Distributed dynamic speed scaling

Rade Stanojevic and Robert Shorten

***Abstract*— In recent years we have witnessed a great interest in large distributed computing platforms, also known as clouds. While these systems offer enormous computing power, they are major energy consumers. In existing data centers CPUs are responsible for approximately half of the energy consumed by the servers. A promising technique for saving CPU energy consumption is dynamic speed scaling, in which the speed at which the processor is run is adjusted based on demand and performance constraints. In this paper we look at the problem of allocating the demand in the network of processors (each being capable to perform dynamic speed scaling) to minimize the global energy consumption/cost subject to a performance constraint. The nonlinear dependence between the energy consumption and the performance as well as the high variability in the energy prices result in a nontrivial resource allocation. The problem can be abstracted as a fully distributed convex optimization with a linear constraint. On the theoretical side, we propose two low-overhead fully decentralized algorithms for solving the problem of interest and provide closed-form conditions that ensure stability of the algorithms. Then we evaluate the efficacy of the optimal solution using simulations driven by the real-world energy prices. Our findings indicate a possible cost reduction of $10-40\%$ compared to power-oblivious $1/N$ load balancing, for a wide range of load factors.**

***Index Terms*— Data center, energy management, distributed coordination, dynamic speed scaling.**

"Physics tells us its easier to ship photons than electrons; that is, its cheaper to ship data over fiber optic cables than to ship electricity over high-voltage transmission lines".

Above the Clouds: A Berkeley View of Cloud Computing.

I. Introduction

Nowadays, many internet services are structured in a "cloud" around a large number of cloud servers that are distributed worldwide to decrease the costs and to improve content availability, robustness to faults, end-to-end delays, and data transmission rates [10]. Examples include most of Yahoo! and Google services, Amazon's Simple Storage Service (S3) and Elastic Compute Cloud (EC2) as well as Akamai's Content Delivery Network (CDN). Applications that currently rely on these distributed platforms include content distribution, search, remote backup, social networking, etc. Utilizing third party resources (located in the cloud) seems to be a major step towards the new generation of powerful and cheap applications (particularly in enterprise environments), often referred to as software-as-a-service (SaaS) model.

The typical size (and consequently, the power consumption) of a network cloud is massive. The U.S. Environmental Protection Agency estimates the energy consumption by the nation's servers and data centers was 61 billion kilowatt-hours (kWh) in 2006 (1.5 percent of total U.S. electricity consumption), and is projected to reach 100 billion kWh by 2011. It is evident that, apart from the direct costs from the massive power consumption, the environmental cost of data centers is abnormally high. This has led a number of research groups to look at various approaches for reducing energy consumption/cost [1], [4], [9], [14], [22], [23], [25].

The worldwide scale[1] of such systems raises important issues as to how to efficiently control power consumption/cost in those large distributed environments. The following reasons make the design of power-aware resource-control highly nontrivial. First, energy prices exhibit large temporal and geographic variations[2] which in turn implies that the optimal operating point is not fixed, but rather changing based on (hard-to-predict) energy-price dynamics, consequently requiring an online solution. Second, the function that relates the demand and power consumption is highly nonlinear, implying that it is hard to predict what would be the global effects of actions performed locally. Finally, the dimensionality of the problem requires solutions that offer high scalability through low communication overhead, high fault robustness and fast convergence times.

In modern data centers, processors are responsible for approximately 50% of energy consumed by high-end servers [14], [17]. Moreover, every watt saved on the servers, results on the extra 0.5-1 watt saved on cooling and power delivery costs. Consequently, in this paper we investigate how balancing the load among processors with different levels of utilization, as well as different energy prices can reduce the overall energy-related costs. Our work is built on the concept of dynamic speed scaling [4]. This is a technique that allows load-aware adaptation of the speed at which processor is run, in order to save the energy. Namely, in order to run a processor at speed s, one needs to supply a power of cs^{α}, with $\alpha = 3$ being the value most commonly used in the literature [1], [4], [29]. Here we look at the problem of demand allocation across a network of processors, with the common goal of minimizing the total cost of power consumed. As we will see in the later sections the key factors affecting the cost gains are (1) the nonlinearity of the performance/energy curve and (2) the large temporal and spatial variance of the energy prices that allows significant gains by avoiding expensive sites at peak hours.

