Reducing Energy Costs in Internet-Scale Distributed Systems Using Load Shifting

Vimal Mathew[†], Ramesh K. Sitaraman[†‡] and Prashant Shenoy[†]
[†]University of Massachusetts, Amherst [‡]Akamai Technologies Inc.
Email: {vml, ramesh, shenoy}@cs.umass.edu

***Abstract*—Energy costs have become a significant fraction of the operational costs of running large Internet-scale distributed systems (IDS). In this work, we propose a demand-response technique where the system temporarily reduces its energy usage in response to pricing signals from a smart grid. Our proposed demand-response technique involves deferring the load from elastic requests to later time periods in order to reduce the server demand and the current energy usage, and hence, energy costs. We propose an optimal offline algorithm for demand response and evaluate it on production workloads from a commercial content delivery network using realistic electricity pricing models. Our optimal offline algorithm can achieve 12% energy cost savings for time-of-use electricity pricing, even when only 40% of the load is elastic and the service delay is at most 6 hours. The savings increase to 32% under peak-based demand pricing and to 23% under a combination of time-of-use and demand pricing. Further, we show that almost all the energy cost savings can be achieved with no increase in the bandwidth cost of the IDS.**

I. Introduction

Large Internet-scale distributed systems (IDS) have emerged in recent years for applications ranging from cloud-based service delivery to content delivery. The infrastructure for an Internet-scale distributed system consists of tens of thousands of servers deployed across a large number of data centers spanning the globe. Such systems are used today to run many popular applications such as news, entertainment and e-commerce [1].

Recent studies have shown that the energy costs have become a significant component of the total cost of operating and owning an Internet-scale distributed system. The energy cost of such a system includes the cost of powering the servers (and associated infrastructure) and the cost of cooling them, and in some systems, these energy costs can be as much as 20-30% of the total operational costs. Consequently the design of energy-efficient distributed systems has emerged as a popular research topic. Energy-efficiency techniques for such systems attempt to reduce the energy usage of the system or the energy costs (i.e., bills), or both [2], [3], [4], [5].

A concurrent trend is the emergence of the smart electric grid that supports many technologies and features to encourage greater adoption of energy-efficiency techniques. These include the availability of novel electricity pricing models to encourage greater energy efficiency, the deployment of smart meters for fine-grain metering and billing needed by such pricing models, and automated demand-response where the grid provides explicit signals to consumers to reduce their usage during peak periods of supply-demand imbalances. While demand-response involves explicit requests to users to reduce usage, we note that variable pricing schemes provide an *implicit* form of demand-response by discouraging users from using "too much" electricity when the electricity prices are high.

In this paper, we study how Internet-scale distributed systems can exploit smart grid features such as demand response to reduce their energy costs. There are two possible methods for reducing energy usage in an IDS in response to explicit or implicit demand-response signals. Both methods involve reducing the load at the data center that receives such a signal and then shutting down a subset of the servers to reduce the total energy usage. One possible approach to reducing energy use is to move a portion of the load to other nearby data centers and then shutting down a portion of the servers; this is achieved by having the IDS redirect some of the incoming requests to other nearby data centers and ensure that data is already replicated to service these requests from alternate sites. This approach was studied in [5] where this mechanism was employed to reduce electricity bills by redirecting load from data centers with higher electricity prices to others with lower prices. This approach, and related ones, implicitly assume that the incoming requests need to be serviced immediately (i.e., in "real-time"). In this paper, we study an alternate approach that moves load in the temporal dimension (rather than spatially or geographically, as has been done in prior work [5]) in order to reduce energy costs. Our approach assumes that *not all of the incoming requests need to be serviced immediately.* While requests to interactive services such as web requests do need immediate service, there are other classes of requests that are elastic and can be delayed if necessary. Examples of such elastic requests include background downloads of software updates by operating systems, distribution of OS-level or security patches and content prefetching for local caching.[1] In addition to elastic content requests, Internet-scale distributed systems also see elastic requests for computation—such as batch jobs like transcoding of videos [6], analytics processing, nightly backups, or book-keeping operations such as accounting and billing. Thus we assume that an IDS sees two types of requests: interactive requests that require immediate service and elastic requests that can be delayed if necessary. We study how such a system can respond to demand-response signals from the smart grid by delaying elastic requests and shutting down some of the servers, thereby temporarily reducing energy usage (and thus, energy costs).

Our paper makes the following contributions:

- In the offline context where the full load sequence is known

[1]All major OS platforms—Mac, Windows and Linux—as well as many phone-based OSes routinely download software updates in the background.

ahead of time, we derive provably optimal algorithms for demand-response that delay load to minimize the overall cost.
- We evaluate our optimal offline algorithm on a large CDN workload using an extensive set of pricing contracts that include time-of-use energy pricing and peak demand pricing. We see savings of 12% even when only 40% of the load is elastic and off-peak usage is charged at half the rate of on-peak usage. We also demonstrate that almost all the energy savings can be attained with no increase in the bandwidth costs.
- For a peak demand pricing contract the algorithm does significantly better, achieving 32% savings under similar constraints.
- For hybrid contracts where both energy usage and demand charges are included in the energy costs, we show that 23% savings are possible for the case when energy and demand contribute almost equally to the total cost.
- We find that upper-bounding the service delay by 6 hours is sufficient to achieve the maximum possible savings for 40% elastic load under various pricing contracts.

The rest of this paper is structured as follows. Section II presents some background and the models assumed for the workload, power consumption and electricity pricing. Our algorithm for optimizing energy costs via demand-response is presented in Section III. Results from our experimental evaluation are presented in Section IV. We present related work in Section V and conclude in Section VI.

