An Approach for QoS-Driven Performance Modeling of Peering Content Delivery Networks

Al-Mukaddim Khan Pathan, James Broberg and Rajkumar Buyya

Grid Computing and **D**istributed **S**ystems (GRIDS) Laboratory
Department of Computer Science and Software Engineering
The University of Melbourne, Australia

{apathan, brobergj, raj}@csse.unimelb.edu.au

ABSTRACT

A Content Delivery Network (CDN) is expected to provide high performance Internet content delivery through global coverage, which might be an obstacle for new CDN providers, as well as affecting commercial viability of the existing ones. Peering of CDNs can be a way to allow cooperation between CDNs in a scalable manner and to achieve better overall service, as perceived by end-users. In this paper, we present a Quality of Service (QoS)-driven performance modeling approach for peering CDNs in order to predict the user performance. We also show that peering between CDNs upholds user perceived performance by satisfying the target QoS. The methodology presented in this paper provides CDNs a way to dynamically redirect user requests to other peering CDNs according to different request-redirection policies. The model-based approach helps an overloaded CDN to return to normal by offloading excess requests to the peers. It also assists in making concrete QoS guarantee for a CDN. Our approach endeavors to achieve scalability for a CDN in a user transparent manner.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems; C.4 [**Performance of Systems**]; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Distributed Systems*; H.3.5 [**Information Storage and Retrieval**]: Online Information service—*Web-based Services*

General Terms

Design, Measurement, Performance

Keywords

Content Delivery Networks, QoS, Modeling, Peering

1. INTRODUCTION

Content Delivery Networks (CDNs) [4] are networks of surrogate servers spanning the Internet, aiming to offer fast and reliable Web services by distributing content to *edge* servers located close to end-users. The main objective of a CDN is to deliver competitive services according to user Quality of Service (QoS) requirements. The current deployment approach for a CDN requires building a global network of surrogate servers to host replicated content. Running a global CDN is challenging in financial, technical and administrative terms, both for deployment and operation of service. Furthermore, commercial CDNs make specific commitments to their customers by signing Service Level Agreements (SLAs) [10]. An SLA is a contract between the provider and the customer to describe the provider's commitment and to specify penalties if those commitments are not met. So, if a CDN is unable to provide QoS for user requests, it may result in SLA violation and end up costing the provider.

The requirements for providing high quality service through global coverage might be an obstacle for new providers, as well as affecting commercial viability of the existing ones. To ensure QoS while serving end-user requests a CDN is required to either provide all necessary distributed computing and network infrastructure (thus massively over provisioning resources during day-to-day operation), or to be able to harness external resources on demand to meet any unexpected resource shortfalls. Therefore, the objective of providing high quality service could be achieved by permitting CDNs to cooperate and thereby provide a means for CDNs to redistribute content delivery between themselves [3].

This large Internet-wide cooperation can be termed as a 'peering arrangement' [1] or internetworking [13] between CDNs, where some CDN providers may team up at some point in time to share resources and form an alliance in order to respond to or exploit a particular niche [2]. It virtualizes multiple providers, and provides flexible resource sharing and dynamic collaboration between autonomous individual CDNs. In such a system, a CDN serves user requests as long as the load can be handled by itself. If the load exceeds its capacity, the excess user requests are offloaded to the Web servers of the peers. Thus, an overloaded CDN can redirect a fraction of the incoming content requests. This approach also provides a means to avoid long-term (i.e. periodic traffic pattern during a particular Web event) or short term (i.e. flash-crowds) bottlenecks [1][2].

Such peering arrangements are appealing, since it allows individual providers to achieve greater scale and network reach cooperatively than they could otherwise attain individually. However, developing a model capturing the characteristics of end-user requests redirection in peering CDNs is challenging for a number of reasons, which include virtualization of multiple providers and offloading end-user requests from the primary CDN provider to peers based on cost, performance and load. In such a cooperative multi-provider environment, users are redirected across distributed set of Web servers deployed by partnering CDNs as opposed to individual servers belonging to a single CDN. Moreover, limited information about response time or service cost is typically available from individual CDNs, and load balancing control is retained by an individual provider within its own Web servers. Therefore, request-redirections must occur over distributed sets of Web servers belonging to multiple CDN providers, without the benefit of the full information available, as in the single provider case.

The general objective of a peering CDNs model is to provide improved QoS performance through minimizing end-user response time. However, the proprietary nature of existing commercial CDNs makes it difficult to predict the performance a given user is expected to experience from a particular CDN.

Furthermore, such a model can be based on a complex combination of attributes such as Web server responsiveness or load, expected network delay, or geographic location. Several of these potential attributes vary over time and there is no single repository for listing the value of attributes such as geographic location or expected delay for all Internet-connected systems. Therefore, the values used in a peering CDNs model are likely to be based on heuristics.

In this paper, we present an approach to perform QoS-driven modeling of the peering CDNs based on the fundamentals of queuing theory. We also demonstrate the performance comparison of four request-redirection policies within the peering CDNs model. Our aim is to show that the cooperation between CDNs through a peering arrangement upholds user perceived performance by providing target QoS according to SLAs. Our performance models can be used to reveal the effects of peering and to predict end-user perceived performance. Our approach endeavors to assist in making concrete QoS guarantees by a CDN provider. The main contributions of this paper are:

- performance models to demonstrate the effects of peering and to predict user perceived performance;
- systematic performance analysis and measurement-based methodology to study the impact of key performance parameters such as server load and measurement errors that can be expected in a realistic system;
- an approach to measure the QoS level of a CDN provider to ensure it provides efficient services; and
- performance comparison of four request-redirection policies within the peering CDNs model.

The rest of the paper is structured as follows: Section 2 presents the related work. Section 3 describes the cooperation between CDNs through a peering arrangement. Section 4 provides the performance models and outlines our approach for measuring QoS performance of a CDN provider. Section 5 demonstrates the results. Finally, Section 6 concludes the paper with a brief summary of contributions and future works.

2. RELATED WORK

Research on Web server selection to redirect end-user requests has thus far focused primarily on techniques for choosing a server from a group administered by a single CDN. Peering of CDNs is gaining popularity among researchers of the scientific community, since such cooperation between CDNs can achieve better overall service, as perceived by end-users. Some projects are being conducted for finding ways to allow peering between CDNs. But many of them lack in virtualizing multiple CDNs for the management and delivery of content in a cooperative environment, and directing end-user requests to different CDNs based on performance to satisfy user QoS requirements.

The internet draft by IETF proposes a Content Distribution Internetworking (CDI) Model [13], which allows CDNs to have a means of affiliating their delivery and distribution infrastructure with other CDNs who have content to distribute. It recommends providing QoS in the cooperative domain either through using a supervision function or an independent third party to supervise and manage all the CDN peers, and to give some guarantees on the QoS of each CDN in the CDI model. However, it does not provide any hint on the type and/or characteristics of the supervision function to be used. The CDI model also does not examine the implications of using an independent third party for ensuring QoS guarantees to end-user requests.

