

DEPARTMENT OF ELECTRICAL ENGINEERING
TECHNICAL REPORT

A QUEUING ANALYSIS OF SERVER SHARING COLLECTIVES FOR
CONTENT DISTRIBUTION

Daniel Villela and Dan Rubenstein

CU/EE/TR 2002-04-121

Columbia University
Department of Electrical Engineering
500 W. 120th Street, MC 4712
New York, NY 10027

<http://www.ee.columbia.edu>

Copyright © 2002 – The Trustees of Columbia University in the City of New York
All Rights Reserved

The technical reports in this series are considered to be semi-formal.
The ideas expressed are solely those of the authors, and questions about the content
should be directed to them.

A Queuing Analysis of Server Sharing Collectives for Content Distribution*

Daniel Villela Dan Rubenstein
Dept. of Electrical Engineering
Columbia University
500W 120th St.
New York NY 10027
{dvillela,danr}@ee.columbia.edu

Columbia University Technical Report EE200412-1
April 2002 [†]

Abstract

A content provider must choose how to provision its server resources to handle client demand for its content. When provisioning is based on typical demand, clients are turned away during peak demand periods. Provisioning based on peak demand or contracting a third party content distribution network service increases costs. An alternative approach is for the various content providers to form a collective in which their server resources are pooled together, servicing the demands of the entire collective. Here we compare the performance of server sharing collectives to content providers who operate their servers in isolation. We demonstrate using analysis and simulations upon fundamental queueing models that by hosting one another's content within a collective, content distributors can reduce the rate at which client requests are turned away. We also analyze a class of thresholding techniques that content providers within the collective can apply to limit the use of their own server resources by other collective members. We show that these thresholding techniques permit a content provider to assist other providers within the collective without sharing its resources to a point where its ability to service its own clients is unsatisfactory.

1 Introduction

Content providers profit by servicing client requests, but must pay for the set of resources (i.e., servers) that provide the service. Profits are therefore maximized by carefully configuring the system of servers to closely meet client demand for the content objects it hosts. However, selecting the best configuration is often difficult because of fluctuations in an object's popularity over time. For example, there is empirical evidence of server workloads of online course offerings whose request rates vary sporadically with time [1].

In such scenarios, the *content provider* (a.k.a., the *provider*) can provision the system to handle the expected average demand. This keeps equipment costs low and provides acceptable service most of the time. However, during periods where the content reaches its peak in popularity, the serving system will be forced to turn away a large number of requests, reducing potential revenues. The provider can pay a third

*SERE, Function: noun, Etymology: Latin series, Date: 1916 : a series of ecological communities formed in ecological succession.

[†]This material was supported in part by the National Science Foundation under Grant No. ANI-0117738 and CAREER Award No. 0133829. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Daniel Villela is a CNPq-Brazil Scholar (Ref. Number 200168/98-3).

party content distribution network (CDN) to host its content during these peak periods or it can overprovision its own set of resources to handle these peak periods. Either way, the provider pays for the extra capacity.

Here, we consider another alternative that we call SERES: providers form collectives and assist one other during periods of overload by hosting each other's content. In SERES, each provider's bandwidth and storage resources are used to host the content of the entire collective. Under this paradigm, a provider's server system can be provisioned to service the expected average demand while additional demands during peak periods are serviced by redirecting requests to servers owned by other members of the collective. It follows intuitively and from the law of large numbers that if all providers provision according to their average loads, then with very high probability, a large collective as a whole will be able to absorb the peak periods of high demands imposed upon the individual providers.

The SERES framework differs from frameworks analyzed previously that leverage off the gains achieved by pooling together resources (e.g., to prevent overloading or to enable load balancing) in that in SERES, a provider only profits when one of its own clients is serviced by the SERES system. As in previous models, a provider "wins" by participating because the additional resources in the system can be used to process its clients' requests. However, unlike previous models, the provider also "loses" because its own resources must also be available to process the requests of other providers' clients. Such processing drains its own available resources without directly contributing any profit.

We perform an analytical examination of the potential benefits and drawbacks that a service provider receives by participating within a SERES collective. Our examination assumes that the limiting resource of a server is its processing power or the bandwidth on the access link to the server. Under this assumption, the geographical positioning of the servers relative to one another as well as to the client does not affect performance. Practically speaking, one can consider this scenario to reflect a network whose core forwarding resources are more than adequately provisioned. We leave consideration of the geographical positioning of clients and servers as future work.

Within this framework, we consider two general classes of traffic. The first class consists of a set of transfers that must be transmitted at a *fixed rate*, such as streamed video. Such transfers require a fixed amount of the server's processing resource to perform the transfer. The second class consists of a set of transfers that are *elastic* and "share" the available processing equally: a server hosting n simultaneous transfers can transmit each session at $1/n$ th the rate that would occur when the server transmits a single session. We model the servers as queueing systems: for fixed-rate transfers, the queueing system is first-come-first-serve (FCFS), for elastic transfers the queueing system is modeled using a processor sharing (PS) discipline where each server bounds the maximum number of requests it will service simultaneously to lower bound the minimum rate of service.

We measure performance of systems servicing fixed-rate transfers as a function of the *blocking probability*: the rate at which requests for transfers are rejected because the entire system's processing resources are in use. Performance of systems servicing elastic traffic is measured as a function of both blocking probability and completion time of a transfer from the time of the request. We evaluate the performance of the SERES system by fixing the arrival process and service configuration (buffer size, processing rates) of each provider's servicing system, and comparing the blocking probabilities and completion times (when applicable) of a provider's system participating in SERES to a similarly configured system that does not participate in SERES.

Our analysis shows that SERES can reduce blocking probabilities and completion times of all customer requests to all participants of the SERES system by several orders of magnitude. Hence, SERES can offer a win-win situation to all participants. Our analysis upon $M/G/k/k$ systems shows this to be the case when each provider participating within SERES contributes the same intensity (i.e., $\rho = \lambda/\mu$, ratio of client request rate to the overall processing rate) and places the same bound on the number of requests that can be serviced simultaneously (i.e., k in an $M/G/k/k$ system). We also consider the blocking probability of the SERES system that services fixed-rate transfers as the number of participating providers grows large.

The interest for this study comes from the fact that when $\rho/k < 1$, the blocking rate of the system converges to 0 but when $\rho/k > 1$, we see the blocking probability converging to $1 - k/\rho$.

Next, we turn our attention to SERES systems where servers contribute different levels of intensity and has different bounds on the number of requests serviced. We show an obvious result that when a lightly loaded system and a heavily loaded system form a SERES collective, the blocking rate on requests for content offered by the lightly loaded system will increase beyond the rate when it operates in isolation. We demonstrate, however, that SERES exhibits a win-win situation over a wider range of system configurations when supporting fixed-rate transmissions than is exhibited when it supports elastic-rate transfer systems.

To protect server systems from overextending resources in these heterogeneous scenarios without dropping active jobs, we introduce a thresholding technique that can be applied to admission requests. Thresholding allows servers to prioritize admission of their own content over requests for content offered by other SERES participants. Via a combination of analysis upon systems where job sizes are exponentially distributed and simulation using empirically observed distributions for job sizes, we demonstrate that thresholding can improve performance of heavily overloaded systems without significantly degrading the performance of the lightly loaded systems that participate in SERES.

The rest of the paper is structured as follows. In Section 2, we briefly overview related work. In Section 3 we present a model and evaluation of SERES for fixed-rate transfers. We develop and evaluate for elastic file transfers in Section 4 and present our analytical solution for this model. Section 5 evaluates a suite of thresholding techniques. We conclude and elaborate on open issues in Section 6.

2 Related work

Several works have previously analyzed systems that pool together server resources to improve various performance aspects of content delivery. For instance, [2, 3, 4] investigate the practical challenge of maintaining consistency among distributed content replicas. The study in [5] considers investigates the placement of content in the network to minimize delivery latencies. Other studies [6, 7] investigate “load sharing” policies that attempt to keep the processing load on a set of hosts relatively balanced while keeping redirection traffic levels low. The Oceano project [8] provides a set of servers that can be spawned and managed to meet additional server resources for customers. After servers are allocated to customers they are use exclusively by contracted customer without a concurrent system sharing. An analytical study of these types of systems was performed in [9]. An essential difference between these works and ours is that in this previous work, increasing the availability of resources always improves the “performance” of those who make the resources available. In contrast, in SERES systems, those who donate their resources to service other content may in fact be penalized by reduced service levels of their own content. This places additional restrictions on how and when resources can be shared.