II. Background

Internet-scale Distributed Systems: Our work assumes an Internet-scale Distributed System (IDS) that provide service delivery or content delivery to its users. Content distribution networks (CDNs) are an example of an Internet-scale distributed system, and so are distributed cloud-based service delivery networks. A large IDS employs tens of thousands of servers that are spread across a large number of data centers; each data center houses a cluster of servers and the size of each cluster can vary from hundreds to many thousands of servers [1]. Incoming requests for service are assumed to be forwarded to an appropriate cluster by the IDS, and the request is then serviced by one of the servers within that cluster. Our work assumes that a request can be one of two types: interactive requests that require immediate service and elastic requests that can be delayed if needed by the system. In this work, we assume that each request, whether interactive or elastic, is always serviced by the cluster to which it is sent by the IDS. That is, we do not consider the ability of the IDS to redirect some of the load to other nearby clusters, and only look at temporal load optimizations for elastic requests. While it is possible to combine techniques for moving load across clusters with those that move load across time, we leave the design of such hybrid techniques to future work.

We are interested in quantifying the potential energy savings that can result by delaying elastic requests when performing smart-grid demand response. Demand response (DR) is a technique by which a customer temporarily reduces electricity usage in response to a signal from the grid; in our context, demand response refers to any technique that the IDS can employ to reduce or defer its energy usage in response to signals from the grid. We assume that the smart grid exposes variable electricity prices to each customer; the exact pricing models considered in this study are detailed later in this section. Since price of electricity is no longer flat, the varying prices serve as implicit signals for demand-response When the electricity price is high or when higher electricity usage will result in higher costs, the consumer (which, in our case, is the IDS) is incentivized to temporarily curtail usage or shift usage to lower-price periods, and thereby reduce costs. Our paper studies an optimization approach for performing such demand-response in an IDS. Our work focuses only on implicit demand-response (that responds to pricing signals) and we do not consider explicit demand response here. Temporary deferral of elastic requests in response to an explicit DR signal from the grid is an easier problem and it is straightforward to incorporate such DR signals into our current work.

Workload Model: The workload of an IDS is generated by users and applications around the world. The global load balancer of the IDS partitions the load and directs a part of the load to each cluster of the IDS. Since our energy cost optimizations do not move load across clusters, we model and optimize the load arriving at each cluster independently. For each cluster, we model the load arriving at that cluster by an *arrival sequence* $\lambda = \langle \lambda_0, \lambda_1, \ldots, \lambda_{T-1} \rangle$, where λ_t is the load that *arrives* at the cluster at time step t. We assume that a fraction κ of the incoming load is elastic and that the elastic load can be served in a delayed fashion. Specifically, we assume that the maximum allowed service delay for elastic load is τ. As a result of our optimizations, the loads are processed by the servers in the cluster at times that are potentially different from when they arrived. The output of our optimization is a *service sequence* that we represent by $\hat{\lambda} = \langle \hat{\lambda}_0, \hat{\lambda}_1, \ldots, \hat{\lambda}_{T-1} \rangle$, where $\hat{\lambda}_t$ represents the load that will be *served* by the cluster at time t.

Power consumption model for clusters: A power consumption model is used to derive the instantaneous power drawn by the cluster, given its service load sequence $\hat{\lambda}$. Our cluster power model is based on our earlier work in [7]. We assume that the cluster is fully power proportional and consumes power that equals $u \cdot P_{peak}$, where the u is the utilization of the cluster defined as the ratio of the load served by the cluster and its peak capacity. P_{peak} is the maximum power that can be drawn by the cluster that equals the product of the number of servers in the cluster and the peak power draw of each server. Based on a typical deployed server used by IDNs, we assume that each server can draw 97W of power at peak. Note that we assume that the cluster is power proportional since a number of techniques such as server shutdown [4] are known to make clusters close to power proportional. We also model the power required for cooling the cluster as below.

$$P^{\mathbf{COOL}} = P^{\mathbf{COOL}}_{peak} \times \left(A + B \cdot u' + C \cdot u'^2\right)$$

where u' is the utilization of the chiller and the constants A, B, and C can be derived from the regression curves provided by the California Energy Commission [8]. We refer to our earlier work [7] for more details on our cooling model.

Electricity Pricing Models: The cost of electricity is often computed on the basis of the four generic metrics described below. These metrics are themselves computed from "instantaneous" measurements of electricity consumption made

throughout the billing period that is typically a month. Each metric below is either a demand metric that is based on peak KW measurements or an energy usage metric that is based on the energy consumed in KWHs. Further, some parts of the day are denoted as peak, when energy consumption is usually high, and other parts of the day are denoted as off-peak, when the energy consumption is usually low. We first derive the integrated thirty-minute values by partitioning the billing period into 30-minute intervals and computing both the average demand (KW) and the energy KWHs) in each 30-minute interval. We then compute the four metrics below.

1) On-peak demand (D_{on}): The maximum integrated thirty-minute demand (in KWs) during on-peak periods.
2) Off-peak demand (D_{off}): The maximum integrated thirty-minute demand (in KWs) during off-peak periods.
3) On-peak energy usage (E_{on}): Energy consumed (in KWHs) during on-peak periods.
4) Off-peak energy usage (E_{off}): Energy consumed (in KWHs) during off-peak periods.

We consider three commonly used pricing models in our work. Let the cost of electricity to serve a load sequence λ under a particular pricing model π be denoted by $cost_\pi(\lambda)$. We compute $cost_\pi(\lambda)$ as follows. First we apply the cluster power model to determine how much instantaneous power is drawn by the cluster to serve a given load sequence. We then compute the four metrics above using the instantaneous power draw and use it as follows.