A protocol architecture [14] for CDI attempts to support the interoperation and cooperation between separately administered CDNs. In this architecture, performance data is interchanged between CDNs before forwarding a request by an authoritative CDN (for a particular group), which adds an overhead on the response time perceived by the users. Moreover, being a point-to-point protocol, if one end-point is down the connection remains interrupted until that end-point is restored. Since no evaluation has been provided for performance data interchange, the effectiveness of the protocol is unclear.

CDN brokering [16] allows one CDN to intelligently redirect end-users dynamically to other CDNs in that domain. Though it provides benefits of increased CDN capacity, reduced cost and better fault tolerance, it does not explicitly consider the end-user perceived performance to satisfy QoS while serving requests. Moreover, it demonstrates the usefulness of brokering rather than to comprehensively evaluate a specific CDN's performance.

Amini et al. [15] present a peering system for content delivery workloads in a federated, multi-provider infrastructure. The core component of the system is a peering algorithm that directs user requests to partner providers to minimize cost and improve performance. But the peering strategy, resource provisioning and QoS guarantees between providers are unexplored in this work.

An approach to model traffic redirection [17] in geographically diverse server sets uses a novel metric Server Set Distance (SSD) to simplify the modeling and classification of redirection scheme. Though this modeling provides a foundation for intelligent server selection over multiple, separately administrated server pools, it does not try to show the effectiveness of any particular policy or evaluate the QoS performance of any given CDN.

WARD (*W*ide *A*rea *R*edirection of *D*ynamic Content) [18] presents a novel architecture for redirecting dynamic content requests from an overloaded Internet Data Center (IDC) to a remote replica. It demonstrates a simple analytical model to characterize the effects of wide area request-redirection on end-to-end delay. WARD can avoid over-provisioning of IDCs and achieve significant performance improvement through reduction in average request response times. But it is mainly targeted to IDCs under the control of single administrative entity. Therefore, it does not virtualize multiple providers while considering request-redirection. Moreover, it also does not provide mechanisms to evaluate the QoS performance of individual IDCs.

Cardellini et al. [19] present an architecture for enhancing QoS in geographically distributed Web systems. The architecture integrates DNS proximity and dispatcher scheduling with an HTTP redirection mechanism in order to achieve a scalable and balanced Web system. Though it aims to minimize the response time experienced by users while accessing geographically distributed Web sites, the use of HTTP request-redirection may lead to increased network impact on latency experienced by the end-users. Moreover, it does not capture the heterogeneity in Web server systems since it only considers homogeneous Web clusters (service distributions are the same) belonging to a single entity.

From a user-side perspective, Cooperative Networking (CoopNet) [12] provides cooperation of end-hosts to improve network

performance perceived by all. This cooperation between users is invoked for the duration of the flash crowd. CoopNet is found to be effective for small Web sites with limited resources. But the main problem of the user-side mechanisms is that they are not transparent to end-users, which are likely to restrict their widespread deployment. Hence, it can not be used as a replacement and/or alternative for cooperation among infrastructure-based CDNs.

Among the deployed peer-to-peer (P2P)-based CDN systems, CoralCDN [20] can be mentioned for its decentralized and self-organizing nature. It replicates content in proportion to the content popularity, regardless of the content provider's resources. Though it is built on top of a set of cooperative Web caches, it does not capture the notion of multi-provider existence for content delivery. Moreover, it gives better performance to most users for accessing participating Websites, but dose not provide mechanism to evaluate the QoS performance of a certain provider.

Other than the above mentioned related research projects/work in content internetworking domain, systems such as CoDeeN [21], Globule [22], and DotSlash [23] address the issue of collaborative content delivery. But some of these systems make strong assumptions on the characteristics of applications. Many of these systems also do not virtualize multiple providers for cooperative management and delivery of content in a peering environment.

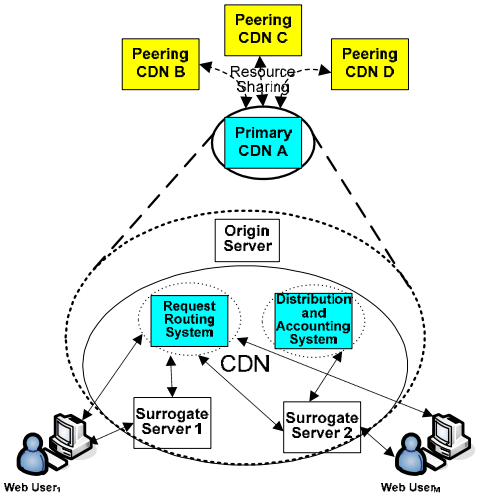


Figure 1: Abstraction of Peering CDNs

3. PEERING BETWEEN CDNs

In our approach [1], a CDN serves end-user requests as long as the load can be handled by itself. If the load exceeds its capacity, the excess end-user requests are offloaded to the Web servers of other cooperating CDNs. For the purposes of this paper, we define a 'peering arrangement' of CDNs as:

Definition of 'peering arrangement' – *A peering arrangement of CDNs is formed by a set of autonomous CDNs {CDN_1, CDN_2, …, CDN_n}, which cooperate through a mechanism **M** that provides facilities and infrastructure for cooperation between multiple CDNs for sharing resources in order to ensure efficient service delivery. Each CDN_i is connected to other peers within **M** through a 'conduit' C_i, which assists in discovering useful resources that can be harnessed from other CDNs. We denote **S** = {S_1, S_2, …, S_m} as the set of services provided by a CDN.*

Our definition of 'service' in this context is in line with the service definition in Service Oriented Computing (SOC) [24] paradigm. We define a 'service' as:

Definition of 'service' – *A service S_i, offered by a cooperating CDN_i is the endpoint of a well-defined, self-contained connection or underlying system (which does not depend on the context or state of other services) to serve a request according to the QoS requirements of end-users. A service request may specify a call for serving request for an individual file or object, a Web page (containing multiple objects) or an application service of a particular script (e.g. CGI, PHP) or any digital content.*

In Figure 1, we provide an abstraction of the peering CDNs. The initiator of a peering negotiation is called a *primary* CDN; while other CDNs who agree to provide their resources are called *peering* CDNs. The endpoint of a peering negotiation between two CDNs is a contract (SLA) that specifies the peer resources (Web servers, bandwidth etc.) that will be allocated to serve content on behalf of the primary CDN. The primary CDN manages the resources it has acquired insofar that it determines what proportion of the Web traffic (i.e. end-user requests) is redirected to the Web servers of the peering CDNs.