Content providers currently rely on third-party content distribution networks (CDNs) to relieve their management of content distribution and to achieve low latency for their services at increased costs. In 2000, a standardization body named Content Alliance [10] was created for producing open standards to foster cooperation between CDNs, with an implementation described in in [11]. This work provides a mechanism that enables providers to redirect content requests and share one another’s resources, but gives no specific guidelines as to how these resources should be shared.

Recent attention has focused on the problem of alleviating rapid and unpredicted spikes in request load, a phenomenon often referred to as a “flash crowd”. Recent proposals in this area [12, 13, 14, 15] approach this problem via a peer-to-peer (P2P) solution, in which clients communicate directly with one another to retrieve the desired content. Here, clients do not have content “of interest” from which they profit. Instead, the effectiveness of these approaches relies on the goodwill of those who participate in order to receive content to also transmit the content to others when requested to do so.

We model elastic file transfers as a queue that utilizes a processor sharing discipline. An early work that investigated this discipline is found in [16]. More recently, Zwart and Boxma [17] studied asymptotic behavior for this queueing discipline with Poisson arrivals. Heavy-tailed distributions are studied by Momcilovic and Jelenkovic [18] and Borst *et al.*[19]. Bansal and Harchol-Balter [20] and Crovella *et al.* [21] demonstrate that expected completion times of jobs in systems that prioritize jobs according to job length are smaller than in systems that employ processor sharing. In its current form, the SERES system analyzed does not queue jobs for processing (if there is no space, they are simply denied service). If queuing is added to a SERES system, it would be interesting to consider how these different scheduling mechanisms further impact delay and blocking probability. Last, our fixed-rate model presented in this paper is solved as a product-form solution. A recent survey of these method appears in [22].

3 SERES for fixed-rate transfers

In this section we present and evaluate a model for a SERES system in which accepted requests receive data at fixed rates (i.e., the server that accepts the request dedicates a fixed amount of processing/bandwidth to handle the request). This model is based on rate-controlled applications such as streamed video or audio transfer and servers.

3.1 Model and Rationale

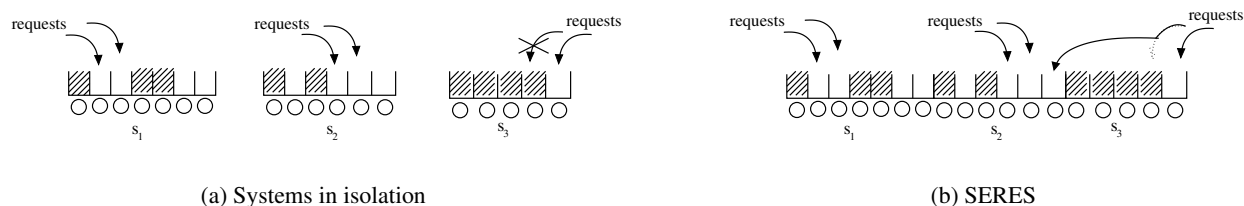


Figure 1: SERES example compared to systems in isolation.

SERES is a paradigm in which a set of providers share their server resources with one another. A provider participating in SERES must store and distribute its own content as well as the content of the other content participating providers. In practice, SERES requires a negotiation phase in which the participants agree to share resources and maintain up-to-date copies of one another’s content. Once content is uploaded to the server collective, client requests for content that cannot be serviced by the originating provider due to overload can be redirected to the other participants of the collective to be serviced by an alternate server. An example of the benefits of a SERES system is depicted in Figure 1. In Figure 1(a) three different providers operate in isolation (i.e., do not host one another’s content). The provider labeled s_3 cannot service all requests and is forced to drop a request. The SERES system involving these three servers is shown in Figure 1(b). Here, s_3 can redirect requests to s_2 , who, at the time of the arrival of the request, has capacity to process the request.

Our investigation in this paper focuses on the performance benefits of the SERES system assuming (a) servers have sufficient disk space to store copies of all content to be distributed by SERES, (b) the redirection mechanism always redirects requests instantaneously to an alternate server with available capacity (when one exists) to process the request, and (c) core network capacity is unbounded, such that either server processing or transmission capacities access links to the network bound the aggregate transmission rates of

each provider. This is a reasonable starting point given that the core of a network is often well-provisioned to handle peak loads and that redirection times are a small overhead for long-lived transfers such as streaming video or large files. There are practical reasons to consider models where these assumptions are relaxed (e.g., short-lived transfers or bottlenecks in the core that result from peering between network providers). We leave the investigation of SERES under these relaxed assumptions as future work.

We develop a SERES system upon an Internet infrastructure that consists of content providers, servers, clients and commodities. *Commodities* are the content/information goods offered by content providers. For instance, multimedia presentations of lectures are the commodities of an online course offering. The *content provider* (also referred to simply as *provider*) is the entity that offers commodities to customers via the Internet. A *client* is a user that requests receipt of a given commodity, and a *server* is the vehicle owned by the provider that interfaces with the network to deliver commodities to clients.

We consider a set of content providers $\mathcal{Y} = \{y_1, y_2, \dots, y_n\}$ that offer a finite set of commodities $\mathcal{V} = \{v_1, v_2, \dots, v_m\}$ through a set of servers, $\mathcal{S} = \{s_1, s_2, s_3, \dots, s_n\}$. The set of commodities owned by a provider is formally expressed as $\phi(y_i) = \{v_j | v_j \text{ is offered as } y_i\text{'s content}\}$; y_i profits from the processing of a request for $v_j \in \phi(y_i)$, regardless of which provider owns the server that accepts and processes the request. We assume $\phi(y_i) \cap \phi(y_j) = \emptyset$ for $i \neq j$, i.e., no two providers profit from the hosting of a particular commodity. We define the set of commodities hosted by server s_i as $\psi(s_i) = \{v_j | s_i \text{ offers } v_j\}$. A *dedicated system*, a.k.a., a *system in isolation* is a system in which a SERES collective is not formed such that each provider hosts only its own content. Thus, if y_i uses $s_{i_1}, s_{i_2}, \dots, s_{i_n}$ to distribute its content then $\phi(y_i) = \cup_{j=1}^n \psi(s_{i_j})$ in a dedicated system. In a SERES system, s_i can host additional commodities on behalf of other content providers, relaxing the relationship between y_i and s_j to $\phi(y_i) \subseteq \cup_{j=1}^k \psi(s_{i_j})$. The notation used throughout this paper is summarized in Appendix A.

Because the geographical position of the servers have no effect on performance in our model, a provider's server system that permits the provider to always direct requests to a server with available capacity (when one exists) can be modeled, without loss of generality, as a single server formed by aggregating the individual servers together. We assume all fixed-rate commodity transfers require the same transfer rate. However, the transfer time (i.e., workload) of each request can be individually described as an i.i.d. random variable drawn from a general distribution (including the case where the r.v. is a constant), and where the choice of commodity requested by a client is i.i.d. Without loss of generality, the blocking probability and transfer times of such a system is equivalent to one that offers a single commodity, where the distribution of the workload of this one commodity is formed by aggregating together the distributions of the various commodities' distributions from the original system in proportion to the frequency with which each commodity is chosen. Thus, without loss of generality, the rest of the paper assumes that each provider y_i 's system consists of a single server, s_i offering a single commodity, v_i .

We model each server that transfers data at a fixed rate as an $M/G/k/k$ queueing system: Server s_i has k_i slots and k_i processors where each slot can be used to actively transmit a single commodity to a single client. We assume for all commodities $v_i \in \mathcal{V}$ that request arrivals are described by a Poisson process with rate λ_i with each request serviced immediately if there is an available slot. Otherwise, the request is forwarded to an available server (when one exists) in the SERES system and is otherwise dropped (and must be dropped when the server operates in isolation). The service time for commodity v_i is an i.i.d. random variable B_i with mean $E[B_i]$ and general distribution. Each accepted request is assigned to an available processor. Note that since each entry in the system is actively being serviced, the service rate of the queueing system is proportional to the number of busy processors.¹

We define $p_i(\{y_1, y_2, \dots, y_n\})$ to be the blocking probability that a request for commodity v_i is denied entry to a SERES system composed of the set of content providers $\{y_1, y_2, \dots, y_n\}$ (employing servers s_1, \dots, s_n). The dedicated system's blocking probability is given by $p_i(\{y_i\})$. For a dedicated system,

¹When the context is clear, e.g., for i.i.d. random variables, we drop a variable subscript for sake of simplicity.

the Erlang loss formula (also known as Erlang B formula) applies directly [23]:²

$$p(\{y_1\}) = \frac{\rho_1^{k_1}/k_1!}{\sum_{j=0}^{k_1} (\rho_1)^j/j!}. \quad (1)$$

where $\rho_1 = \lambda_1 E[B_1]$.