1) The first model we consider is the *time-of-use (TOU) pricing model*[9] where the utility computes the electricity bill based only on energy usage and does not explicitly impose a demand price that depends on the peak consumption. If π is a tariff that uses the TOU model then

$$cost_\pi(\lambda) = \alpha_{on} E_{on} + \alpha_{off} E_{off},$$

where α_{on} the on-peak unit price (in \$/KWH) and is more expensive than the off-peak unit price α_{off}. Of particular interest is the ratio of off-peak to on-peak energy prices $\rho_E = \frac{\alpha_{off}}{\alpha_{on}}$. Small values of ρ_E imply a cheap off-peak price, while $\rho_E = 100\%$ is equivalent to flat pricing.

2) The second model we consider is the *demand pricing model* where the utility computes the electricity bill based only on the demand and does not explicitly charge for the energy consumed. If π is a tariff that uses demand pricing then

$$cost_\pi(\lambda) = \beta_{on} D_{on} + \beta_{off} D_{off},$$

where β_{on} the on-peak unit price (in \$/KW) is more expensive than the off-peak unit price β_{off} (in \$/KW). Of particular interest is the the ratio of off-peak to on-peak demand prices $\rho_D = \frac{\beta_{off}}{\beta_{on}}$. Small values of ρ_D imply a much cheaper off-peak price, while $\rho_D = 100\%$ is equivalent to flat demand pricing.

3) In the most general model which we call the *hybrid pricing model* [10], [11] all four metrics above are used to compute the energy cost. In particular, $cost_\pi(\lambda) = \alpha_{on} E_{on} + \alpha_{off} E_{off} + \beta_{on} D_{on} + \beta_{off} D_{off}$. We define the *mixing coefficient* as the ratio $\rho_M = \frac{\beta_{on}}{\alpha_{on}}$, where a value of 0 implies a pure energy usage pricing, while ∞ implies a pure demand pricing. Note that we can rewrite the incurred cost as $\beta_{on}\left(D_{on} + \rho_D D_{off} + \rho_M \left\{E_{on} + \rho_E E_{off}\right\}\right)$.

III. An Optimal Algorithm for Demand Response

We describe our algorithm for demand response that optimally delays load to minimize the total energy cost of an IDS. The optimal offline algorithm works individually for each cluster of the IDS and does not move load across clusters. Let the incoming load at a cluster be represented by an arrival sequence $\lambda = \langle \lambda_0, \lambda_1, \ldots, \lambda_{T-1} \rangle$, where λ_t is the load that arrives at the cluster at time step t. Further, let the fraction of the incoming load that is elastic be κ and let the maximum allowed service delay for elastic load be τ.

Our algorithm works in two steps. First, our algorithm creates a modified load sequence called the service load sequence that we represent by $\hat{\lambda} = \langle \hat{\lambda}_0, \hat{\lambda}_1, \ldots, \hat{\lambda}_{T-1} \rangle$, where $\hat{\lambda}_t$ represents the load that will be *served* by the system at time t. Note that $\hat{\lambda}$ represents the load sequence obtained after the algorithm moves around the load to optimize energy costs. (For simplicity, assume that $\lambda_t = \hat{\lambda}_t = 0$, for $t < 0$ and $t \geq T$.) Next, our algorithm uses the service sequence $\hat{\lambda}$ and produces a set of specific load movements $L_{t,t'} \geq 0$ that transforms the arrival sequence λ to the service sequence $\hat{\lambda}$. Specifically, $L_{t,t'}$ is the amount of elastic load that is moved from time t to time t', for all $0 \leq t \leq T-1$ and $t \leq t' \leq t+\tau$. We describe each step in detail below.

A. Constructing the service load sequence $\hat{\lambda}$

The algorithm delays processing some of the elastic load to minimize the energy cost, while ensuring that no elastic load is delayed more than τ time steps and further the cluster's capacity bounds are met. Let f_t be the elastic load that arrived at time step t but was postponed to be processed at a later step by our algorithm. Since the amount of elastic load arriving at time t is at most $\kappa\lambda_t$, the following holds.

$$f_t \leq \kappa \cdot \lambda_t, \forall t \tag{1}$$

The load that is delayed at a time step is assigned by the algorithm to be processed at a later time step. Let p_t represent the total elastic load that arrived at the cluster at some time in the past but is assigned to be served at time t. We can write the load served by the cluster at time t as

$$\hat{\lambda}_t = \lambda_t + p_t - f_t, \forall t \tag{2}$$

For simplicity, for values of t outside of our time window we set both p_t and f_t to be zero, i.e., $p_t = f_t = 0$ for $t < 0$ and $t \geq T$. Since the algorithm can only move elastic load to a future time slot and never back to a past time slot, we require that the total load served in every prefix in the service load sequence is upper bounded by the corresponding load from the arrival load sequence. In other words,

$$\sum_{i=0}^{t} \hat{\lambda}_i \leq \sum_{i=0}^{t} \lambda_i, \forall t \tag{3}$$

By substituting for $\hat{\lambda}_i$ from Equation 2, we get

$$\sum_{i=0}^{t} f_i - \sum_{i=0}^{t} p_i \geq 0, \forall t \tag{4}$$

Since service delay is at most τ, we require that the load in the arrival sequence $\lambda_1, \cdots, \lambda_t$ should be served by the cluster within time $t + \tau$. In other words

$$\sum_{i=0}^{t+\tau} \hat{\lambda}_i \geq \sum_{i=0}^{t} \lambda_i, \forall t. \tag{5}$$

Substituting for $\hat{\lambda}_i$, we get

$$\sum_{i=0}^{t+\tau} f_i - \sum_{i=0}^{t+\tau} p_i \leq \sum_{i=t+1}^{t+\tau} \lambda_i, \forall t \tag{6}$$

Let cluster capacity C represent the maximum load that a cluster can serve at any given time . The cluster capacity is a function of server resources available at each cluster. Since the served load cannot exceed C at any time step, we have

$$\hat{\lambda}_t \leq C, \forall t \tag{7}$$

Finally, we need the following variables to be non-negative.

$$\hat{\lambda}_t, p_t, f_t \geq 0, \forall t \tag{8}$$

Let $cost_\pi(\hat{\lambda})$ represent the energy cost of serving load sequence $\hat{\lambda}$ using energy pricing policy π. We minimize $cost_\pi(\hat{\lambda})$ subject to the linear constraints represented in Equations 1, 2, 4, 6, 7, and 8. Since the constraints are linear and we know that the cost function $cost_\pi$ described in Section II is also linear for the tariffs π that we consider, we can solve the minimization problem as a linear program (LP).

