power in watts, P_{idle} is idle power, and U is average server utilization.

To generate our estimates, we assumed: average PUE was 1.7 [25]; average server utilization was 30% [43, 88]; average peak server power usage was 250 watts (based on measurements of actual servers at Akamai); and idle servers drew 65% of their peak power [56, 50]. The server numbers were compiled from public disclosures, industry presentations, blog posts, and earnings reports: Facebook [67]; Akamai [2]; Rackspace [16]; and eBay [100].

It is worth noting that Akamai's electricity costs represent indirect costs not seen by the company itself. Like others who operate their clusters in multi-tenant facilities, Akamai seldom pays directly for electricity. Power is mostly built into the billing model, with charges based on provisioned capacity rather than consumption. In chapter 7 we discuss why our ideas are relevant even to those not directly charged according to the volume of electricity used.

[1] In reality, non-generation charges at different locations can range from 45% to 70% [33, 31].

Microsoft Our total energy number for Microsoft is an estimate for 2008 based on company statements [86] and energy figures mentioned in a promotional video [89].

Google To estimate Google's power consumption, we assumed 800K total servers [53, 66], used the Google server model detailed earlier (145W peak server power; 80% of peak power at 30% load; see chapter 3), a PUE of 1.3 [83], and an average server utilization of around 30% [43]. Such a system would consume more than 1100 GWh annually, at a generation cost of over $67M. Google may well have more than a million servers [76], so an annual generation cost close to $100M would not be surprising.

These numbers are consistent with an independent calculation we can make. comScore estimated that Google sites handled about 3 billion searches per day in December 2009 [30], and Google has stated that each search takes 1 kJ of energy on average [71] (presumably amortized to include indexing and other costs). Thus, if we assume search traffic is roughly constant throughout the year, search alone works out to 300 GWh. Google's servers handle GMail, YouTube, background data crunching, and many other applications, so our earlier estimates seem reasonable.

5.2 Wholesale Electricity Markets

Although market details differ regionally, this section provides a high-level view of deregulated electricity markets, providing a context for the work that follows. The discussion is based on markets in the United States.

Generation. Electricity is produced by government utilities and independent power producers from a variety of sources. In the United States, coal dominates (nearly 50%), followed by natural gas (~20%), nuclear power (~20%), and hydroelectric generation (6%) [111].

Different regions may have very different power generation profiles. For example, in 2007, hydroelectric sources accounted for 74% of the power generated in Washington state, while in Texas, 86% of the energy was generated using natural gas and

RTO	Region	Some Regional Hubs
ISONE	New England	Boston (MA-BOS), Maine (ME), Connecticut (CT)
NYISO	New York	NYC, Albany (CAPITL), Buffalo (WEST), PJM import (PJM)
PJM	Eastern	Chicago (CHI), Virgina (DOM), New Jersey (NJ)
MISO	Midwest	Peoria (IL), Minnesota (MN), Indiana (CINERGY)
CAISO	California	Palo Alto (NP15), LA (SP15)
ERCOT	Texas	Dallas (N), Austin (S)

Figure 5-2: The different regions studied in this dissertation. The listed hubs provide a sense of RTO coverage.

coal. Generation diversity is one factor giving rise to a lack of correlation between prices at different locations.

Transmission. Producers and consumers are connected to an electric *grid*, a complex network of high-voltage transmission lines. The quantities of electricity that are transported over the grid cannot be stored easily[2], so supply and demand must continuously be balanced.

In addition to connecting nearby nodes, the grid can be used to transfer electricity between distant locations. The United States is divided into eight *reliability regions*, with varying degrees of inter-connectivity. Congestion on the grid, transmission line losses (est. 6% in 2006 [114]), and boundaries between regions introduce distribution inefficiencies that limit how electricity can flow and affect prices.

Consumers usually do not directly draw power from the grid transmission lines. Electricity from the grid is transformed to lower voltages—possibly passing through multiple substations and transformers—and transferred using a secondary distribution network. Like other industrial consumers, large data centers draw power from dedicated substations, a more direct connection to the grid than residential and typical business consumers.

[2] Pumped hydro (pumping water to a higher altitude to store, then letting it flow down through a turbine to recover energy) is the most widely used grid storage technique. Compressed air (pumping air into underground caverns) is another option, but is hard to implement. Flywheels (large discs that spin in a vacuum) and batteries can also be used, but have scaling issues. Newer technologies like ultracapacitors and fuel cells show promise, but are as yet unproven.

Market Structure. In each region, a pseudo-governmental body, a Regional Transmission Organization (RTO), manages the grid. An RTO provides a central authority that sets up and directs the flow of electricity between generators and consumers over the grid. RTO's also provide mechanisms to ensure the short-term reliability of the grid. Figure 5-2 lists the main RTO's and regions we study in this dissertation.

Additionally, RTO's administer *wholesale* electricty markets. While bilateral contracts account for the majority of the electricity that flows over the grid, wholesale electricity trading has been growing rapidly, and presently covers about 40% of total electricity usage.

Wholesale market participants can trade forward contracts for the delivery of electricity at some specified hour. In order to determine prices for these contracts, RTO's often use an auctioning mechanism [75, 90]: power producers present supply offers (possibly price sensitive), consumers present demand bids (possibly price sensitive); and the coordinating body sets prices and determines how electricity should flow. The market clearing process sets hourly prices for the different locations in the market. The outcomes depend not only on bids and offers, but also account for a number of constraints (grid-connectivity, reliability, etc.).

Each RTO usually operates multiple parallel wholesale markets. There are two common market types:

Day-ahead markets (futures) provide hourly prices for delivery during the following day. The outcome is based on expected load[3].

Real-time markets (spot) are balancing markets where prices are calculated every five minutes or so (hourly prices are integrals), based on actual conditions, rather than expectations. Typically, this market accounts for a small fraction of total energy transactions (e.g., less than 10% of the total in NYISO).

Generally speaking, the most expensive active generation resource determines the market clearing price for each hour. The RTO attempts to meet expected demand while minimizing total operating cost, activating a subset of production resources and

[3] Hour-ahead markets, not discussed here, are analogous.

selecting the most appropriate power-plant loads. Consequently, when demand is low, the base-load power plants, such as coal and nuclear can be used to fulfill it; when demand rises, additional resources, such as natural gas turbines, may be activated, driving up prices.

Security constraints, line losses and grid *congestion costs* also impact prices. When transmission network restrictions, such as line capacities, prevent the least expensive energy supplier from serving demand, *congestion* is said to exist. More expensive generation units will then need to be activated, elevating prices. Some markets include an explicit congestion cost component in their prices, partly to incentivize behaviour that will reduce congestion.

Surprisingly, negative prices can show up for brief periods, representing conditions where if energy were to be consumed at a specific location at a specific time the overall efficiency of the system would increase.

Furthermore, market boundaries introduce transmission inefficiences and economic discontinuities. As we shall see later, even geographically close locations in different markets tend to see uncorrelated prices. Part of the problem is that different markets have evolved using different rules, pricing models, etc.

Clearly, the market for electricity is complex. In addition to the factors mentioned here, many local idiosyncrasies exist. In this dissertation, we use a relatively simple market model that assumes the following:

Visible Variation. Prices are known and vary hourly.

Metered Billing. The electric bill paid by the service operator is proportional to consumption and indexed to hourly wholesale prices.

Price-taking. The request routing behavior induced by our method does not significantly alter prices and market behavior.

The validity of the second assumption depends upon the extent to which companies hedge their energy costs by contractually locking in fixed pricing. The third assumption is a reasonable starting point, but may not hold when a cluster accounts for a large fraction of power consumption at a market location. In chapter 8 we will

explore the implications of relaxing these assumptions. For more details on markets, the reader may refer to the large body of economic literature that deals with the structure and evolution of energy markets [112, 90, 75], market failures, and arbitrage opportunities for securities traders (e.g. [107, 65]).

5.3 Empirical Market Analysis

We posit that imperfectly correlated variations in electricity prices give rise to economic opportunities that can be exploited by the operators of large geographically distributed systems. Rather than presenting a theoretical discussion, we build an argument using empirical evidence.

Before introducing a system model, in this chapter we focus exclusively on market price data. Of the economic opportunities that we identify, not all of them may be exploitable by real systems. We ground our analysis in historical market data aggregated from government sources [112, 111], trade publication archives [96], and public data archives maintained by the different RTO's. We use price data for 30 locations, covering January 2006 through March 2009.

Our work mostly focuses on real-time markets. Our eventual goal is to embrace and exploit geographically uncorrelated price volatility. As we shall see, real-time markets tend to be more volatile than day-ahead markets.

5.3.1 Price Variation

Geographic price differentials are what really matter to us, but it is useful to first get a feel for the behaviour of individual prices.

Daily Variation. Figure 5-3 shows daily average prices for six locations[4], from January 2006 through April 2009. Although prices are moderately stable at long time scales, they exhibit a significant amount of day-to-day volatility, short-term

[4] The Northwest is an important region, but lacks an hourly wholesale market, forcing us to omit the region from the remainder of our analysis. Similarly, a lack of hourly data leads us to omit the Southeast from our later analysis.

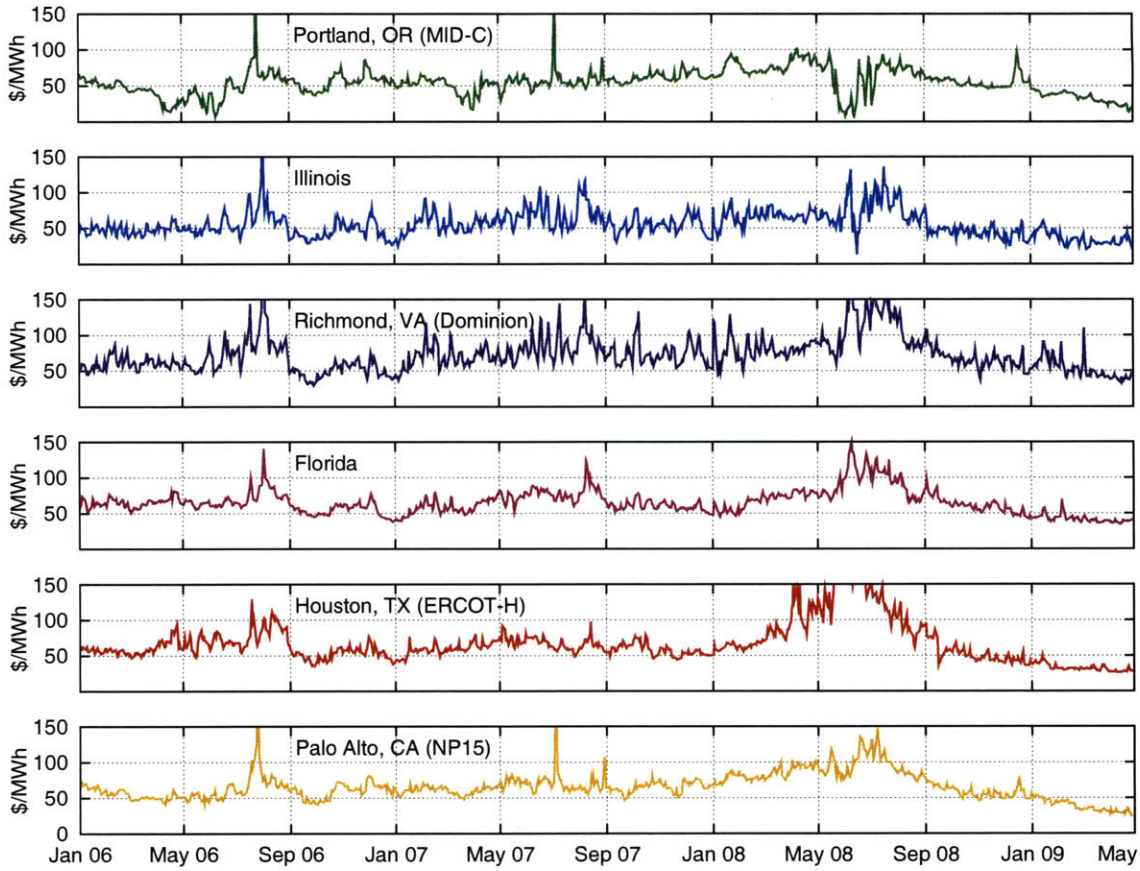


Figure 5-3: Daily averages of day-ahead peak prices at different locations [96]. The elevation in 2008—everywhere but the hydroelectric dominated Northwest—correlates with record high natural gas prices. The Northwest consistently dips near April (this seems to be correlated with seasonal rainfall). Correlated with the global economic downturn, all prices shown here exhibit a downward trend.

spikes, seasonal trends, and dependencies on fuel prices and consumer demand. Some locations in the figure are visibly correlated, but we will see later that hourly prices are not well correlated (§5.3.2).

Different Market Types. Spot and futures markets have different price dynamics. Figures 5-4 and 5-5 illustrate the difference for NYC. Compared to the day-ahead market, the hourly real-time (RT) market is more volatile, with more high-frequency variation, and a lower average price. The underlying five-minute RT prices are even more volatile. Figure 5-6 provides statistics for hourly prices at other locations.

From this point forward, we focus exclusively on the more volatile RT markets.

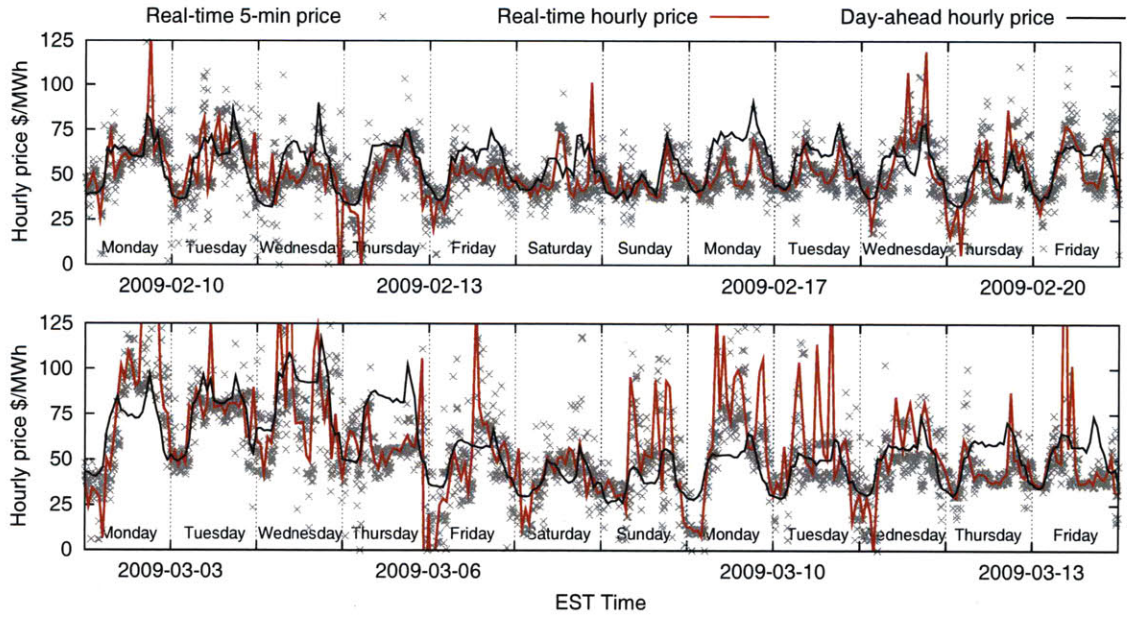


Figure 5-4: Comparing price variation in different wholesale markets, for New York City. The top graph shows a period when prices were similar across all markets; the bottom graph shows a period when there was significantly more volatility in the real-time market.

Window	**5 min**	**1 hr**	**3 hr**	**12 hr**	**24 hr**
Real-time σ	28.5	24.8	21.9	18.1	15.6
Day-ahead σ	N/A	20.0	19.4	17.1	16.0

Figure 5-5: The real-time market is more variable at short time-scales than the day-ahead market. Standard deviations for Q1 2009 prices for NYC are shown, averaged using different window sizes.

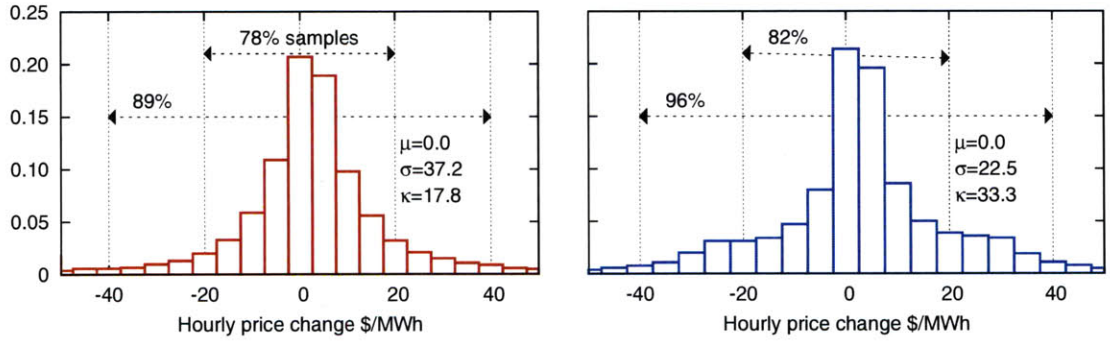
We restrict ourselves to hourly prices, but speculate that the additional volatility in five-minute prices provides further opportunities.

Hour-to-Hour Volatility. As seen in figure 5-4, the hour-to-hour variation in NYC's RT prices can be dramatic. This degree of volatility is common at a number of different locations. For example, figure 5-7 shows the distribution of the hourly change for Palo Alto and Chicago. At each location, the price per MWh changed hourly by \$20 or more roughly 20% of the time. A \$20 step represents 50% of the mean price for Chicago. Furthermore, the minimum and maximum prices during a single day can easily differ by 2×.

The existence of rapid price fluctuations reflects the fact that short term demand for electricity is far more elastic than supply. Electricity cannot always be efficiently

Location	RTO	Mean*	StDev*	Kurt.*
Chicago, IL	PJM	40.6	26.9	4.6
Indianapolis, IN	MISO	44.0	28.3	5.8
Palo Alto, CA	CAISO	54.0	34.2	11.9
Richmond, VA	PJM	57.8	39.2	6.6
Boston, MA	ISONE	66.5	25.8	5.7
New York, NY	NYISO	77.9	40.26	7.9

Figure 5-6: Real-time market statistics, covering hourly prices from January 2006 through March 2009 (*statistics are from the 1% trimmed data).



(a) Palo Alto (NP15) **(b) Chicago (PJM)**

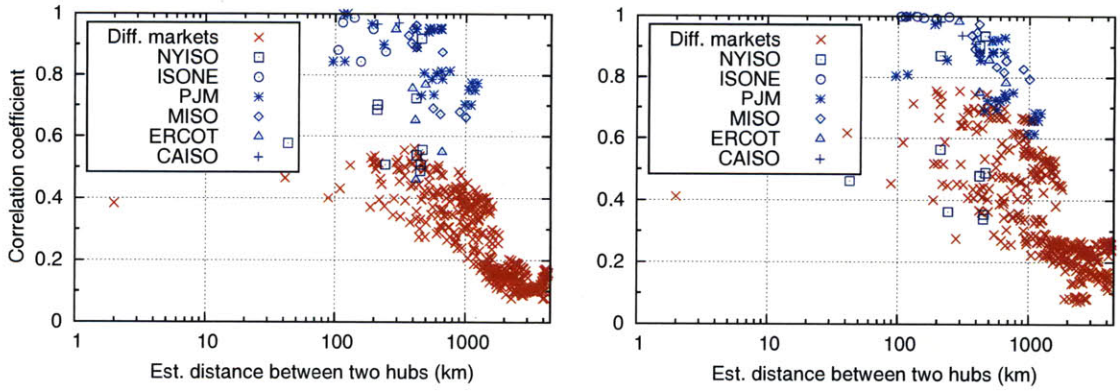
Figure 5-7: Histograms of hour-to-hour change in real-time hourly prices for two locations, over the 39-month period. Both distributions are zero-mean, Gaussian-like, with long tails.

moved from low demand areas to high demand areas, and producers cannot always ramp up or down easily.