A. Problem formulation

Let D be the demand intensity (say in requests per second) that needs to be processed by N processor clusters, with

This work is supported by the SFI grant 07/IN.1/I901.

R.Stanojevic is with Telefonica Research, Barcelona, Spain and R. Shorten is with Hamilton Institute, NUIM, Ireland.

[1]For example, Google's and Microsoft's services run on up to a million servers distributed worldwide [24], [7], [14] across dozens of data centers, across all continents. Akamai's content distribution network utilizes tens of thousands of servers [10].

[2]For example, in the UK&Ireland energy market, the energy price is created once every 30 minutes with the ratio of daily highest and lowest prices varying between 3 and 10 [5]; see Figure 3 (bottom) for a time series of energy prices during a week in Jan 2009.

cluster i has P_i processors capable of performing dynamic speed scaling based on the offered load. For $i \in \{1, 2, \dots, N\}$ the cluster i processes a fraction, $D_i \geq 0$, of total demand, so that

$$\sum_{i=1}^{N} D_i = D. \tag{1}$$

With cluster i serving a load with intensity D_i, the power consumption/cost required for ensuring certain QoS level (measured through some relevant performance metrics) is a convex function of D_i: $f_i(D_i)$. Function f_i depends on the local cost of energy (in $\$/KWh$) and the cluster structure (eg. the number of CPUs, the type of application, etc.); see Section II for more details. Note here that, with given temporal and geographic diversity in the prices, the cost function f_i is location dependent and varies with time (at cluster i) of the day, seasonal changes, etc. Our design objective is obtaining a fully decentralized architecture for power-cost minimization. In terms of communication infrastructure, we allow clusters to exchange local information over the edges of the communication (undirected) graph $G = (\mathbf{N}, E)$, where $\mathbf{N} = \{1, 2, \dots, N\}$. Given this our performance goal is to design an algorithm that allows nodes (processors) to collaborate without any global information to achieve the minimum aggregate cost:

$$\min_{D_i \geq 0,\ \sum_{i=1}^{N} D_i = D} V(\mathbf{D}) = \min_{D_i \geq 0,\ \sum_{i=1}^{N} D_i = D} \sum_{i=1}^{N} f_i(D_i). \tag{2}$$

Fully distributed algorithms for solving convex problems with multiple linear constraints have been studied in [21]. The method relies on the distributed algorithm for the SUM computation which is executed at each step of the underlying gradient ascent algorithm. As we shall see, this distributed SUM computation step imposes a large communication overhead that may discourage its usage in commercial settings. In the Section III we present a method that avoids computation of aggregate SUM, and therefore significantly reduces the communication overhead (and consequently reduces the convergence time).

Comment 1: While here we model functions f_i through simple polynomials, in reality, these functions might not be parameterizable through a handful of parameters. That in turn implies that centralized approach would require each node to continuously communicate fine-grained representation of time-varying function f_i to the centralized controller. This approach would not scale to large number of nodes. Therefore, we seek for decentralized solutions that scale well with large number of nodes, through the framework described above. However, in small networks, centralized architecture is feasible, and would provide a solution of the optimization problem in a straightforward manner.

B. Our contributions

Motivated by the need for reducing power costs through distributed speed scaling, the main concern of this paper is investigation of the potential benefits that power-aware load balancing can have on the electricity costs across a network of distributed servers. In particular we seek for efficient and fully distributed solution for minimization of the aggregate costs (2). Briefly, the main contributions of our work are following:

- We propose a framework for power reduction in the cloud through distributed dynamic speed scaling that takes into account the temporal and spatial variability of the power prices.
- Two fully distributed algorithms (synchronous and asynchronous) for solving the optimization problem of interest are proposed, and sufficient conditions are derived which guarantee that both algorithms drive system to the desired state.
- Several illustrative simulations are presented to evaluate the behavior of the algorithm and support our analytical findings. In particular, we show on a synthetic example driven by the real-world energy prices, that one can expect a reduction of (processor energy related) costs in the range of $10 - 40\%$.

While the presented work has been motivated mainly by concerns related to energy reduction in the cloud, we note that the general problems of type (2) arise in many related domains. In particular load-aware WLAN spectrum sharing [8], distributed throttling of high-bandwidth aggregates for DDoS protections [28], and cost-optimal multihoming [13] share the same underlying problem (see Section V).