Theorem 1. *For a given arrival load sequence λ, our linear program produces a feasible service load sequence $\hat{\lambda}$ that has the minimum energy cost.*

Proof: Our LP formulation has a feasible solution since the input arrival sequence λ satisfies the capacity constraints of Equation 7. Here we make the reasonable assumption that the load balancer of the IDS distributes load to each cluster such that arriving load satisfies the capacity constraint. Thus, $\hat{\lambda}_t = \lambda_t$ and $p_t = f_t = 0$, for all t, is a feasible solution for the LP. It follows that our algorithm yields a feasible service sequence with minimum cost. ■

B. Constructing the load movement schedule L

The first step of our algorithm does not explicitly produce a schedule for how much elastic load moves from each time t to each time t', $t' > t$. However, such a schedule $L_{t,t'}$ can be computed given the output service sequence $\hat{\lambda}$ and the input arrival sequence λ as follows. We create a directed graph $G = (V, E)$ with capacities assigned to each edge as follows. The vertex set $V = \{s\} \cup U \cup V \cup \{s'\}$, where s is a source node, s' is a sink node, $U = \{u_0, u_1, \cdots, u_{T-1}\}$, and $V = \{v_0, v_1, \cdots, v_{T-1}\}$. The edge set E has an edge (s, u_t) for each $u_t \in U$ with capacity $w(u, s_t) = \lambda_t$. Likewise, it has an edge (v_t, s') for each $v_t \in V$ with capacity $w(s_t, s') = \hat{\lambda}_t$. Finally, we add edges $(u_t, v_{t'})$ with capacity $+\infty$ as long as $t \leq t' \leq t + \tau$. We then compute the maximum flow from source s to sink s' in graph G and compute the required load movement schedule $L(t, t')$ to equal the flow routed on edge $(u_t, v_{t'})$.

Theorem 2. *The above process finds a valid load movement schedule L that corresponds to the arrival sequence λ and service sequence $\hat{\lambda}$ in time $O(\tau T^2)$.*

Proof: First, we establish that all the load is successfully reassigned without any being dropped. That is, the maximum flow routed equals the total load $\sum_i \lambda_i$ that arrived at the cluster. Since the maximum flow equals the minimum capacity of a cut that separates the source s and sink s' vertices, we compute the capacity of the minimum cut of G. Note that the minimum cut will not contain any edge in $U \times V$ since those edges have infinite capacity. Therefore, it suffices to consider cuts that place vertices $\{s\} \cup \{u_0 \cdots u_t\} \cup \{v_0, \cdots, v_{t+\tau}\}$ on one side and rest of the vertices on the other side, for some $0 \leq t \leq T - 1$. Such a cut has capacity

$$\sum_{i=t+1}^{T-1} \lambda_i + \sum_{i=0}^{t+\tau} \hat{\lambda}_t,$$

which using Equation 5 is at least $\sum_{i=0}^{T-1} \lambda_i$. Now noting there exists a cut of size $\sum_{i=0}^{T-1} \lambda_i$, namely the cut with source s on one side and all other vertices on the other side, we can conclude that the capacity of the minimum cut is $\sum_{i=0}^{T-1} \lambda_i$ which in turn equals the routed flow through G. Thus, all load that arrived at the cluster is routed through G. Further, note that L constructed in this fashion obeys the delay bound of τ, since we added only edges from a vertex u_t to vertices $\{v_t, \cdots, v_{t+\tau}\}$ when constructing G. Thus, the load movement schedule L is valid and when L is applied to the arrival load sequence λ we obtain the service load sequence $\hat{\lambda}$. Finally, note that using Orlin's max flow algorithm, computing the load assignment L takes $O(|V||E|) = O(\tau T^2)$ time. ■

IV. Evaluating the Benefits of Demand Response

To evaluate the cost benefits of demand response (DR) in an IDS we used extensive traces from Akamai [1], the largest commercial CDN, and ran the optimal demand response algorithm presented in Section III for each Akamai cluster. We used each of the three electricity pricing models described in Section II and analyzed the energy cost benefits for the IDS. For all our evaluations, we report on system-wide cost savings for the IDS by aggregating our results across all clusters. The system-wide metrics capture the situation where demand response is implemented in all the clusters of the IDS. As a baseline we compute the energy cost incurred by the IDS when no demand response is implemented in any of the clusters, i.e., in the baseline no load is shifted and the arrival load sequence and service load sequence are identical for each cluster. Energy cost savings is the percent reduction in cost due to DR, i.e.,

$$100 \times ((\text{baseline cost}) - (\text{cost with DR})/(\text{baseline cost})).$$

A. Empirical Data from the Akamai Network

For our analysis, we used extensive load traces collected over 25 days from a large set of Akamai clusters deployed in data centers in the US. The 22 clusters captured in our traces are distributed widely within the US and had 15439 servers in total, i.e., it is a representative sampling of Akamai's US deployments. Our load traces account for a peak traffic of 800K requests/second and an aggregate of 950 million requests

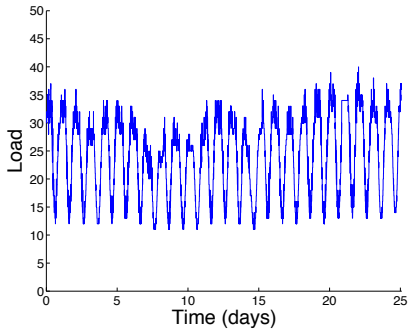


Fig. 1: Average load per server measured every 5 minutes across 22 Akamai clusters in the US over 25 days.