3.1 QoS in peering CDNs

QoS performance can be measured based on the user's experience of a service to compare the 'promise' against the 'delivery'. The definition of quality varies from different perspectives and views. In this paper, we adopt the conformance view and define QoS as the experience perceived by a user when being served by a CDN. Therefore, quality can be defined in the following way:

Definition of 'quality' – *Let A be a CDN provider and **S** = {S_1, S_2, …, S_m} be the set of services provided by it. Assume that for each service S_i, S_i^p is the quality that A promised to offer to the users and S_i^d is the actual quality delivered by A. Then the QoS for CDN A is given by,*

$$QoS_A = f(S_i^p, S_i^d)$$

where f is the function that measures the conformance between S_i^p and S_i^d.

In this definition, the notion of conformance is captured generally, but it is not specified how S_i^p and S_i^d can be measured. We anticipate measuring the QoS level of a CDN in terms of the performance perceived by the end-users while being served. It can be measured in terms of *expected waiting time* of a request to be served or arrival rates (requests/second).

3.2 SLAs to ensure QoS

Ensuring QoS guarantees requires a means of establishing a set of common quality parameters and establishing which attributes are needed by a particular customer to describe its QoS requirements. These factors are combined to an SLA that both a customer and provider agree to and that the provider refers to when monitoring its QoS performance. Examples of QoS parameters that an SLA may specify are:

- The maximum response time for a service request will not exceed 0.5 seconds.
- 95% of user requests will be completed in less than 2 seconds.
- A service will be available for at least 99.9% of the time.

From a service management and business perspective, the fulfillment and assurance of SLAs is of key importance. In our context, the importance of an SLA lies in its encoding of performance obligations, so that a CDN can manage its workload. For example, the existing SLAs that a CDN currently holds permit it to determine if a new SLA can be accepted as it is. Secondly, the SLAs are used to determine if the CDN is meeting user QoS requirements. Thirdly, a CDN uses its SLAs to quantify which resources it requires in a peering arrangement. Thus, the collected SLAs are critical in establishing the runtime performance metrics for the CDN and as a basis for establishing peering arrangements. Examples of attributes that an SLA encodes are:

- *Service type* – What service (e.g. content and/or application delivery) a user is requesting from a CDN.
- *Service requirements* – The processing and/or capacity requirements for a CDN to serve user requests.
- *Duration* – The maximum time duration within which content requests are to be served from a CDN.
- *Guaranteed QoS level* – The level of guarantee (in terms of response time) that requests will be served within a delay threshold. For example, three levels of QoS guarantees can be specified: *Gold* = user requests will be served timely in *95%* of cases; *Silver* = user requests will be served timely in *60%* of cases; *Bronze* = no guarantee (best effort) will be provided for serving user requests.

4. PERFORMANCE MODELS

In this section, we develop the performance models based on the fundamentals of queuing theory to demonstrate the effects of peering between CDNs and to characterize the QoS performance of a CDN.

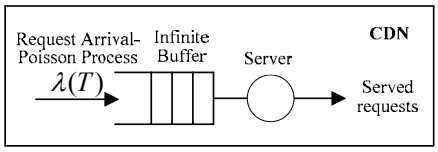


Figure 2: Model of an M/G/1 queue

4.1 Single CDN model

Let us model a CDN as an M/G/1 queue as shown in Figure 2. The request streams coming to the Web servers of a CDN are abstracted as a single request stream. User requests arrive following a Poisson process with mean arrival rate λ. All requests in its queue are served on a first-come-first-serve (FCFS) basis with mean service rate μ. It is assumed that the total processing of the Web servers of a CDN is accumulated through the server and the service time follows a general distribution. The term 'task' is used as a generalization of a request arrival for service. We denote the processing requirements of an arrival as 'task size'.

Web workloads often follow a *heavy-tailed* distribution (a very small fraction of the largest files determines a large fraction of the load) [5][6], characterized by the function, $\Pr\{X > x\} \sim x^{-\alpha}$, $0 \le \alpha \le 2$. In a CDN, users request for content of varying sizes (ranging from small to large) and thus processing requirements (i.e. task size) also vary. Hence, we model the task size on a given CDN's service capacity to follow a Bounded Pareto distribution for which the probability density function (P.D.F) is defined as

$$f(x) = \frac{\alpha k^{\alpha}}{1-(k/p)^{\alpha}} x^{-\alpha-1},$$

where α represents the task size variation, k is the smallest possible task size, and p is the largest possible task ($k \le x \le p$). By varying the value of α, we can observe distributions that exhibit moderate variability ($\alpha \approx 2$) to high variability ($\alpha \approx 1$).

By Little's law, the mean queue length $E[N_q] = \lambda E[W]$ and load on the server, $\rho = \lambda E[X]$, where $E[W]$ is the waiting time and $E[X]$ is the mean service time. Let $E[X^j]$ be the j-th moment of the service distribution of the *tasks*. We have,

$$E[X^j] = \begin{cases} \dfrac{\alpha p^j((k/p)^{\alpha}-(k/p)^j)}{(j-\alpha)(1-(k/p)^{\alpha})} & \text{if } j \neq \alpha \\ \dfrac{\alpha k^{\alpha}\ln(p/k)}{(1-(k/p)^{\alpha})} & \text{if } j = \alpha \end{cases}$$

Hence, using P-K formula, the expected waiting time $E[W] = \lambda E[X^2]/2(1-\rho)$. It can be used to measure the waiting time with respect to varying server load and task sizes.

4.1.1 Hyper-exponential approximation

In order to quantify the performance perceived by the users while being served by a CDN, we need to find the P.D.F of waiting time distribution. The Bounded Pareto distribution has all moments finite; however advanced analysis is complex due to the difficulties in manipulating the Laplace transforms of the queuing metrics (e.g. waiting time, busy period). Hence, the 'heavy-tailed' Bounded Pareto distribution can be approximated with a series of exponential distributions (known as Hyper-exponential), while still maintaining the main characteristics of the original distribution, such as heavy tail, first and second moments [9]. An n part Hyper-exponential distribution has the following P.D.F

$$h_n(t) = \sum_{i=1}^{n} P_i \lambda_i e^{-\lambda_i t}, \text{ where } \sum_{i=1}^{n} P_i = 1$$

which can be used for our purpose.