5.3.2 Geographic Correlation

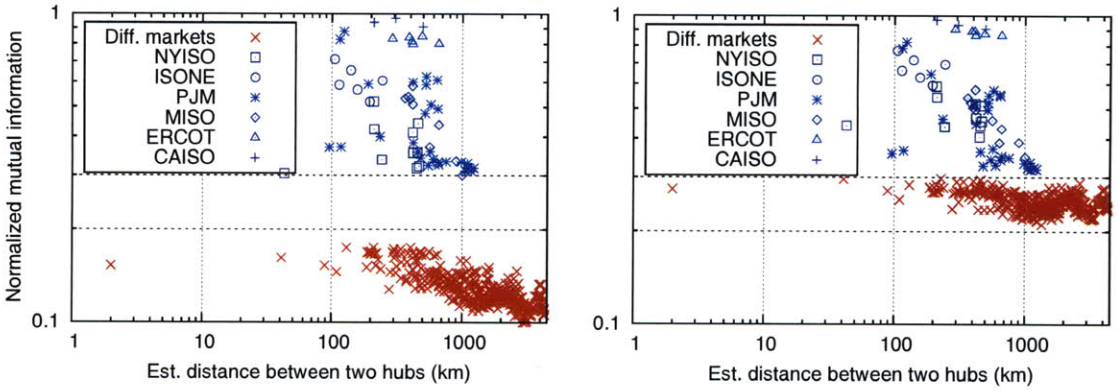
Our approach would fail if hourly prices are well correlated at different locations. However, in our data, we found that locations in different regional markets were never highly correlated, even when nearby, and that locations in the same region were not always well correlated. We also found that correlation tended to decrease with increasing distance.

Figure 5-8 shows scatter-plots of pairwise correlation and geographic distance, calculated over two time periods. No pairs were negatively correlated. Note how correlation decreases with distance. Further, note the impact of RTO market boundaries: for the longer time period, most pairs drawn from the same market lie above the 0.6 correlation line (except for locations in the NYISO market), while all pairs



(a) all data (01/2006-04/2009) **(b) 9 months (7/2008-04/2009)**

Figure 5-8: The relationship between price correlation, distance, and parent market. Each point represents a pair of locations (29 locs, 406 pairs), and the correlation coefficient of their hourly prices over the specified period. Red represents pairs with sites from different markets; blue points are labeled with the shared market.



(a) all data (01/2006-04/2009) **(b) 9 months (7/2008-04/2009)**

Figure 5-9: The relationship between price MI, distance, and parent market. Each point represents a pair of locations (29 locs, 406 pairs), and the normalized MI of their hourly prices over the specified period. Red represents pairs with sites from different markets; blue points are labeled with the shared market.

from different regions lie below it. For the shorter time period, the division is less clear. We also see a surprising lack of diversity within some regions, like California: LA and Palo Alto have a correlation coefficient close to 1.0.

Correlation coefficients only test for a linear relationship between two random variables. In contrast, a metric like *Mutual Information* (MI) also tests for non-linear relationships[5]. Figure 5-9 shows scatter-plots of pairwise MI and distance, calculated

[5] Given two random variables X and Y, their MI quantifies how knowing the value of one decreases our uncertainty about the other's value.

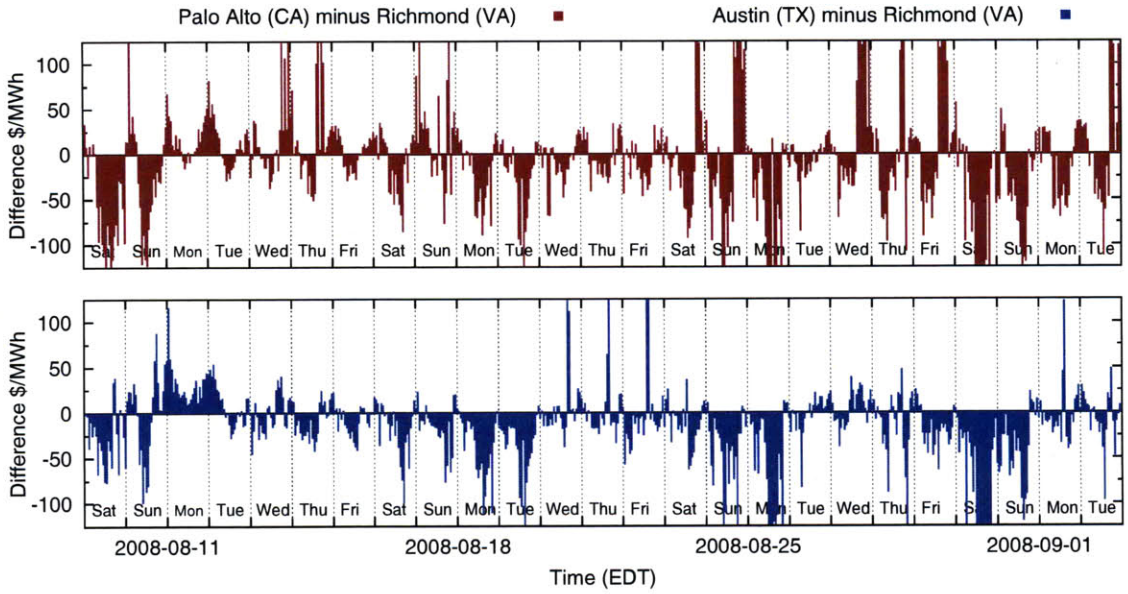


Figure 5-10: Variation of price differentials with time.

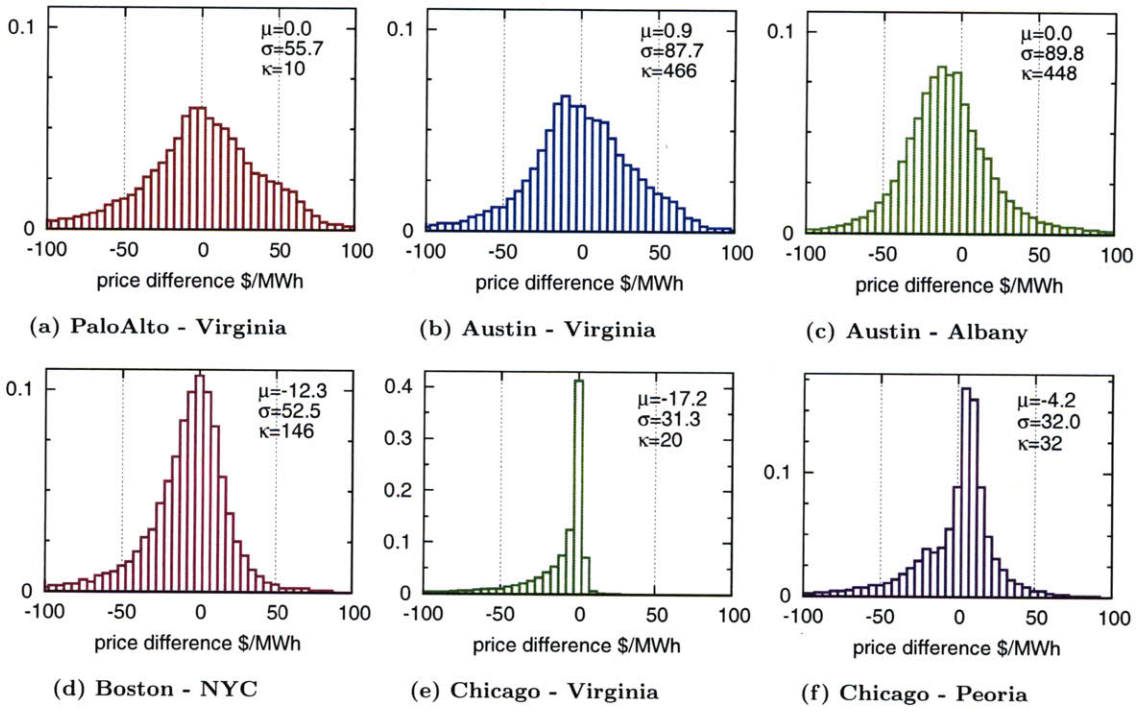
over the same two time periods. MI much more clearly divides the data between single-market and multiple-market pairs. This suggests that the small overlap in figure 5-8 is due to the existence of non-linear relationships within NYISO, not detected by the correlation coefficient.

Thus, we can conclude that hourly prices from different markets are not well correlated at short time-scales, and that correlation tends to decrease with distance, except in some markets like California. We have verified our results using other time ranges from the data, shifted signals, etc.

We have not tested for lower-frequency correlations. We expect that some such relationships exist. Natural gas prices, for example, will introduce some coupling (see figure 5-3) between distant locations.

5.3.3 Price Differentials

Figure 5-10 shows hourly price differentials for two pairs of locations over an eight day period (both pairs have mean differentials close to zero). The three locations are far from each other and in different RTOs. We see price spikes (some extend far off the scale) and extended periods of price asymmetry. Sometimes the asymmetry



(a) PaloAlto - Virginia (b) Austin - Virginia (c) Austin - Albany

(d) Boston - NYC (e) Chicago - Virginia (f) Chicago - Peoria

Figure 5-11: Hourly price differential histograms for six location pairs (2006-2009).

favours one, sometimes the other. This suggests that a pre-determined assignment of clients to servers is not optimal.

Differential Distributions. Consider a system with clusters at two locations. In order for our dynamic approach to yield substantial savings over a static solution, the price differential between those locations must vary in time, and the distribution of this differential should ideally have a zero mean and a high variance. Such a distribution would imply that neither site is strictly better than the other, but also that a dynamic solution—always buying from whichever site is least expensive that hour—could yield meaningful savings.

Figure 5-11 shows the pairwise differential distributions for some locations, for the 2006-2009 RT market data. The California-Virginia (figure 5-11a), Texas-Virginia (figure 5-11b), and Texas-NewYork (figure 5-11c) distributions are zero-mean with a high variance. There are many other such pairs[6].

[6] There are 60 pairs (a set of 16 hubs) with $|\mu| \leq 5 \wedge \sigma \geq 50$; and 86 pairs (a set of 28 hubs) with $|\mu| \leq 5 \wedge \sigma \geq 25$.

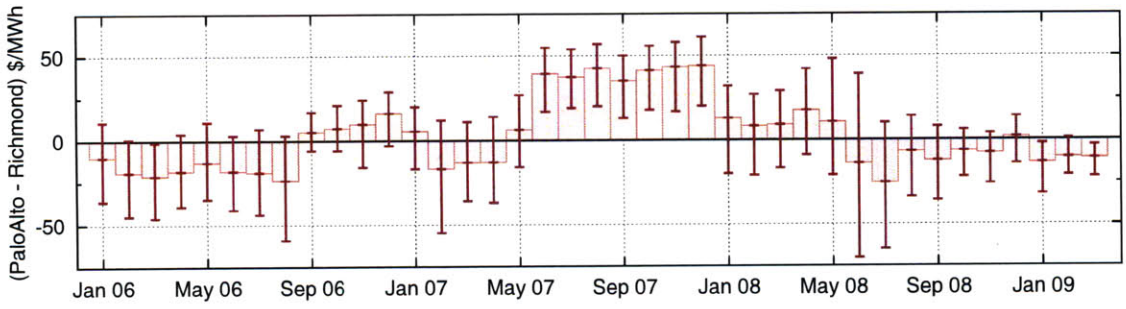


Figure 5-12: PaloAlto-Virginia price differential distributions for each month. The monthly median prices and inter-quartile range are shown.

Even with significantly skewed distributions, there may exist opportunities to dynamically exploit differentials for meaningful savings. For example, Boston-NYC (figure 5-11d) is skewed, since Boston tends to be cheaper than NYC, but NYC is less expensive 36% of the time (the savings are greater than \$10/MWh 18% of the time).

Unsurprisingly, a number of pairs exist where one location is strictly better than the other, and dynamic adaptation is unnecessary. Chicago-Virginia (figure 5-11e) is an example: Virginia is less expensive 8% of the time, but the savings almost never exceed \$10/MWh.

The dispersion introduced by a market boundary can be seen in the dynamically exploitable Chicago-Peoria distribution (figure 5-11f).

Evolution in Time. The price differential distributions do not remain static in time. Figure 5-12 shows how the PaloAlto-Virginia distribution changed from month to month. A sustained price asymmetry may exist for many months, before reversing itself. The spread of prices in one month may double the next month.

Time-of-Day. Price differentials can depend on the time-of-day. For instance, because California and Virginia are in different time zones, peak demand is out of phase. This is likely a factor shaping the price differential.

Figure 5-13 shows how the hour of day affects the differentials for three location pairs. For California-Virginia, we see a reasonably strong dependency on the hour.

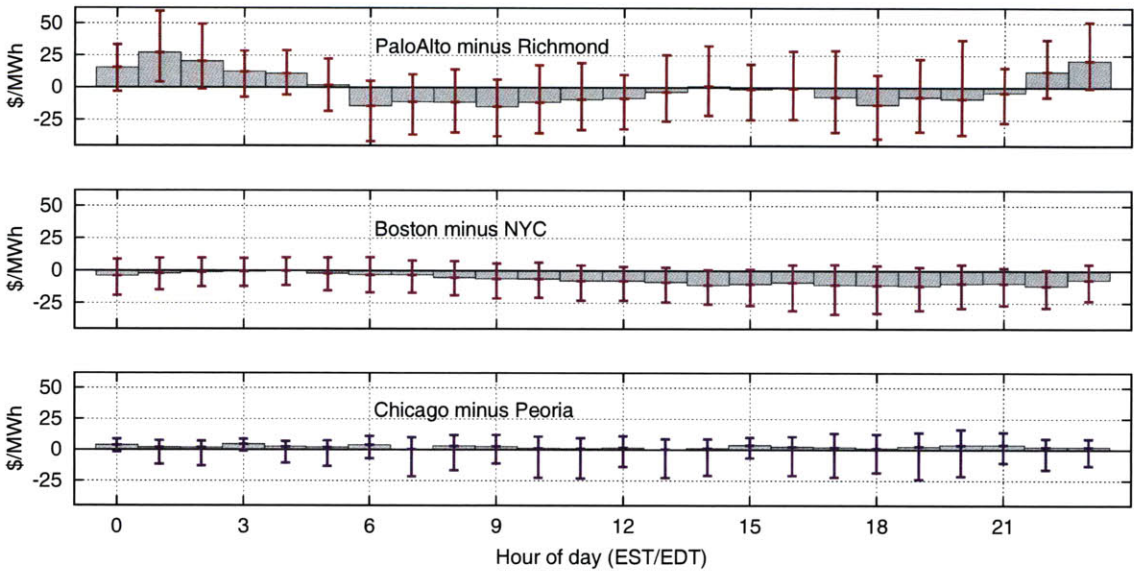


Figure 5-13: Price differential distributions (median and inter-quartile range) for each hour of the day.

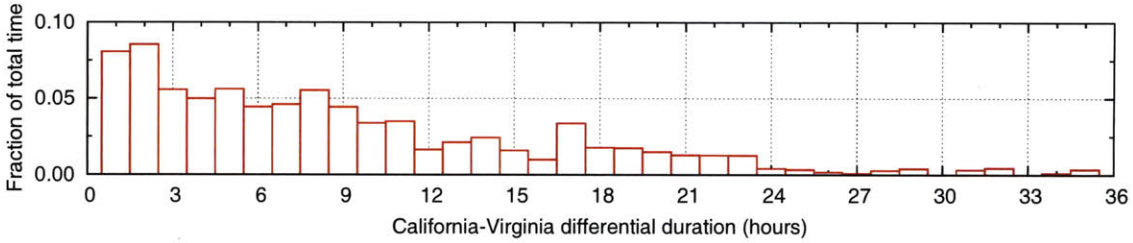


Figure 5-14: For PaloAlto-Virginia, short-lived price differentials account for most of the time.

Before 5am (eastern), Virginia has a significant edge over California; by 6am the situation has reversed; from 1-4pm neither is better. For Boston-NYC we see a different kind of dependency: from 1am-7am neither site is better, at all other times Boston has the edge. The effect of hour-of-day on Chicago-Peoria is less clear, which is not surprising, since the spread is likely the result of a market boundary effect.

Differential Duration. We define the *duration* of a sustained price differential as the number of consecutive hours one location is favoured over another by more than \$5/MWh. As soon as the differential falls below this threshold, or reverses to favour the other location, we mark the end of the differential.

Figure 5-14 shows how much time was spent in short-duration price-differentials,

for PaloAlto-Virginia. Short differentials (<3 hrs) are more frequent than other types. Medium length differentials (<9 hrs) are common. Differentials that last longer than a day are rare for a balanced pair.

Summary

This chapter introduced how power-demand routing can be coupled with electricity market price signals to lower electric bills. We provided motivation for lowering bills by estimating the server electricity expenditures of some large Internet companies, and showing that even 1% reductions in their bills could exceed a million dollars a year. We also provided a primer on wholesale electricity markets. Finally, we described an empirical analysis of historical market price data. We identified several features in the data that could be exploited using PDR. This sets the stage for the next chapter, in which we combine market data with workload and energy models to quantify the savings in electricity costs that PDR could achieve.

Chapter 6.

Quantifying Savings

One of the central theses of this dissertation is that PDR can be coupled to real-time market prices and used to reduce the electricity expenses of large distributed systems. In order to test this thesis, we conducted a number of simulations, quantifying and analyzing the impact of different routing policies on energy costs and client-server distances. Our simulations were driven by empirical data and models and combined many of the pieces described earlier in this dissertation. We show that:

Existing systems can benefit from PDR. Existing systems like Google's can save more than $2M annually without a noticeable impact on network latencies, or up to $6.5M annually if latency is not a concern.

PDR will become more relevant as technology progresses. Savings accelerate with energy proportionality and current technology trends will lead to increasing proportionality. With our most futuristic model, energy generation costs can be cut by over 30%.

One can trade-off performance and savings. Savings are a non-decreasing function of the average client-server distance.

We also explore some other issues: a complex network cost model; different server geo-distributions; and the performance of a PDR with imperfect knowledge.

6.1 Simulation Strategy

We constructed a simple trace-driven simulator to evaluate power-demand routing. The simulator was built using the pieces we have described earlier: cluster energy models (chapter 3); request workload models (chapter 4); and an historical electricity price database (chapter 5). We also implemented two different PDR algorithms in the simulator (discussed below). We focused on how energy costs with these algorithms compared to the energy costs with routing schemes that were unaware of cluster cost differentials. By simulating different energy models and several variations of the distributed system model, we explored the sensitivity of our PDR algorithms to different factors. Before presenting our results, we provide some details about our simulation setup and PDR algorithms.

6.1.1 Simulation framework.

Each simulation represented a specific historical time period. Our simulations ranged from 24-day periods to 39-month periods. We simulated each hour independently. A simulator would generate a traffic volume matrix for the hour and pass that matrix to a routing module with a global view of the system. Using the $w_{i,j}$ ingress traffic splits returned by the router[1], the simulator would determine the load on each cluster, and use an energy model to map load to energy consumption. We then calculated the energy generation cost[2] by using our electricity price database (for brevity, we will refer to generation cost simply as energy cost from here on). When looking up the electricity price for a cluster, we used that cluster's geographic location and the exact date and time of the simulated hour. We also calculated average client-server distances for each simulated hour. For the most part, we assume network bandwidth prices are equal everywhere (except in §6.7).

[1] This is the router formulation described in chapter 2.

[2] Generation cost = number of kWh consumed × price per kWh. This is the component of the electric utility bill that depends on market prices and total consumption, roughly half the utility bill. See §5.1 for details.

In addition to the time period, a simulation needed three more parameters: a router; an energy model; and a cluster geo-distribution.

Router: a router implementation (described in the next section). Some routers have multiple parameters, allowing us to simulate a wide range of routing policies.

Energy Model: one of the 25 cluster models from chapter 3. Figure 6-1 tabulates these models again. Each cluster used the same base energy model, with the model's output scaled by the cluster's server count and a peak server power of 150W.

Geo-distribution: the geographic locations of the clusters and the numbers of servers in each cluster. In actual systems, there is no dominant distribution pattern.

Most of our simulations used a geographic server distribution based on Akamai's actual distribution[3] The distribution we used consisted of 9 clusters and 12.5K servers[4]. Servers were spread unevenly: the smallest cluster had less than 0.5% of the servers; the largest had a little over 18%. The maximum request processing rate for a server varied with its cluster (we calculated these rates from data provided by Akamai). Finally, the system's total request processing capacity was about 3× the peak traffic volume in the raw 24-day trace.

We also used some synthetic cluster distributions (described later). The synthetic distributions were based on what we know to be popular data center locations. Unlike the Akamai distribution, our synthetic distributions had balanced clusters: every cluster had the same number of servers and all servers had the same request rates. The aggregate request capacity was the same as the Akamai geo-distribution's capacity, to avoid having to scale the traffic workload.

[3] In the course of our collaboration, we obtained detailed information about the placement of a large subset of Akamai's servers: almost 16K servers, or about 33% of their servers, located in 25 cities and 12 states. Akamai has many small clusters; we grouped clusters by city. We cannot publicly disclose details about this server distribution—this information is considered confidential by Akamai.

[4] These nine clusters cover 18 of the 25 cities, with cities grouped by their electricity market hubs. We excluded 7 cities because we did not have electricity prices for those cities.