We shall also see that dynamic system underlying our algorithm is nonlinear and implicit. This makes the task of analysis challenging. In particular, the standard theory of distributed coordination algorithms (see [18] and references therein) cannot be employed in our case. The convergence results established in Section III-A are highly nontrivial and represent the main theoretical contribution of this paper.

From the practical side, the proposed solution requires only a few lines of code, is easy to debug, and requires configuration of at most one parameter (η). In contrast, the MRS algorithm from [21] that solves the same type of problems has a set of non-trivial rules for parameter setting, involved logic behind it, making it hard to implement and debug in a real-world setting. Finally because our schemes avoid computation of the global state, the resulting communication overhead can be several orders of magnitude smaller compared to MRS (see [27]).

II. LOAD BALANCING WITHIN ONE CLUSTER

Before we proceed with the problem of distributed power-aware load distribution, we first investigate what can be done locally in terms of load balancing among the processors within one cluster under standard CPU speed vs. power model. Let P be the number of processors within a cluster, each being able to run at speed $s \in (0, 1]$. Dynamic speed scaling literature [1], [4], [29], models the processor power consumption running at speed $s > 0$ as a function of the following form:

$$p(s) = \gamma + (1 - \gamma)s^{\alpha},$$

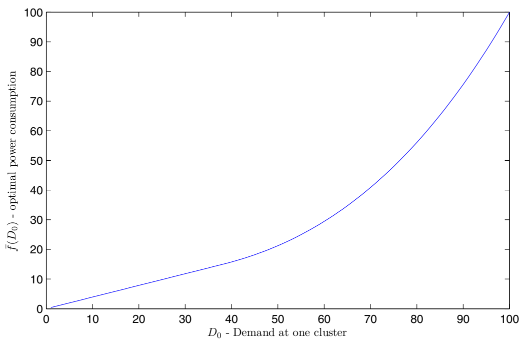

Fig. 1. Local load vs. optimal energy consumption.

```
1   InitializeDemands()
2     for $i = 1 : N$
3       $D_i \leftarrow D \frac{P_i}{\sum_{j=1}^{N} P_j}$
4     endfor

5   UpdateDemands()
6     Once every $\Delta$ units of time do
7       for $i = 1 : N$
8         $D_i \leftarrow D_i + \eta \sum_{(i,j)\in E}(h'_j(D_j) - h'_i(D_i))$
9       endfor
10    enddo
```

Fig. 2. Pseudo-code of IDC

with the power exponent $\alpha > 1$, most commonly the value $\alpha = 3$ is used. Note also that amortization parameter γ lies in the interval $(0, 1)$, as certain amount of energy is spent for running the processor even when it is idle. We assume also that when the processor is turned off the power consumption is $p(0) = 0$. Since the capacity (maximum speed) of each processor is one unit, the maximum processing speed of the whole cluster is P. For $D_0 \in [0, P]$, the following Proposition characterizes the strategy for allocating the demand among those P processors, $(s_1, \ldots, s_P)$ (with $s_i \geq 0$ and $\sum_{i=1}^{P} s_i = D_0$) that minimizes the total power:

$$A(s_1, \ldots, s_P) = \sum_{i=1}^{P} p(s_i).$$

Proposition 1: In the model described above, the load allocation that minimizes the consumed power at one cluster of P processors is the one for which k^* servers are turned on, and each of them processes the same load: $\frac{D}{k^*}$, where

$$k^* = \min(P, D_0 \left(\frac{(1-\gamma)(\alpha-1)}{\gamma}\right)^{\frac{1}{\alpha}}) \tag{3}$$

In that case, the energy consumed by the cluster is given by:

$$\bar{f}(D_0) = b(k^*) = k^*\gamma + (1-\gamma)(k^*)^{1-\alpha} D_0^{\alpha}. \tag{4}$$

The graph $(D_0, \bar{f}(D_0))$ is depicted in Figure 1 for $\alpha = 3$, $\gamma = 0.1$ on a cluster of $P = 100$ processors.