4.1.2 Service distribution and waiting time

The Laplace transform of the service distribution $h_n(t)$ is

$$L_{h_n}(s) = \int_0^{\infty} e^{-st} h_n(t)dt = \sum_{i=1}^{n} \frac{P_i \lambda_i}{\lambda_i + s}$$

The moments of the service distribution can be obtained as

$$E[X^n] = (-1)^n \frac{d^n L_{h_n}(s)}{ds}\bigg|_{s=0}$$

where X is a continuous random variable with P.D.F $h_n(t)$. The first moment (mean) $E[X]$ and the second moment $E[X^2]$ are

$$E[X] = \sum_{i=1}^{n} \frac{P_i}{\lambda_i} \text{ and } E[X^2] = \sum_{i=1}^{n} \frac{2P_i}{\lambda_i^2}$$

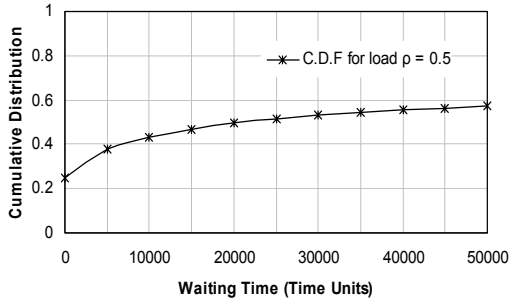
The Laplace transform of the waiting time, $L_W(s)$ for an M/G/1 queue with the hyper-exponential approximation of a Bounded Pareto distribution is defined as follows:

$$L_W(s) = \frac{s(1-\rho)}{s-\lambda+\lambda L_{h_n}(s)} = \frac{s(1-\rho)}{s-\lambda+\lambda\sum_{i=1}^{n} \frac{P_i\lambda_i}{\lambda_i+s}}$$

This result can be numerically inverted to obtain the P.D.F of the waiting time distribution, $w(t)$. It can also be used to obtain the cumulative distribution function (C.D.F),

$$W(t) = \Pr[T \le t] = \int_0^t w(t)dt$$

Using the C.D.F, concrete QoS guarantees can be made regarding the waiting time experienced by a certain percentage of user requests. Figure 3 shows the C.D.F of waiting time of a CDN for the system load $\rho = 0.5$. Here, the hyper-exponential approximation of a Bounded Pareto distribution is used with $\alpha = 1.5$, $k = 1010.15$ and $p = 10^{10}$. From the figure we observe that there is about *50%* probability that the waiting time experienced by the users will be less than *20000* time units. Thus, concrete guarantees can be made regarding the waiting time experienced by a certain percentage of user requests on a given CDN.



(a) – W(t)

Figure 3: Cumulative distribution of waiting time of a CDN modeled as an M/G/1 queue

4.2 Peering CDNs Model

A CDN's inability to meet user QoS requirements according to the SLAs may lead to a collaboration of CDNs, so that it may redirect excess requests to the Web servers of the peers. In Figure 4, a conceptual view of the peering CDNs is provided where each CDN is modeled as an M/G/1 queue.

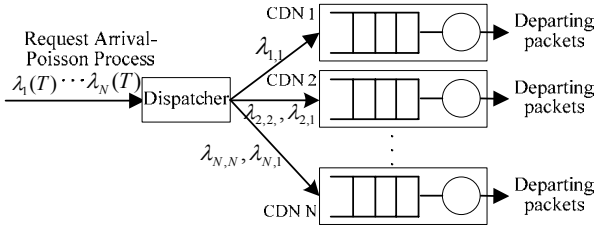


Figure 4: Conceptual view of the peering CDNs

It is abstracted that N independent streams of end-user requests arrive at a conceptual entity, called *dispatcher*, following a Poisson process with the mean arrival rate λ_i, $i \in \{1,2,...,N\}$. The dispatcher acts as a centralized scheduler in a particular peering relationship with independent mechanism to distribute content requests among partnering CDNs in a user transparent manner. If, on arrival, a user request can not be serviced by CDN i, it may redirect excess requests to the peers. Since this dispatching acts on individual requests of Web content, it endeavors to achieve a fine grain control level. We anticipate that it can also pave the ways in performing the *request assignment* and *redirection* at multiple levels – at the DNS, at the gateways to local clusters and also (redirection) between servers in a cluster. Thus, end-users can be assigned via DNS (by the CDNs updating their DNS records regularly) and also via redirection at the gateway (through dispatching) when appropriate [1]. The dispatcher follows a certain policy that assists to assign a fraction of requests of CDN i to CDN j. The request stream to a CDN in the peering CDNs model is defined as $\lambda_{j,i}$ = request to CDN j for CDN i's content. For $\forall j \neq i$, $\lambda_{j,i}$ denotes redirected user requests, where CDN i is the primary where CDN j is a peer. On the other hand, for $\forall j = i$, $\lambda_{j,i}$ denotes the user requests to a primary CDN i. For example, request to CDN B for CDN A's content can be denoted as $\lambda_{B,A}$.

The service times of the CDNs are defined as independent of interarrival times and of one another, and have a general P.D.F. Inside each request stream, there is FCFS service. Each request stream is assigned priority. Here, $p = 1,2,...,P$ priority classes of user-requests are assumed. A peer always prioritizes the requests from the primary CDN over its own user requests. However, if a redirected request (higher priority) arrives to a peer when its own user request (lower priority) is being served, it never interrupts the current service. Thus, this priority discipline is non-preemptive during service quantum of end-user requests.

4.2.1 Waiting time

The classical result [7] for non-preemptive head-of-the-line (HOL) priority queue can be used to find the expected waiting time for the p-th ($p = 1,2,\ldots,P$) priority user request,

$$W_p = \frac{W_0}{(1-\sigma_p)(1-\sigma_{p+1})}, \text{ where } \sigma_p = \sum_{i=p}^{P} \rho_i \qquad (1)$$

W_0 is the average delay to a particular priority user request due to other requests found in service. It can be expressed as,

$$W_0 = \sum_{i=1}^{P} \frac{\rho_i E[X_i^2]}{2}$$

where $E[X_i^2]$ is the second moment of service time for a customer from class i. It can be mentioned that W_k does not depend on user requests from lower priority class (i.e. $i = 1, 2,\ldots, p\text{-}1$), except for their contribution to the numerator W_0 [11].

Let us assume that the user requests for the primary CDN belongs to the p-th priority class. The Laplace transform of the waiting time for the primary CDN is denoted as $W^*_p(s)$. Using the known solution [8] for the distribution of waiting time for each priority group in a priority queue, it can be expressed as,

$$W_p^*(s) = \frac{(1-\rho)s + \lambda_L[1 - \sum_{j=1}^{p-1} \frac{\lambda_{j,j}}{\lambda_L} B_j^*(s)]}{s - \lambda_{p,p} + \lambda_{p,p} B_p^*(s)}, \text{ where } \lambda_L = \sum_{j=1}^{p-1} \lambda_{j,j} \qquad (2)$$

Similarly, for any peer with the priority in the range *1,2 ...,(p-1)* the Laplace transform of waiting time is found as,

$$W_j^*(s) = \frac{(1-\rho)[s + \lambda_{p,p} - \lambda_{p,p} G_p^*(s)]}{s - \sum_{j=1}^{p-1} \lambda_{j,j} + \sum_{j=1}^{p-1} \lambda_{j,j} \sum_{j=1}^{p-1} B_j^*(s + \lambda_{p,p} - \lambda_{p,p} G_p^*(s))} \qquad (3)$$

Here, $G^*_p(s)$ is the transform for the M/G/1 busy period distribution for a p-priority class, which is expressed as,

$$G_p^*(s) = B_p^*(s + \lambda_{p,p} - \lambda_{p,p} G_p^*(s)) \qquad (4)$$

These solutions can be used to measure the expected waiting time for end-user requests to each of the participating entities in the peering arrangement.