We extend this formula to the blocking probability of a two-server SERES system $p(\{y_1, y_2\})$. Since such a loss system is a symmetric queue [24], the stationary distribution for each state $P(N_1 = i, N_2 = j)$, where $i(j)$ is the number of commodities of type v_1 (v_2) actively being processing, can be expressed in product-form: $P(N_1 = i, N_2 = j) = \pi_i \pi_j c_{\{y_1, y_2\}}$, where $\pi_i = \rho_1^i/i!$, $\pi_j = \rho_2^j/j!$, and $c_{\{y_1, y_2\}}$ is a normalizing constant such that $\sum_{i \geq 0, j \geq 0, j+i \leq k_1+k_2} \pi_i \pi_j = 1$. Hence we find the blocking probability of a two-server SERES system to be

$$\begin{aligned} p(\{y_1, y_2\}) = P(X_1 + X_2 = k_1 + k_2) &= \sum_{i=0}^n P(X_1 = k_1 + k_2 - i, X_2 = i) \\ &= \sum_{i=0}^n \frac{1}{i!} \rho_1^i \frac{1}{(k_1 + k_2 - i)!} \rho_2^{k_1+k_2-i} c_{\{y_1, y_2\}} \\ &= (\rho_1 + \rho_2)^{k_1+k_2} \frac{1}{(k_1 + k_2)!} c_{\{y_1, y_2\}}, \end{aligned} \quad (2)$$

where $\rho_i = \lambda_i E[B_i]$. We refer to ρ_i as the *provider intensity* of provider i . Noting that (2) is in the form of (1) with ρ_1 replaced by $\rho_1 + \rho_2$ and k_1 replaced by $k_1 + k_2$ we can repeat this process recursively to compute the blocking probability for a set of n servers cooperating within a SERES system.

$$p(\{y_1, y_2, \dots, y_n\}) = \frac{1}{(\sum_j^n k_j)!} \left(\sum_j^n \rho_j \right)^{\sum_j^n k_j} c_{\{y_1, y_2, \dots, y_n\}}. \quad (3)$$

where $c_{\{y_1, y_2, \dots, y_n\}}$ is once more the normalizing constant for this distribution such that

$$\sum_{i_1 \geq 0, i_2 \geq 0, \dots, \sum_{j=1}^n i_j < \sum_{j=1}^n k_j} \pi_{i_1, i_2, \dots, i_n} = 1$$

Note that if $\forall i, j E[B_i] = E[B_j]$ and $\lambda_i = \lambda_j$, the SERES system can be viewed as a larger server with the same intensity per processor, but with $\sum_{i=1}^n k_i$ processors for more effective load balancing.

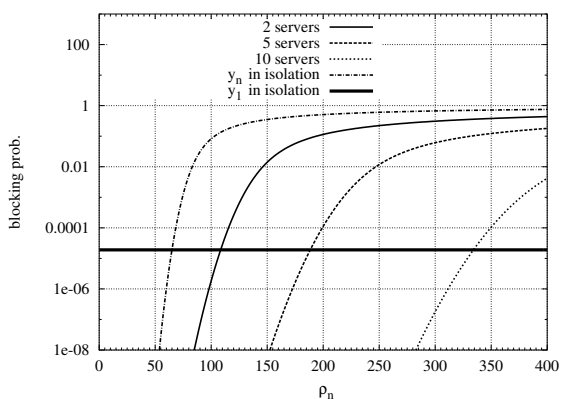
3.1.1 Criteria for SERES participation

We assume that a provider will be willing to participate in a SERES system as long as there is no strong incentive to operate in isolation. More formally, a provider is *favorable* if either of the following two properties hold:

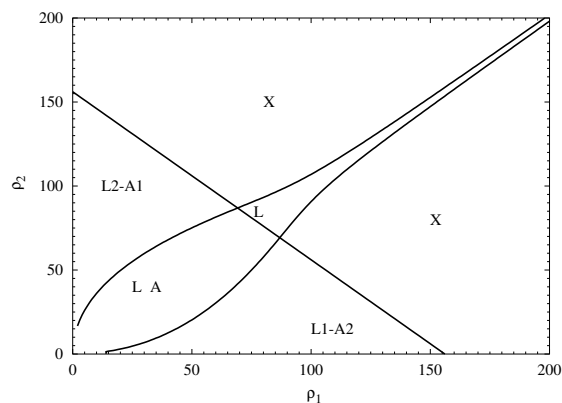
- The blocking probability of a provider's commodity is decreased when the provider participates in SERES.
- The blocking probability of a provider's commodity increases by participating in the SERES system but lies beneath a *tolerance limit*, l_i . For instance, if $l_i = 10^{-5}$ for provider s_i , s_i will contribute its resources to SERES so long as the SERES system keeps the blocking rate for commodity v_i below 10^{-5} .

²Note the fact that the formula can be applied to these systems where the processing time is described by a general distribution is due to the model assuming that jobs are not queued, i.e., they are accepted immediately or else turned away.

3.2 Evaluation of a Simple Example Scenario



(a) Blocking probability in SERES scenario compared to servers in isolation.



(b) Areas of comparison between two-server SERES and dedicated systems.

Figure 2: Evaluation of a SERES system under the processor sharing model.

We begin by evaluating SERES as a function of number of providers participating and the provider intensities. Figure 2(a) plots blocking probabilities of n -provider systems, $n = 2, 5, 10$, where each server can simultaneously service 100 commodities ($k_i = 100, i = 1, 2$).³ The results are a direct application of equation (2).

The value of $\rho_i = 65$ is chosen such that requests for v_i exhibit a blocking probability of approximately 10^{-5} when y_i operates in isolation, $1 \leq i < n$. On the x -axis, we vary ρ_n , the provider intensity for provider y_n . The y -axis plots blocking probabilities for various system configurations. The curve labeled “ y_n in isolation” depicts the blocking probability of requests for commodities v_n within the dedicated system for y_n . The constant line at $p(\{y_1\}) \approx 10^{-5}$, labeled “ y_1 in isolation”, plots the blocking probability of requests for commodity v_1 for provider y_1 when y_1 operates in isolation (results for providers y_2, \dots, y_{n-1} are identical). The remaining curves labeled “ n servers”, $n = 2, 5, 10$, depict the blocking probability for all providers that participate in an n -provider SERES system. From the figure, we observe that:

- For the shown range of values of ρ_n considered, the blocking rate that occurs in the SERES system for commodity v_n is smaller than its blocking rate in isolation. Therefore, regardless of its own provider intensity, a provider is willing to participate in SERES when the other providers have low provider intensities.
- A range for ρ_n exists, $0 \leq \rho_n \leq 110, 175, 330$ for $n = 2, 5, 10$, respectively, where the blocking rate in the SERES system for commodity $v_i, i < n$ is smaller than the blocking rate when s_i operates in isolation. Hence, multiplexing across servers in SERES benefits provider y_i even when all other providers in the SERES system have larger provider intensities.

³The value of a maximum of 100 is intended to be a modest but reasonable value for a provider. For instance, a streaming server might have limitations depending on the quality of the sessions it carries. Another reason is the fact that commercial licenses are often limited. Today RealNetworks [25], one of the major developer of streaming media technologies, offers its basic streaming server (free of charge) with maximum capacity of 25 concurrent clients. Their \$1,995-dollar license server (RealServer Plus) has maximum capacity of 60 concurrent sessions.

- Beyond the range described above, requests for commodity $v_i, i < n$ are dropped with greater probability in the SERES system. However, as long as $\rho_n < 120, 200, 350$, for $n = 2, 5, 10$, respectively, the blocking probability in the shared system remains small, i.e., below the threshold $l_n = 10^{-4}$. Hence, content provider y_i is favorable.
- When ρ_n assumes higher values, the dropping probability for commodities becomes excessive, and y_i would prefer to withdraw its participation from the SERES system.
- For the system to achieve a particular blocking probability, the value of ρ increases significantly with increases in n .

Figure 2(b) graphically demonstrates the “areas of SERES willingness”. In this figure we consider a system with two providers, y_1 and y_2 where $k_i = 100, i = 1, 2$. We vary ρ_1 and ρ_2 on the x - and y -axis, respectively. The diagonal line that separates the bottom-left portion of the graph from the top right indicates values of ρ_1 and ρ_2 where the SERES system drops requests at a rate of 10^{-4} . Below this line, the SERES system offers a lower aggregate intensity, resulting in a drop rate below 10^{-4} . The top curve that goes from the bottom-left to the top-right of the graph is formed from the set of values of (ρ_1, ρ_2) where content provider y_1 experiences the same blocking probability regardless of whether it operates in isolation or participates within the SERES system, i.e., $p(\{y_1\}) = p(\{y_1, y_2\})$. Above this curve, $p(\{y_1\}) < p(\{y_1, y_2\})$: blocking probability is reduced for y_1 by operating in isolation, and below, $p(\{y_1\}) > p(\{y_1, y_2\})$. Similarly, the lower curve that runs from the bottom left to the top right is formed from the points where $p(\{y_2\}) = p(\{y_1, y_2\})$. Below this curve, $p(\{y_2\}) < p(\{y_1, y_2\})$, and above, $p(\{y_2\}) > p(\{y_1, y_2\})$.