III. Load balancing across distributed clusters

In the previous section we showed how to balance the load allocated to one cluster in order to minimize the energy consumption at the cluster. In this section we look at the problem formulated in Section I-A: balancing the load D on the set of distributed clusters to minimize the aggregate cost of power (2). For cluster i serving load D_i, the energy consumption is given by $\bar{f}_i(D_i)$ (Proposition 1), and the cost of unit of power is given by ν_i. The power cost is then $f_i(D_i) = \nu_i \bar{f}_i(D_i)$. The key observation of this section is that solving convex problem (2) can be reduced to solving a distributed coordination problem (in which function values among all nodes in the network need to be equalized) that in turn helps us design algorithms with low communication overhead and fast convergence times. In order to avoid directly enforcing the nonnegativity constraints ($D_i \geq 0$ and $D_i \leq P_i$), we consider the cost functions with a logarithmic barrier

$$h_i(D_i) = f_i(D_i) - \theta \log(D_i) - \theta \log(P_i - D_i),$$

for a small parameter $\theta > 0$. Now, consider the following optimization problem, parameterized with θ:

$$\min_{\sum_{i=1}^{N} D_i = D} V_\theta(\mathbf{D}) = \min_{\sum_{i=1}^{N} D_i = D} \sum_{i=1}^{N} h_i(D_i). \tag{5}$$

For the analysis of the error introduced by this logarithmic barrier see [21]. Basically, for any $\theta > 0$, the solution of (2) is $O(N\theta)$ away from solution of (5). In the evaluation section we set $\theta = 10^{-10}$. For given Lagrange multiplier λ, the Lagrange function is:

$$\Lambda(\lambda, D_1, \ldots, D_N) = V_\theta(D_1, \ldots, D_N) - \lambda(\sum_{i=1}^{N} D_i - D).$$

Therefore the point $\mathbf{D} = (D_1, \ldots, D_N)$ that minimizes V_θ must satisfy $\frac{\partial \Lambda}{\partial D_i} = 0$ for all i, which is equivalent to:

$$h'_i(D_i) = f'_i(D_i) - \frac{\theta}{D_i} + \frac{\theta}{P_i - D_i} = \lambda.$$

Thus, finding the solution of (5) is **equivalent** to finding the point $(D_1, \ldots, D_N)$ for which $\sum_{i=1}^{N} D_i = D$ and

$$\forall(i,j) \quad h'_i(D_i) = h'_j(D_j). \tag{6}$$

Now we describe a fully decentralized algorithm that allows every cluster to obtain the optimal load it needs to process. The observation that finding the optimal point is equivalent to finding a point that satisfies (1) and (6) greatly simplifies the design. The algorithm, that we call Implicit Distributed Coordination (IDC) is given by the pseudocode in Figure 2, and parameterized by parameter η. It is fully decentralized, in that every agent exchanges only local information with its neighbors and has the following logic behind it. Let i and j be nodes connected in the communication graph G. If $h'_i(D_i)$ is greater than $h'_j(D_j)$ then this indicates that allocating some demand to node j from node i would potentially decrease the difference between $h'_i(D_i)$ and $h'_j(D_j)$. The parameter $\eta > 0$ determines the responsiveness and stability properties of IDC and the next section provides a sufficient condition under which the system converges to the optimal state.

A. Model and analysis of IDC

In this section we analyze the dynamics of IDC. We provide a bound on η that ensures the stability (convergence) of the system, describing the dynamics of the state vector, $(D_1, \ldots, D_N)$, and show that the convergence is exponentially fast (it converges with time t as e^{-at}). We model the IDC system in discrete time t. At the t-th iteration, denote by $D_i(t)$ the fraction of load, served by processor i and by

$$q_i(t) = h_i'(D_i(t)),$$

the value of h_i' at point $D_i(t)$. Being the sum of three convex functions, h_i is itself a convex function. Therefore, h_i' is an increasing function and has a well defined inverse. Let us denote by $g_i = (h_i')^{-1}$ the inverse function of h_i'. Then

$$D_i(t) = g_i(q_i(t)) = g_i(h_i'(D_i(t))). \tag{7}$$

Equation (7) represents the key relationship between $D_i(t)$ and $q_i(t)$. Given this, the dynamical system describing the evolution of $D_i(t)$, is given by:

$$D_1(0) = D_2(0) = \ldots = D_N(0) = D/N, \tag{8}$$

$$D_i(t+1) = D_i(t) + \eta \sum_{(i,j)\in E} (q_j(t) - q_i(t)). \tag{9}$$

We also denote

$$m(t) = \min_{1\le i\le N} q_i(t), \quad \text{and} \quad M(t) = \max_{1\le i\le N} q_i(t).$$

The following lemma is a straightforward consequence of the fact that G is an undirected graph.