4.2.2 QoS performance

The ability to gauge the QoS of a CDN provider is crucial for

achieving effective service from it. The P.D.F of the waiting time distribution (through numerical inversion) for each given (primary) CDN, with independent priority class can be used to observe the expected waiting time perceived by majority of users in the peering CDNs system. Since a primary CDN's request has priority over any peer's own user requests, we can consider using (2) for a primary CDN, while (3) for any peering CDN. Though these equations are useful for computation, the iterative expression for $G^*_i(s)$ in (4) is impossible to invert numerically. Therefore, the waiting time experienced by a primary CDN's user requests is found using (2), while the classical result presented in (1) is used to find the average expected waiting time for the peer's user requests.

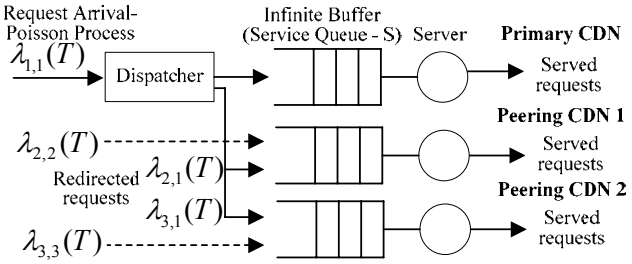


Figure 5: A peering scenario with three CDNs

Table 1: Workload model

Category	Distribution	P.D.F	Range	Parameters
Primary CDN, $0.1 \le \rho \le 0.9$	Hyper-exponential	$h_n(t) = \sum_{i=1}^{n} P_i \lambda_i e^{-\lambda_i t}$ approximating $f(x) = \frac{\alpha k^\alpha}{1-(k/p)^\alpha} x^{-\alpha-1}$	$x \ge k$	$\alpha = 1.5$ $k = 1010.15$ $p = 10^{10}$
Peer 1, $\rho = 0.5$	Hyper-exponential	$h_n(t) = \sum_{i=1}^{n} P_i \lambda_i e^{-\lambda_i t}$ approximating $f(x) = \frac{\alpha k^\alpha}{1-(k/p)^\alpha} x^{-\alpha-1}$	$x \ge k$	$\alpha = 1.5$ $k = 1010.15$ $p = 10^{10}$
Peer 2, $\rho = 0.4$	Hyper-exponential	$h_n(t) = \sum_{i=1}^{n} P_i \lambda_i e^{-\lambda_i t}$ approximating $f(x) = \frac{\alpha k^\alpha}{1-(k/p)^\alpha} x^{-\alpha-1}$	$x \ge k$	$\alpha = 2$ $k = 1500.23$ $p = 10^{10}$

5. RESULTS

In this section, we present the performance results obtained using the models presented in Section 4. We consider a peering CDNs system consisting of three CDNs, as shown in Figure 5. Each CDN is modeled as an M/G/1 queue with highly variable Hyper-exponential distribution which approximates a heavy-tailed Bounded Pareto service distribution (α, k, p) with variable task sizes. Thus, the workload model incorporates the high variability and self-similar nature of Web access. Table 1 shows the distributions, probability density functions and parameter ranges for the workload model.

For our experiments, we consider the expected waiting time as an important parameter to evaluate the performance of a CDN. In our peering scenario, we also assume an SLA of serving all user requests by the primary CDN in less than *20000* time units.

5.1 QoS performance of the primary CDN

First, we attempt to provide the evidence that a peering arrangement between CDNs is able to assist a primary CDN to provide better QoS to its users. The C.D.F of the waiting time distribution of the primary CDN can be used as the QoS performance metric. In a highly variable system such as peering CDNs it is more significant than average values. The waiting time corresponds to the time elapsed by a user request before being served by the CDN.

Figure 6 shows the C.D.F of waiting time of the primary CDN without peering at different loads. From the figure we see that for a fair load $\rho = 0.6$ there is about *55%* probability that users will have a waiting time less than the threshold of *20000* time units. For a moderate load $\rho = 0.7$, there is about *50%* probability for users to have waiting time below the threshold, while for a heavy load $\rho = 0.9$ the probability reduces to > *24%*.

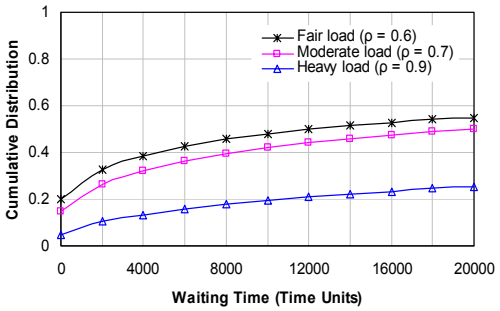


Figure 6: Cumulative distribution of waiting time of the primary CDN without peering

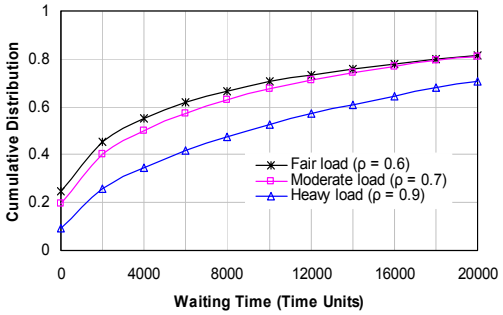


Figure 7: Cumulative distribution of waiting time of the primary CDN in a peering arrangement

The peering CDNs model arranges the participating providers according to a non-preemptive HOL priority queuing system (Section 4.2). It is an M/G/1 queuing system in which we assume that user priority is known upon their arrival to a CDN and therefore they may be ordered in the queue immediately upon entry. Therefore, various priority classes receive different grades of service and requests are discriminated on the basis of *known* priority. Thus, in our model an incoming request (with priority p) joins the queue behind all other user requests with priorities less than or equal to p and in front of all the user requests with priority greater than p. Due to this nature of the peering CDNs model, the effect of peering can be captured irrespective of any particular request-redirection policy.