Each area between the various curves is labeled in Figure 2(b) to indicate the “willingness” of y_1 and y_2 to participate in SERES. A label of ‘L1’ indicates an area in which blocking probability for y_1 is smaller in SERES than in isolation, i.e., $p(\{y_1, y_2\}) < p(\{y_i\}), i = 1, 2$. An ‘A1’ label indicates an area where $p(\{y_1, y_2\}) < l_1$. A similar labeling applies to mark the various areas for provider y_2 . When both providers mark an area similarly, the pair of labels ‘A1’, ‘A2’ are replaced by ‘A’, and ‘L1’, ‘L2’ are replaced by ‘L’. ‘X’ marks the areas where a content provider is not favorable. We see that there is a significant region in which both providers can benefit simultaneously by participating in SERES. In addition, we note that there is no area in which the blocking probabilities of both content providers is higher in the shared system than in the dedicated system, i.e., at least one provider is willing to participate in SERES in any area marked by an ‘X’.

3.3 Asymptotic limits of SERES: The ρ/k factor

We turn our attention to examining the performance of an n -server SERES system as the number of servers n tends to ∞ . We assume that there are n_c different classes of providers, where all provider systems in the same class exhibit the same provider intensity and have the same bound, k_i , on the number of jobs that can simultaneously be serviced. We let f_i represent the fraction of providers in class $i, 1 \leq i \leq n_c$. Equation 3 can be reformulated as

$$p(\{y_1, y_2, \dots, y_n\}) = \frac{(\sum_{j=1}^{n_c} (n f_j \rho_j))^{\sum_{j=1}^{n_c} n k_j f_j}}{(\sum_{j=1}^{n_c} n k_j f_j)!} \sum_{i=0}^{\sum_{j=1}^{n_c} n f_j k_j} \frac{(\sum_{j=1}^{n_c} (n f_j \rho_j))^i}{i!} \quad (4)$$

Equation 4 can be simplified via the constants $\hat{\rho} = \sum_{j=1}^{n_c} f_j \rho_j$ and $\hat{k} = \sum_{j=1}^{n_c} f_j k_j$:

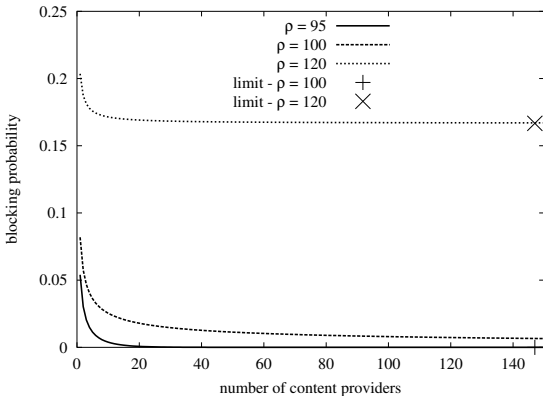
$$p(\{y_1, y_2, \dots, y_n\}) = \frac{(n \hat{\rho})^{n \hat{k}} / (n \hat{k})!}{\sum_{j=0}^{n \hat{k}} (n \hat{\rho})^j / j!} \quad (5)$$

The asymptotic behavior of loss systems such that the ratio of workload by the total capacity is finite when the number of servers is infinite is known [26]:

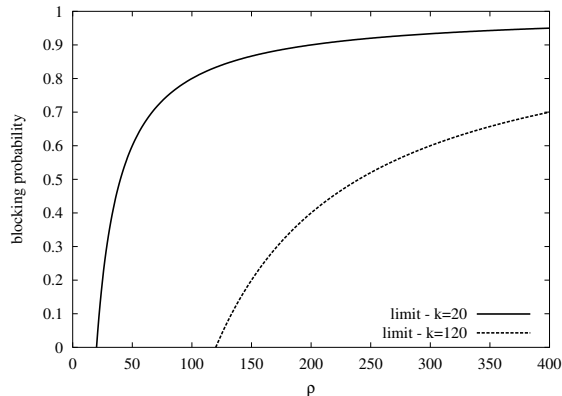
$$\lim_{n \rightarrow \infty} p(\{y_1, y_2, \dots, y_n\}) = \begin{cases} 0, & \text{if } \hat{\rho}/\hat{k} \leq 1 \\ 1 - \hat{k}/\hat{\rho}, & \text{if } \hat{\rho}/\hat{k} > 1 \end{cases} \quad (6)$$

Figure 3 illustrates the behavior of an n -server SERES system as n grows large. For simplicity, we show only the case where only one class of servers exists, i.e., $n_c = 1$ in which each provider's system can simultaneously handle $k = 100$ transmissions. In Figure 3(a), we vary the number of participating providers along the x -axis, plotting the blocking probability along the y -axis, where each curve depicts a system where a given provider intensity, ρ , is exhibited by all providers in the system. The asymptotic limits observed here fit the claims of Equation 6. In particular, for $\rho < k$, the blocking probability converges to 0, whereas for $\rho > k$, we observe the blocking probability converging to the limit given by Equation 6. The figure demonstrates (for a homogeneous collection of providers) that the most dramatic benefits from SERES result from clustering small numbers of providers together, and that incorporating additional providers further reduces blocking probabilities, but at a quickly diminishing rate.

In Figure 3(b) we plot the asymptotic blocking probability of a SERES system as a function of ρ . The curve on the left is the asymptotic blocking probability for the case where $k = 20$. The curve on the right is for the case where $k = 120$. We notice for $k = 120$ a slower increase in the asymptotic blocking probability.



(a) Blocking probability increasing the number of providers.



(b) Asymptotic blocking probability for $n \rightarrow \infty$.

Figure 3: Asymptotic observations on blocking probability for fixed-rate transfer SERES systems.

4 SERES for Elastic Transfers

4.1 Model and rationale

In this section we present and evaluate a model for a SERES system in which accepted requests receive data at rate that is proportional to $1/m$, where m is the aggregate number of customers currently being serviced across all servers that comprise the SERES system. Such a model fits an environment in which load is equally balanced across servers participating in the SERES system. This can be accomplished in

practice, for instance, with parallel downloading technology [27]. Such technology is commonplace today in peer-to-peer downloading tools such as Morpheus.

We again consider a model in which client arrivals to each provider are described by a Poisson process and the load imposed by each commodity is described by a general distribution. Using the same argument as before, we can collapse the model to the case where each provider offers a single commodity and uses a single server. We apply a processor sharing (PS) model to capture the behavior of the transmission rate as a function of the number of requests currently being serviced. We assume that each server s_i bounds the minimum rate at which it will transmit data to clients by bounding the number of clients accepted by a constant, k_i , and will turn away (or redirect) any additional clients requesting service.⁴ Effectively, this is an $M/G/1/k_i/PS$ queue. As before, we assume that jobs are never queued when service is unavailable, but are simply turned away (dropped). In addition to blocking probability as a metric we also measure job completion time.

The blocking probability of sessions under this model is obtained and evaluated with numerical results in the next session. Whereas for an unbounded system the distribution of the queue under a processor sharing discipline is shown to be geometric [23], in our case we find the following law (derivation is shown in Appendix B):

$$p(\{y_1, y_2, \dots, y_n\}) = (\lambda E[B])^k \frac{1 - \lambda E[B]}{1 - (\lambda E[B])^{k+1}} \quad (7)$$

where $\lambda = \sum_{j=1}^n \lambda_j$ (where λ_j is the request rate for commodity v_j), $k = \sum_{j=1}^n k_j$, and $E[B]$ is the mean size of the request to the SERES system. The amount of work for each individual job $B(\{y_1, y_2, \dots, y_n\})$ introduced in SERES is related to the amount of work B introduced in isolation for $\{y_1\}$ through the expression $B(\{y_1, y_2, \dots\}) = k_1 B / (\sum_{j=1}^n k_j)$.

Assuming $\lambda_i = \lambda_j = \lambda$, $k_i = k_j$, $\forall i, j$, it is easy to show that, for $\rho = \lambda E[B]$, when the number of content providers n goes to ∞ , the blocking probability tends to

$$\lim_{n \rightarrow \infty} p(\{y_1, y_2, \dots, y_n\}) = 1 - \frac{1}{\rho}. \quad (8)$$

The expected value of the delay in the system is obtained using Little's Law [28]. As a fraction of customers enters the system we need to consider $\lambda_f = (1 - p(\{y_1\}))\lambda$. Therefore,

$$d(\{y_1\}) = E[D] = E[N] / \lambda_f = \frac{E[N]}{(1 - p_v)\lambda}, \quad (9)$$

where $E[N] = \sum_{n=0}^k n P(N = n)$.