ID	Idle server power	PUE Peak load	PUE Idle	Comments	EPG	EAP
Fixed Overhead Models						
Linear IT Models ($P_{IT} = m \cdot u + c$)						
L1	65%	1.90	2.35		0.15	0.18
L2	50%	1.70	2.33		0.24	0.28
L3	33%	1.50	2.38		0.42	0.36
L4	25%	1.30	2.04		0.47	0.55
L5	20%	1.10	1.42		0.59	0.69
Google IT Models ($P_{IT} \sim u^{1.4}$)						
G1	65%	1.90	2.35	e.g., EPA 2011 (current-trends).	0.21	0.22
G2	50%	1.70	2.33		0.33	0.36
G3	40%	1.50	2.16		0.45	0.49
G4	33%	1.50	2.38		0.50	0.54
G5	22.5%	1.50	2.90		0.58	0.63
G6	25%	1.30	2.04	e.g., EPA 2011 (best-practices).	0.64	0.70
G7	22.5%	1.20	1.76	e.g., Microsoft containers.	0.72	0.78
G8	20%	1.10	1.42		0.81	0.88
G9	10%	1.10	1.69	Best current tech.	0.91	0.99
Variable Overhead Models						
X1	25%	1.70	1.54	IT $\sim u^{1.4}$; cooling is 52% of total power at peak; inefficient UPS.	0.75	0.84
X2	20%	1.30	1.24	IT $\sim u^{1.4}$; cooling is 17.5%.	0.83	0.92
X3	47%	1.50	1.35	IT $\sim$ Google curve; 36% on cool.	1.17	1.16
X4	22%	1.50	1.37	IT $\sim$ TX150 curve; 36% on cool.	0.63	0.69
X5	26%	1.50	1.35	IT $\sim$ R610 curve; 36% on cool.	0.83	0.88
Dynamic Server Scaling (cooling is 30% at peak)						
D1	25%	1.43	1.91	IT $\sim u^{1.4}$; 50-server blocks.	0.68	0.85
D2	25%	1.43	1.56	IT $\sim u^{1.4}$; 250-server blocks.	0.69	0.83
D3	25%	1.43	1.40	IT $\sim u^{1.4}$; 500-server blocks.	0.67	0.84
D4	26%	1.43	1.78	IT $\sim$ R610; 100-server blocks.	0.69	0.80
Cutting-Edge Clusters (IT $\sim u^{1.4}$; distributed UPS; 17.5% on cooling)						
E1	20%	1.21	1.15	e.g., Google data center.	0.84	0.93
E2	20%	1.21	1.41	with 50-server block DSS.	0.74	0.91

Figure 6-1: A collection of cluster energy models. See §3.7 for EPG and EAP. This is a reproduction of figure 3-14. Models with DSS turn off idle servers.

6.1.2 Routing Algorithms

***nlp* router.** This router sets up the optimization problem from chapter 2 as a non-linear programming problem and calls a generic NLP solver to find a low-cost traffic allocation. We incorporate performance goals using a strict geo-distance constraint

parameter. Figure 6-2 provides pseudocode.

In the implementation, for performance reasons, we transform the ingress-cluster graph in order to reduce the number of variables used by the NLP solver. We first remove edges that will never be used because of the performance constraint, then group ingress nodes together whenever they have the same edge profile[5]. The *nlp* router is implemented in Python, using the `OpenOpt` framework's `ralg` solver [14]. We selected this solver because it can handle non-smooth problems (some of our energy models result in non-smooth cost functions). The `ralg` solver does not guarantee that it will find an optimal solution. It uses heuristics and convergence hasn't been proven. Nonetheless, the solver performs quite well in our simulations, consistently yielding lower-cost allocations than the other routers we tested. Furthermore, our implementation is efficient. Using a single Core2 core, it requires hours to simulate weeks. Thus our *nlp* router should be usable in practice. However, this router was not efficient enough to allow us to run the many multi-year simulations we needed for our analysis. We had to develop another router for that purpose.

***radial* router.** The *radial* router is an iterative greedy allocator that is about 500× faster than the *nlp* router. In our simulations it consistently obtained energy cost reductions that were within 5% of the *nlp* router.

This router is inspired by the following approach. Given an ingress, first find the geographically closest cluster. Suppose the distance between the ingress and this cluster is D. Then consider all other clusters in an area of radius $(D + \tau)$ around the ingress, where τ is a performance constraint parameter. Of these candidate clusters, select the cluster with the best energy cost characteristics and assign a portion of the ingress's traffic to it. Repeat the process until all traffic has been allocated.

Pseudocode for the *radial* router is shown in figure 6-3. There are two variations of this router: one selects the candidate cluster with the lowest price (*radial-price*); and the other selects the cluster with the minimal marginal increase in energy cost

[5] e.g., if ingress 1 and 2 both have edges to clusters 3, 4, and 5, the ingress nodes are combined into an aggregate ingress with the sum of their traffic volumes; while ingress 3, with edges to only 3 and 4, is left unchanged.

parameters :
$maxDist \leftarrow$ maximum allowed client-server distance (e.g., 1000km)

GenerateRoutesNLP() :
C is the set of clusters
I is the set of ingresses
$volume(i)$ is the request traffic volume originating at ingress i
$capacity(j)$ is the maximum traffic volume cluster j can serve

$W_0 \leftarrow$ generate initial solution

$X_0 \leftarrow$ weight constraints
$\forall i \in I, \forall j \in C$: $0 \leq w_{i,j} \leq 1$
$\forall i \in I$: $\sum_{j \in C} w_{i,j} = 1$

$X_1 \leftarrow$ cluster capacity constraints
$\forall j \in C$: $\sum_{i \in I} w_{i,j} \cdot volume(i) \leq capacity(j)$

$X_2 \leftarrow$ performance constraints (strict)
$\forall i \in I, \forall j \in C$: **if** $GeoDist(i,j) > maxDist$ **then** $w_{i,j} = 0$

// optimization cost function
$Cost(W')$: // W' is a candidate solution: $\{w'_{i,j}\}$
$\forall j \in C$: $alloc_j = \sum_{i \in I} w'_{i,j} \cdot volume(i)$
return $\sum_{j \in C} ModeledEnergyCost(j, alloc_j)$

$W \leftarrow$ **call** NLP optimizer with W_0, X_1, X_2, X_3 and $Cost$
if optimizer did not find a valid solution **then**: $W \leftarrow GenerateRoutesRadial()$
return W

Figure 6-2: Pseudocode for the *nlp* router. *GeoDist* calculates the great-circle geographic distance, and *ModeledEnergyCost* uses an energy model and the local electricity prices to calculate a cluster's energy cost from a traffic volume.

(*radial-gradient*). The marginal cost variant performs no worse than the price variant and promises to perform better with non-linear energy models and unevenly sized clusters. The price variant does not rely on an energy model, so it can be used when cluster energy curves are unknown. From this point on, unless otherwise specified, *radial* refers to the *gradient* variation.

In addition to the distance threshold, a price (or gradient) threshold can be set, so that the router will ignore small cost differentials. In §6.7 we also show how to modify the radial router to incorporate a more complex network cost model.

parameters :

$\tau_D \leftarrow$ client-server distance threshold (e.g., 1000km)
$\tau_P \leftarrow$ price threshold (e.g., \$5)
$\tau_\Delta \leftarrow$ cost gradient threshold

GenerateRoutesRadial() :

while some unallocated traffic **do**
for all i such that i is an ingress with some unallocated traffic **do**
$V \leftarrow$ total traffic volume at i
$vol \leftarrow$ unallocated traffic volume for i

// pick best cluster
do not consider any clusters operating near their capacity
$c* \leftarrow$ cluster nearest to i
$C' \leftarrow \{$cluster c such that $GeoDist(c,i) \leq GeoDist(c*,i) + \tau_D\}$
$j \leftarrow BestCost(C', i, vol)$ // one of the two *BestCost* functions shown below

// allocate some traffic to cluster j
$cap \leftarrow$ remaining capacity at c'
$w_{i,j} \leftarrow min(vol, cap)/V$
end for
end while
return $\{w_{i,j}\}$

BestCostPrice(C, i, vol) :

$p* \leftarrow$ minimum energy price for clusters in C
$C' \leftarrow \{$cluster c if energy price of $c \leq p* + \tau_P\}$
return cluster in C' with minimum $GeoDist(c,i)$

BestCostGradient(C, i, vol) :

$c* \leftarrow None$; $\Delta c* \leftarrow None$ // goal is to find cluster with min. Δc
for all $c \in C$ **do**
$cvol \leftarrow$ traffic volume currently allocated to cluster c
$cap \leftarrow$ remaining capacity at c'
$\Delta v \leftarrow min(vol,\ cap - cvol)$ // max. possible volume we can alloc.
$\Delta c \leftarrow ModeledEnergyCost(c, cvol + \Delta v) - ModeledEnergyCost(c, cvol)$
if $|\Delta c - \Delta c*| \leq \tau_\Delta$ **then**
if $cvol >$ remaining capacity at $c*$ **then**
$c* \leftarrow c$; $\Delta c* \leftarrow \Delta c$
end if
else if $\Delta c < \Delta c*$ **then**
$c* \leftarrow c$; $\Delta c* \leftarrow \Delta c$
end if
end for
return $c*$

Figure 6-3: Pseudocode for the *radial* router.

Cost unaware schemes. In order to determine whether PDR is cutting electric bills or not, we need a baseline bill to compare against. For each simulation we ran with a *radial* or *nlp* router, we ran a simulation with almost identical parameters, different only in its use of a router that was unaware of cluster energy costs.

One of our cost unaware routers replays Akamai's traffic allocation (from the raw traces). This only works for the 24-day period those traces cover. For other time periods, we used the *radial* router and replaced *BestCost* with a function that cycles through available clusters. We set performance constraints so that the average client-server distances were similar to those in Akamai's traffic allocation.

6.2 At the Turn of the Year: 24 Days of Traffic

We begin by asking the question: what would have happened if an Akamai-like system had used power-demand routing at the end of 2008? How would PDR have compared in cost and client-server distance to the current routing methods employed by Akamai?

To quantify savings, we ran simulations with the 24-day traffic traces and the Akamai geo-distribution. We find that a 10% or higher reduction in electricity generation costs is plausible, without a meaningful increase in average client-server distance. The degree to which costs can be reduced hinges on the energy characteristics of the system's clusters. Savings could be below 1% or more than 30%. We will discuss the relationship between proportionality and savings in some detail (§6.5), but, for now, we restrict ourselves to the `G6` cluster energy model (see figure 6-1). This model represents advanced energy efficient server clusters, but does not factor in state-of-the-art techniques (e.g., dynamic server scaling) that help PDR.

Different Routing Policies. We explored the sensitivity of savings to a number of different routing policies. Figure 6-4 shows some of our results. Both the *nlp* and *radial* routers yield substantial savings.

We simulated the energy cost associated with Akamai's routing scheme and used this cost as a baseline to calculate percentage savings. We also simulated an energy

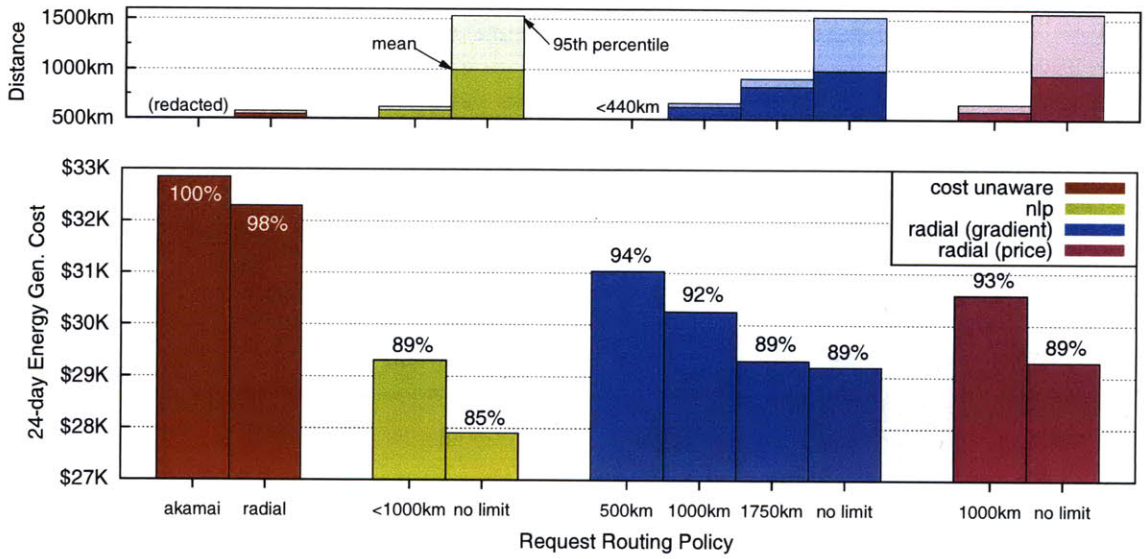


Figure 6-4: 24-day simulation results with the G6 energy model, comparing energy costs (per 10K servers) and client-server distances for several routing policies and the Akamai server geo-distribution.

cost unaware variant of the *radial* router that had acceptably low client-server distances. This later scheme resulted in a lower cost than Akamai's, implying that we need to be careful when our analysis projects small reductions in cost (e.g., $\leq 2\%$).

As the figure shows, the lowest-cost policy—the *nlp* router without any distance constraints—resulted in a reduction in cost of 15%. Extrapolating, this represents an annual savings of almost $6M for an 800K server system like Google's. However, the policy almost doubled average client-server distances and almost trebled the 95-th percentile distances, a measure of worst-case performance[6]. This increase in distance may be acceptable. Recall that a 1000km increase in distance may only represent a $20ms$ increase in average network RTT's (§4.2.2).

When we constrained the *nlp* router to only consider clusters within 1000km of clients, it still achieved savings of 11%. The client-server distances were comparable to those with Akamai's routing policy.

In general, allowing distances to increase, amplifies our ability to exploit cost differentials, and increases achievable savings. This trade-off is shown clearly by how

[6] This is not the 95-th percentile request distance. This is the 95-th percentile of the hourly samples. We only calculated *mean* client-server distance in each hour (see chapter 4's discussion of the census-weighted geo-distance function to understand why).

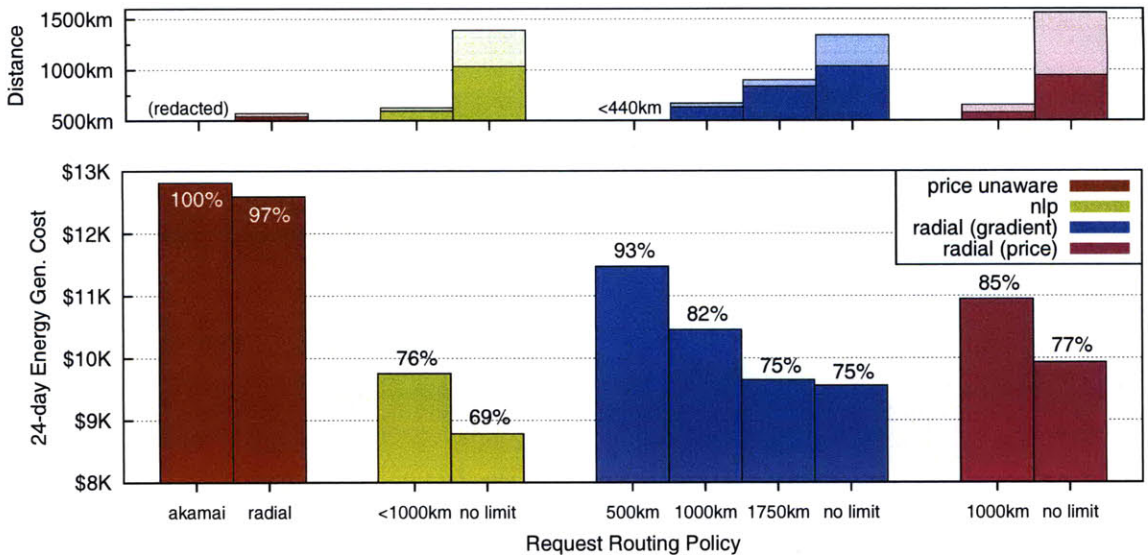


Figure 6-5: 24-day simulation results with the D1 energy model, comparing energy costs (per 10K servers) and client-server distances for several routing policies and the Akamai server geo-distribution.

the *radial* router performed at different distance thresholds (see figure 6-4). We will discuss this relationship in more detail later (§6.4).

Finally, note that the *nlp* router always does better than the *radial* router. However, for a given client-server distance target, the difference in savings between the two is never more than 5%. The price based variant of the *radial* router performs almost as well as the gradient based variant here. The G6 model is close to linear, so this is not surprising.

How sensitive are these results to our choice of G6 as the underlying energy model? Figure 6-5 shows results from a different set of simulations that used the D1 energy model instead (D1 is similar to G6, except it uses DSS).

The D1 model represents clusters that are very efficient at low utilizations, so the baseline generation cost is dramatically lower under D1 than it is under G6. While the baseline cost drops by almost $20K, the *nlp* savings drop by much less ($4.9K for G6; $4K for D1). This is because both models have a similar EPG score (we explore EPG and savings further in §6.5).

The percentage savings are much larger, but the earlier trends remain unchanged. The *radial* gradient-based variant more noticeably outperforms the price-based vari-

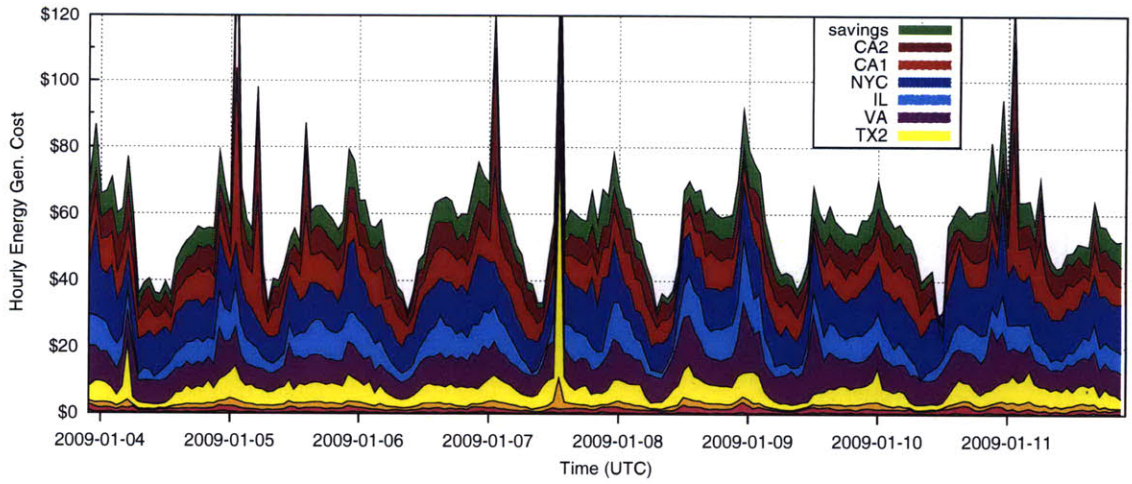


Figure 6-6: Variation in hourly energy costs and savings, with the *nlp* router. Clusters are not evenly sized in the Akamai geo-distribution; the thin slivers at the base of the graph represent the small clusters.

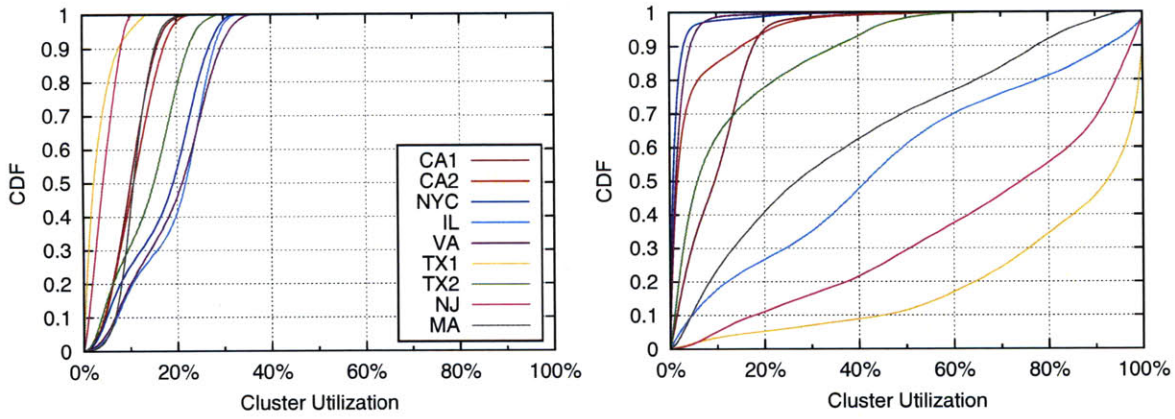
Figure 6-7: CDF of hourly savings over 24 days, for the *nlp* router.

ant, because `D1` deviates further from linear than `G6` does.

Costs in Time. Each cluster's share of the total cost varies in time, depending on its electricity price and the traffic assigned to it. Figure 6-6 illustrates how hourly generation costs vary for the *nlp* router, operating under a 1000km distance constraint.