Figure 7 shows the C.D.F of the primary CDN with peering for different loads. By comparing Figure 6 and Figure 7, it can be

found that for a fair load $\rho = 0.6$ there is about *80%* probability that users will have a waiting time of less than a threshold of *20000* time units. Therefore, in our scenario, peering assists the primary CDN to achieve a QoS performance improvement of about *31%*. For a moderate load $\rho = 0.7$, there is $> 81\%$ probability for users to have waiting time below the threshold, an improvement of about *38%*. For a heavily loaded primary CDN with $\rho = 0.9$ the probability reaches to about *70%*, which lead to an improvement of $> 65\%$. Moreover, for loads $\rho > 0.9$, still higher improvement can be predicted by the model. Based on these observations, it can be stated that peering between CDNs irrespective of any particular request-redirection policy achieves substantial QoS performance improvement when comparing to the non-peering case.

5.2 Request-redirection policies

In the peering CDNs model, no redirection is assumed until primary CDN's load reaches a *threshold* load ($\rho = 0.5$). This load value is also used as the *baseline* load for comparing waiting times at different primary CDN loads. Any load above that will be 'shed' to peers. A request-redirection policy determines which requests have to be redirected to the peers. Each peer is ready to accept only a certain fraction (*acceptance threshold*) of the redirected requests. Any redirected request to a given peer exceeding this acceptance threshold is simply dropped to maintain the system equilibrium. In face of sudden surge in demand, the load on a given primary CDN i, $i \in \{1,2,...,N\}$ becomes, $\rho^*_i = \rho_i - \rho^i_{redirect}$ and the redirected load is distributed among the peering CDNs. The value of $\rho^i_{redirect}$ varies depending on the dispatcher chosen redirection policy. The initial and new load on the given primary CDN i is measured by, $\rho_i = \lambda_{i,i} E[X_i]$ and $\rho^*_i = \lambda^*_{i,i} E[X_i]$ respectively. $\lambda_{i,i}$ is the initial arrival rate, whereas, $\lambda^*_{i,i} = \lambda_{i,i} - \lambda^i_{redirect}$ is the new arrival rate.

We define four request-redirection policies for evaluation within the peering CDNs model:

- *Uniform Load Balanced (ULB)* request-redirection policy distributes the redirected content requests uniformly among all the peering CDNs.
- *Minimum Load Balanced (MLB)* request-redirection policy assigns the redirected content requests to the peer with minimum expected waiting time.
- *Probabilistic Load Balanced (PLB)* request-redirection policy distributes redirected content requests to the peers according to certain probability. This probability depends on the load threshold for the fraction of redirected requests that a peer can accept from the primary. In our case, considering a peering system of three CDNs, we use a probability of *0.4* and *0.6* to assign a certain fraction of the redirected requests to peer 1 and peer 2 respectively. We measure this probability based on the acceptance threshold and the service capacity of the peers.
- *Weighted Load Balanced (WLB)* request-redirection policy assigns *80%* of redirected content requests to the peer with minimum expected waiting time. The remaining *20%* of traffic is uniformly distributed over all other participating peers.

The amount of redirected requests is denoted as the *redirection ratio*, which is quantified in percent of the primary CDN's load. We express the redirection ratio as a fraction of the primary CDN's load. The influence of different primary CDN loads on the redirection ratio to the peers is shown in Figure 8. From the figure, we can see that the redirection ratio (in percentage, %) increases with increase in the primary CDN's load, independent of any particular request-redirection policy.

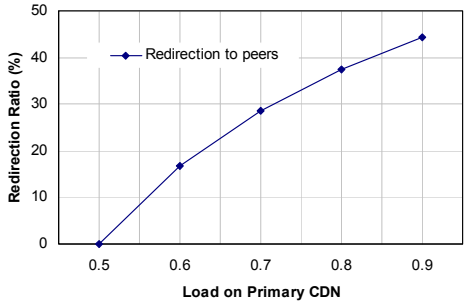


Figure 8: Redirected ratio at different primary CDN loads

The redirected requests are distributed (by the dispatcher) to the peers according to a particular request-redirection policy. In Figure 9, the redirection ratios assigned to the peers are shown. Each curve denotes a different request-redirection policy and x-axis denotes the load on the primary CDN. In ULB request-redirection policy, both peer 1 and peer 2 receive the same amount of redirected requests from the primary CDN. The use of MLB request-redirection policy by the dispatcher assigns all the redirected requests to peer 2, which has the minimum expected waiting time. Hence, no redirected request is assigned to peer 1. If PLB request-redirection policy is used by the dispatcher, it leads to a distribution of *40%-60%* of the redirected requests to peer 1 and peer 2 respectively. A dispatcher following the WLB request-redirection policy assigns *80%* of redirected requests to peer 2 (with minimum expected waiting time) and rest *20%* to peer 1.

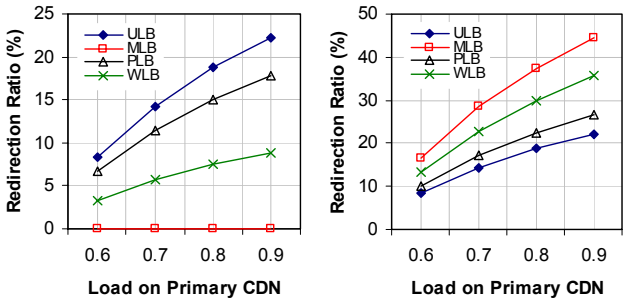


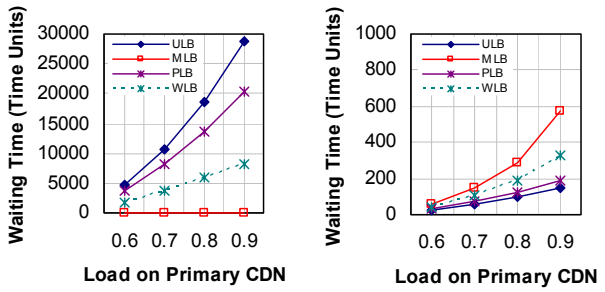
Figure 9: Distribution of redirected requests to the peers for different request-redirection policies and primary CDN loads

5.3 Impact of request redirection

Next, we study the impact of request-redirection on the expected waiting time of users on the primary CDN for different request-redirection policies. Without request-redirection when the primary CDN's load approaches to *1.0*, the user perceived performance (in terms of waiting time and queue length) for service by the primary CDN tends to infinity. On the other hand, with request-redirection the waiting time of the primary CDN decreases as the requests are redirected to the peers. However, request-redirection may lead to temporary overload on certain peer(s).

Figure 10 shows the expected waiting time experienced by the redirected requests on the peering CDNs for different request-

redirection policies. From the figure it can be seen that as more requests are redirected to the peers they realize higher waiting time due to the peers' own load.