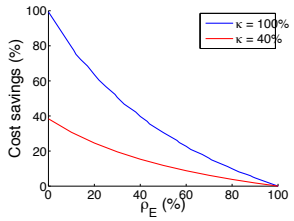
delivered to clients. The traces consist of a snapshot of total load served by each cluster collected every 5-minute interval from Dec 19th 2008 to January 12th 2009, a time period that includes the busy holiday shopping season for e-commerce traffic (Figure 1). In the figure, one may note load variations due to day, night, weekday, weekend, and holidays (such as low load on day no. 8, which was Christmas).

B. Time-of-use (TOU) Pricing Model

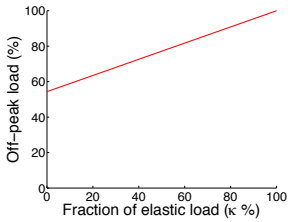
We evaluate energy cost benefits of DR on a typical time-of-use energy contract where the energy usage charge is a function of the time of day. The energy consumed between 9 AM to 9 PM on weekdays is charged at the on-peak energy rate of α_{on} dollars per kWh. The energy consumed during the remaining duration is charged at the off-peak rate of α_{off} dollars per kWh.

Varying ρ_E. Electric utility companies incentivize off-peak usage by providing discounted pricing. We capture this through $\rho_E = \frac{\alpha_{off}}{\alpha_{on}}$, the ratio of off-peak to on-peak energy usage charge. $\rho_E = 1$ corresponds to flat pricing where the energy charge is independent of the time of day. $\rho_E = 0$ corresponds to the case where off-peak usage is free (such as in underutilized renewable sources of energy). To study the impact of discounted pricing, we varied ρ_E and plotted it against the savings obtained by our algorithm for $\tau = 12$ hours. (Figure 2a). A service delay of half a day allows us to move almost the entire load from peak periods to off-peak hours ($\rho = 0$), saving 99% when $\kappa = 100\%$, and 38% savings with $\kappa = 40\%$. The savings drop to 0 when the incentive is removed and off-peak is charged at the same rate as on-peak ($\rho_E = 1$). For a typical value of $\rho_E = 50\%$ where off-peak energy charge is half of the on-peak charge we are able to save 12% even when only 40% of the load is elastic.

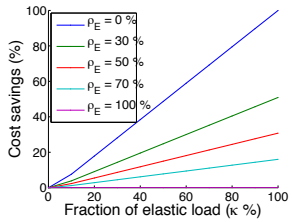
Varying elastic load fraction κ. Any increase in the fraction of elastic load κ is exploited by the algorithm by moving a larger fraction of the overall load to off-peak hours. Figure 2b quantifies this by plotting κ against the fraction of overall traffic served during off-peak hours over the duration of the entire trace. For interactive loads, where $\kappa = 0$, about 55%



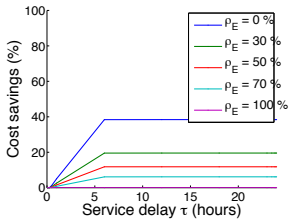
(a) Energy cost savings as a function of ρ_E for different fractions of elastic load (κ). 12% savings when $\rho_E = 50\%$ for 40% elastic load with $\tau = 12$ hours.



(b) 72% of the total load is served at off-peak hours when $\kappa = 40\%$ of the load is elastic with $\tau = 12$ hours.



(c) Cost savings increase linearly with the fraction of elastic load.



(d) At 40% elastic load, max service delay $\tau = 6$ hours is sufficient to get maximum savings.

Fig. 2: Time-of-use pricing

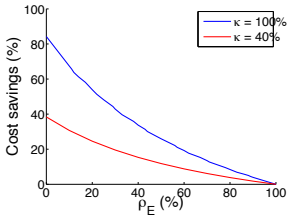
of the entire load is handled during off-peak hours. With increasing flexibility to delay load, the fraction of off-peak load increases linearly with κ. For typical values of $\tau = 12$ hours, $\kappa = 40\%$ the algorithm serves 72% of the entire load during off-peak hours.

The linear relation between κ and the off-peak load gets reflected in the cost savings as well, as seen in Figure 2c. Individual curves in the figure correspond to different values of the energy usage pricing ratio ρ_E. The lower the value of ρ_E, the higher the discount for off-peak usage and thus the greater savings.

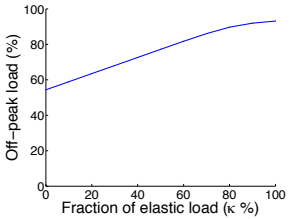
Varying maximum allowable delay τ. Different elastic tasks processed by an IDS have different delay sensitivities. A task such as billing is relatively insensitive to delay, since it suffices that the monthly bills for customers of the IDS is ready by the end of the month. However, other elastic tasks like a software update or video transcoding is expected to complete within hours.The relation between maximum allowable service delay τ and cost savings obtained by the algorithm are shown in Figure 2d for $\kappa = 40\%$. Individual curves in the figure correspond to different values of the energy usage pricing ratio ρ_E. It is interesting to note that increasing τ beyond a threshold provides little additional cost savings. In particular, a service delay τ = 6 hours is sufficient to obtain the maximum possible savings. Thus, adding elastic loads with more laxity than 6

hours does not provide larger benefits. The six hour threshold is a consequence of the time duration of the on-peak and off-peak time periods in the TOU pricing.