We see persistent hourly savings. PDR aggregates lots of little pieces, rather than occasionally yielding large chunks. This is also illustrated by figure 6-7, which shows the CDF of hourly savings over the 24-day period.

Cluster Utilization PDR works by shifting load away from high cost regions and, consequently, it increases the utilization of clusters operating in low cost regions.



(a) Akamai **(b) nlp** (1000km constraint)

Figure 6-8: Cluster utilization under different routing policies. In each region PDR increases the average utilization of clusters with low costs. These clusters have unequal capacity: e.g., the MA cluster is about 2% the size of the NYC cluster; the IL cluster is about half the size of the NYC cluster.

Figure 6-8 compares cluster utilizations under Akamai's routing policy to the *nlp* router operating under a 1000km distance constraint.

Akamai's policy keeps every cluster at a similar utilization level. In contrast, the PDR policy raises the average utilization levels of low-cost clusters (e.g., IL) and reduces the utilization levels of high-cost clusters (e.g., NYC). Even though the MA cluster is relatively expensive and very small compared to some of the others, it is used to shift traffic away from NYC whenever possible.

6.3 Synthetic Workload: 39 Months of Prices

The 24-day analysis covered a very small subset of our price data. Using the synthetic network traffic model we developed earlier (§4.1.2) we ran simulations covering January 2006 through March 2009. We replicated our savings analysis for the 3 year time period, using the Akamai server distribution and the `G6` power model. Our results show that savings over the 39-month period are comparable to those during the 24-day period.

Additionally, we analyzed a synthetic cluster distribution as a first step to ensure that our results are not overly sensitive to Akamai's distribution (we consider in detail how server distribution differences can impact savings later, in §6.6). We assumed

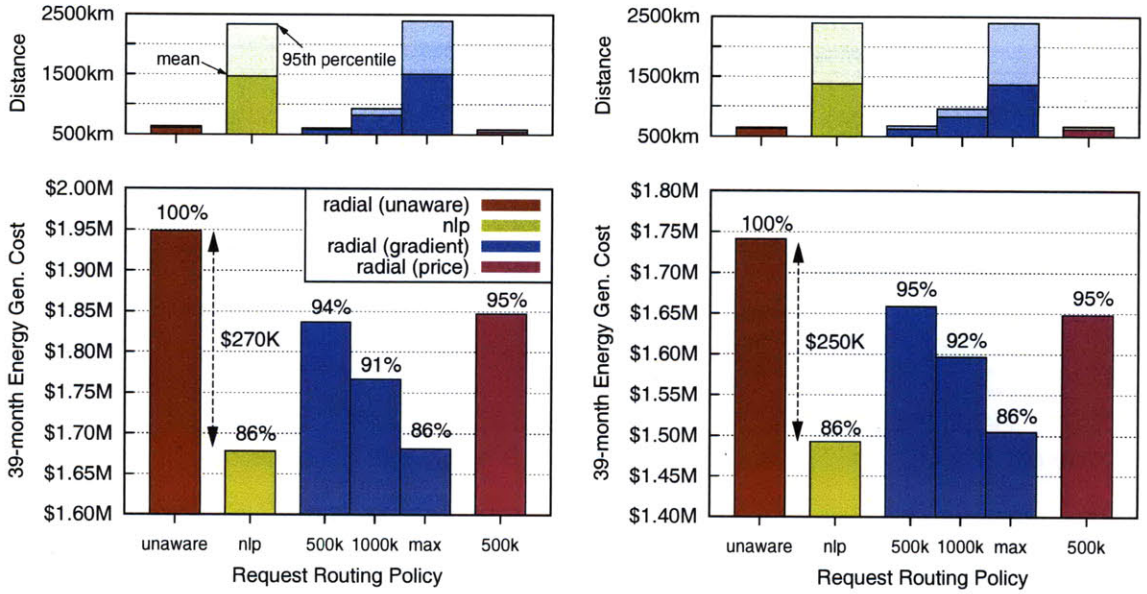


Figure 6-9: 39-month simulation results with the G6 energy model, comparing energy costs (per 10K servers) and client-server distances for several routing policies and two different server geo-distributions.

that servers were evenly divided among clusters (unlike Akamai) and selected five locations, each one a popular data center location for which we had price data:

- Northern California (San Jose)
- Virginia (Richmond)
- Illinois (Chicago)
- Texas (Dallas)
- New York (NYC)

As before, we analyzed the savings achieved by different routing policies. Figure 6-9 summarizes our results for both cluster distributions. For a baseline cost, we used a cost-unaware *radial* router with average client-server distances close to $500km$.

The unconstrained *nlp* router was able to reduce costs by 14% for both distributions. However, distances increased dramatically, with the mean rising to about $1500km$, and the 95-th percentile almost reaching $2500km$. Again, this may be acceptable, since the increase in distance represents a less than $50ms$ increase in network

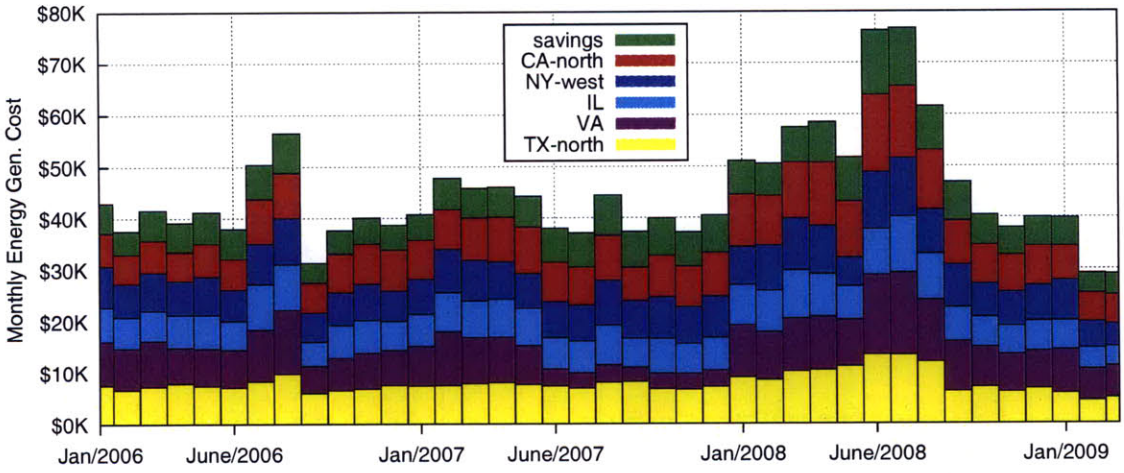


Figure 6-10: Monthly energy costs over 39 months, with the unconstrained *nlp* router. This shows cost per 10K servers, for the synthetic-5 distribution and the `G6` model.

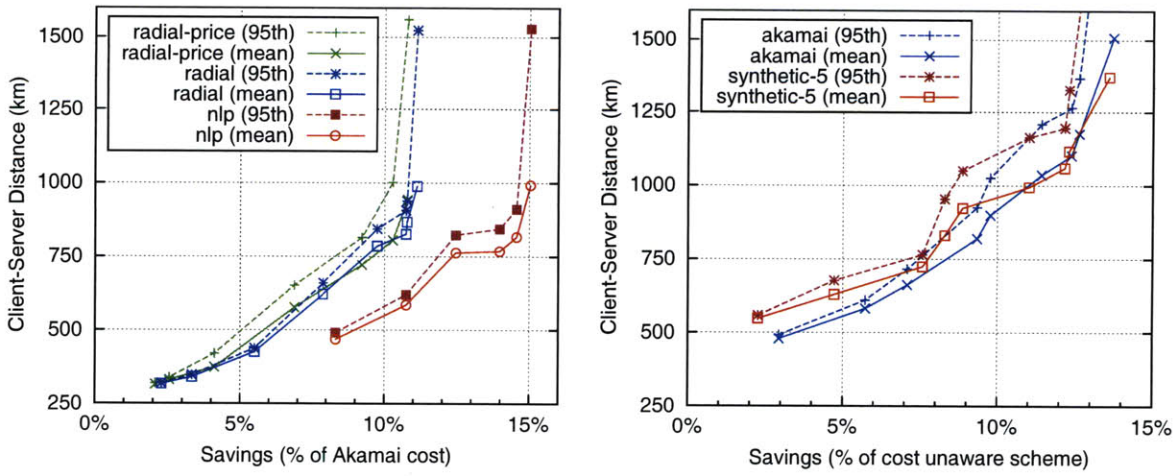
latency (§4.2.2). As we noted earlier (§4.2.3), latency increases of $50ms$ or lower are often considered to be tolerable, as they cause no adverse impact on revenue for services such as web search.

We did not simulate the distance-constrained version of the *nlp* router, since a single 39-month simulation would have taken over a day on our machines. However, we see that the *radial* router can reduce costs by more than 8%, with acceptable client-server distances (less than $1000km$). For the synthetic distribution, the unconstrained *radial* router achieves savings very close to the *nlp* router.

An interesting feature of these results is that the unconstrained *radial* router performs almost as well as the unconstrained *nlp* router. This could be due to differences between the raw traffic workload used earlier and the synthetic workload used here. Alternatively, it may be that over the longer time period most cost differentials were equally accessible to both routers.

While both geo-distributions have strikingly similar percentage savings, they differ in their absolute costs. The 39-month cost per 10K servers for the Akamai distribution approaches $1.95M, but the synthetic distribution's cost is below $1.75M.

The savings are spread out in time, as before. We see that PDR consistently lowers the system's monthly electricity costs. Figure 6-10 illustrates this, showing how monthly costs and savings varied over the 39 months.



(a) Different Routers **(b) Different Geo-Distributions**

Figure 6-11: The average client-server distance required to achieve a certain level of savings: (a) depends on the routing policy (24-day savings, Akamai geo-distribution); and (b) depends on the clusters' geo-distribution (39-month savings, with the radial-gradient router). These curves are for the G6 energy model.

6.4 Performance and Savings

We find that savings rise if client-server distances are allowed to increase, up to a point. Relaxing distance constraints allows clients to consider more clusters, increasing the likelihood that they will be routed to the lowest cost location. Both the 24-day and 39-month results, presented earlier, demonstrated this relationship.

In order to map how savings depend on distance, we used a number of additional simulations in which we varied the routers' distance constraint parameters. Figure 6-11 shows some of our results. Each point in these curves is the result of one simulation: the achieved savings and the corresponding mean (or 95-th percentile) distance.

The shape of a savings-vs-distance curve depends on the routing policy. Figure 6-11a plots 24-day costs for different routers. In the case of the *radial* router: the relationship is approximately linear between 4% and 11% savings; reducing the savings below 4% does not result in a meaningful reduction in distances; and marginal increases in savings beyond 11% require large jumps in distances. The *nlp* router's curve is not linear. The flat region in the middle exists because of the way in which Akamai's geo-distribution unevenly spreads out servers. This effect is not visible in the *radial* router curves because that router uses relative distance constraints, while

the *nlp* router uses strict absolute distance constraints.

Differences in geo-distributions also lead to different savings-vs-distance curve shapes. Figure 6-11b plots the *radial* router's 39-month savings against distances, for the two geo-distributions we have described so far. As in the 24-day case, the curve for the Akamai distribution is approximately linear in the middle. In contrast, the synthetic 5-cluster distribution results in a kinked curve. The hump in the middle is likely the result of the router having exhausted nearby economic opportunities, and lasts until the router becomes unconstrained enough to shift clients away from Northern clusters, like Virginia, to Texas in the South.

6.5 Different Energy Models

It is obvious that the savings from PDR will be sensitive to the energy-vs-utilization characteristics of a system's clusters. Clusters that have a high degree of energy proportionality will be able to relocate large fractions of their power consumption. The larger the relocatable fraction, the better a system can take advantage of electricity price differentials, and the lower its energy cost with PDR should be.

To confirm this, we simulated all 25 energy models from figure 6-1. Figure 6-12 summarizes our results. The simulations in the figure covered three non-consecutive months[7] (July 2008, October 2008 and March 2009), use the *nlp* router (without a distance constraint), and the 5-cluster server geo-distribution described earlier. Percentage savings are calculated relative to a cost-unaware variation of the *radial* router. Because distances were unconstrained, these results represent the maximum savings from PDR for each model.

Let us first focus on the results for the `G` models[8]. Generally, higher model numbers represent more energy efficient systems (see figure 6-1 for details). The percentage savings rise rapidly as the energy efficiency of the model increases. The `G9` model uses

[7] The three simulated months straddle three seasons, so we capture some seasonal market variations. We did not simulate the entire 39-month period because of the inefficiency of the *nlp* router.

[8] The PUE and idle server power vary; the IT energy model is based on Google's data center study; infrastructure power is constant; and there is no-DSS.

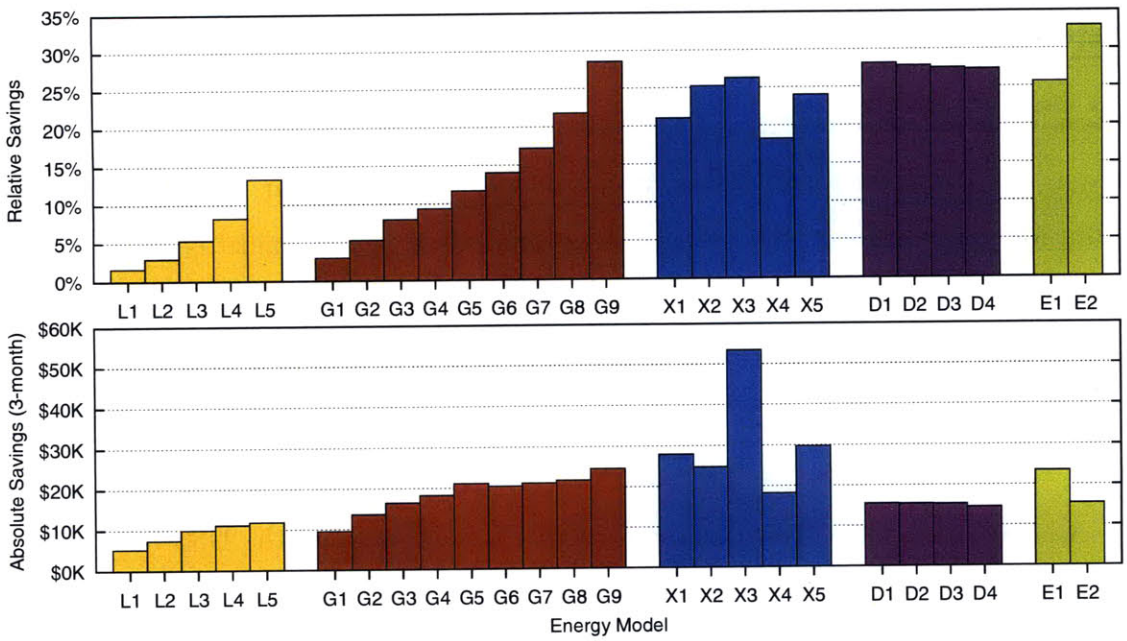


Figure 6-12: Savings achieved by the unconstrained *nlp* router for different energy models, with the synthetic-5 geo-distribution. The absolute 3-month savings are for 10K servers.

the best known PUE and idle server power for actual data centers, and its savings almost reach 30%. Even with the `G2` model, which represents the average case for recent data centers, savings are at 5%.

Absolute savings begin to level off after the `G4` model, even as percentage savings continue to rise. As we move to higher models, their increasing energy efficiency is driving down the energy per unit of work, so PDR has to move larger volumes of requests for every marginal increase in absolute savings. The stagnation in savings implies that only a certain degree of energy proportionality is needed by PDR to capture almost all of the savings. Recall that `G5` is less efficient than the EPA's 2011 best-practices target for data centers. Therefore, the current direction of technological evolution curve will lead to clusters that are perfectly poised to exploit PDR.

From these results we estimate that a system with 800K servers (e.g., Google) would be able to save about $6.5M per year, if network latency were ignored. From our earlier simulations, we conclude that at least two-thirds of these savings should be achievable with acceptable client-server distances.

The `X` models use nonlinear energy curves, based on different server hardware

types, and treat the infrastructure power as varying with load. We see substantial savings with these models, demonstrating that the *nlp* router will work with clusters whose energy characteristics do not match the simpler models from Google's data center study. `X3` has the highest absolute savings among our models, more than $16M for 800K servers. The server energy curve used in this model is quite different from most other curves we have studied (see §3.2.2, and figures 3-3 and 3-2). Its idle server power is 47% of peak, but power jumps rapidly, reaching 80% of peak by the time cluster utilization is 0.2. The high savings for `X3` are the result of *nlp* aggressively pushing clusters to either a utilization of 0.0, or as close to 1.0 as possible.

All the `D` models achieve similar savings. This shows that the DSS block size[9] is not important. As long as DSS exists, idle power is low and PDR can exploit that. The absolute savings (about $5.5M for 800K servers) are lower than with `G5` because DSS has very good energy efficiency characteristics, so baseline energy costs are low.

The `E` models represent cutting edge clusters. The highest percentage savings are achieved by the `E2` model, which represents cluster designs that will not become common for another five to ten years.

Since PDR's effectiveness varies so widely, is there a way to predict if a system can extract meaningful savings using PDR? We asserted earlier that systems whose energy curves have higher EPG and EAP scores would be able to extract more savings from PDR than systems with lower scores. Recall that these metrics are a measure of energy proportionality. Therefore, to gauge the usefulness of PDR to a new system, we can measure the energy-vs-utilization curve, and calculate the EPG or EAP score (see figure 6-1 for our models' scores).

We ran a number of simulations to determine how well these metrics predict savings. Figure 6-13 shows a scatterplot of absolute savings versus energy model EPG scores. The simulations with the unconstrained *nlp* routers represent the maximum possible PDR savings. A linear function fits these results well. Thus we can predict absolute savings using a linear function of the EPG score. The *radial* router results represent a sub-optimal router constrained by moderately strict performance goals.

[9] The number of servers DSS turns on or off in discrete units.

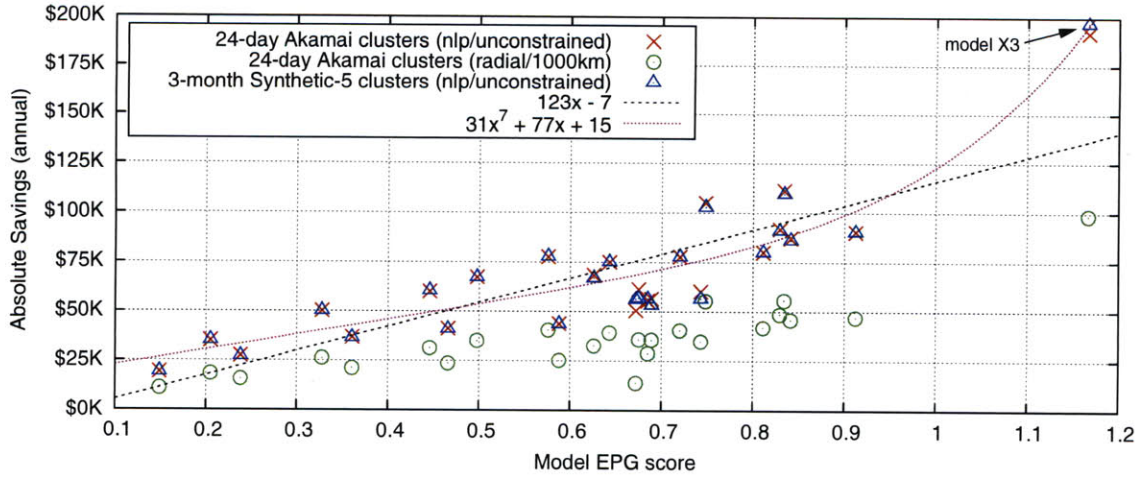


Figure 6-13: Simulations with different energy models, geo-distributions, and time periods show that the EPG score is a good predictor for absolute savings (per 10K servers).

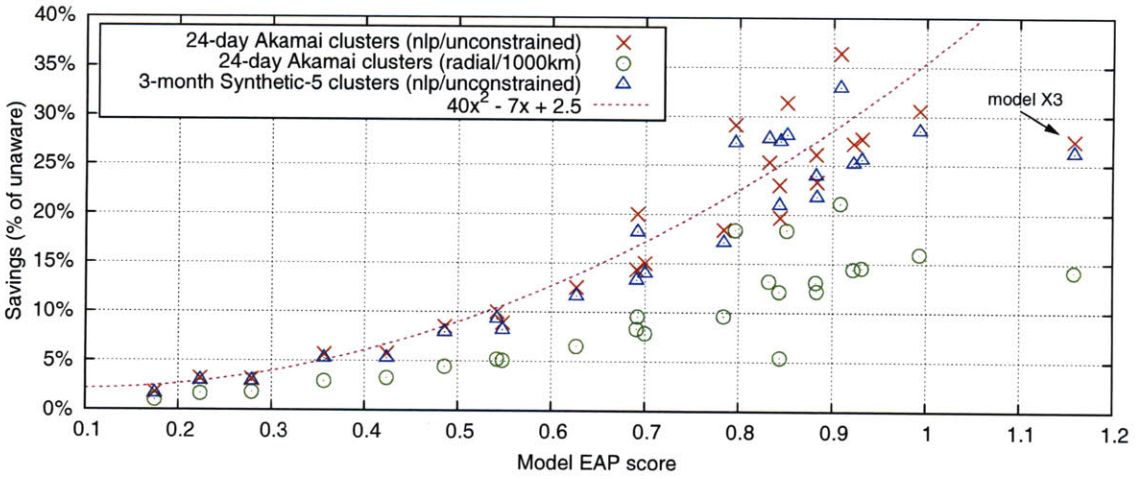


Figure 6-14: Simulations with different energy models, geo-distributions, and time periods show that the EAP score is a good predictor for relative savings.