Lemma 1: At all times t, the constraint (1) is satisfied: $D_1(t) + D_2(t) + \ldots + D_N(t) = D$.

Proof: See Technical report [27]. ■

The following theorem gives a sufficient condition under which the system (8)-(9) converges.

Theorem 1: Let d_i be the degree of node i in the communication graph. If all g_i are differentiable on $(0, \infty)$, then for every η that satisfies:

$$0 < \eta \le \frac{1}{2} \min_{1\le i\le N} \big(\inf_{y\in[m(0),M(0)]} g_i'(y) \big) \min_{1\le i\le N} \frac{1}{d_i} \tag{10}$$

the following limits exist

$$\lim_{t\to\infty} D_i(t) =: D_i^*$$

and

$$\lim_{t\to\infty} h_i'(D_i(t)) = \lim_{t\to\infty} h_j'(D_j(t)) =: q^*.$$

Proof: See Technical report [27]. ■

IV. Evaluation

In this section we present results from several representative simulations to illustrate the behavior of the algorithms discussed above. In particular we investigate the following: (1) The potential benefits in terms of cost reduction of the power-aware load balancing compared to oblivious $\frac{1}{N}$ load balancing; (2) The relationship between the aggregate load of the system and the cost reduction. Tech report [27] investigates also the scalability to large networks of IDC.

N	number of servers
P_i	number of CPUs per server i
D	aggregate demand
D_i	demand allocated to cluster i
ν_i	cost of unit of power
$f_i(D_i)$	energy consumed by cluster i
$f_i(D_i)$	cost of energy consumed by cluster i
μ	aggregate load $(D/(\sum_{i=1}^N P_i))$
α	exponent in dynamic speed scaling
γ	amortization parameter
η	gain parameter in IDC
c	overhead factor in MRS

TABLE I
SYMBOL MAP

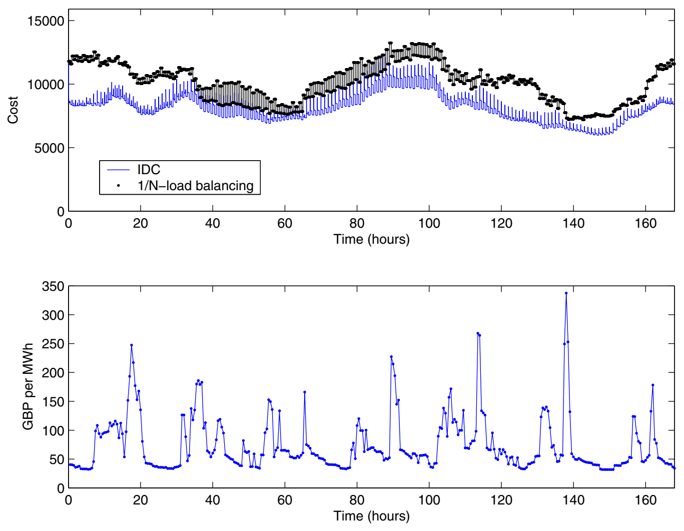

Fig. 3. Bottom: energy prices in GBP per MWh. Top: aggregate cost of running IDC algorithm compared to the aggregate cost of the dynamic speed scaling with $\frac{1}{N}$-load balancing (top). Time span of one week, 168 hours, from Monday 19/01/2009, 00:00:00 to Sunday 26/01/2009 23:59:59 (GMT).