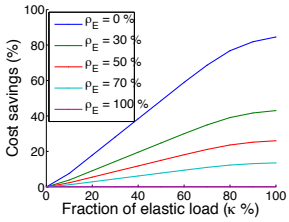
Optimizing electricity costs without increasing bandwidth costs: The TOU pricing model does not explicitly charge for the maximum power demand of a cluster. So the cost optimizations we saw earlier in this section could potentially create new load peaks when moving load from on-peak to off-peak hours. In fact, such peaks could cause the maximum load of the service load sequence to be higher than that of the arrival load sequence! Such a situation is untenable from the standpoint of other operational costs incurred by an IDS. Besides electricity, a primary operating cost for an IDS is bandwidth. Bandwidth is often priced using a 95/5 contract where the billing period is divided into 5-minute intervals and the average bandwidth used by the cluster is computed over each such interval. The bandwidth cost of the cluster is then proportional to 95^{th} percentile of the 5-minute averages [12]. We use the maximum load of the service load sequence of a cluster as a reasonable proxy for bandwidth costs incurred in that cluster. In particular, we assume more load means proportionally more bandwidth usage. Further, as we did in [12], we use "maximum" as a proxy for the "95^{th} percentile" as the latter is difficult to analyze and optimize. Note that if our energy cost optimization increases the bandwidth cost, that could negate the economic incentive[2] for the IDS performing such an optimization.



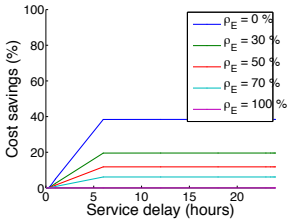
(a) Energy cost savings as a function of ρ_E for different fractions of elastic load (κ). 12% savings when $\rho_E = 50\%$ for 40% elastic load with $\tau = 12$ hours.



(b) 72% of the total load is served at off-peak hours when $\kappa = 40\%$ of the load is elastic with $\tau = 12$ hours.



(c) Cost savings increase linearly for $\kappa < 70\%$ and then slowly plateaus



(d) At 40% elastic load, max service delay $\tau = 6$ hours is sufficient to get maximum savings.

Fig. 3: Energy cost optimization without increasing bandwidth costs using the max-load constraints.

We now optimize demand response in the TOU pricing model with the additional constraint that the bandwidth costs are not increased. To achieve this we add a new constraint to our optimization algorithm mandating that the maximum load of the output service load sequence $\hat{\lambda}$ is no more than the maximum load of the input arrival load sequence λ. Specifically, let the maximum load in the arrival sequence be $\lambda_{max} = \max_{i=0}^{T-1} \lambda_i$. We require that $\forall i, \hat{\lambda}_i \leq \lambda_{max}$.

A limit on the maximum load decreases the ability to run at higher utilization and thus exploit energy discounts effectively. Therefore we would expect cost savings to decrease with the max-load constraints. Figure 3a shows that savings drops to 84% when off-peak energy is free ($\rho_E = 0$) for pure elastic load ($\kappa = 100\%$). It is interesting to note however that the additional constraints have no impact for a lower fraction of elastic load ($\kappa = 40\%$). Comparing figures 2c and 3c we see that the max-load constraint has no impact on the behavior of the algorithm for $\kappa < 70\%$.

C. Demand Pricing

Demand pricing is an important component of most realistic electricity pricing contracts, allowing electric utilities to directly manage the peak power demand by charging on the basis of it. A demand pricing contract consists of an on-peak demand charge β_{on} and an off-peak demand charge β_{off}. The on-peak charge is applied to the maximum integrated thirty-minute demand during on-peak periods (D_{on}) seen over the billing period. Similarly the off-peak charge is applied to the maximum integrated thirty-minute demand during off-peak periods (D_{off}). The electricity cost for a demand pricing policy π for a load sequence λ is

$$cost_\pi(\lambda) = \beta_{on} D_{on} + \beta_{off} D_{off}.$$

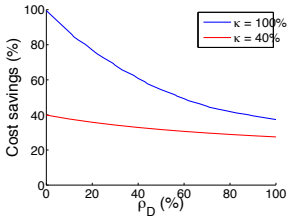
Varying relative off-peak ratio ρ_D. Electric utilities are underutilized during off-peak hours and can support higher demands from individual consumers and incentivize them by discounted off-peak pricing. We capture this discounting through $\rho_D = \frac{\beta_{off}}{\beta_{on}}$, the relative price of off-peak demand. $\rho_D = 0$ corresponds to free usage during off-peak hours, and $\rho_D = 1$ corresponds to time-insensitive demand pricing. Figure 4a plots cost savings as a function of the relative off-peak price ρ_D. For a maximum service delay of half a day the savings resemble those seen earlier for pure energy usage contracts when $\rho_D = 0$. But for $\rho_D = 100\%$ savings are still possible by smoothing out the peaks. When the entire load is capable of withstanding service delays of $\tau = 12$ hours, we see savings of 37%. For a lower value of $\kappa = 40\%$, we still get savings of 27% at $\rho_D = 100\%$. For typical values of $\rho_D = 50\%$ and $\kappa = 40\%$ we get 32% savings.

Varying percent of elastic load κ. Since pricing depends on peak demand, substantial savings can be obtained by smoothing out the largest peaks with relatively low movement

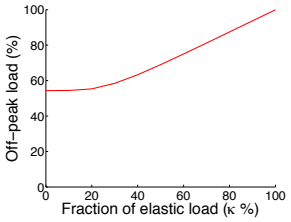
[2]It is also worth noting that any scheme that increases maximum load also increases the maximum power demand, instead of decreasing it. This negates a primary purpose of an utility offering TOU pricing to incentivize reduction in peak power demand.

in load. As the peaks and valleys get shallower, more load needs to be moved for incremental savings. We see this in Figures 4b and 4c where for low values of κ, savings grow rapidly without moving load from on-peak to off-peak hours. As κ increases beyond 30%, the gains obtained by local valley filling are exhausted and additional gains are obtained by moving traffic to off-peak hours.