4.2 Numerical evaluation

Here we study conditions under which providers would be *favorable* participants within systems that consist of providers whose transfer rates are elastic. Figure 4 depicts areas of interest computed via application of Equation 7. The parameters $\rho_1 = \lambda_1/k_1$ and $\rho_2 = \lambda_2/k_2$ are varied respectively along the x - and y -axis. In comparison to Figure 2(b), we see that SERES in elastic environments is favorable to both servers for a wider variation between their respective intensities when both intensities are small (i.e., the bubble in the bottom-left corner is bigger). However, when intensities are large, the difference in provider intensities over which both providers are favorable is reduced. Intuitively, this may be due to the fact that when the system is under high loads, the average completion time of a job increases. Bringing in additional capacity (but with a proportional load) does not reduce completion times of admitted jobs significantly when load is full.

⁴In the context of parallel downloading, the sum of all fractional components serviced should add up to k_i .

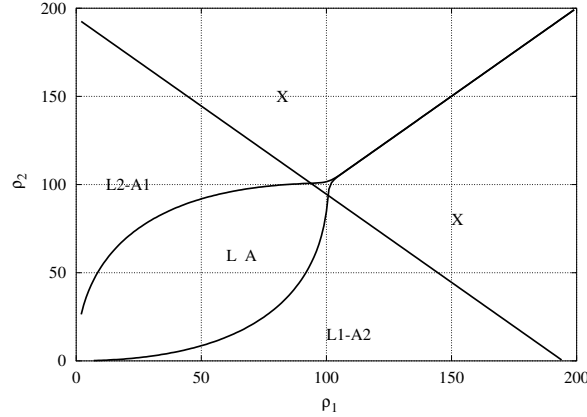


Figure 4: Area of benefit for a SERES system under processor sharing discipline

It will, however, reduce completion times when load is light. Since SERES is most useful when intensities are high (e.g., in Figure 4, the bulb lies almost completely within a range of blocking probabilities that are below the providers’ tolerance limits), we conclude that SERES can tolerate more heterogeneity in systems when servicing fixed-rate requests than when servicing elastic-rate requests.

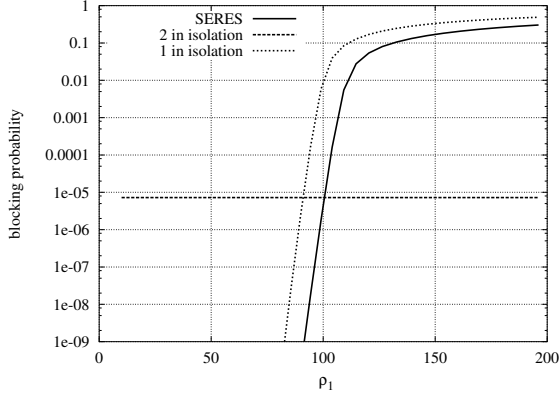
Figure 5 illustrates a scenario in which we vary ρ_1 and maintain ρ_2 fixed at $\rho_2 = 91$. Figure 5(a) plots the blocking probabilities of the shared SERES system as well as of each dedicated system as a function of ρ_1 . The curve labeled “SERES”, “ s_1 ”, and “ s_2 ” respectively plot $p(\{y_1, y_2\})$, $p(\{y_1\})$, and $p(\{y_2\})$ (the last being constant). We observe the blocking probability of the SERES system to be almost 4 orders of magnitude smaller than $p(\{y_1\})$ when ρ_1 and ρ_2 are approximately equal. However, the benefits become marginal with increasing $|\rho_1 - \rho_2|$. This again supports the conclusion that SERES is useful for elastic transfers only when the intensities imposed by providers are approximately the same.

5 Resource Bounding with Thresholds

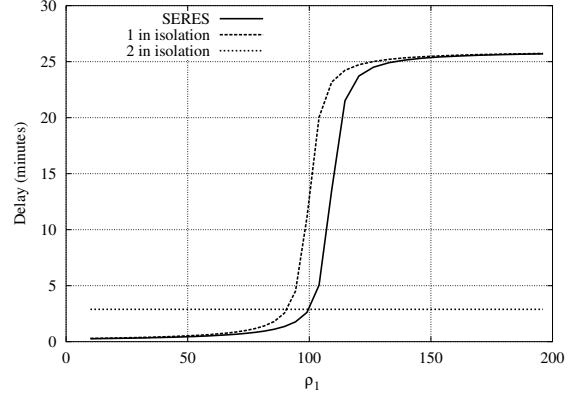
We noted in the previous two sections that when content providers openly share their resources, and individual provider configurations differ, situations can arise in which subsets of providers within the SERES system will not be favorable participants. Here, we evaluate a *thresholding* technique as a means to limit the amount of their own server resources that they provide to the SERES system. We show here that thresholding can be used to bound the blocking probabilities of otherwise unfavorable participants, making them favorable participants.

5.1 Threshold type definitions

Let h_i be a *threshold*, $0 \leq h_i \leq k_i$ for server s_i such that s_i refuses any requests to service other provider’s commodities whenever it is currently servicing h_i commodities that are not v_i . Such a threshold guarantees that at least $k_i - h_i$ slots will be used exclusively for servicing the content provider’s own commodity. We call this threshold type D1. We also evaluate a slight variant on the thresholding technique. A D2 type threshold is one in which an arriving request for another provider’s commodities is denied whenever the number of available slots (i.e., $k_i - x_i$ where x_i is the number of slots that are actively in use) falls below h_i . For both types of thresholding, setting $h_i = 0$ for is equivalent to a dedicated system (i.e., the provider does not participate in SERES) and setting $h_i = k_i$ for all i is equivalent to a SERES system described in previous sections.



(a) Blocking probability under a processor sharing discipline.



(b) Mean delay under a processor sharing discipline.

Figure 5: Evaluation of a SERES system under the processor sharing model.

5.2 Analytical evaluation of D1 thresholding

In order to simplify our analysis we make an assumption that servers participating in SERES enable *swapping*: for any i , if there are fewer than k_i jobs servicing commodity v_i , then all these jobs are serviced by s_i . From a practical perspective, if a slot within s_i becomes available, some ongoing transmission for commodity v_i is swapped from an s_j to s_i if such a j exists, $j \neq i$. We shall see shortly (comparing simulation results) that enabling swapping has little impact on blocking probability.

For the sake of analysis, we assume that service times are exponentially distributed.⁵ This allows us to model the 2-provider SERES system as a truncated Markov chain with states described by the pair (N_1, N_2) , where N_i is the number of sessions servicing commodity v_i in the system, $i = 1, 2$, and $0 \leq N_1 \leq k_1 + \min(k_2 - N_2, h_2)$, and $N_2 \leq k_2 + \min(k_1 - N_1, h_1)$. A crucial difference from previous models is that here, since the decision to accept a request depends on the identity of the commodity being requested, the blocking probabilities for the differing commodities can differ within a SERES system.

This transitions of the described Markov chain occur as follows:

- From $(i - 1, j)$ to (i, j) with rate λ_1 , for $1 \leq i \leq k_1 + \min(k_2 - j, h_2)$.
- From (i, j) to $(i - 1, j)$ with rate $i\mu_1$, for $1 \leq i \leq k_1 + \min(k_2 - j, h_2)$.
- From $(i, j - 1)$ to (i, j) with rate λ_2 , for $1 \leq j \leq k_2 + \min(k_1 - i, h_1)$.
- From (i, j) to $(i, j - 1)$ with rate $j\mu_2$, for $1 \leq j \leq k_2 + \min(k_1 - i, h_1)$.

The above conditions satisfy the requirements that allow computation of the steady-state probabilities as a product-form solution for truncated Markov chains [29].

$$\pi_{i,j} = P(N_1 = i, N_2 = j) = \pi_i \pi_j c_{\{y_1, y_2\}}, \quad (10)$$

where $\pi_i = \rho_1^i / i!$, $\pi_j = \rho_2^j / j!$, and $c_{\{y_1, y_2\}}$ is a normalizing constant such that $\sum_{i=0}^{k_1+h_2} \sum_{j=0}^{k_2+\min(h_1, k_1-i)} \pi_{i,j} = 1$.

⁵Note that our reduction of a provider's server system to a single server with a single commodity still holds WLOG.

