Performance Analysis for Overflow Loss Systems of Processor-Sharing Queues

Yin-Chi Chan, Jun Guo, Eric W. M. Wong and Moshe Zukerman
Department of Electronic Engineering, City University of Hong Kong, Hong Kong
Email: ycchan26-c@my.cityu.edu.hk, {j.guo, eeewong, m.zu}@cityu.edu.hk

***Abstract*—Overflow loss systems have wide applications in telecommunications and multimedia systems. In this paper, we consider an overflow loss system consisting of a set of finite-buffer processor-sharing (PS) queues, and develop effective methods for evaluation of its blocking probability. For such a problem, an existing approximation of the blocking probability is based on decomposition of the system into independent PS queues. We provide a new approximation which instead performs decomposition on a surrogate model of the original system, and demonstrate via extensive numerical results that our new approximation is more accurate and robust than the existing approach. We also examine the sensitivity of the blocking probability to the service time distribution, and demonstrate that an exponential distribution is a good approximation for a wide range of service time distributions.**

I. Introduction

Overflow loss systems are an important class of teletraffic models with wide applications in telecommunications and multimedia systems including gradings [1] and video-on-demand (VoD) systems [2]. They have also found applications in service sectors for modeling emergency response systems [3], [4], call centers [5] and intensive care units [6]. In general, overflow loss systems are characterized by requests requiring service in a system comprising multiple nodes [7]. Requests are divided into classes. Each node is a finite-buffer queue capable of serving requests from some of the classes. If a node does not have sufficient idle capacity to serve a request, the request *overflows* immediately to another node. The request is blocked and cleared from the system if none of the nodes capable of serving it has sufficient idle capacity. Accordingly, the probability that the request is blocked and cleared from the system is called the *blocking probability*.

The notion of a *processor-sharing* (PS) queue arose from the modeling of time-shared computing systems [8]. In time-sharing, a processor constantly cycles through its buffer of requests, giving each request Q seconds of service time per cycle. PS refers to the idealized case of time-sharing where $Q \to 0$ and there is no time penalty for swapping between requests. PS can also be applied to the allocation of bandwidth in web servers (see for example [9]).

In this paper, we consider an overflow loss system consisting of a set of finite-buffer PS queues. The model, with the assumptions of Poisson arrivals and exponentially distributed service times, is motivated from [10] in the context of a peer-to-peer (P2P) VoD system. It was observed in [10] that such a system of single-server PS queues, where the buffer size of each queue is K and the service capacity of each server is B, has a lower blocking probability than a system of K-server first-come-first-serve (FCFS) queues with no waiting room and where the service capacity of each server is B/K. An approximation method was presented in [10] for evaluation of the blocking probability. It is based on decoupling the given system into independent PS queues and treating each queue as an independent M/M/1/K-PS queue. (Here, the use of "-PS" to indicate PS queueing discipline in Kendall notation is consistent with common practice, see e.g. [11], [12].) This decomposition technique inherently gives rise to a fixed-point solution of the blocking probability. We shall call it the PS *fixed-point approximation* (PS-FPA), as it is similar to a classic method first introduced in [13] and now widely known as the Erlang fixed-point approximation (EFPA) [14] for evaluation of blocking probabilities in loss networks.

In this paper, we extend and apply two recently established approximation frameworks, namely, *overflow priority classification approximation* (OPCA) [15] and *information exchange surrogate approximation* (IESA) [16], for blocking probability evaluation in the overflow loss system of PS queues. As a result, we obtain PS-OPCA and PS-IESA, and demonstrate that these new methods provide increased accuracy over PS-FPA, with PS-IESA being generally the most robust of the three approximations. We also examine the sensitivity of the request blocking probability to the service time distribution, demonstrating that an exponential distribution is a good approximation for a wide range of service time distributions.

The rest of the paper is organized as follows. In Section II, we review the literature of PS queues and overflow loss systems. The model of the overflow loss system of PS queues is described in Section III. In Section IV, we provide details of the three approximations. Numerical results are presented in Section V, with concluding remarks given in Section VI.

II. Related Work

A. PS queues

A theoretical treatment of a *single* M/M/1-PS queue was given in [17], in which formulas are derived for the mean sojourn time and the mean number of jobs in the system. An extension of the M/M/1-PS queue to one with priorities was also given in [17].

In many applications, a finite-buffer PS queue is of interest, in which there is an upper bound on the total number of jobs that the PS queue can accept [11], [18]. An optimal design

of finite-buffer PS queues was provided in [19]. A study of the mean sojourn time of requests to an M/M/1/K-PS queue, conditioned on the service requirement, was done in [11]. For evaluation of the blocking probability of an M/M/1/K-PS queue, it suffices to note that an M/M/1/K-PS queue has the same Markov chain representation as an M/M/1/K (FCFS) queue.