A. Potential benefits

In our first simulation, we look at potential savings that can be expected if the cloud-based service providers employed power-aware distributed speed scaling. Figure 3 (bottom) contains the graph of energy prices in (GBP per MWh) UK for a week in January 2009 [5]. Energy prices are formed once every 30 minutes, based on the demand and available supply. The setup is following. We consider $N = 12$ identical server clusters, with $P_i = P = 100$ CPUs at each cluster, located in even time zones (GMT, GMT+2,..., GMT+22) serving a constant load $D = 600$. We assume also that the energy-price at local time t is the same across all time zones and are driven by the values depicted in Figure 3 (bottom). We use the cubic power/speed dependance ($\alpha = 3$), with amortization factor $\gamma = 0.1$, with cost at each location being simply the product of energy consumed and power price. We compare the cost of the optimal operating point with the cost of the oblivious load balancing where each location processes $\frac{1}{N}$ of the total demand. The offered load is $\mu = \frac{D}{NP} = 50\%$. The results are depicted in Figure 3 (top).

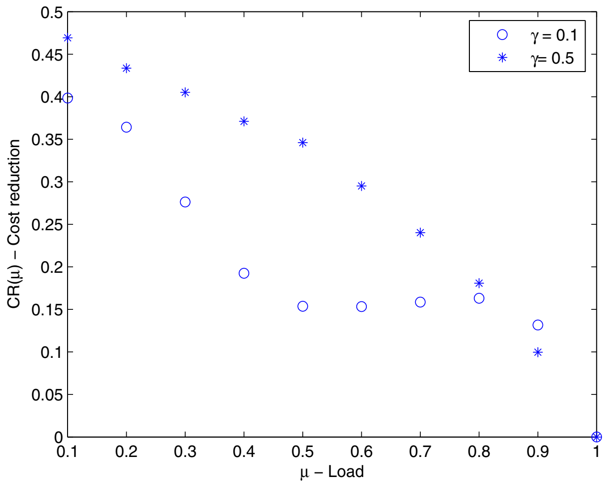

Fig. 4. Cost reduction vs. the aggregate load.

B. Aggregate load vs. cost reduction

In this subsection we evaluate the relationship between the aggregate load and the cost reduction compared to the oblivious $\frac{1}{N}$ load balancing in which each cluster processes $1/N$ of the total demand. For the setup described in Section IV-A, we vary the aggregate load μ and plot it against the cost reduction defined as

$$CR(\mu) = 1 - \frac{V(\mathbf{D}^*(\mu))}{V(\frac{D_\mu}{N}, \ldots, \frac{D_\mu}{N})},$$

where $\mathbf{D}^*(\mu)$ is the optimal operating point (solution of (2)) and $(\frac{D_\mu}{N}, \ldots, \frac{D_\mu}{N})$ is the point corresponding to the power-oblivious $1/N$ load allocation. The value of $CR(\mu)$, for scenario described in previous subsection, is depicted in Figure 4 for two values of amortization parameter: $\gamma = 0.1$ and $\gamma = 0.5$. As expected, we can see that for low loads, the improvements are greater than for heavy loads, as in low-load cases, there is more space for balancing and (almost fully) avoiding the expensive clusters. However, we observe a non-monotonic dependance between the load and the cost reduction (in $\gamma = 0.1$ case) for which we do not have an analytic explanation.

V. Conclusions

Issues related to service reliability, service availability, and fault tolerance, have encouraged many service providers in the Internet to shift from traditional centric services to cloud based services. This trend appears to be a sustained mechanism for ensuring robustness of internet services with many “big players”, such as Google, Yahoo!, Akamai, Amazon, offering a suit of cloud-based services.

While offering a huge computational power, those systems are major energy consumers. One of the techniques for CPU energy reduction is dynamic speed scaling, that adjusts the speed at which the processor is run based on current load. In this paper we propose a framework for utilizing dynamic speed scaling in a distributed setting. We propose a distributed algorithm for solving the optimization problem arising in this context that has significantly smaller communication overhead compared to state-of-the-art. Our evaluation supports our analytical findings and shows promise in non-negligible cost savings possible with schemes that exploit temporal and geographical diversity of energy prices and nonlinearity of the performance/power curve. However, the presented evaluation is limited to synthetic scenarios, and for better understanding of the benefits of large scale power-aware load-balancing, we need some realistic traffic models of load/power dependence in real-world implementations of dynamic speed scaling.