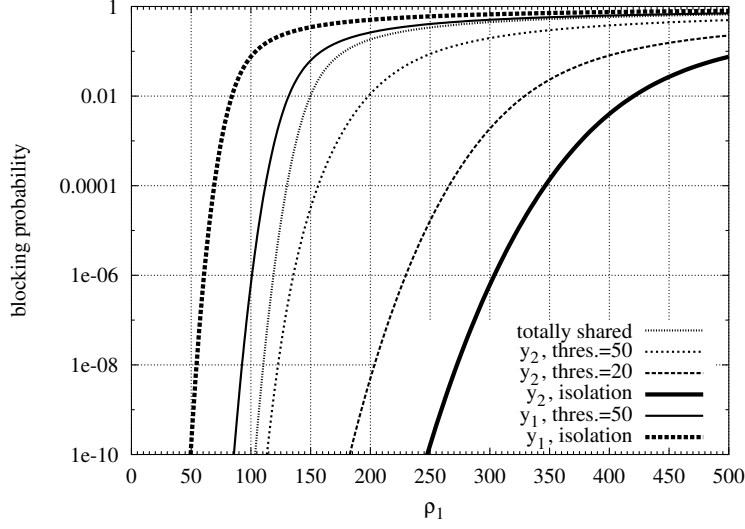


Figure 6: Blocking probability experienced on requests for commodities v_1 and v_2 and different thresholds.

We use Equation 10 to compute the probabilities of all states for which $N_i = k_i + \min(k_j - m, h_j)$, given that $N_j = m, j \neq i$. This yields $p_1(\{y_1, y_2\})$, the blocking probability for commodity v_1 in the SERES system:

$$\begin{aligned}
p_1(\{y_1, y_2\}) &= P(N_1 + N_2) \sum_{i=k_1-h_1}^{k_1+h_2} \pi_{i, k_1+k_2-i} + \sum_{j=0}^{k_2-h_2-1} \pi_{k_1+h_2, j} = \\
&= \sum_{i=k_1-h_1}^{k_1+h_2} \left(\frac{\lambda_1}{\mu_1} \right)^i \frac{1}{i!} \left(\frac{\lambda_2}{\mu_2} \right)^{k_1+k_2-i} \frac{1}{(k_1+k_2-i)!} c_{\{y_1, y_2\}} \\
&\quad + \left(\frac{\lambda_1}{\mu_1} \right)^{k_1+h_2} \frac{1}{(k_1+h_2)!} \left(\sum_{j=0}^{k_2-h_2-1} \left(\frac{\lambda_2}{\mu_2} \right)^j \frac{1}{j!} \right) c_{\{y_1, y_2\}}
\end{aligned} \tag{11}$$

where $c_{\{y_1, y_2\}}$ is the same as in Equation 10. $p_2(\{y_1, y_2\})$, the blocking probability for commodity v_2 , is computed in a similar manner.

Figure 6 depicts respective blocking probabilities experienced for requests of commodities v_1 and v_2 for various intensities and D1 (with swapping) threshold levels, where providers y_1 and y_2 apply the same threshold, i.e., $h_1 = h_2$. We once again consider two servers with a total of $k_i = 100$ slots. On the x -axis, we vary ρ_1 . Instead of fixing ρ_2 , we set $\rho_2 = 0.2\rho_1$ such that the intensities that both providers contribute to SERES increase along the x -axis, but y_1 's intensity is remains much larger than that of y_2 . The various curves depict blocking probabilities for the two commodities for differing threshold levels. The curves for the systems in isolation are represented with thicker lines. The other curves are labeled according to which content the blocking probability refers to and the threshold value. The curve labeled "totally shared" is the blocking probability for both commodity requests (when the thresholds are set to maximum values $h_1 = h_2 = 100$). The remaining curves' labels indicate the provider whose content's blocking probability is being plotted and the value to which the threshold is set.

The most important conclusion here is that variation in the threshold can lead to significant variations in blocking probability for a SERES system. In fact, whenever the load on a server is small enough that the server can meet its desired blocking probability as a dedicated system, that server can then use thresholds in

a SERES system to allow other content providers the use of its resources without raising its own blocking probability above the undesired level.

5.3 Comparison of Thresholding Techniques / Swapping

We now evaluate the blocking probability of SERES system employing thresholding when swapping is disabled, as well as when type D2 thresholding is employed. We resort to simulation to explore these remaining cases. As before, we assume arrivals are described by a Poisson process. Service times are described by a lognormal distribution, as has been observed in practice by [1, 30, 31, 32]. The probability density function of the lognormal distribution is defined as $f_{\text{lognormal}}(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp(-\frac{1}{2}(\frac{\log(x)-\mu}{\sigma})^2)$, where $\log(x)$ is the natural logarithmic function and μ and σ are the standard parameters used within the lognormal distribution. We used the mean and standard deviation of 26 and 46 (minutes) observed in [1], respectively, to derive the parameters μ and σ .

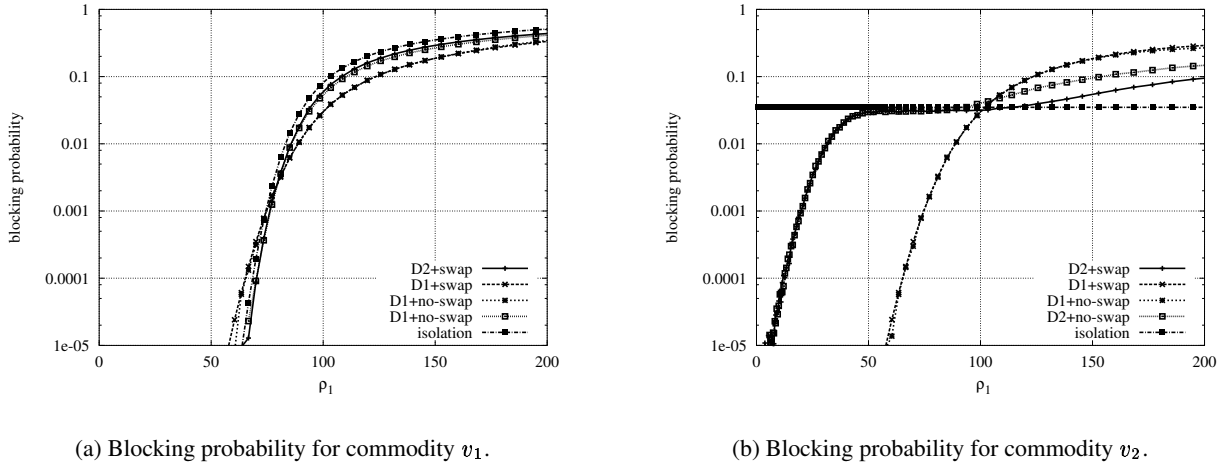


Figure 7: The use of D1-thresholding and D2-thresholding/swapping.

Based on measurements from [1], we conduct simulations with $\lambda = 3.5$ requests per minute and $E[B] = 26$ giving a value $\rho_2 = 91$ for commodity v_2 . Figure 7 plots blocking probabilities obtained from the previous analysis (for the case of D1 thresholding with swapping) and simulations (for the other cases) in which $h_i = 20$ for $i = 1, 2$. Figure 7(a) plots blocking probabilities for commodity v_1 . Figure 7(b) plots blocking probabilities for commodity v_2 . ρ_1 is varied along the x -axis in both Figure 7(a) and Figure 7(b). The curves in each figure labeled “D1+swap”, “D1+non-swap”, “D2+swap”, and “D2+non-swap” depict the various SERES thresholding configurations formed by alternating between the use of non-swapping and swapping methods, and between the use of D1 and D2 thresholding techniques.