(a) Peer 1 (b) Peer 2

Figure 10: Expected waiting time of redirected requests on peers for different redirection policies

Typically a burst of redirected requests improves performance on the primary CDN. In Figure 11, we present this evidence by showing the performance improvement (in terms of waiting time) the primary CDN gains for all the request-redirection policies. Here, we compare the expected waiting time as a function of system load under the four request-redirection policies, by considering lightly loaded peers (load of peer 1 and peer 2 are set to $\rho = 0.5$ and $\rho = 0.4$ respectively), while tuning the primary CDN's load ($0.1 \leq \rho \leq 0.9$). It can be noted that a weighted average value of waiting time is presented in order to capture the effect of request-redirection.

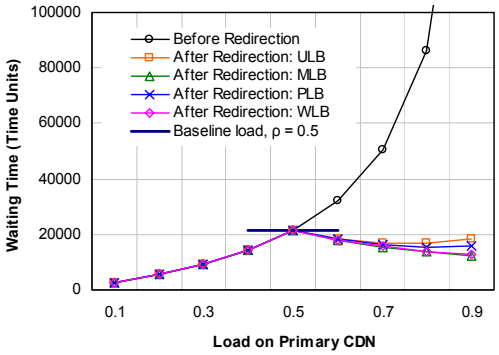


Figure 11: Impact of request-redirection on waiting time of the primary CDN for different request-redirection policies

Table 2 summarizes the reduction on the expected waiting time for the primary CDN in peering for different request-redirection policies. For all the four policies, it is also observed that substantial performance improvement is achieved on the expected waiting time when compared to the non-peering case. Among all the four policies, ULB, PLB and WLB effectively have *redirection ratio* as the common performance parameter. For all these three policies, redirected requests are distributed among peers according to certain percentage. Therefore, to some extent they exhibit similar characteristics. Whereas MLB assigns all redirected requests to a single peer. Though results for MLB may show as good performance as the other three policies (due to light load on peers), there is a possible concern for the peer with minimum expected waiting time to become overloaded with the redirected requests (herd effect [25]). Therefore, it is preferable to spread the load of redirected requests among multiple CDNs rather than assigning all redirected requests to a single peer.

Table 2: Reduction on waiting time for the primary CDN under different request-redirection policies

Load on primary CDN	Reduction in waiting time %			
	ULB	**MLB**	**PLB**	**WLB**
Fair load, $\rho = 0.6$	*43.20%*	*44.41%*	*43.66%*	*44.24%*
Moderate load, $\rho = 0.7$	*66.31%*	*69.31%*	*67.50%*	*68.91%*
Heavy load, $\rho = 0.9$	*90.52%*	*93.70%*	*91.94%*	*93.39%*

From the results, it is clear that all the request-redirection policies guarantee that the maximum waiting time is below *20000* time units. This confirms that redirecting only a certain fraction of requests reduces instability and overload in the system because the peers are not overwhelmed by bursts of additional requests.

5.4 Measurement errors

The dispatcher bases its redirection decision on the measured value of the primary CDN's load. So far we have assumed that perfect information is available for this decision. However, the dispatcher can have inaccurate information about the load on the primary CDN e.g. due to delays in receiving the measurements. Therefore, the impact of measurement errors on the effectiveness of the redirection policies can be measured. Let us denote the measured load of the primary CDN at the dispatcher by $\tilde{\rho} = \tilde{\lambda}E[X]$, with $\tilde{\lambda} = \lambda(1 \pm \varepsilon)$, where ε is the percentage of the correct load ρ.

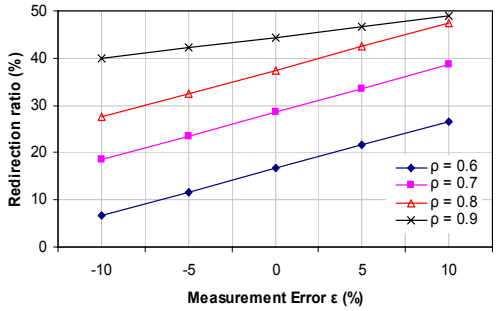


Figure 12: Impact of measurement errors on request-redirection ratio at different primary CDN load

The effects of measurement error in primary CDN's load on the redirection ratio are shown in Figure 12. Each line in the figure denotes a different primary CDN load ρ, and the x-axis denotes the measurement error ε, in percent of ρ. A value of *0* on the x-axis corresponds to perfect primary CDN load information. From the figure we see that the redirection ratio changes more for positive ε than for the corresponding negative ε.

Figure 13 shows the impact of primary CDN's load measurement error on the waiting time for different request-redirection policies. It can be observed that in all the four cases, for measurement error $\varepsilon > 0$, the dispatcher assumes the primary CDN's load to be higher than what it is and hence it redirects more requests than the actual load. The extra redirections incur additional waiting time for the user requests and hence it increases linearly. For negative ε, the dispatcher assumes the primary CDN's load to be less than the actual and hence redirects pessimistically. As a result, requests on the primary CDN experience greater expected waiting time for being processed. Thus, it can be concluded that greater accuracy is needed in load measurement of the primary CDN.

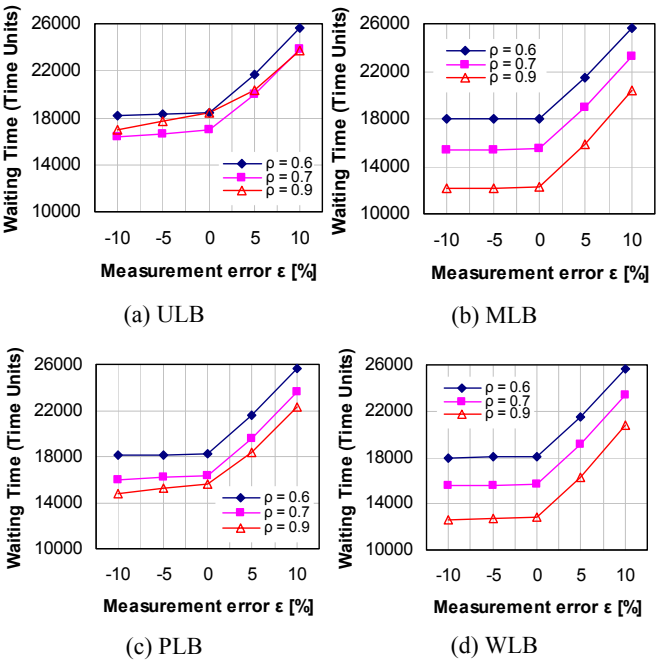


Figure 13: Waiting time for load measurement errors ε for different request-redirection policies

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed an innovative approach to model the peering CDNs. Through the presented performance models we demonstrated the effects of peering and predicted end-user perceived performance from a primary CDN. We outlined a measurement-based methodology which endeavors to assist in making concrete QoS guarantees by a CDN provider. Our approach assists an overloaded CDN to immediately stabilized by offloaded a fraction of the incoming content requests to the peers.