The *radial* router's savings lie below the *nlp* line but exhibit a similar relationship to EPG scores. Even with performance constraints, we can use EPG to predict savings.

While the EPG score can predict absolute savings, the EAP score can predict relative savings. Figure 6-14 shows a scatterplot of percentage savings versus model EAP scores. We see that a quadratic curve fits this data well, for EAP values below 1. Recall that the EAP metric is specified so that the power curve $P(u) = u$ has a score of 1.

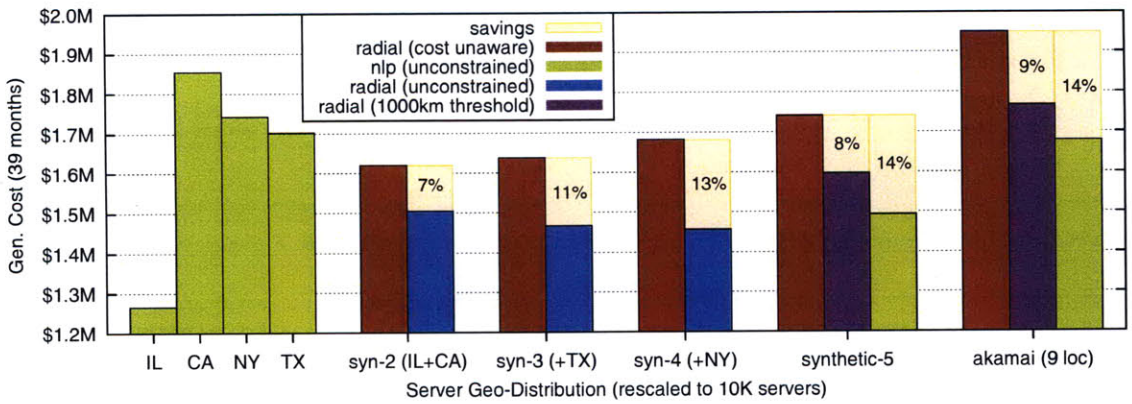


Figure 6-15: 39-month energy costs for several cluster distributions, with the G6 model. The first four distributions place all servers in one cluster. Even with two clusters, PDR can yield a meaningful cost reduction.

6.6 Different Geo-Distributions

In the previous sections, we have not seen any notable differences between simulations that used Akamai's server geo-distribution and those simulations that used a synthetic 5-cluster distribution. In this section we investigate over a hundred geo-distributions.

We start by asking the question: how many clusters does one need to leverage PDR? Companies like Akamai and Google spread their servers across many data centers, but others, like Microsoft, use a small number of primary sites (on the order of 3). If PDR can only extract meaningful savings when there are 5 or more clusters, its utility will be limited.

Figure 6-15 summarizes simulation results for five geo-distributions. The synthetic distributions shown here were based on what we know to be popular data center locations[10]. We see that PDR can extract meaningful savings even with only two clusters (one on the east coast and one in the mid-west). However, in this case the latency characteristics can be quite bad, worse than having a single cluster in the mid-west. Without distance constraints, PDR will shuffle clients between both coasts to take advantage of inexpensive energy. With four or more clusters, we find that the maximum savings (with the G6 model) are around 14%, and that savings of about

[10] The exact cluster locations: IL is near Chicago; CA is near Palo Alto; NY is near Buffalo; TX is near Abilene.

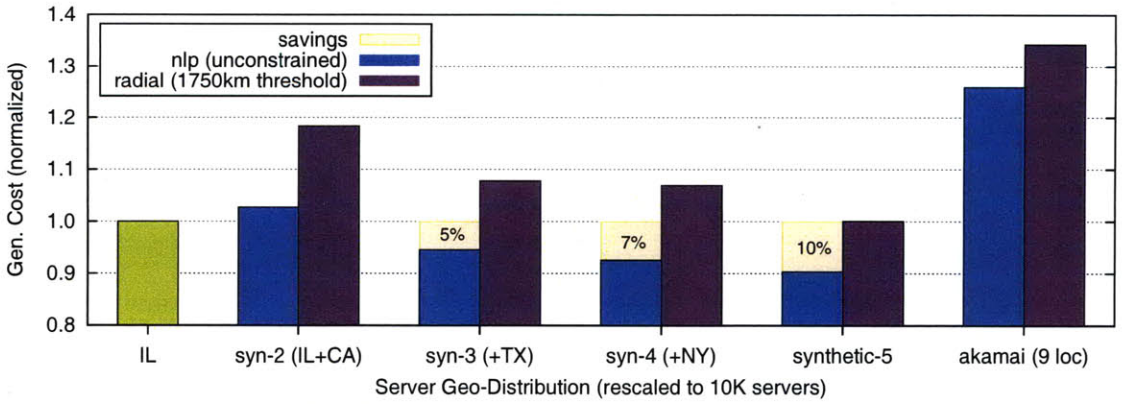


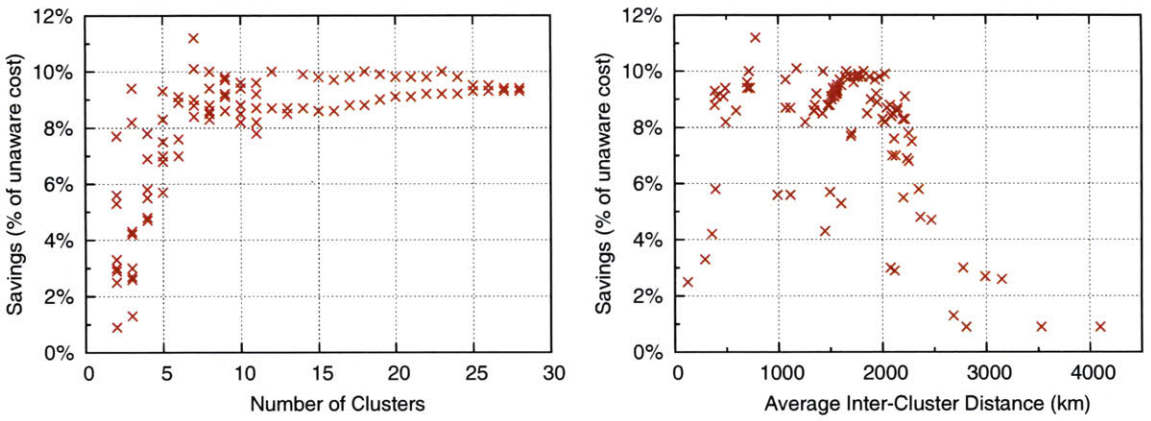
Figure 6-16: 39-month energy costs for the least expensive market and several cluster distributions, with the E2 model. When three or more clusters exist, the *nlp* router can lower costs below the least expensive market.

9% can be achieved within acceptable performance bounds.

In these simulations, placing all the servers in the least expensive market (IL) resulted in the lowest electricity cost. We find, however, that with more energy proportional clusters it is possible to beat the least expensive market by using multiple clusters and PDR.

Figure 6-16 shows the results for the E2 energy model. Recall that this is our most energy efficient model. With two clusters, energy costs with *nlp* match Illinois, the least expensive market. With five clusters, *nlp* achieves costs that are 10% below Illinois's costs. As before, however, performance is worse than when a single cluster in Illinois is used. We have found it impossible to beat the least expensive market with tight performance constraints. We do find that if distances are allowed to be moderately high, the five cluster distribution can match Illinois. In contrast, the Akamai distribution never comes close to Illinois. This is because too much of Akamai's capacity lies in expensive markets.

We have extensively explored the impact of geo-distribution on savings. Figure 6-17 summarizes simulation results for over a hundred different geo-distributions. Our electricity price database contains prices for 28 locations and we generated geo-distributions that spread servers evenly across subsets of these locations. The number of clusters ranged from 2 to 28. Some geo-distributions were generated randomly,



(a) Savings vs. Cluster Count **(b) Savings vs. Geo-Spread**

Figure 6-17: Simulation results for over a hundred different server geo-distributions, with between 2 and 28 clusters. The results are from one year costs, and compare the costs of a radial router (1000km threshold) with the cost unaware variant. All these simulations used the `G6` energy model.

others were generated using heuristics[11], and some were picked by hand.

We can see from figure 6-17a that with 5 or more clusters, the savings level off. The best possible distribution has seven clusters and savings above 11% (recall that this is with performance constraints in place). With a small number of clusters, savings vary widely, but savings of 7% or higher are frequently achievable.

Figure 6-17b illustrates that clusters need to be spread out geographically, but not spread out too much. When they are too close together, there are fewer market price differentials to exploit. When they are too far apart, the router's distance constraint is too limiting.

We conclude that PDR does not require prior planning to be useful. Existing geo-distributions should be able to reduce their energy costs, as long as there are a moderate number of clusters and performance constraints are not too strict.

6.7 Accounting for Complex Network Costs

A reduction in a system's electric bill may be overshadowed by an increase in its network usage costs. By redirecting traffic to regions with low energy costs, PDR may

[11] e.g., start with a geo-distribution picked by hand and iteratively remove hubs to minimize the sum of the pairwise market price variances.

be increasing the system's network costs by sending traffic to regions where bandwidth is expensive. Thus far, we have assumed that a cluster's per bit network cost is independent of its traffic volume, and that costs are equal everywhere.

However, there can be large differences between costs charged by different network providers at different locations, and sometimes by the same provider over time. Network costs often overshadow energy costs. For instance, network bandwidth costs for Akamai exceed their energy costs, and thus their system is aggressively optimized to account for locational variation to reduce overall network costs. Moving away from such a system's default assignment of clients to clusters, in order to reduce energy costs, could increase the system's network costs.

We cannot therefore ignore network costs in our analysis. The complication is that network cost details are considered to be proprietary information by companies. Our treatment of network costs in this dissertation will be relatively abstract.

In our analysis, we consider how the introduction of a 95/5 network billing model, a typical scheme used by ISP's, affects energy savings. In 95/5 billing, prices are usually set per network port: traffic is divided into five minute intervals and the 95-th percentile traffic volume is used for billing. We will discuss 95/5 billing in more detail in chapter 7.

Our simulation approach is to estimate 95-th percentiles from the observed Akamai traffic data, and then to constrain the *radial* router so that it does not increase any location's 95-th percentile. We modified the *radial* algorithm from figure 6-3 as follows: the routing loop starts by initially setting the cluster capacity constraints to their 95-th percentile values; once all the clusters have been saturated, these constraints are reset to the actual capacities, and the routing loop continues.

Figure 6-18 shows how savings were affected once we introduced this additional bandwidth constraint into our simulations. We see that adding 95/5 constraints caused energy savings to drop down to about a third of their earlier values.

The good news, however, is that these savings are reductions in energy cost without any increase in network costs. By jointly optimizing network and energy costs, it should be possible to acquire part of the economic value represented by the difference

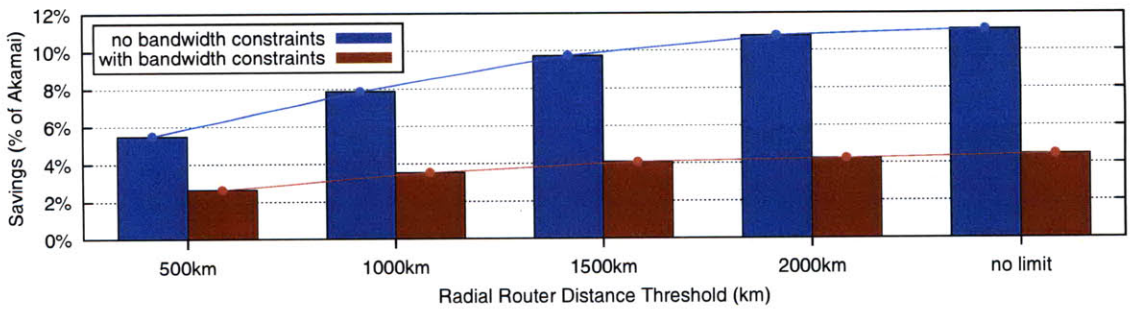


Figure 6-18: The impact on savings when we incorporate 95/5 bandwidth constraints to prevent network costs from rising. These simulations used the G6 model.

between savings with and without 95/5 constraints. Later (in §7.6), we discuss how to modify the earlier optimization framework to incorporate complex network billing models, such as 95/5 billing.

6.8 Imperfect Knowledge

What happens when a router's knowledge of cluster costs is imperfect? This could happen because of inaccurate energy models; because it takes enough time for new routes to be deployed that the electricity prices have changed by the time the routes are in place; or because of market inefficiencies that cause delays in price propagation.

In order to analyze the effect of imperfect knowledge on savings, we constructed some simulations in which the router saw artificial prices that deviated from the actual market prices. Market prices were still used to calculate system costs. We ran some control simulations in which the router was given a randomly generated number every time it asked the database for a price. We also investigated the effect of delaying prices, so that the router saw the previous hour's prices, or the previous day's prices. Finally, we used static prices that did not vary in time. One set of static prices used the 39-month average price for each location. Another set used a partial order of prices, assigning a 'price' of 1 to the location with the lowest average market price, a price of 3 to the location with the highest average, and a price of 2 to all other locations. Finally we picked a set of prices that ordered the locations inversely, so that the actual highest-cost location had the lowest artificial price.

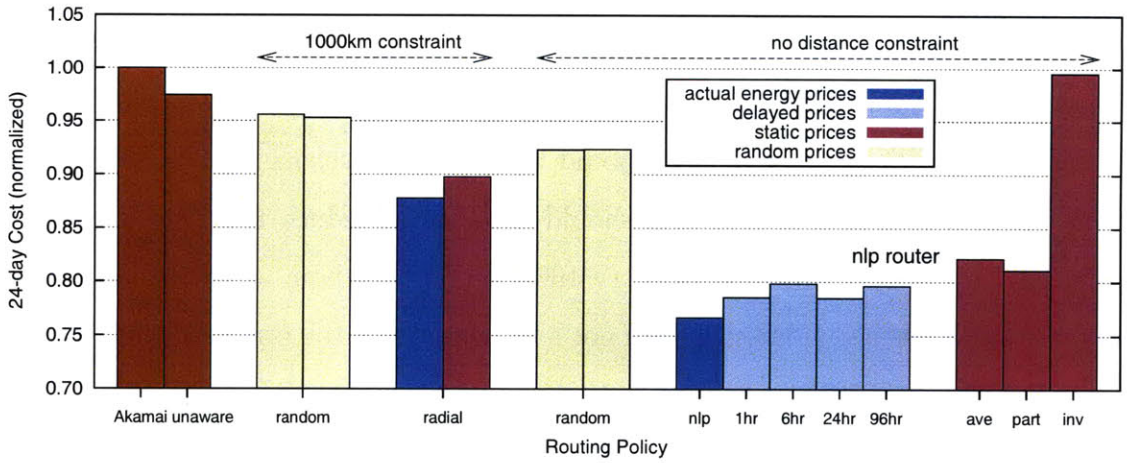


Figure 6-19: Simulation results when the electricity prices seen by the router deviated from actual prices (Akamai geo-distribution, 24-day raw traffic traces, and G9 model). In some simulations prices were delayed so that the router saw hour-old or day-old prices. In other simulations the router was passed randomly generated prices. We also passed in artificial prices that did not vary in time: the mean price for a location (ave); a partial order (part); and prices that ordered locations inversely (inv).

Figure 6-19 summarizes the results of several simulations. There are two sets of simulations: those that used distance constraints and those that did not. In all the cases, the routers that saw the true market prices had the lowest costs.

Optimization based on static prices works surprisingly well. When using either the average or the partial-order prices more than half the maximum savings can be captured. This implies that statically ranking clusters and slowly optimizing routes is still a good strategy for reducing operating costs. The partial order worked better than using average prices. Inverse static prices resulted in the router exceeding the cost unaware router.

Static optimization will be easier to implement. However, optimizing hourly captures the economic value of transient price differentials. This value may just be a few percentage points, but in large systems a few percentage points can represent millions of dollars in annual energy costs.

Delaying prices eliminates most of the savings achieved beyond the static optimization. Even a single hour's delay has a big impact. Once price delays reach 6 hours, savings more or less stabilize at the static optimization level. A 24 hour delay is special, because market prices exhibit some hour-of-day correlation.

Summary

In this chapter we built upon earlier parts of the dissertation to quantitatively estimate the monetary savings that should be achievable through PDR. We conclude that today's cutting-edge systems should be able to reduce their electric bills by about 8% (or $300K annually for every 100K servers) without an appreciable impact on network latencies. If latencies are not a concern, then the potential savings rise to around 15% (or $750K annually per 100K servers). Older systems can also benefit, but their savings could be about half this. Furthermore, network cost concerns may reduce savings to about a third of these values. After exploring the sensitivity of our results to several different modeling factors, we are confident that these projected savings are realistic.

Chapter 7.

Other Considerations

We have made a number of simplifying modeling assumptions to make the earlier analysis tractable. In this chapter, we revisit some of these assumptions and discuss the implications of relaxing them. We also touch upon some extensions of our work.

We first describe complications caused by existing electricity billing contracts (§7.1) and whether we should expect market prices to remain unaffected by power-demand routing (§7.2). We then explore how a distributed system can actively engage with the power grid, instead of reacting to market prices (§7.3). Following that we discuss how our PDR optimization framework (chapter 2) can be augmented to model partially replicated systems (§7.4) and multihomed clusters (§7.5). We then describe how to integrate more realistic network cost models into our optimization framework (§7.6). Finally, we conclude by linking our work on energy cost differentials to resource pricing in multi-tenant clouds (§7.7).

7.1 Existing Electricity Contracts

In this dissertation, we assume that power bills are proportional to energy consumption and indexed to hourly market prices. Additionally, we assume that the decisions of server operators will not affect market prices.

The strength of this approach is that we can use historical price data to quantify how much money would have been saved had one used PDR. In reality, however,

achieving these savings would probably require a renegotiation of existing utility contracts. Furthermore, using PDR to move power-demand between regions may affect market prices (we will cover this in the following section).

It is safe to say that most current contractual arrangements would reduce the potential savings below what our analysis indicates. That said, server operators should be able to negotiate deals that allow them to capture at least some of this value.

Data center owners often sign long-term contracts with power producers, bypassing the wholesale market and guaranteeing a fixed electricity price over the duration of the contract (e.g., Google's recent 20-year commitment to buy wind power in Iowa [72]). Even with fixed-rate contracts, cost differentials may exist between different data centers and these differentials should be exploited with PDR. Our earlier simulation results show that optimizing with static prices is still useful (§6.8).

Furthermore, even when a data center operator is buying electricity at a fixed rate, they may not be able to ignore spot prices in the regional wholesale market. When market prices are above the private contract's rate, an opportunity cost exists: the system could shift its power-demand to another location and resell the freed power in the local market, netting a profit. In fact, Google has explicitly stated that they will resell the wind power they have bought in the regional spot market [72]. In this case, their motivation is to encourage renewable energy, by contracting more power than they can use, but one can easily imagine adding PDR to the mix.

Companies such as Akamai and Facebook that rent spaces in multi-tenant facilities will almost certainly have to negotiate a new billing structure to obtain any advantage from our approach. Electricity charges in most multi-tenant facilities are based on provisioned power capacity, not on the amount of electricity used. Akamai, for example, pays by the rack, and each rack has a maximum power rating. We speculate that as energy costs rise relative to other costs, it will be in the interest of data center owners to charge based on consumption and possibly indexed to spot market prices. There is evidence that network bandwidth costs are falling, but energy costs are not.

Utilities throughout the US are increasingly offering wholesale-indexed bills. For example, in the mid-west RTO Commonwealth Edison offers a *Real-Time Pricing* program [4]. Customers enrolled in it are billed based on hourly consumption and corresponding wholesale PJM-MISO locational market prices. We expect that similar metered power bills will become common in data centers as well.

Moving away from fixed-rate contracts is appealing to electricity providers because this transfers risk to consumers. When electricity prices rise, the additional burden can be passed on. In 2001, the Pacific Gas & Electric company filed for bankruptcy because it was tied to fixed-rate contracts and had to buy electricity at a higher rate than it could sell to consumers.