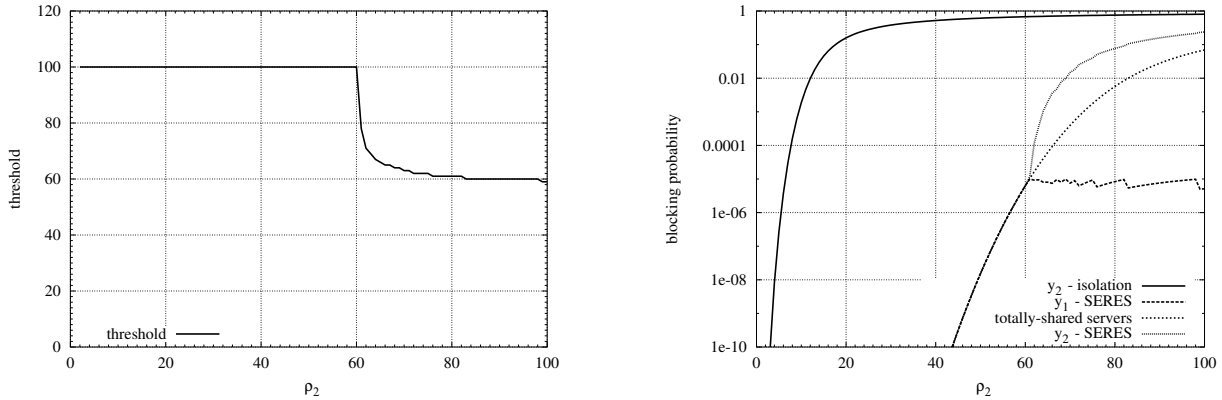
In Figure 7(a), we observe little difference in blocking probability for commodity v_1 as we vary the configuration. In Figure 7(b), we observe the blocking probabilities for the server s_2 . The horizontal line depicts the blocking probability for commodity v_2 when y_2 operates in isolation. These two curves with sharp increases that occur for $\rho_1 < 40$ correspond to the SERES system with D2-thresholds. The two curves are indistinguishable except for $\rho_1 > 100$ where the swapping curve exhibits a blocking probability that is slightly lower. The two curves with sharp increases in the range $60 < \rho_1 < 100$ correspond to the SERES system with D1-thresholds. When operating in the SERES system, the blocking rate for commodity v_2 drops by as much as three orders of magnitude in the range where $\rho_1 < 50$ using D2 and $\rho_1 < 100$ using D1-

thresholds. In the range $\rho_1 > 100$ the blocking probabilities for commodity v_2 increase at a slower rate when using D2 thresholds than when using D1-thresholds. This is because under D2 thresholding, commodity v_2 is less likely to be serviced by server s_1 . In contrast, the blocking probabilities exhibited when using D1-thresholding provide a notable improvement over those exhibited when using D2 thresholding when $\rho_1 < 100$. Hence, D1 thresholding should be used when the competing system's intensity is expected to be higher, and D2 thresholding should be used when the competing system's intensity is expected to be lower.

We further observe that there is little difference in the results obtained when swapping is enabled from when it is not. This suggests that analytical results for blocking probability from a swapping system can be used to approximate blocking probability within non-swapping systems.

5.4 Adaptive Thresholding

Motivated by the results in previous sections, we consider an ideal scenario in which a provider, y_i , can adapt its threshold as a function of the provider intensities imposed on the SERES system so that it contributes the maximum amount of its own server resources (i.e., applies the highest threshold possible) without the blocking probability for its own commodity, v_i , exceeding its tolerance limit, l_i .



(a) Threshold adjustment.

(b) Blocking probabilities.

Figure 8: Ideal algorithm for setting thresholds.

We apply our analysis of D1-type threshold systems with swapping to a 2-provider SERES system in which server s_2 has $k_2 = 20$ slots for processing and server s_1 contains $k_1 = 100$ slots and a fixed provider intensity of $\rho_1 = 20$. Provider y_1 adjusts its threshold h_1 such that the blocking probability for commodity v_1 remains below its tolerance limit of $l_1 = 10^{-5}$. Provider y_2 enables its server to share its resources, i.e., its threshold is set to the total number of slots, $h_2 = k_2 = 20$.

Figure 8(a) plots the value of h_1 as a function of ρ_2 , the provider intensity brought into the system by provider y_2 to maintain $p_1(\{y_1, y_2\}) < l_1$. We see that until $\rho_2 = 60$, the threshold remains at 100. As ρ_2 crosses 60, the threshold drops rapidly, then continues to reduce, but at a much slower rate. Figure 8(b) shows blocking probabilities of various configurations as a function of ρ_2 . The left-most curve plots the blocking probability of commodity v_2 when y_2 operates in isolation. The remaining three curves (which differ only when $\rho > 60$) plot, from top to bottom, the blocking probability for commodity v_2 when SERES with the thresholding described above is applied, the blocking probability for all commodities when SERES is applied without thresholding, and the blocking probability for commodity v_1 when the thresholding de-

scribed above is applied.

The bottom curve verifies that with the thresholding, blocking probabilities for commodity v_1 remain below $l_1 = 10^{-5}$. By comparing the remaining two curves from the SERES system to the curve for the case where y_2 uses a dedicated system, we see that, even when thresholding is deployed, participating in the SERES system provides significant benefits to provider y_2 . We see that, while thresholding increases the blocking probability for commodity v_2 in comparison to a threshold-free SERES system, provider y_1 is favorable only when the thresholding is applied, and commodity v_2 experiences a blocking probability that is orders of magnitude better than in isolation.

6 Conclusion and Future Directions

We have analyzed the performance of resource sharing for content distribution through server collectives (SERES) as a means of reducing blocking probabilities and delays within stochastic environments. Our analysis and simulation upon fundamental queueing models has led to the following results and insights:

- We developed analytical models of server collectives that enable content providers to host one another's content. Separate models are developed for fixed-rate and elastic-rate transmissions. A preliminary queueing analysis upon these models demonstrates that, for homogeneous provider configurations, all providers can simultaneously reduce the blocking probabilities of requests for their content. Similar observations hold for transmission delays of elastic files.
- The most dramatic decreases in blocking probability are observed as the number of participants in small collectives is increased. For instance, we observe a 4-order-of-magnitude reduction in blocking probabilities when comparing an isolated system to a two-server SERES system, while a 10-server SERES system has a 7-order-of-magnitude reduction in comparison to an isolated system.
- We found asymptotic results for a SERES system as the number of participating providers grows to ∞ . If the ratio of the average provider intensity ρ to the average number of processing slots is less than one, then the system's blocking probability is 0 in the limit. Otherwise, the blocking probability converges to $1 - k/\rho$.
- When loads on servers are high, SERES systems (without thresholding) can improve the blocking probability of all participants for a greater variation in intensities among participating systems supporting fixed-rate transfers than can be tolerated within systems supporting elastic-rate transfers.
- We analyzed two thresholding techniques that enable heterogeneous sets of server systems (different intensities and numbers of processing slots) to form a SERES system in which requests for all participants' commodities are dropped at a rate lower than when the systems operate in isolation. We show that, in conjunction with thresholding, the ability to dynamically swap a transmission to the server that profits directly from the servicing of the content has little impact on the blocking probability.

Our future research will explore on on-line algorithms for adaptive thresholding. We are also interested in furthering the development of analytical models for thresholding. We believe that matrix-geometric analysis may prove to be fruitful in this area. We are also interested in extending our model to cases where the "usefulness" of a particular server is a function of its proximity within the network to the client issuing the request. Policies and protocols used by providers to perform SERES organization and efficient redirection are important issues that must be examined before SERES can be deployed in practice. As a long-term goal we envision implementation and deployment of SERES.

Acknowledgements

We would like to thank Ed Coffman and Predrag Jelenkovic, and Vishal Misra for valuable discussions regarding this work. We are also thankful to Ward Whitt for pointing out a reference on asymptotic behavior of loss systems.

A Notation

$\mathcal{V} = \{v_1, v_2, \dots, v_m\}$	- set of commodities
$\mathcal{S} = \{s_1, s_2, \dots, s_n\}$	- set of servers
$\mathcal{Y} = \{y_1, y_2, \dots, y_n\}$	- set of content providers
$\psi(s_i)$	- set of commodities offered by server s_i
$\phi(y_i)$	- set of commodities offered by content provider y_i
$p_i(\{s_1, s_2, \dots, s_n\})$	- blocking probability of commodity v_i in a SERES system with n servers
N_i	- number of sessions of commodity v_i in the SERES system
k_i	- maximum capacity on server s_i
λ_i	- arrival rate of requests for commodity v_i
B_i	- time for processing commodity v_i
ρ_i	- $\rho_i = \lambda_i E[B_i]$
π_i	- $\pi_i = \rho_i / i!$
$\hat{\rho}$	- $\hat{\rho} = \sum_{j=1}^{n_c} f_j \rho_j$
\hat{k}	- $\hat{k} = \sum_{j=1}^{n_c} f_j k_j$
γ	- $\gamma = \hat{\rho} / \hat{k}$
h_i	- threshold imposed by server s_i on non-primary commodities
D1-threshold	- threshold definition based on active number of non-primary sessions
D2-threshold	- threshold definition based on remaining available capacity
$d_i(\{y_1, y_2, \dots, y_n\})$	- overall time of v_i -session in SERES
l_i	- maximum blocking probability in the tolerance range of provider y_i
$\Gamma_i = (x_1, x_2, \dots, x_n)$	- state given by remaining fractions x_i of work for a given job i
$B_i(x), b_i(x)$	- distribution and density function of B_i , respectively
$\bar{B}_i(x)$	- $\bar{B}_i(x) = P(B_i > x)$
$w(x)$	- hazard function $b_i(x) / \bar{B}_i(x)$

B Number of sessions under processor sharing discipline

In order to find the distribution of number of sessions in the system, we follow a procedure similar to the derivation described in [23] (pp. 171) for an unbounded system. The work introduced to the system by an accepted session is a random variable B . We denote its distribution $B(x)$ and the density probability function $b(x)$. Furthermore, we make use of the function $\bar{B}(\cdot)$ defined as $\bar{B}(x) = P(B > x)$. The hazard function is defined as the ratio $w(x) = b(x) / \bar{B}(x)$.