References

[1] S. Albers, H. Fujiwara. “Energy-efficient algorithms for flow time minimization”. In Proc. of STACS 2006,
[2] M. Armbrust, et al. “Above the Clouds: A Berkeley View of Cloud Computing”. Technical Report No. UCB/EECS-2009-28.
[3] Amazon Web Services. http://aws.amazon.com/.
[4] N. Bansal, K. Pruhs, C. Stein. “Speed scaling for weighted flow times”. In Proc. of SODA 2007.
[5] UK energy prices. http://www.bmreports.com/bwx_-reporting.htm.
[6] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah. “Gossip algorithms: Design, analysis and applications”. In Proc. of IEEE Infocom, 2005
[7] D. F. Carr. “How Google works”. Baseline Magazine, July 2006.
[8] R. Chandra et al. “A Case for Adapting Channel Width in Wireless Networks”. In Proc. of ACM SIGCOMM 2008.
[9] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, F. Zhao. “Energy-aware server provisioning and load dispatching for connection-intensive internet services. In Proc. of NSDI 2008.
[10] Akamai, State of the Internet, 2008. Available online: http://www.akamai.com/stateoftheinternet/
[11] EPA Report on Server and Data Center Energy Efficiency. U.S. Environmental Protection Agency, 2007.
[12] A. Gandhi, M. Harchol-Balter, R. Das, C. Lefurgy. “Optimal Power Allocation in Server Farms”. In Proc. of ACM SIGMETRICS 2009.
[13] D. Goldenberg, L. Qiu, H. Xie, Y. Yang, Y. Zhang. “Optimizing cost and performance, for multihoming”. In Proc. of ACM SIGCOMM 2004.
[14] A. Greenberg, et al. “The Cost of a Cloud: Research Problems in Data Center Networks”. ACM Com. Comm. Rev. vol 39(1) 2009.
[15] R. Gummadi, H. Balakrishnan. “Wireless Networks Should Spread Spectrum Based On Demands”. In Proc. of Hotnets 2008.
[16] K. Church, A. Greenberg, J. Hamilton. “On Delivering Embarrassingly Distributed Cloud Services”. In proc of Hotnets 2008.
[17] W. Hammond. “Efficient power consumption in the modern Datacenter”. http://www.research.ibm.com/aceed/2005/proceedings/hammond.ppt
[18] A. Jadbabaie, J. Lin, and A. S. Morse, “Coordination of groups of mobile autonomous agents using nearest neighbor rules”. IEEE Transactions on Automatic Control, vol. 48(6), 2003
[19] N. Laoutaris, P. Rodriguez, L. Massoulie. “ECHOS: Edge Capacity Hosting Overlays of Nano Data Centers”. ACM CCR, vol 38(1) 2008.
[20] T. Moscibroda, R. Chandra, Y. Wu, S. Sengupta, P. Bahl. “Load-aware spectrum distribution in wireless LANs”. In Proc. of ICNP 2008.
[21] D. Mosk-Aoyama, T. Roughgarden, D. Shah. “Fully Distributed Algorithms for Convex Optimization Problems”. In Proc. of DISC 2007.
[22] S. Nedevschi et al. “Reducing network energy consumption via sleeping and rate-adaptation”. In Proceedings of NSDI, 2008.
[23] A. Qureshi. “Plugging Into Energy Market Diversity”. Hotnets 2008.
[24] B. Raghavan, K. Vishwanath, S. Rambhadran, K. Yocum, A. Snoeren. “Cloud Control with Distributed Rate Limiting”. ACM SIGCOMM 2007.
[25] S. Srikantaiah, A. Kansal, F. Zhao. “ Energy Aware Consolidation for Cloud Computing”. In Proc. of Hotpower 2008.
[26] R. Stanojevic, R. Shorten. “Fully decentralized emulation of best-effort and processor sharing queues”. In Proc. of ACM SIGMETRICS 2008.
[27] R. Stanojevic, R. Shorten. “Distributed dynamic speed scaling”. Technical report. Online: http://www.hamilton.ie/person/rade/DDSS.pdf.
[28] C.W. Tan, D.M. Chiu, J.C.S. Lui, D.K.Y. Yau. “A Distributed Throttling Approach for Handling High Bandwidth Aggregates”. IEEE Transactions on Parallel and Distributed Systems, vol 18(7), 2007.
[29] A. Wierman, L.L.H. Andrew, A. Tang. “Power-Aware Speed Scaling in Processor Sharing Systems”. In Proc. of IEEE Infocom 2009.