Studies with residential consumers have also shown that metered bills benefit consumers. The Department of Energy conducted a year-long study with metered real-time-pricing and concluded that homeowners who participated saved approximately 10% on their electricity bills [26]. The homes in the study were outfitted with software that automatically cut consumption when prices rose, e.g., by automatically lowering thermostats to levels specified by the individual homeowners.

In contrast to residential consumers, distributed systems have more flexibility and so can benefit more from metered billing. Using PDR they can quickly reduce power in a location where prices are rising. We therefore advocate that these systems should embrace market price volatility, instead of avoiding it with long-term contracts.

If metered contracts are not available, server operators may be able to sell their load-flexibility through a side-channel like *demand response*—as discussed later (§7.3)—bypassing inflexible contracts.

7.2 Impact on Market Prices

We assume that shifting power-demand from one location to another does not affect market prices in either location. This sort of price-taking assumption is common in economic analysis.

We have reason to believe that prices will remain unaffected. The energy con-

sumption of a data center is a small fraction of the total electricity flowing over the grid. Large data centers have power capacities on the order of 10MW. In comparison, New York state had an average demand of over 18GW in 2009, with a summer peak that almost reached 31GW [34].

On the other hand, there are two clear cases where PDR will affect prices. First, data centers have very high power densities compared to homes. This concentrated consumption can cause congestion on the grid [25] and if data centers were to rapidly ramp their power up or down, they could cause localized disturbances that affect prices. Second, prices will be affected if the power-demand being shifted represents the marginal load in the region, i.e., if adding that load to the region will require the activation of an additional power plant or if removing it will allow a running plant to be shut down.

Furthermore, if many data centers in a region implement PDR, a collective herding behaviour could emerge, with the different systems shifting their demand in sync. Even when one operator's power-demand is not enough to affect prices, when multiple operators move in concert the aggregate demand could be large enough to affect prices.

If circumstances are such that the price-taking assumption begins to break down, operators should decouple routing policy from market prices and actively engage with the grid operators, as described in the next section.

7.3 Actively Engaging with the Grid

Recall that Regional Transmission Organizations (RTO's) manage different parts of the power grid. An RTO provides a central authority that sets up and directs the flow of electricity between generators and consumers over the grid. RTO's decide which power plants are active at any given time, based on the demand load and an auction between power producers and consumers that sets market prices (§5.2).

Some RTO's allow consumers to participate in these auctions by bidding *negawatts* (negative demand, or load reductions). The Federal Energy Regulatory Commission has recently directed regional operators in the US to value a negawatt at the market

price of a megawatt of additional production [74]. Under California's Proxy Demand Resources program, a consumer who can turn off 100MW or more of power on a moment's notice is treated as if they could generate 100MW of electricity [73].

Distributed systems with energy proportional clusters can be far more flexible than traditional consumers: operators can quickly and precipitously reduce power usage at a location (by suspending servers, and routing requests elsewhere). The current generation of data centers will not cross the 100MW threshold for California's program, but data centers may grow to this level in time, or multiple data centers could coordinate and bid as an aggregate entity, or the 100MW threshold could be lowered in the future.

The best strategy for constructing auction bids is not obvious. Recall that a geo-distributed system will be participating in multiple market auctions simultaneously. A discussion of bidding strategies is beyond the scope of this dissertation.

Alternatively, a distributed system could enroll its clusters in triggered *demand response* programs, agreeing to reduce its power usage in a region in response to a request by that region's RTO. Load reduction requests are sent out when electricity demand is high enough to put grid reliability at risk, or rising demand requires the imminent activation of expensive/unreliable generation assets. The advance notice given by the RTO can range from days to minutes. Participating customers are compensated based on their flexibility and load.

Demand-response variants exist in every market we cover in this dissertation and are seen as increasingly important tools to improve grid efficiency [108]. For example, Texas has been increasing its use of wind power, and one cold still day in 2008 the state's power grid went into emergency mode (the wind wasn't blowing and homeowners cranked up their heat). By cutting power—about 1.1GW within 10 minutes—to demand-response participants the Texas RTO was able to continue without any problems [27].

Small consumers can also be aggregated into large blocs that reduce load in concert. This is the approach taken by EnerNOC, a company that collects many consumers, packages them, and sells their aggregate ability to make on-demand reduc-

tions to the RTO. A package of hotels could, for example, reduce laundry volume in sync to ease power demand on the grid. Even consumers using as little as 10kW (a few racks) can participate in such programs.

EnerNOC's network has sprung into action multiple times this year, responding to plant failures and abnormally high demands due to a heat wave [69]. In June 2010, two power plants failed in New England and energy prices momentarily spiked to $1,000/MWh. EnerNOC's network was able to coordinate some 1,000 assets and reduce demand by 380MW. Later, in response to similar failures in the mid-west, EnerNOC was able to reduce load by 2.5GW.

There is anecdotal evidence that data centers have participated in demand response programs [25]. However, the applicability of demand response to single data centers is not widely accepted. Participating data centers may face additional down-time or periods of reduced capacity. Conversely, when we look at large distributed systems, participation in such programs is attractive. Especially when the barriers to entry are so low—only a few racks per location are needed to construct a multi-market demand response system in coordination with aggregators like EnerNOC.

However, the fact that the distributed system operator is giving up some control to the RTO may cause problems. We have ignored what happens when an operator is told to reduce power consumption at a location, when there is a concentration of active clients nearby. In systems like Akamai, demand is generally predictable, but there will be heavy traffic days that are impossible to predict. As with auctions, a discussion of demand response strategies is beyond the scope of this dissertation.

We speculate that there exists a spectrum of achievable savings, depending on the degree to which server operators interact with RTO markets. As we have demonstrated earlier, operators can passively exploit spot market price-differentials and reduce costs. Operators should be able to increase their savings by selling their ability to reduce load in the day-ahead market auctions and real-time demand-response programs. The good thing about selling flexibility as a product, is that this is valued even where wholesale markets do not exist. It even works if price-differentials don't exist (e.g. fixed price contracts or in highly regulated markets).

7.4 Partial Replication

The PDR optimization framework we presented in chapter 2 assumed that the system was fully replicated at all its clusters. This assumption is accurate for large web services, such as search, and greatly simplifies our analysis. In practice, however, systems tend to be only partially replicated, e.g.: user data is replicated at multiple clusters, but not at all clusters; so a request may be serviced by more than one cluster, but not by any cluster [37]. In this section we show how to adapt our framework to take partial replication into account.

Our model of partial replication is based on the common practice of *sharding*. The data the service depends on—a set of objects or database records—is partitioned. Each partition is referred to as a *shard*, and is replicated at multiple locations. Ideally, objects are distributed so that a request has access to all the data it needs at a single location. Thus, requests with read-only operations involve only one cluster. When an operation mutates an object, the changes need to be propagated to the other replicas. We ignore write propagation here, assuming the request stream is dominated by read-only requests. 6:1 read-write ratios have been reported for services like Facebook [40].

Not all shards are replicated at every cluster. For instance, `gmail` replicates user data at two locations [99]; and Yahoo's PNUTS data platform is designed to maintain multiple replicas [51]. The policy used to partition objects into shards and to determine how shards are distributed varies by service. Shards may be generated and spread out randomly; a goal could be to balance load across clusters; locality to users that own objects in those shards could be considered. The process of finding a good shard distribution can be formulated as a dynamic optimization problem [37].

When we constructed the PDR model earlier, we based it on a bipartite flow graph. The optimization goal was to find the minimum cost flow in that graph. Sticking with this strategy, we model partial replication by constructing a more complicated graph. The new optimization is to find the minimum cost flow in the new graph. Figure 7-1 shows an example of this new graph, and we describe its features below.

Earlier we had one set of nodes representing ingress points, and one set of nodes

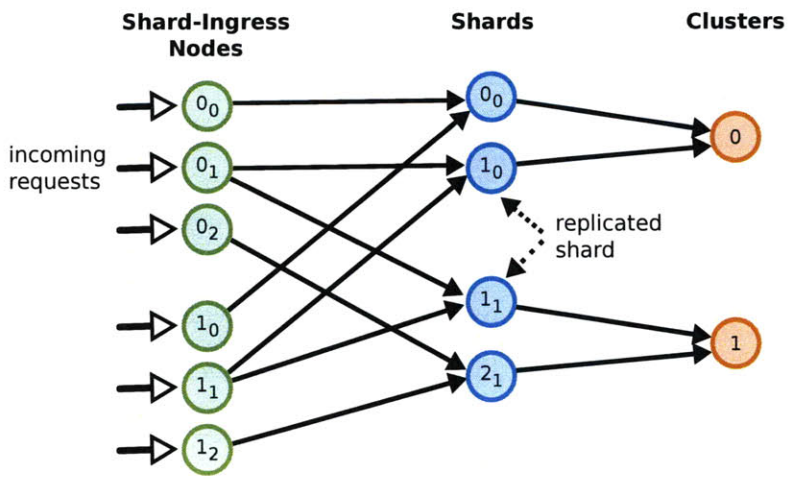


Figure 7-1: A PDR graph for a partially replicated system. There are two clusters, three shards, and two ingress points. Shard #1 is the only shard replicated at both clusters.

representing clusters. Suppose the system has n' non-overlapping shards. We start by splitting the request streams based on the shard for which they are destined. Instead of one node per ingress, we now have n' nodes per ingress, one for each shard (the shard-ingress nodes).

Between these nodes and the cluster nodes, we then add a layer of shard nodes. A node for shard k is added before cluster j iff shard k is replicated at cluster j in the system. The shard nodes express the partial replication characteristics of the system; we can continue to model link latency and cluster energy as before. We may need to add capacity constraints for the shard nodes, if the hardware at a cluster is statically partitioned among shards.

Note that although partial replication complicates the problem, it can still be solved using the optimization methods we used earlier.

We can extend this further, to model the case where not all requests are identical. If there are m' classes of requests, each shard-ingress node in figure 7-1 will explode into m' nodes, one for each request class. Different request classes may have different resource requirements or different latency SLA's. SLA's can continue to be modeled by labeling edges with latency constraints. However, cluster capacity and energy consumption can no longer be modeled by simply using constraints and properties at the cluster notes. More complicated cost functions will need to be constructed.

7.5 Multihomed Clusters

Our basic PDR model assumes that each cluster is connected to a single Internet Service Provider (ISP). In practice, multihomed clusters are common [115, 61]. Connecting to multiple networks: makes clusters more robust, capable of surviving network failures; can reduce cost, through ISP peering agreements; and can improve performance, by allowing for more direct routes to clients.

However, when connected to multiple networks, a cluster is billed for network usage by multiple entities. Each connection will have an independent cost function, proportional to the traffic volume on that connection. The system operator may also own some network infrastructure, resulting in some zero-cost connections. Additionally, systems that drive a significant amount of Internet traffic may be able to negotiate peering arrangements with ISP's, possibly resulting in other zero-cost links. Some regional providers may also provide transit for free to large companies (e.g., Akamai). Finally, even if all connections are charged using the same pricing function, if they use P95 billing (see §7.6) the composite network cost function cannot be expressed using the earlier bipartite graph model.

In the basic PDR model, the routing problem is to decide where to send requests. With multihomed clusters, not only must one decide between clusters, one must also determine the best path to a cluster. Different path choices could lead to substantially different network costs [115].

Furthermore, earlier we had assumed that network links are overprovisioned (or, equivalently, that the cluster's capacity is reached before the network link becomes overloaded). This allowed us to avoid including link capacity constraints in our model. However, when connected to multiple ISP's, links may have asymmetric capacities. This could be because of different kinds of network connections (e.g., DS3 45 MBps vs. OC3 155 MBps), or because of peering agreements that limit rates. Therefore, we should include link capacity constraints in the multihomed cluster model.

To adapt our formulation of PDR to handle multihoming, we convert the earlier bipartite graph problem to a slightly more complicated graph. As shown in figure 7-2,

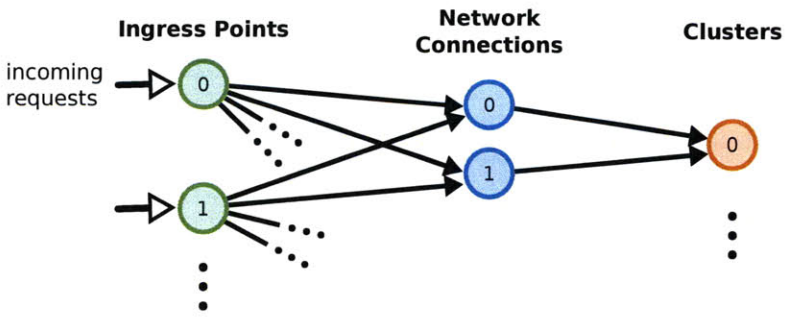


Figure 7-2: Modeling multihomed clusters with a modified flow graph.

we add a node for each network connection at a cluster; clusters do not share these intermediary nodes. The cluster's network cost is now the sum of the individual ISP costs. Each link may also be labeled with a capacity constraint. It is easy to see how the partial replication graph (§7.4) can also be modified in this way.

Note that an ISP may not provide transit to all routes, choosing instead to only advertise some subset of Internet routes to a cluster. Therefore not all ingress points will necessarily have connections to all ISP's.

7.6 Complex Network Cost Models

Previously, we modeled a cluster's network cost as a linear function of the cluster's traffic volume. This model significantly simplifies the optimization problem, but can be inaccurate. In this section we examine more realistic network cost models based on how ISP's typically bill their clients for network usage. However, as we note later in this section, optimizing using the earlier simplified cost model has been shown to be reasonably effective in reducing network costs, for a related traffic engineering problem.

We only discuss network costs in broad strokes here. Bandwidth prices paid by large distributed systems to their ISP's are considered proprietary information. We are not aware of any public disclosures that provide specific information about the geographic and temporal variation in bandwidth prices. In our discussions with the operators of large systems, we have gathered that ISP prices in the public domain differ substantially from actual prices privately negotiated between large system op-

erators and their ISP's. We rely on the literature and on second-hand information from researchers at Akamai and Microsoft.

Network costs are an important contributor to data center costs, and there may be large differences between costs levied by different network providers, and sometimes on the same network provider over time. Bandwidth costs are significant for Akamai, and thus their system is aggressively optimized to reduce network costs. Changing Akamais current assignments of clients to clusters could increase its network costs. Presently, the portion of Akamai's data center rents attributable to energy is less than but still a significant fraction of the cost of bandwidth. The relative cost of energy versus bandwidth has been rising, primarily due to decreases in bandwidth prices.

Akamai does not view network bandwidth prices as being geographically differentiated. In some instances, a company as large as Akamai can negotiate contracts with carriers on a nationwide basis. Smaller regional providers may provide transit for free. However, the network cost at a location for Akamai is often a non-linear function of the network traffic volume at that location—a function of the *95-th percentile* traffic volume—as described later.

More generally, system operators rent links from one or more ISP's for each cluster (or rent network ports in multi-tenant spaces)[1]. Rental charges for a link are levied at the end of a billing period (e.g., a month) and are a function of the traffic volume signal during that *entire* period. One consequence of this is that we may not be able to optimize for each hour independently (an *online* approach); the optimization may need to be conducted over the entire billing period (an *offline* approach, based on traffic expectations).

The cost is typically defined in terms of a non-decreasing function (F) of the *charging volume* (ν):

$$c = F(\nu)$$

F may be a piece-wise linear function of ν or it may be a step function (different

[1] In this discussion, we ignore the case where operators own their own network infrastructure. Owned links can be modeled as being zero-cost, or a more complete model can account for downstream ISP transit costs. If necessary, we can account for downstream costs by expanding the flow graph, adding intermediate nodes as we did to account for multihomed clusters.

prices for different volume ranges) [61]. Charging volume ν is a function of the traffic signal over the billing period. Different approaches for calculating ν can result in very different network cost functions. Two common approaches are:

Total-volume based charging : the charging volume is defined as the total traffic volume over the billing period. In this case, we can optimize each hour independently, iff the minimum value of $F(\sum \nu_t)$ is the same as the minimum value of $\sum F(\nu_t)$ over the traffic optimization's set of feasible choices for the $w_{i,j}$. This property of F is true if it is a linear function of ν_t, i.e. if the marginal cost-per-bit is constant.

Percentile-volume based charging : the charging volume is defined in terms of a statistical property of the traffic signal. A typical scheme uses the 95-th percentile traffic volume as ν (95/5 or P95 billing). The ISP tracks traffic volume for every non-overlapping 5-minute interval, and uses the 95-th percentile volume[2]. One motivation for this model is to tolerate bursty traffic. Since the top 5% intervals are ignored, the system operator may occasionally raise the traffic on a link, without affecting network cost.

In the case of percentile-based ν or complicated forms of F, the optimization must be solved offline—instead of deriving optimal weights for an hour, we must derive the optimal weight vector $[W_t]$ for the entire billing period. Others have developed efficient algorithms to optimize the network costs of multihomed clusters with P95 billing [61], a related problem. That work can be leveraged to build an offline PDR mechanism that works with P95 network billing. The offline algorithm will need expected traffic volumes for every hour, and the weight vector $[W_t]$ will minimize *expected* cost.

Accurately predicting traffic volume for every hour is a hard problem; it is easier to derive good 95-th percentile goals for every cluster for every hour. We can then optimize hourly, using our basic approach, but adding these 95-th percentile goals as

[2] Precisely: the ISP sorts the volume samples, and uses the sample ranked $\lceil 0.95x \rceil$ (x is the total number of samples in a billing period). With 5-minute samples, and a 30 minute interval, the sample ranked 8208-th is used.

cluster capacity constraints. This is the approach we took earlier, when we simulated Akamai's P95 capacity constraints (§6.7).

Note that with P95 billing, we can add traffic to a cluster without increasing network cost, as long as we do not increase the monthly 95-th percentile. P95 billing allows for 36 hours in a month during which we can exceed constraints based on 95-th percentiles without affecting cost. We can use heuristics to determine when it is okay to burst past these constraints. In this way, we can construct an online solution that approximately minimizes cost in the presence of P95 network billing. This is similar to a proposed online approach for minimizing network costs for multihomed clusters [61]. A more drastic approach is to reformulate the PDR optimization problem as an online stochastic control problem, to apply optimal control theory techniques.

There is some evidence that this complexity may not be necessary, and that the network cost function presented in our basic formulation—while not accurate—will work reasonably well in practice. In a recent paper, researchers from Microsoft tackled a problem similar to PDR [115]. Given a large system with multihomed clusters, they explored ways to jointly optimize network cost and performance. Although some details differ—in particular, they did not account for energy costs—their problem setup is similar enough to our own that some of their conclusions apply. Importantly, they had access to network billing data for a large system, so they were able to do the sort of in-depth network cost analysis that we cannot.

Their strategy, like ours, was to consider traffic during a short period (minutes or hours) and to jointly optimize for latency and the network *pseudo cost*. The pseudo cost at a cluster was defined as $F(vol)$, where vol was the traffic volume. They did not compare this approach to the optimal solution, but they did compare it to pure-bandwidth and pure-performance optimizations, and concluded that optimizing for *pseudo cost* performed well as strategy for driving down actual costs.

In summary, it is worth keeping in mind that network costs in real systems may prove so problematic that either sequential optimization will be necessary (we know operators such as Akamai have effective but proprietary network cost optimization mechanisms), or we will have to accept sub-optimal solutions.

7.7 Valuing Resources in Clouds

With the rise of web-based computing and the computing-as-a-utility model, many companies are renting out their infrastructure to third-party applications. Examples include Amazon's EC2, Google's AppEngine and Microsoft's Azure platform. Tenant applications are billed by the resources they consume: computation cycles, network I/O and storage.

How much does it cost a cloud provider to perform one unit of work on behalf of a hosted application? How much does it cost Amazon to handle a single client request on behalf of a hosted web application?

Cost depends on the location where the request is serviced. We have already established that marginal service costs can differ radically with location and in time. Large cloud providers (Amazon and Google) will already need to absorb their fixed costs. They need to build multiple data centers, and keep machines up and running, to support their own primary services. The cost to them of performing some incremental work on behalf of a hosted application will be dominated by the marginal cost, mainly the cost of the additional joules they consume.

By charging fixed resource prices, as they currently do, while being able to decide where to buy electricity, cloud providers are missing an opportunity. With a price structure that embraces energy cost diversity, and by using a cost-conscious replication strategy, cloud providers could increase their margins or lower their prices.

Hosted applications care about how much they are charged and what performance their users receive. Infrastructure providers could build energy cost differences into some pricing plans, allowing buyers to make trade-offs. For example, free applications should always be hosted in the lowest cost locations, capacity permitting. Additionally, some buyers may be willing to pay premiums for regionally optimized performance. The Boston Globe's website, having regionally concentrated demand, values proximity, and could be billed to compensate for elevated electricity prices.