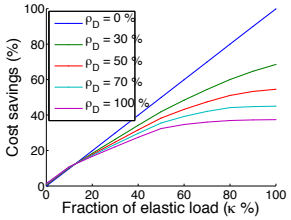
Varying maximum allowable service delay τ. The relationship between the maximum allowed service delay and cost savings are shown in Figure 4d for $\kappa = 40\%$. As in the case for time-of-use contracts, we see that maximum possible savings are achieved by a service delay of at most 6 hours.



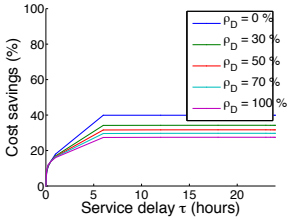
(a) 32% savings when $\rho_D = 50\%$ for $\kappa = 40\%$ elastic load with $\tau = 12$ hours.



(b) 63% of the total load is served at off-peak hours for 40% elastic $\tau = 12$ hours



(c) Cost savings flatten out as the fraction of elastic load increases



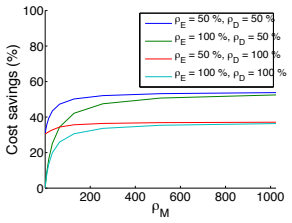
(d) At 40% elastic load, 6 hours of service delay is sufficient to get maximum savings

Fig. 4: Demand Pricing

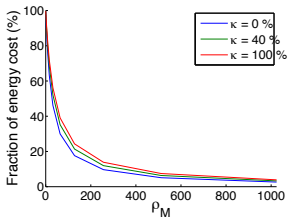
D. Hybrid Pricing

Electric utilities use a combination of energy usage and demand charges to increase the usage during off-peak hours and at the same time decrease the peak power usage. We capture this through a *mixing coefficient* $\rho_M = \frac{\beta_{on}}{\alpha_{on}}$, the ratio of on-peak demand charge to on-peak energy charge. $\rho_M = 0$ corresponds to a pure energy usage contract such as time-of-use, while as ρ_M tends to infinity the contract gets closer to a pure demand pricing model.

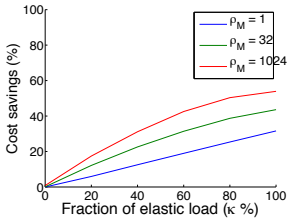
Varying mixing coefficient ρ_M. In Figure 5a, we study the impact of demand response as ρ_M is increased with $\kappa = 100\%$. When energy usage costs dominate at low values of ρ_M we see savings as observed earlier in Figure 2a with 0 savings for $\rho_E = 100\%$ and 31% for $\rho_E = 50\%$. When the contribution of demand charges dominates for large values of ρ_M we see



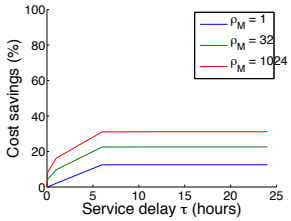
(a) 43% savings when $\rho_M = 32$ for $\rho_D = 50\%$, $\rho_E = 50\%$, $\kappa = 100\%$ elastic load with $\tau = 12$ hours.



(b) Energy costs contribute roughly half of the total cost when $\rho_M = 32$ for $\kappa = 40\%$ with $\tau = 12$ hours



(c) The relation between savings and the fraction of elastic load κ becomes non-linear as ρ_M increases



(d) At 40% elastic load, 6 hours of service delay is sufficient to get maximum savings of 23% at $\rho_M = 32$

Fig. 5: Hybrid Pricing

savings rise to roughly 37% and 53% respectively for $\rho_D = 100\%$ and $\rho_D = 50\%$ respectively, comparable to values seen in Figure 4a. It is interesting to note that savings increase as the contract tends towards a demand pricing model.

We consider a typical hybrid contract where $\rho_E = \rho_D = 50\%$ in greater detail. Figure 5b shows the contribution of energy charges as a fraction of the total cost paid to the utility. We see the curve drop-off asymptotically from 99.6% to 3.3% as ρ_M increases from 0.1 to 1024. Energy utilization charges contribute about the same as demand charges when $\rho_M = 32$.

Varying fraction of elastic load κ. Figure 5c shows the relation between the fraction of elastic κ and cost savings for different values of the mixing coefficient ρ_M. Low values correspond to linear relation, similar to pure energy contracts (Figure 2c) while high values mirror the non-linear relation observed in Figure 4c.

Varying maximum allowable delay τ. A high service delay τ allows greater freedom to the algorithm to postpone load and thus increase savings. It is interesting to note though that a maximum service delay of 6 hours is sufficient to obtain the maximum possible savings through demand response. The savings obtained increases with the value of ρ_M when the demand pricing component begins to dominate. Savings increase from 13% to 31% as ρ_M increases from 1 to 1024 for $\kappa = 40\%$ elastic load.

V. Related Work

Recently the area of energy-aware ("green") distributed system design has seen significant research attention. Design of energy-aware techniques for data centers has involved power management mechanisms at a server level [13] as well shutting down servers when not needed [14], [15], [16]. Thermal-aware placement of workloads across servers to reduce energy and cooling costs has also been studied [17]. FAWN uses "wimpy" nodes to serve simple content and reduce cluster energy costs [18]. More recent work has studied how to incorporate intermittent renewable energy to power data center clusters [2], [3]. Design of energy-aware Internet-scale systems has also seen recent attention. The use of server shutdown and cluster shutdown have been proposed as mechanisms to turn off less utilized servers or clusters in a CDN and reduce energy costs [4], [19], [7]. Separately techniques to move incoming load to other nearby data centers with lower electricity prices has been proposed as a mechanism to reduce the energy bills of an IDS [5]. Our approach is complementary since we propose moving the load in the temporal dimension—by delaying elastic requests—and thereby reducing electricity bills. Integration of demand-response in data centers has been studied previously in [20], [21]. Our work differs in the focus on CDN workloads. We go beyond service delay constraints and also look at bandwidth costs which contribute to the operating costs of a CDN. We also evaluate our algorithms over a wide range of pricing models.