We define states of the system as the current completed fraction of service relative to the individual jobs in increasing order. For example, given n jobs we have the state $\mathbf{x} = (x_1, x_2, x_3, \dots, x_n)$. Since each job receives $1/n$ -th of the processing, a transition from state $\mathbf{x} = (x_1, x_2, \dots, x_i, \dots, x_n)$ to state $\mathbf{e}_i(\mathbf{x}) = (x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$ occurs with probability intensity (loosely speaking "rate") $w(x_i) / n$. On the other hand for the reverse process the transition from $\mathbf{e}_i(\mathbf{x})$ to the state \mathbf{x} is given by $\lambda b(x_i)$. In fact we conjecture that we may construct a reverse process with balancing equations given by this transition rates

and use the theorem shown in [23] (pp. 164). We shall have

$$p(\mathbf{x}) \frac{w(x_i)}{n} = p(\mathbf{e}_i(\mathbf{x})) \lambda b(x_i) \quad (12)$$

We assume without loss of generality that $x_1 \leq x_2 \leq x_3 \dots \leq x_n$. From the definition of the hazard function we rewrite Equation 12 as

$$p(\mathbf{x}) = n p(\mathbf{e}_i(\mathbf{x})) \lambda \bar{B}(x_i) \quad (13)$$

Using the definition of $w(\cdot)$ we are able to find the probability of states as recursively as the number in the system decreases:

$$P(\mathbf{x}) = P(\mathbf{x} = (x_1, x_2, \dots, x_n)) = P(N = 0) n! \lambda^n \prod_{i=1}^n \bar{B}(x_i). \quad (14)$$

We find the expression for the number of sessions in the system as:

$$\begin{aligned} P(N = n) &= n! \lambda^n P(N = 0) \int_{x_1} \int_{x_2} \dots \int_{x_n} \bar{B}(x_i) dx_1 dx_2 \dots dx_n \\ &= \lambda^n P(N = 0) \int \int \dots \int_{x_1 \leq x_2 \leq \dots \leq x_n} \bar{B}(x_i) dx_1 dx_2 \dots dx_n \\ &= P(N = 0) (\lambda E[B])^n. \end{aligned} \quad (15)$$

We can easily find $P(N = 0)$ by applying the normalizing condition $\sum_{n=1}^k P(N = n) = 1$, thus finding:

$$P(N = 0) = \frac{1 - \lambda E[B]}{1 - (\lambda E[B])^{k+1}} \quad (16)$$

Hence, the blocking probability is the probability of the event of maximum utilization which is given by

$$p(\{y_1\}) = (\lambda E[B])^k \frac{1 - \lambda E[B]}{1 - (\lambda E[B])^{k+1}}. \quad (17)$$

Since we verify that $p(\mathbf{x}) = p(0, \mathbf{x}) / (n + 1)$, we validate the initial conjecture.

References

- [1] J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon, "Analysis of educational media server workloads," in *Proc. of NOSSDAV*, pp. 21–30, June 2001.
- [2] M. Dahlin, R. Wang, T. Anderson, and D. Patterson, "Cooperative caching: Using remote client memory to improve file system performance," in *Proc. of OSDI'94*, 1994.
- [3] G. Voelker, H. Jamrozik, M. Vernon, H. Levy, and E. Lazowska, "Managing server load in global memory systems," in *Proc. of 1997 ACM Sigmetrics*, June 1997.
- [4] G. Voelker, E. Anderson, T. Kimbrel, M. Feeley, J. Chase, A. Karlin, and H. Levy, "Implementing cooperative prefetching and caching in a global memory system," in *Proc. of the 1998 ACM Sigmetrics Conference*, June 1998.
- [5] J. Kangasharju, J. W. Roberts, and K. W. Ross, "Object replication strategies in content distribution networks," in *Proceedings of Sixth International Workshop on Web Caching and Content Delivery*, June 2001.

- [6] D. Eager, E. Lazowska, and J. Zahorjan, "A comparison of receiver-initiated and sender-initiated adaptive load sharing," *Performance Evaluation*, vol. 16, May 1986.
- [7] D. Eager, E. Lazowska, and J. Zahorjan, "Adaptive load sharing in distributed systems," *IEEE Transactions on Software Engineering*, vol. 12, May 1986.
- [8] I. Research, "Oceano project." <http://www.research.ibm.com/oceanoproject/>.
- [9] L. Golubchik and J. C. S. Lui, "Bounding of performance measures for threshold-based systems: Theory and application to dynamic resource management in video-on-demand servers," *IEEE Transactions of Computers*, to appear.
- [10] C. Alliance, "<http://www.content-peering.org/>."
- [11] A. Biliris, C. Cranor, F. Douglass, M. Rabinovich, S. Sibal, O. Spatscheck, and W. Sturm, "CDN brokering," in *Proceedings of WCW'01*, June 2001.
- [12] V. N. Padmanabhan and K. Sripanidkulchai, "The case for cooperative networking," in *Proc. of First IPTPS*, Mar. 2002.
- [13] T. Stading, P. Maniatis, and M. Baker, "Peer-to-peer caching schemes to address flash crowds," in *1st International Workshop on Peer-to-Peer Systems (IPTPS 2002)*, Mar. 2002.
- [14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of ACM SIGCOMM 2001*, (San Diego, CA), Aug. 2001.
- [15] R. J. B. Jr., A. Somani, D. Gruhl, and R. Agrawal, "Youserv: A web hosting and content sharing tool for the masses," in *Proc. of WWW-2002*, 2002.
- [16] E. G. C. Jr., R. R. Muntz, and H. Trotter, "Waiting time distributions for processor-sharing systems," *J. Assoc. Comp. Mach.*, vol. 17, pp. 123–130, 1970.
- [17] A. P. Zwart and O. J. Boxma, "Sojourn time asymptotics in the M/G/1 processor sharing queue," *Queueing Systems*, vol. 35, no. 1–4, pp. 141–166, 2000.
- [18] P. Jelenkovic and P. Momcilovic, "Heavy-tailed waiting times in a processor sharing queue," in *Proc. of 11th INFORMS Applied Probability Society Conference*, July 2001.
- [19] S. Borst, O. Boxma, and P. R. Jelenkovic, "Asymptotic behavior of generalized processor sharing with long-tailed traffic sources," in *Proc. of INFOCOM'00*, Mar. 2000.
- [20] N. Bansal and M. Harchol-Balter, "Analysis of srpt scheduling: Investigating unfairness," in *Proceedings of ACM Sigmetrics 2001*, 2001.
- [21] M. Crovella, M. Harchol-Balter, and C. Murta, "Task assignment in a distributed system: Improving performance by unbalancing load," in *Proceedings of ACM Sigmetrics'98*, 1998.
- [22] R. D. Nelson, "The mathematics of product form queuing networks," *ACM Computing Surveys*, vol. 25, pp. 339–369, Sept. 1993.
- [23] S. Ross, *Stochastic Processes*. John Wiley & Sons, Inc., 1983.
- [24] R. W. Wolff, *Stochastic Modeling and the Theory of Queues*, ch. 6, pp. 334–341. Prentice Hall, 1988.
- [25] I. RealNetworks, "<http://www.real.com/>" web-site,

- [26] K. Ross, *Multiservice loss models for Broadband Telecommunications Networks*. Springer, 1995.
- [27] J. Byers, M. Luby, and M. Mitzenmacher, "Accessing multiple mirror sites in parallel: Using tornado codes to speed up downloads," in *Proc. of IEEE INFOCOM'99*, 1999.
- [28] L. Kleinrock, *Queueing systems*. Wiley, 1974.
- [29] D. Bertsekas and R. Gallager, *Data Networks*. Prentice-Hall, second edition ed., 1992.
- [30] J. Padhye and J. Kurose, "An empirical study of client interactions with a continuous-media courseware server," *IEEE Internet Computing*, Apr. 1999.
- [31] M. Chesire, A. Wolman, G. M. Voelker, and H. M. Levy, "Measurement and analysis of a streaming-media workload," in *Proceedings of the Third USENIX Symposium on Internet Technologies and Systems*, Mar. 2001.
- [32] S. Jin and A. Bestavros, "Gismo: Generator of streaming media objects and workloads," *Performance Evaluation Review*, 2001.