Our model provides a foundation for performing effective peering between CDNs though achieving target QoS in service delivery to end-users. Since the peering CDNs retain load-balancing control within their own Web server sets, using our approach a primary CDN can realize the QoS performance it can provide to the end-users, without requiring individual partners to provide expected service performance from it. Our model-based approach is important since having each CDN provider communicate how it would service millions of potential end-users would introduce significant scalability issues, and requesting this information from each partnering provider at the user requests time would introduce substantial delays. Thus, we believe that our approach seeks to achieve scalability for a CDN in a user transparent manner.

Our future work includes performing an advanced system analysis to study the impact of other performance parameters such as network latency and cost of peering. It also includes developing a proof-of-the-concept implementation for demonstrating the real-time application of our approach for peering between CDNs.

We expect that our methodology for modeling peering CDNs and predicting performance of a CDN provider in a peering arrangement will be a timely contribution to the content networking trend in the infrastructure-based CDNs domain. For more information about our efforts on peering CDNs, please visit the project Web site at www.gridbus.org/cdn.

REFERENCES

[1] Pathan, A. M. K, Broberg, J., Bubendorfer, K., Kim, K. H., and Buyya, R. An Architecture for Virtual Organization (VO)-Based Effective Peering of Content Delivery Networks, UPGRADE-CN'07, In Proc. of the *16th IEEE International Symposium on High Performance Distributed Computing (HPDC)*, Monterey, California, USA, Jun. 2007.

[2] Pathan, A. M. K. and Buyya, R. Economy-Based Content Replication for Peering CDNs. TCSC Doctoral Symposium, In *Proc. of the 7th IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007)*, Brazil, May 2007.

[3] Buyya, R., Pathan, A. M. K., Broberg, J., and Tari, Z. A Case for Peering of Content Delivery Networks. *IEEE Distributed Systems Online*, 7(10), USA, Oct. 2006.

[4] Pathan, A. M. K. and Buyya., R. A Taxonomy and Survey of CDNs, Technical Report, GRIDS-TR-2007-4, The University of Melbourne, Australia, Feb. 2007.

[5] Crovella, M. E., and Bestavros, A. Self-similarity in World Wide Web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6), pp. 835-846, 1997.

[6] Crovella, M. E., Taqqu, M. S., and Bestavros, A. Heavy-Tailed Probability Distributions in the World Wide Web. *A Practical Guide To Heavy Tails*, Birkhauser Boston Inc., Cambridge, MA, USA, pp. 3-26, 1998.

[7] Cobham, A. Priority Assignment in Waiting Line Problems. *Journal of the Operations Research Society of America*, Vol. 2, No. 1, pp. 70-76, 1954.

[8] Conway, R. W., Maxwell, W. L., and Miller, L. W. *Theory of Scheduling*, Addison-Wesley (Reading, Mass.), 1967.

[9] Broberg, J., Zeephongsekul., P., and Tari, Z. Approximating Bounded General Service Distributions, In Proc of IEEE Symposium on Computers and Communications, Aveiro, Protugal, Jul. 2007.

[10] Bouman, J., Trienekens, J., and Zwan, M. Specification of Service Level Agreements, Clarifying Concepts on the Basis of Practical Research. In *Proc. of the Software Technology and Engineering Practice Conference*, pp. 169, 1999.

[11] Kleinrock, L. Queuing Systems. Vol. II: Computer Applications. John Wiley & Sons, pp. 15-19, 1975.

[12] Padmanabhan, V. N. and Sripanidkulchai, K. The Case for Cooperative Networking. In *Proc. of International Peer-To-Peer Workshop (IPTPS02)*, 2002.

[13] Day, M., Cain, B., Tomlinson, G., and Rzewski, P. A Model for Content Internetworking. IETF RFC 3466, Feb. 2003.

[14] Turrini, E. An Architecture for Content Distribution Internetworking. Technical Report UBLCS-2004-2, University of Bologna, Italy, March 2004.

[15] Amini, L., Shaikh, A., and Schulzrinne, H. Effective Peering for Multi-Provider Content Delivery Services. In *Proc. of 23rd Annual IEEE Conference on Computer Communications (INFOCOM'04)*, pp. 850-861, 2004.

[16] Biliris, A., Cranor, C., Douglis, F., Rabinovich, M., Sibal, S., Spatscheck, O., and Sturm, W. CDN Brokering. *Computer Communications*, 25(4), pp. 393-402, 2002.

[17] Amini, L., Shaikh, A., and Schulzrinne, H. Modeling Redirection in Geographically Diverse Server Sets. In *Proc. of the 12th International Conference on World Wide Web (WWW)*, ACM Press New York, NY, USA, pp. 472-481, 2003.

[18] Ranjan, S., Karter, R., and Knightly, E. Wide Area Redirection of Dynamic Content by Internet Data Centers. In *Proc. of 23rd Annual IEEE Conference on Computer Communications (INFOCOM'04)*, 2004.

[19] Cardellini, V. and Colajanni, M. and Yu, P.S., Geographic Load Balancing for Scalable Distributed Web Systems. In *Proc. of the International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2000.

[20] Freedman, M. J., Freudenthal, E., and Mazières, D. Democratizing Content Publication with Coral. In *Proc. of 1st Symposium on Networked Systems Design and Implementation*, San Francisco, CA, pp. 239-252, Mar. 2004.

[21] Wang, L., Park, K. S., Pang, R., Pai, V. S., and Peterson, L. Reliability and Securrity in the CoDeeN Content Distribution Network. In *Proc. of Usenix Annual Technical Conference*, Boston, MA, Jun. 2004.

[22] Pierre, G. and Steen, M. van. Globule: A Platform for Self-Replicating Web Documents. In *Proc. of the 6th International Conference on Protocols for Multimedia Systems (PROMS'01)*, The Netherlands, pp. 1-11, 2001.

[23] Zhao, W. and Schulzrinne, H. DotSlash: A Self-configuring and Scalable Rescue System for Handling Web Hotspots Effectively. In *Proc. of the International Workshop on Web Caching and Content Distribution (WCW)*, Beijing, China, pp. 1-18, 2004.

[24] Papazoglou, M. P. and Georgakopoulos, D. Service-Oriented Computing. *Communications of the ACM*, 46(10), ACM Press New York, NY, USA, pp. 25-28, 2003.

[25] Dahlin. M. Interpreting stale load information. *IEEE Transactions on Parallel and Distributed Systems*, 11(10), pp. 1033–1047, Oct. 2000.