These ideas can also be mapped to content distribution networks. For instance, a CDN provider could charge a premium for hosting content in expensive markets.

Chapter 8.

Greener Systems

Not all joules are created equal. In power pools, like the grid, that aggregate electricity from diverse providers, the environmental impact per joule varies in time and space. Sometimes more wind power is available, for example. Other times, more of the energy is being generated using coal.

Power-demand routing works best when different locations have unequal costs and those costs vary in time in an uncorrelated manner. When we express cluster costs in terms of their environmental footprints, we see the sort of cost variation that PDR can productively exploit.

Therefore, rather than attempting to minimize the dollar cost of the energy consumed, a service operator may instead choose to use power-demand routing with an environmental cost function (e.g., carbon cost). In this chapter we show how to construct such cost functions and project how well this approach would perform in practice. This chapter is more speculative than our earlier work with electricity markets because we lack ground-truth data.

The first half of this chapter discusses why pollution may vary in space and time. The second half describes some carbon cost functions that can be used with PDR to reduce a system's carbon footprint.

8.1 Generation and Pollution

Electricity is generated in a number of ways and the different generation processes release pollutants in dissimilar ways. To determine the environmental impact of a system—its carbon emissions, for example—it is not enough to count the total energy consumed. We need to track how the energy used at each cluster was produced. The *carbon intensity* at a location is the average CO_2 pollution generated per unit of energy consumed there (grams of CO_2 per kWh). When we know the cluster consumptions and carbon intensities, we can calculate the overall emissions for which the system is responsible. In this dissertation we focus exclusively on CO_2 emissions, but other air pollutants are also associated with electricity generation (e.g., sulfur and nitrogen oxides, linked to acid rain; and methane, a potent greenhouse gas).

Power Generation Methods. Most power plants produce electricity by using thermal energy to turn turbines. Nuclear and coal-fired plants produce steam and use it to drive the blades of giant turbines. Combined cycle gas turbines use the expanding gases produced by a natural gas combustion reaction to drive their turbines, then use the waste heat for secondary steam-powered turbines. Hydroelectric power plants use gravity, pushing their turbines with water falling from elevated reservoirs. Wind farms use giant turbines whose blades are kept pointed into the wind using computer-controlled motors.

The EPA provides average emission intensities for the different kinds of power plants [24]: e.g., on average natural gas-fired generation releases 515 gCO_2/kWh; and coal-fired generation releases almost double that, 1020 gCO_2/kWh. We assume that nuclear, hydro, wind and solar power account for no pollutant emissions[1].

For individual power plants, one can acquire emissions intensity data that is more accurate than these averages. In the US, fossil fuel power plants above a threshold

[1] This is not true if we do a lifecycle analysis. Wind turbine construction involves a considerable amount of energy and raw material extraction. The uranium mining and enrichment process also results in substantial CO_2 emissions, in addition to the issue of disposing of spent nuclear fuel. When a dam is built, large amounts of vegetation may be trapped under the water, eventually decay, and release methane.

State	Emissions (gCO_2/kWh)		Generation source (%)					
	overall	non-baseload	Coal	Oil	Gas	Nuc.	Hydro.	Wind
CA	328	491	1	1	47	18	20	2
CO	854	733	72	0	24	0	3	2
FL	598	614	28	17	38	13	0	0
IL	697	903	48	0	4	48	0	0
NY	369	691	14	16	22	29	17	0
TX	600	507	37	1	49	10	0	1
WA	409	604	10	0	8	8	71	0

Figure 8-1: Annual average 2005 carbon emissions and generation fuel mixes for some different states. These numbers are from the EPA's eGRID data [24]. The emissions are calculated from consumption, not from generation (the states also imported electricity, so the consumption fuel mix may not match the generation fuel mix).

production capacity are mandated to report hourly pollutant emissions to the EPA. Although this data is not available in real-time, it is public and can be used to construct power plant emissions models [95, 36].

The generation profile of the US is dominated by coal. Coal plants produced almost 50% of US electricity in 2008. Nuclear plants produced about 20%. Natural gas turbines provide 21%. Oil based power plants, the dirtiest of the fossil fuel plants, are also used but provided less than 0.5% of the total. Hydroelectric plants accounted for a little under 7%. Electricity from other renewable sources (wind, solar, biomass, geothermal, etc.) accounted for a little over 4% of the total. Renewables are growing, and of these wind power has the most momentum.

Carbon Intensity Variability. In power pools, like the grid, that aggregate electricity from diverse providers, the carbon intensity varies in time and space, depending upon what generating assets are active and how much power they are each contributing to the pool. The combination of fuels being used to supply the grid is often referred to as the *fuel mix*.

Regional differences in generation profiles give rise to geographic variation, differentiating a system's clusters from one another. Figure 8-1 tabulates information for some states (the *baseload* power covers the minimum daily demand, as we describe later). Different states can have very different profiles: Colorado generates 72% of its

electricity from coal, while Washington generates 71% of its electricity using hydroelectric plants. Wind power has been ramping up since 2005, and some estimates put it above 4% for Texas in 2009.

Because demand for electricity fluctuates, not all power plants are continuously active. Daily demand typically follows a bell-curve-like shape with its peak in the late afternoon, and the exact shape varies by season (e.g., winter demand has two peaks) [52]. The minimum daily demand (about 35-40% of maximum demand) is covered using *baseload* power plants that run all the time. Typically coal, nuclear and hydroelectric plants are used for this purpose. When load is at high levels, *peaking* power plants are activated. The power plants used for this purpose must be able to ramp up production quickly and vary their output by the minute. These plants are typically active 10-15% of the time in regions like New York, or they may only operate a few hours per year. Natural gas, oil and hydroelectric plants are commonly used for this purpose. Between base and peak load, other generators are activated as needed. These intermediate-load power plants run 30-60% of the time. Wind and solar power is intermittent, but whenever such power is available it can be used to deactivate some intermediate or peaking plants.

As we noted during our discussion of wholesale electricity markets (chapter 5), the market auction may also play a part in determining which intermediate and peaking power plants are activated. Plants that are willing to provide power at lower prices than others will tend to be activated first.

When consuming power at a single site, these activation dynamics will cause temporal variation in that site's carbon intensity. The variation occurs at multiple time scales, e.g., seasonal (is there water to power hydro systems), weekly (what are the relative prices of various fossil fuels), and hourly (is the wind blowing).

Since carbon intensity differentials exist, both in time and space, we can adapt PDR to shrink carbon footprints. Rather than attempting to minimize the dollar cost of the energy consumed, we now use a carbon cost function. In the sections that follow, we show how to construct such cost functions and speculate on how well PDR would perform in practice.

8.2 Carbon Cost Functions

We use cost functions that measure CO_2 emissions. We assume a linear relationship between total pollution and emission intensity and assume that intensity is independent of cluster energy consumption. Thus cost at a location is:

$$CO_2 \text{ emissions} \approx \text{energy consumption} \times \text{carbon intensity}$$

The advantage of this approach is that the above formula is analogous to the monetary cost function we used in chapter 6. Instead of electric bills we are now calculating emissions; and instead of electricity market prices, we are now using carbon intensities. Linearity assumptions are widely used to estimate emissions[2].

However, as in the case of our earlier price-taking assumption (§7.2), there are situations in which this linearity assumption will deviate from the ground truth. For instance, if the power-demand being shifted represents the marginal load on the grid, the shift will cause the activation or deactivation of a power plant. This could lead to a step change in the carbon intensity.

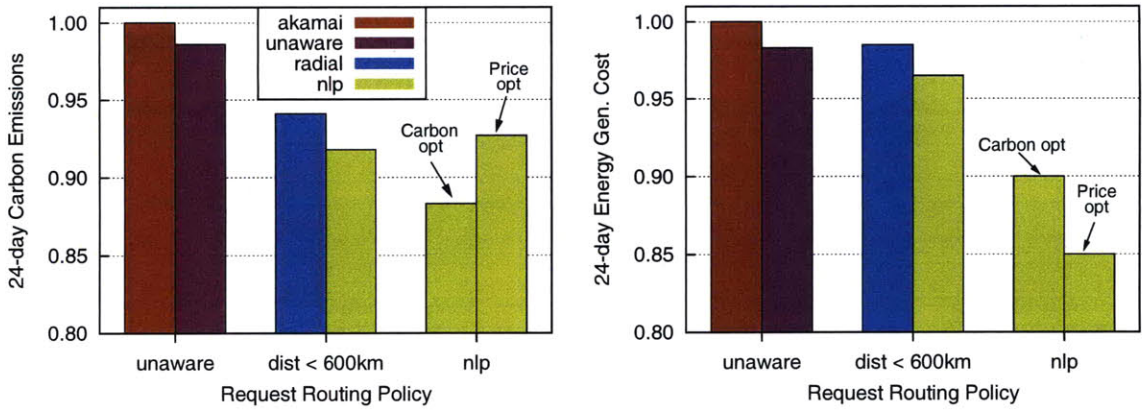
A more accurate alternative approach is to ignore carbon intensity and instead use the *marginal* carbon cost, the change in carbon emissions as we add or remove load in the region. In order to do this, one would have to coordinate with the RTO. Existing demand response programs offer precedents for such coordination (§7.3). In this dissertation, we restrict ourselves to the simpler linear approach.

8.2.1 Annual Average Intensities

Different locations have different annual average intensities. Figure 8-1 lists averages provided by the EPA for some US states. We can construct cost functions for each location with these values as constant multipliers. This will direct PDR to try to exploit geographic variations in pollution rates, but will ignore any temporal variation.

To investigate how well this would perform, we ran some simulations similar to

[2] e.g., the use of emissions factors in reporting of national greenhouse gas inventories.



(a) Carbon Dioxide Emissions **(b) Energy Costs**

Figure 8-2: Normalized costs and carbon emissions from simulations with static carbon cost functions, the Akamai geo-distribution, the 24-day traffic trace, and the G6 energy model. In both graphs, the left-most pair of bars represent routing policies unaware of carbon costs and market prices; and the right-most bar shows the *nlp* router set up to optimize for market prices instead of carbon.

those in chapter 6. The PDR routers were set up to use average carbon intensities. Using the Akamai geo-distribution and the 24-day traffic trace, we simulated several routers and calculated both CO_2 emissions and generation costs for each routing policy. Figure 8-2 summarizes our results.

We see that the market-price-based PDR policies already have smaller carbon footprints than the price-unaware policies. When the *nlp* router is unconstrained and optimizes using prices, we see a 7% reduction in CO_2 emissions. This is partly because PDR's load skewing improves the system's energy efficiency. Another factor is California, which has both the second-cheapest energy and the cleanest energy in this set of locations.

Carbon-based optimization shrinks footprints even further. When the *nlp* router is unconstrained and optimizes for CO_2 emissions, we see an almost 12% reduction in emissions. When the *nlp* router is constrained to provide comparable performance to Akamai's policy, it still manages to cut the CO_2 emissions by a little over 8%. In an 800K server system, every 1% cut in CO_2 emissions is roughly equivalent to taking 725 cars off the road[3].

[3] From our simulation results: the Akamai routing policy results in annual emissions of 4.7×10^6 $kgCO_2$, or about 905 cars worth of emissions, for every 10K servers. However the G6 model represents a system with advanced energy efficiency, with a PUE of 1.3 and peak server power of only 150W

Reducing the system's CO_2 emissions, however, results in a monetary cost to the system operator. We can see this by comparing the two unconstrained *nlp* routers in the figure. When *nlp* optimizes for prices, it can cut bills by 15%, but when it optimizes for carbon, bills fall by only 10%. At the same time, the carbon optimization cuts emissions by about 12%, while the price optimization cuts emissions by about 7%. In this case the trade-off between emissions and bills is about 125¢/kgCO_2. This is orders of magnitude more expensive than the price of carbon offsets in emissions trading markets [7]. Consequently, it seems that a system operator's best strategy is to buy emissions credits and optimize for energy bills.

8.2.2 Wind Power

Wind turbines do not output a constant amount of electricity over time. Wind power is thus characterized by variability on many time scales: seasonal, daily, hourly, even minute-to-minute variations. Studies of wind power [70] have found that there can be a distinct seasonal variation (e.g., summer production being consistently lower than winter production). Weather fronts and sunlight variations can lead to diurnal patterns. During a day there can be long lulls when power drops to zero, or power can ramp up quickly, doubling in a matter of hours, and fall just as fast [70, 42].

At the same time, wind power is growing in importance. In November 2009, wind farms produced more than half the electricity in Spain [104]. About 20% of generation in Western Denmark is from wind [70]. And while wind in the US currently provides less than 5% of the power, the Department of Energy is targeting a growth curve that will increase it to 10% by 2020 and to 20% by 2030 [57]. In California, utilities are required to generate 33% of their electricity from renewable sources by 2030 [63].

Whenever wind power is being produced, we should take advantage of it, since it releases no pollution. To do this, we can use the following cost function with PDR:

$$\text{emissions} \approx (1 - \text{wind fraction}) \times \text{energy consumption} \times \text{carbon intensity}$$

(we measured 250W peaks at Akamai). So 1% could represent much higher CO_2 emissions.

Where the wind fraction varies in time and is the share of total power being provided by wind turbines. The carbon intensity is the location's annual average.

To calculate the wind fraction, a data center using grid power may be able to use wind power forecasts provided by the grid operators. Many RTO's in the US (e.g., CAISO, MISO, and ERCOT) have provided wind power forecasts since 2008 [46]. These range from forecasts for several days ahead to real-time forecasts.

Alternatively, a data center operator may have a private contract with a wind farm (e.g., as Google does [72]). In this case, the operator will have access to a real-time signal of how much wind power is being produced.

It is easy to see how this approach can be generalized to other forms of intermittent renewable power, such as solar.

8.2.3 Real-Time Pollution Data

In theory we should be able to calculate the instantaneous carbon intensity for the grid. The RTO's know which power plants are active and what fraction of grid electricity each plant is providing. By combining this information with data from power plant sensors, or power plant emission models, one could derive a time-varying carbon intensity function.

This kind of instantaneous carbon intensity data is available in some locations. For example, one can get access to real-time emissions data for the UK national grid [17, 3]. AMEE is a web services company that helps track and measure carbon consumption [3]. For the UK, AMEE determines the instantaneous fuel mix on the grid, calculates a weighted average CO_2 emissions rate for each 5 minute period, and streams the data to its customers.

If one had access to these kinds of emissions signals for multiple locations, one could easily use them with our optimization framework. Unfortunately, these signals are only available in a few places and, where they are available, the spatial resolution is quite low (e.g., AMEE's data computes an intensity for the *entire* UK). Furthermore, it is not clear how accurate these numbers are. Studies have shown that there are inaccuracies in the power plant CO_2 estimation methodology used by the US

government: data sets from different government agencies do not agree [36].

Looking at the current trends in the energy sector, we expect that real-time carbon intensity data will eventually be available, be accurate and have a high enough spatial resolution to be usable by PDR implementations.

8.2.4 Extrapolating from Grid Load

In the meantime, we describe a way to construct *pseudo* carbon intensity functions—approximations of the true intensity functions—that vary in time and in space. Our construction relies on a real-time signal that is routinely provided by grid operators: the current demand load in a region. We transform that load signal into an emissions estimate by using a regional model.

The idea is to coarsely model the dynamic that different sets of power plants will be active at different load levels. In regions with nuclear or hydro baseload power and natural gas turbines that activate at higher loads, the carbon intensity rises at higher loads. On the other hand, in a region with coal baseload power instead of nuclear, the carbon intensity falls at higher loads, because natural gas is cleaner than coal. In figure 8-1, we see that some US states use baseload power that is cleaner than non-baseload power (e.g., WA and CA), while others use baseload power that is dirtier (e.g., coal-heavy states CO and TX).

Our model divides power plants into two groups: baseload and non-baseload. For each region, three model parameters are required: an average baseload level (L_B in kWh), the average carbon intensity for baseload power (CI_{base}), and the average carbon intensity for non-baseload power ($CI_{nonbase}$). The pseudo carbon intensity is specified as a function of current load L_t:

$$pCI(L_t) = \frac{max(L_t - L_B, 0) \times CI_{nonbase} + min(L_B, L_t) \times CI_{base}}{L_t} \quad gCO_2/kWh$$

Any fraction of load above the average baseload is assumed to be powered using none-baseload resources; and when load is below L_B, we assume that only baseload power is being used (this is the reason for the *max* and *min* terms).

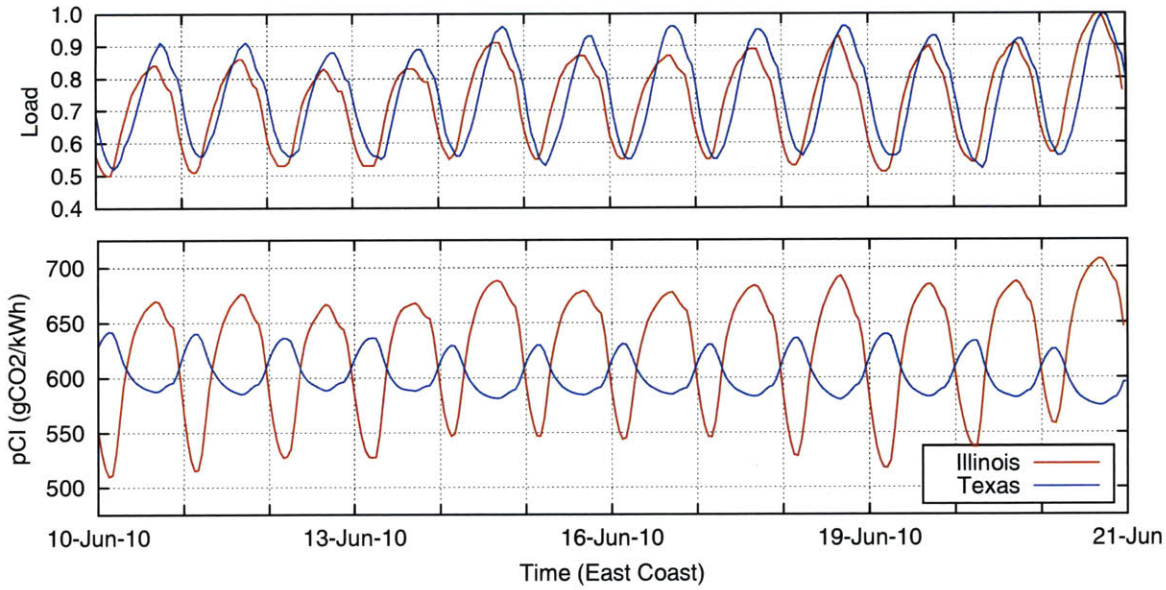


Figure 8-3: Normalized load and pCI values for two regions with different generation profiles. Power in Illinois gets dirtier at higher loads, the opposite of what happens in Texas. The load signals are aggregate loads for the local RTO and are each localized to their RTO's maximum.

We use the EPA's annual averages (figure 8-1) for the CI parameters. The EPA data gives the non-baseload CI and an overall CI. By processing the historical load data for a RTO and calculating the average fraction of non-baseload power, we can derive CI_{base}. However, the model parameters need not remain constant in time. For instance, the average summer baseload for a region is likely to be more than the average winter baseload.

Figure 8-3 shows load and pCI for two US states during some recent summer days. Note that although the load signals are highly correlated, the pCI signals are negatively correlated. Both regions selected for this figure have similar overall emissions rates. Generation in Illinois is split between nuclear and coal (both around 48%). Thus as load rises, coal is used more, making the power dirtier on average. In contrast, baseload in Texas is coal, and higher loads are mostly handled using natural gas. Thus as load in Texas rises, more gas is used, and energy gets cleaner on average.

It is also worth noting that expected demand signals are also available, so we can easily calculate and optimize using near-future pCI expectations. Additionally, the pCI functions can be combined with the earlier wind-based cost functions.

Summary

In this chapter we argued that power-demand routing could be used to reduce a system's environmental footprint. We provided evidence that environmental costs vary across clusters and in time, in a manner that PDR can productively exploit. We focused on optimizing for CO_2 emissions and noted that we expect RTO's to provide real-time carbon intensity data streams in the future. Until that happens, we showed that approximate carbon intensity functions can be constructed and used to rank a system's clusters based on the *greenness* of their energy.

Chapter 9.

Related Work

We leverage a large body of related work in this dissertation. Most of that work has already been cited inline. In this chapter we describe some related work that we did not get a chance to cover earlier.

Power-demand routing precursors. An early proposal advocated 'following the moon', a primitive form of PDR. A cloud provider with data centers at several geographical locations, it was posited, could transfer applications from the side of the world in daylight to the dark side of the planet, where power would be cheaper [94, 113]. Network latency was not viewed as an important concern.

A technique recently implemented by Google is also closely related to PDR. Google has begun operating a data center in Belgium that has no chillers and relies on external air for cooling. On days that are too warm for this sort of air cooling, Google plans to shut down equipment in this data center and shift load to other data centers [9].