An extension of the M/M/1/K-PS queue to one with two priority classes was studied in [20], where the queue is offered a mix of long-lived real-time traffic and short-lived non-real-time traffic, with the real-time traffic given strict (preemptive) priority. An approximation for the blocking probabilities of both traffic classes was provided.

B. Overflow loss systems

It is well known in teletraffic engineering that evaluation of blocking probabilities in overflow loss systems is a difficult problem [21]. This is particularly true for non-hierarchical models where overflow from any node may directly or indirectly affect the load of any other node. In particular, the *mutual overflow* effect [22] refers to a situation where there is congestion on a specific node causing overflow to the other nodes, and where this overflow loads up the other nodes so that they in turn yield overflow back to the original node. Such models, in many practical cases, are not amenable to an exact analysis because they exhibit significant state dependencies. McNamara [23] showed that non-hierarchical overflow loss systems do not have product-form solutions for the blocking probability. The problem, with the assumptions of Poisson arrivals and exponentially distributed service times, can only be solved exactly by a multi-dimensional Markov process. Although the blocking probability can, in principle, be obtained by solving numerically a set of steady-state equations, this approach is not scalable because of the curse of dimensionality.

In the model of [10], each node is an PS queue, and requests seek service from the nodes in random order until an available node is found. This forms a non-hierarchical overflow loss system. The approximation in [10] assumes that the arrival process to each node is Poisson and independent of the other nodes; thus each node can be treated as an independent M/M/1/K-PS queue. However, the mean of the arrival process to each node still depends on the blocking probability of the other nodes, creating a fixed-point dependency that can be solved via iterative substitution.

The approximation method of [10] is similar to the classic EFPA method for loss networks [14] in which each node is treated as an M/M/K/K (FCFS) queue. (In contrast to an overflow loss system, in a loss network requests must be routed between *specific* origin and destination nodes.) EFPA is itself based on an earlier approximation [13] that does not require that the input to each node be Poisson, instead using a two-moment approximation known as the equivalent random method [24]. However, it was shown by Kelly [14] that for fixed-routing circuit-switched networks EFPA is asymptotically exact. We will use the term EFPA to refer specifically to Kelly's version without moment-matching, and PS-FPA for its adaptation to systems of PS queues.

Despite its success in certain applications, EFPA can often lead to large errors [15], [16], due to the major assumptions that the offered traffic to each queue follows a Poisson process and that the queues are mutually independent. These assumptions lead to two types of errors called the Poisson and independence errors respectively.

While various moment-matching techniques exist for reducing the Poisson error in systems of FCFS queues [13], [25]–[27], no simple moment-matching technique exists for systems of PS queues. We will therefore not use moment matching in this paper and leave it as an open research problem for future work. Techniques to reduce the independence error for overflow loss systems have also been developed in recent years [15], [16], [28]. The key philosophy behind these new approximation frameworks is to establish a certain surrogate model that in a systematic way approximately captures the state dependencies due to mutual overflow in the original model. Ideally, the difference between the blocking probability of the surrogate model and that of the original model would cancel out the error introduced in approximation of the surrogate model.

Wong *et al.* [15], [28] first developed the OPCA framework, using a surrogate model in which a preemptive priority scheme is deployed based on the number of times a call is overflowed. The more recently developed IESA framework [16] uses an alternate type of surrogate model that replaces preemption with information exchange. Two IESA algorithms were presented in [16]: while IESA1 is numerically equivalent to OPCA, IESA2 is shown to be more accurate and robust for overflow loss systems of FCFS queues. We shall thus use the term IESA to refer solely to IESA2 for the remainder of this paper.

III. Model

Our model is motivated from a P2P VoD system [10] and considers two types of nodes: *peers* and *leeches*. Peers both generate and serve requests, while leeches generate requests only. Let $\mathbf{X} = \{1, \ldots, X\}$ be the set of peers, working together to serve a catalog $\mathbf{C} = \{1, \ldots, C\}$ of contents. For each peer $x \in \mathbf{X}$, let $\mathbf{C}_x \subseteq \mathbf{C}$ be the set of *pre-fetched* contents available in x (we assume a static file assignment). Conversely, for each content $c \in \mathbf{C}$, let $\mathbf{X}_c \subseteq \mathbf{X}$ be the set of peers with a copy of c, with $k_c = |\mathbf{X}|$. Thus $\mathbf{C}_x = \{c : x \in \mathbf{X}_c\}$, and $\mathbf{X}_c = \{x : c \in \mathbf{C}_x\}$. Finally, define $k_x^\star = \max_{c \in \mathbf{C}_x} k_c$. We shall also define Y as the number of leeches.