VI. Conclusions

In this paper we studied techniques for reducing the energy costs in an IDS by performing demand-response to respond to variable electricity prices. Our proposed demand response approach consists of moving a portion of the incoming load—comprising elastic requests—to a later point in time, thereby temporarily curtailing the server demand and reducing energy costs. Such an approach is best suited for elastic requests such as background downloads of software updates or background computational tasks that do not always require immediate service. We presented an optimization-driven algorithm for our demand-response approach and evaluated the potential benefits of this approach for realistic workloads from a commercial CDN and realistic electricity pricing models. Our results showed that our algorithm can achieve 12% savings in the presence of time-of-use electricity pricing when only 40% of the demand is elastic. The savings increase to 32% under peak-based demand pricing and to 23% under a combination of time-of-use and demand pricing. Further, most of the energy savings can be obtained without an increase in bandwidth costs.

As part of future work, we plan to study hybrid techniques that combine the ability to move load in the spatial dimension (by moving some load to nearby data centers) as well as the temporal dimension (by deferring a portion of the load) to achieve greater energy savings. It is likely that geographically separated data centers will differ not just in the price of power but also the type of contract imposed by the utility, which can provide a greater scope for cost savings. While our work provides an *upper bound* on energy savings possible through demand response, we plan to extend it to the online setting where future loads are not known in advance and the impact of delaying requests may be difficult to predict.

Acknowledgements: This work was supported in part by NSF grants CNS-1117221, CNS-1143655, CNS-0916972 and CNS-0855128.

References

[1] E. Nygren, R. Sitaraman, and J. Sun, "The Akamai Network: A platform for high-performance Internet applications," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 3, pp. 2–19, 2010.

[2] N. Sharma, S. Barker, D. Irwin, and P. Shenoy, "Blink: managing server clusters on intermittent power," in *ASPLOS*, 2011, pp. 185–198.

[3] I. Goiri, W. Katsak, K. Le, T. Nguyen, and R. Bianchini, "Parasol and greenswitch: Managing datacenters powered by renewable energy," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2013.

[4] V. Mathew, R. K. Sitaraman, and P. Shenoy, "Energy-aware load balancing in content delivery networks," in *Proc. INFOCOM*, 2012, pp. 954–962.

[5] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*. ACM, 2009, pp. 123–134.

[6] *VOD Transcoding*, Akamai Technologies, 2013, http://www.akamai.com/dl/brochures/sola_vision_transcoding_brief.pdf.

[7] V. Mathew, R. Sitaraman, and P. Shenoy, "Energy-efficient content delivery networks using cluster shutdown," in *Proceedings of the IEEE International Green Computing Conference (IGCC), Arlington, VA*, June 2013.

[8] *Nonresidential Alternative Calculation Method (ACM) approval manual for the 2008 building energy efficiency standards*, California Energy Commission, December 2008.

[9] *Ontario Hydro Rates*, http://www.ontario-hydro.com/index.php?page=current_rates.

[10] *Duke Energy : Schedule OPT*, http://www.duke-energy.com/pdfs/scscheduleopt.pdf.

[11] *Wisconsin Electric Rates*, http://www.we-energies.com/pdfs/etariffs/wisconsin/elecrateswi.pdf.

[12] M. Adler, R. K. Sitaraman, and H. Venkataramani, "Algorithms for optimizing the bandwidth cost of content delivery," *Computer Networks*, vol. 55, no. 18, pp. 4007–4020, 2011.

[13] K. Kant, M. Murugan, and D. H. C. Du, "Willow: A control system for energy and thermal adaptive computing," in *Procedings of the 25th IEEE IPDPS*, 2011.

[14] J. Chase, D. Anderson, P. Thakar, A. Vahdat, and R. Doyle, "Managing energy and server resources in hosting centers," in *Proceedings of ACM SOSP*, October 2001, pp. 103–116.

[15] A. Chen, W. Das, A. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing server energy and operational costs in hosting centers," in *Proceedings of ACM SIGMETRICS*, June 2005.

[16] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. Kozuch, "Optimality analysis of energy-performance trade-off for server farm management," in *Proc. of Performance 2010 Namur, Belgium*, November 2010.

[17] J. Moore, J. Chase, and P. Ranganathan, "Making scheduling "cool": Temperature-aware workload placement in data centers," in *Proc. USENIX ATC (USENIX '05)*, 2005.

[18] D. Anderson, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan, "Fawn: A fast array of wimpy nodes," in *Proceedings of ACM SOSP*, October 2009.

[19] Z. Liu, M. Lin, A. Wierman, S. Low, and L. Andrew, "Greening geographical load balancing," in *Proc. ACM Sigmetrics*, 2012.

[20] J. Luo, L. Rao, and X. L. Liu, "Data center energy cost minimization: a spatio-temporal scheduling approach," in *Proceedings of the INFOCOM 2013*.

[21] Z. Liu, A. Wierman, Y. Chen, B. Razon, and N. Chen, "Data center demand response: Avoiding the coincident peak via workload shifting and local generation," *SIGMETRICS Perform. Eval. Rev.*, vol. 41, no. 1, pp. 341–342, Jun. 2013. [Online]. Available: http://doi.acm.org/10.1145/2494232.2465740