Related Traffic Engineering Projects. Although not much work has been done on optimizing system energy costs, there exists a body of work related to ours that has investigated how to jointly optimize for bandwidth costs and performance. Goldenberg et al [61] proposed a method to optimize network cost and performance for multihomed users. They developed algorithms to optimize for 95/5 network costs, among other cost models, using an optimization framework similar to ours. Zhang et al [115] described a method to jointly optimize for performance and bandwidth

costs in the sort of geo-distributed systems we study. Their optimization framework is similar to ours, but they focus on multihomed systems, and do not include energy costs. Finally, Agarwal et al [37] described a method to optimize the placement of objects and replicas (or shards) in a large geo-distributed cloud service.

Energy Costs and Carbon Footprints. Closely related to PDR is the recent work of Le et al [79, 80]. They offer a general optimization framework for minimizing energy costs and maximizing the usage of green energy in a multi-data-center system. This work is fairly new, comprising a workshop paper and a technical report, and our initial papers predate theirs. They model system aspects we do not (e.g., SLA's) and their cost functions are composite cost functions that combine monetary and environmental costs. Parts of our optimization framework borrow from theirs. Unlike our real-time market model, however, they use a more simplistic bimodal on-peak/off-peak price model and a very simple linear energy model.

Chapter 10.

Conclusion

A great deal of contemporary research is concerned with increasing the energy efficiency of servers and large disturbed systems by reducing their energy consumption. In this dissertation we charted out a new—but complementary—direction, exploring another aspect of efficiency. Our approach promises to increase economic efficiency by preferentially using the least expensive energy, and environmental efficiency by preferentially using the greenest energy.

We proposed power-demand routing (PDR); offered a traffic engineering framework that clarifies the trade-offs between energy costs, network costs, and performance goals; and described algorithms that could be used to implement PDR. We showed how routing policy could be coupled to signals from spot electricity markets and—through extensive market analysis, empirical modeling and simulation—we argued that existing systems could save millions annually by implementing PDR. The bounds derived in this dissertation should not be taken too literally. Our cost and traffic models are based on actual data, but they incorporate a number of simplifying assumptions. Despite the many caveats, it seems clear that the nature of geographical and temporal differences in the price of electricity offers operators of large distributed systems an opportunity to reduce the cost of servicing requests.

More generally, we focused on identifying factors that dictate the effectiveness of PDR. Systems may evolve in an unexpected direction, and their behaviour may deviate from the models we studied in this dissertation. Still, our work offers a

template for how to evaluate an actual system to determine whether PDR can be used to improve that system's operational efficiency.

Future Work

Our work is foundational and thus a number of open questions remain. There is much potential for future work in this area. Some possible avenues are:

Implementation. In this dissertation we analyzed PDR using models and simulation. Clearly, the next step is to evaluate an implementation. We feel that the results of our analysis provide more than enough motivation to attempt an implementation.

Prediction. We assumed that the routing layer responds instantaneously to market price fluctuations, but our simulations show that response delays could diminish savings. Being able to accurately predict near-future prices may therefore be important in practice.

Data Migration. The partial replication model we discussed in chapter 7 assumes that shards remain fixed in space. However, if a system can predict that a location will experience an extended period of high energy costs (e.g., due to a heat wave) popular shards should be moved out of that region. The data migration problem has previously been studied in the context of performance alone [37]. Dealing with energy cost dynamics poses challenges but also promises benefits.

Background Processing. A significant fraction of the work in geo-distributed systems is not triggered by user requests. For instance web search infrastructures use background processes that rebuild search indices. Energy costs could play a role in deciding where to start these processes. Furthermore, for a long-running process, the system may benefit by periodically suspending and migrating the process to a different location, to exploit a cluster cost differential.

Interfacing with Power Producers. The operators of distributed systems can interact with power producers in many ways. This dissertation has focused on a relatively passive approach: operators are connected to a grid, monitor spot prices and react to favourable conditions. As we discussed in chapter 7, there are other mechanisms in place that service operators may be able to exploit. It is unclear which mechanism is the best option. Demand response is one possibility, but the associated transfer of control from the distributed system operator to the grid operator may cause complications. Another possibility is to bid *negawatts* in the electricity markets, but here the optimal bidding strategy is not obvious.

How to design a good interface between power producers and the operators of geo-distributed systems remains an open question. A good interface would allow producers to exploit the potential of space-shifting power consumption, to improve the efficiency of the grid, and at the same time allow server operators to extract considerable economic value from this relationship.

Weather Differentials. Data centers expend a lot of energy running air cooling systems, sometimes more than 30% of their total energy. In newer facilities, when ambient air temperatures are low enough, external air can be used to radically reduce the power draw of the chillers. At the same time, weather temperature differentials are common for some cluster geo-distributions. This suggests that significant energy savings can be achieved by dynamically routing requests to sites where the heat generated by serving the request is most inexpensively removed. Unlike price or carbon intensity differentials, which reduce cost or carbon but not energy, routing requests to cooler regions may also be able to reduce energy.

Multi-Tenant Clouds. Our work is also relevant to multi-tenant clouds, like Amazon's EC2, where many independent applications co-exist, leasing server and storage resources on a shared geo-distributed infrastructure. We briefly touched upon the relevance of energy cost differentials to resource pricing in these clouds (§7.7), but our work in this area barely scratches the surface.

Final Thoughts

Our work started from the observation that we could dynamically displace a geo-replicated system's power consumption in space by redistributing traffic. We showed how this process could be productively driven by volatile market prices and unstable carbon intensity signals, and described an approach for estimating the usefulness of this technique in a given scenario. We also noted that their ability to shift demand for power in this way set geo-replicated systems apart from other systems connected to the electric grid, and hinted at a promising symbiotic relationship between the grid and these replicated systems.

We believe that power-demand routing has the potential to extract substantial economic benefits for the operators of today's massive geo-distributed systems, and—given current trends in system growth, data center technology and the energy sector—its utility will increase significantly in the future.

Bibliography

[1] VMware DRS: Dynamic Scheduling of System Resources.

[2] Akamai: Facts and Figures. http://www.akamai.com/html/about/facts_figures.html.

[3] AMEE: Real Time Electricity. http://wiki.amee.com/index.php/Real_Time_Electricity.

[4] Commonwealth Edison. http://www.comed.com.

[5] Data Center Containers. http://www.datacentermap.com/blog/datacenter-container-55.html.

[6] Eaton: UPS Efficiency Calculator. http://powerquality.eaton.com/calculator/.

[7] European Climate Exchange. http://www.ecx.eu.

[8] Facebook Follows Google to Data Center Savings. http://www.datacenterknowledge.com/archives/2009/11/27/facebook-follows-google-to-data-center-savings/.

[9] Googles Chiller-less Data Center. http://www.datacenterknowledge.com/archives/2009/07/15/googles-chiller-less-data-center/.

[10] HP Performance-Optimized Datacenter. Data sheet, Hewlett-Packard.

[11] Liebert Economizer Cooling Solutions. brochure, Emerson Network Power.

[12] Microsoft: 300,000 Servers in Container Farm. http://www.datacenterknowledge.com/archives/2008/05/07/microsoft-300000-servers-in-container-farm/.

[13] One Way Active Measurement Project: Internet2 Latency Data. http://www.internet2.edu/observatory/archive/data-collections.html.

[14] OpenOpt: ralg solver. http://openopt.org/ralg.

[15] Power Assure Dynamic Power Management. http://www.powerassure.com.

[16] Rackspace Earnings Report: Q1 2009.

[17] RealtimeCarbon website. http://realtimecarbon.org/.

[18] Resource Consumption Shaping. http://perspectives.mvdirona.com/2008/12/17/ResourceConsumptionShaping.aspx.

[19] SPECpower_ssj2008. http://www.spec.org/power_ssj2008/.

[20] Sustainability at MIT: MIT's Energy Footprint. http://sustainability.mit.edu/content/mits-energy-footprint.

[21] Uptime Institute. http://uptimeinstitute.org.

[22] Harmonic Currents in the Data Center: A Case Study. White paper, American Power Conversion (APC) Corp., 2003.

[23] TIA-942: Data Center Standards Overview. White Paper, ADC, 2006.

[24] EPA: The Emissions and Generation Resource Integrated Database (eGRID), 2007.

[25] Server and Data Center Energy Efficiency. Final Report to Congress, U.S. Environmental Protection Agency, 2007.

[26] Department of Energy putting power in the hands of consumers through technology . report, Pacific Northwest National Laboratory, January 2008.

[27] ERCOT Demand Response Program Helps Restore Frequency Following Tuesday Evening Grid Event. Press Report, ERCOT, February 2008.

[28] The Value of a Millisecond: Finding the Optimal Speed of a Trading Infrastructure. Technical report, TABB Group, April 2008.

[29] Energy Efficiency Baselines for Data Centers. report, Pacific Gas and Electric Co., October 2009.

[30] comScore Reports Global Search Market Growth of 46 Percent in 2009. Press Release, comScore Inc., January 2010.

[31] Electric Bill Breakdown: Understanding your Electric Bill. Pamphlet, Pennsylvania Public Utility Commission, January 2010.

[32] Electric Sales, Revenue, and Average Price 2008. Annual report, EPA, January 2010. Table 5.

[33] Explanation of Bill: Electric Time-of-Use Energy Bill. Web page, Pacific Gas and Electric Company, 2010.

[34] New Yorks Power Grid is Ready for Summer. Press Release, NYISO, May 2010.

[35] Dennis Abts, Michael R. Marty, Philip M. Wells, Pete Klausler, and Hong Liu. Energy Proportional Datacenter Networks. In *ACM ISCA*, 2010.

[36] Katherine V. Ackerman and Eric T. Sundquist. Comparison of Two U.S. Power-Plant Carbon Dioxide Emissions Data Sets. *Environ. Sci. Technol.*, 2008.

[37] Sharad Agarwal, John Dunagan, Navendu Jain, Stefan Saroiu, Alec Wolman, and Harbinder Bhogan. Volley: Automated Data Placement for Geo-Distributed Cloud Services. In *NSDI*, April 2010.

[38] Sharad Agarwal and Jacob R. Lorch. Matchmaking for Online Games and Other Latency-Sensitive P2P Systems. *SIGCOMM Comput. Commun. Rev.*, 39(4):315–326, 2009.

[39] Daniel Alvarez. Improving Data Center (Server Room) Energy Efficiency at Caltech. Technical report, Caltech, August 2008.

[40] David G. Andersen, Jason Franklin, Michael Kaminsky, Amar Phanishayee, Lawrence Tan, and Vijay Vasudevan. FAWN: A Fast Array of Wimpy Nodes. In *SOSP*, October 2009.

[41] Don Atwood and John Miner. Reducing Data Center Cost with an Air Economizer. Brief, Intel, August 2008.

[42] Hadi Banakar, Changling Luo, and Boon Teck Ooi. Impacts of Wind Power Minute-to-Minute Variations on Power System Operation. *IEEE Transactions on Power Systems*, February 2008.

[43] Luiz A. Barroso and Urs Hölzle. The Case for Energy Proportional Computing. *IEEE Computer*, 2007.

[44] Luiz André Barroso and Urs Hölzle. The Datacenter as a Computer. Synthesis Lectures on Computer Architecture, Google, 2009.

[45] Monem H. Beitelmal and Chandrakant D. Patel. Model-Based Approach for Optimizing a Data Center Centralized Cooling System. Technical report, HP Laboratories, 2006.

[46] Audun Botterud and Jianhui Wang. Wind Power Forecasting and Electricity Market Operations. White paper.

[47] Kenneth G. Brill. The Invisible Crisis in the Data Center: The Economic Meltdown of Moore's Law. White paper, Uptime Institute, 2007.

[48] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright. Power Awareness in Network Design and Routing. *INFOCOM*, 2008.

[49] Jeffrey S. Chase and Ronald P. Doyle. Balance of Power: Energy Management for Server Clusters. In *HotOS*, 2001.

[50] Gong Chen, Wenbo He, Jie Liu, Suman Nath, Leonidas Rigas, Lin Xiao, and Feng Zhao. Energy-Aware Server Provisioning and Load dispatching for Connection-Intensive Internet Services. In *NSDI*, 2008.

[51] Brian F. Cooper, Raghu Ramakrishnan, Utkarsh Srivastava, Adam Silberstein, Philip Bohannon, Hans-Arno Jacobsen, Nick Puz, Daniel Weaver, and Ramana Yerneni. Pnuts: Yahoo!'s hosted data serving platform. *VLDB*, 2008.

[52] Matthew Cordaro. Understanding Base Load Power. White paper, October 2008.

[53] Jeff Dean. Underneath the Covers at Google: Current Systems and Future Directions. In *Google I/O*, 2008.

[54] Roman Targosz et al. The Potential for Global Energy Savings from High Efficiency Distribution Transformers. White paper, Leonardo Energy, 2005.

[55] William Tschudi et al. Data Centers and Energy Use Lets Look at the Data. In *ACEEE Summer Study on Energy Efficiency*, July 2003.

[56] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andr Barroso. Power Provisioning for a Warehouse-sized Computer. In *ACM International Symposium on Computer Architecture*, 2007.

[57] Katie Fehrenbacher. U.S. Now the World Leader in Wind Electricity Generation. Technical report, September 2008.

[58] David Filo. Serving up greener data centers. Blog post, Yahoo, June 2009. `http://ycorpblog.com/2009/06/30/serving-up-greener-data-centers/`.

[59] Michael J. Freedman, Karthik Lakshminarayanan, and David Mazires. OASIS: Anycast for Any Service. In *NSDI*, 2006.

[60] Doug Garday. Reducing Data Center Energy Consumption with Wet Side Economizers. White paper, Intel, May 2007.

[61] David K. Goldenberg, Lili Qiu, Haiyong Xie, Yang Richard Yang, and Yin Zhang. Optimizing Cost and Performance for Multihoming. In *SIGCOMM*, 2004.

[62] Google Inc. Efficient Computing: Data Centers. `http://www.google.com/corporate/green/datacenters/`.

[63] Government of California. California Renewables Portfolio Standard.

[64] Albert Greenberg, James Hamilton, David A. Maltz, and Parveen Patel. The cost of a cloud: research problems in data center networks. *SIGCOMM Comput. Commun. Rev.*, 39(1):68–73, 2009.

[65] Lester Hadsell and Hany A Shawky. Electricity Price Volatility and the Marginal Cost of Congestion: An Empirical Study of Peak Hours on the NYISO Market. *The Energy Journal.*

[66] James Hamilton. Jeff Dean on Google Infrastructure. Blog post, June 2008.

[67] James Hamilton. 60,000 servers at Facebook. Blog post, June 2010.

[68] James Hamilton. Cloud Computing Economies of Scale. Presentation, Amazon, 2010.

[69] Pedro Hernandez. EnerNOC: What Heat Wave? *earth2tech blog*, July 2010.

[70] Hannele Holttinen. Hourly Wind Power Variations in Nordic Countries. *Wind Energy*, 2005.

[71] Urs Hölzle. Powering a Google Search. *Official Google Blog*, January 2009.

[72] Urs Hölzle. Reducing our carbon footprint with the direct purchase of renewable energy. *Official Google Blog*, July 2010.

[73] Jeff St. John. Cali Opens the Market to Demand Response. *earth2tech blog*, August 2010.

[74] Jeff St. John. When Negawatts Equal Megawatts, Demand Response Blooms. *earth2tech blog*, March 2010.

[75] Paul L. Joskow. Markets for Power in the United States: an Interim Assessment, August 2005.

[76] Randy H. Katz. Tech Titans Building Boom. *IEEE Spectrum*, February 2009.

[77] Jonathan G. Koomey, Kenneth G. Brill, Pitt Turner, John Stanley, and Bruce Taylor. A Simple Model for Determining True Total Cost of Ownership for Data Centers. White paper, Uptime Institute, 2007.

[78] Craig Labovitz. How Big is Google? Technical report, Arbor Networks, March 2010. `http://asert.arbornetworks.com/2010/03/how-big-is-google/`.

[79] Kien Le, Ricardo Bianchini, Margaret Martonosi, and Thu D. Nguyen. Cost- and Energy-Aware Load Distribution Across Data Centers. In *HotPower*, 2009.

[80] Kien Le, Ozlem Bilgir, Ricardo Bianchini, Margaret Martonosi, and Thu D. Nguyen. Managing the Cost, Energy Consumption, and Carbon Footprint of Internet Services. Technical report, Rutgers University and Princeton University, December 2009.

[81] Greg Linden. Make Data Useful. Presentation, Amazon, November 2006. At Stanford.

[82] Philip J.A. Ling. Transformers and Associated Losses - The Opportunity for Savings. White paper, Powersmiths International Corps., 2001.

[83] Chris Malone and Ben Jai. Insights into Google's PUE Results. Presentation, Google, 2009.

[84] Michael Manos. Chicago Area Data Center begins its journey. Blog post, Microsoft, October 2008.

[85] Lou Marchant. Data Center Growing Pains. In *USENIX Large Installation System Admin. Conference*, 2007. Presentation.

[86] John Markoff and Saul Hansell. Hiding in Plain Sight, Google Seeks an Expansion of Power. *the New York Times*, June 2006.

[87] Mitch Martin, Mukesh Khattar, and Mark Germagian. High Density Heat Containment. In *ASHRAE Journal*, December 2007.

[88] David Meisner, Brian T. Gold, and Thomas F. Wenisch. PowerNap: Eliminating Server Idle power. In *ACM ASPLOS*, 2009.

[89] Microsoft Environmental Sustainability group. Q&A with Rob Bernard. Video.

[90] Midwest ISO. Market Concepts Study Guide. 2005.

[91] Dean Nelson. EBay's flagship data center is open for business in Utah. Blog post, EBay, May 2010. `http://datacenterpulse.org/blogs/geekism/bullet_proof`.

[92] Official Twitter blog. Inauguration Day on Twitter. `http://blog.twitter.com/2009/01/inauguration-day-on-twitter.html`.

[93] Steven Pelley, David Meisner, Thomas F. Wenisch, and James W. VanGilder. Understanding and Abstracting Total Data Center Power. In *Workshop on Energy-Efficient Design*, 2009.

[94] Geva Perry. On Clouds, the Sun and the Moon. *GigaOM blog*, June 2008.

[95] Gabrielle Pétron, Pieter Tans, Gregory Frost, Danlei Chao, and Michael Trainer. High-resolution emissions of CO_2 from power generation in the USA. *Journal of Geophysical Research*, 2008.

[96] Platts. Day-Ahead Market Prices. In *Megawatt Daily*. McGraw-Hill. 2006-2009.

[97] Asfandyar Qureshi. Plugging Into Energy Market Diversity. In *HotNets*, October 2008.

[98] Asfandyar Qureshi, Rick Weber, Hari Balakrishnan, John Guttag, and Bruce Maggs. Cutting the Electric Bill for Internet-Scale Systems. In *SIGCOMM*, August 2009.

[99] Rajen Sheth. Disaster Recovery by Google. *Offical Google Blog*, March 2010.

[100] Randy Shoup. Scalability Best Practices: Lessons from eBay.

[101] Anil Rao. SeaMicro SM10000 System Overview. White paper, June 2010.

[102] Neil Rasmussen. Electrical Efficiency Modeling for Data Centers. White paper, American Power Conversion (APC) Corp., 2007.

[103] Neil Rasmussen and James Spitaels. A Quantitative Comparison of High Efficiency AC vs. DC Power Distribution in Data Centers. White paper, American Power Conversion (APC) Corp., 2008.

[104] Michael T. Reese. Lessons From a Wind Power Milestone. Technical report, December 2009.

[105] Richard L. Sawyer. Making Large UPS Systems More Efficient. White paper, American Power Conversion (APC) Corp., 2006.

[106] Eric Schurman and Jake Brutlag. Performance Related Changes and their User Impact. Presentation, Google and Bing. At Velocity 2009.

[107] Severin Borenstein. The Trouble With Electricity Markets: Understanding California's Restructuring Disaster. *Journal of Economic Perspectives*, 2005.

[108] Rebecca Smith. Incentives Prove Powerful. *The Wall Street Journal*, September 2007.

[109] Jeremy Stribling. All-Pairs-Pings for PlanetLab, December 2005. `http://pdos.csail.mit.edu/~strib/pl_app/`.

[110] My Ton and Brian Fortenbury. Uninterruptible Power Supplies. White paper, LBNL affiliates, 2005.

[111] United States Department of Energy, Official Statistics. `http://www.eia.doe.gov`.

[112] United States Federal Energy Regulatory Commission, Market Oversight. `http://www.ferc.gov`.

[113] James Urquhart. 'Follow the law' computing. *The Wisdom of Clouds blog*, June 2008.

[114] World Bank. World Development Indicators Database.

[115] Zheng Zhang, Ming Zhang, Albert Greenberg, Y. Charlie Hu, Ratul Mahajan, and Blaine Christian. Optimizing Cost and Performance in Online Service Provider Networks. In *NSDI*, 2010.