When a leech requests content $c \in \mathbf{C}$, it attempts to download content c from each peer in $\mathbf{X}_c$ in random order until one of the peers accepts the download attempt. We shall call this scheme *repeated random trials* or RRT. Finally, if a request is not served by any peer, the request is blocked.

On the other hand, if a peer x requests content $c \in \mathbf{C}$, it first checks if c is available on x, that is, if $c \in \mathbf{C}_x$. If yes, the request is a *local request* and the peer uses its own local copy of c. Otherwise, the request is a *non-local* request and

the requesting peer downloads the content from another peer using RRT. We define the blocking probability of the system as the probability that a non-local request is blocked. All requests made by leeches are also non-local requests.

Requests are served in a PS manner. Let U_x be the upload bandwidth of peer x. This bandwidth is shared equally among all requests at x. Let d be the minimum required transfer bandwidth for immediate playback; the maximum number of *non-local* requests each peer can serve is thus $N_x = \lfloor U_x/d \rfloor$.

We assume that the arrival process of requests for each content c is Poisson with rate λ_c, and the originating node for each request is uniformly random, ignoring the playback/idle status of each peer. It was shown in [29] that a Poisson process is an accurate portrayal of request arrivals in a P2P system. As the random thinning of a Poisson process is also a Poisson process, the arrival process of non-local requests for each content c is a Poisson process, with rate $\lambda_c \left(1 - \frac{k_c}{X+Y}\right)$.

We define the *mean service time* $(\mu_{x,c})^{-1}$ as the mean time it would take peer x to upload content c to another node if *no other requests* are being processed by x.

IV. Approximations

As PS-FPA, PS-OPCA, and PS-IESA are unscalable when applied directly to a scheme based on RRT, we instead use, for approximation purposes only, an alternate *round robin with random start* (RRRS) scheme for download attempts. Define an ordering on $\mathbf{X}_c$ such that $X_c(i) < X_c(j)$ for all $0 \le i < j < k_c$. Let $\mathbf{X}_{c,s}$ be a leftwise circular shift of $\mathbf{X}_c$ by s positions, such that $X_{c,s}(i) = X_c((s+i) \mod k_c)$. A request for content c will thus attempt, in order, peers $\mathbf{X}_{c,s} = \{X_{c,s}(0), \ldots, X_{c,s}(k_c-1)\}$ for some random $s \in \{0, \ldots, k_c-1\}$. We also define $\mathbf{X}_{c,s}^{(n)} = \{X_{c,s}(0), \ldots, X_{c,s}(n-1)\}$ as the first n terms of $\mathbf{X}_{c,s}$, $n \le k_c$.

In Section V, we compare PS-FPA, PS-OPCA and PS-IESA for RRRS directly against the simulation results for RRT. A similar approach was used in [16] for systems of FCFS queues, where it was shown that there is little difference between RRRS and RRT for cases where EFPA, OPCA, and IESA are tractable for both routing schemes.

A. PS-FPA

With the assumptions of independent Poisson input to each queue, each peer $x \in \mathbf{X}$ can be seen of as an M/M/1/N_x-PS queue. By changing each queue from an M/M/K/K queue to an M/M/1/K-PS queue, we convert EFPA to PS-FPA. In PS-FPA, as in EFPA, the offered traffic to each peer affects its blocking probability, which in turn affects the amount of overflow traffic from that peer to other peers. This creates a cyclic dependency with a fixed-point solution that can be found via repeated substitution.

Let $\lfloor a, b \rceil$ be the set of integers between a and b inclusive, thus $\lfloor a, b \rceil = [a, b] \cap \mathbb{Z}$. Define:

- $\lambda_{c,s,n}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 0, k_c - 1 \rceil$, as the arrival rate to peer $X_{c,s}(n)$ of requests for content c that have overflowed n times along the path $\mathbf{X}_{c,s}^{(n)}$.
- $a_{c,s,n} = \lambda_{c,s,n} (\mu_{x,c})^{-1}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 0, k_c - 1 \rceil$, as the offered load to peer $X_{c,s}(n)$ of requests for content c that have overflowed n times along the path $\mathbf{X}_{c,s}^{(n)}$, where $x = X_{c,s}(n)$.
- $a_{x,n}$, $x \in \mathbf{X}$, $n \in \lfloor 0, k_c - 1 \rceil$, as the offered load to peer x made up of requests that have overflowed n times.
- A_x, $x \in \mathbf{X}$, as the total offered load to peer x, namely $A_x = \sum_{n=0}^{k_x^\star - 1} a_{x,n}$.
- B_x, $x \in \mathbf{X}$, as the probability that all upload slots of peer x are full.

Based on the Poisson traffic assumption, the blocking probability of peer x is given by

$$B_x = \frac{A_x^{N_x}(1 - A_x)}{1 - A_x^{(N_x+1)}},$$

which is the blocking probability of an M/M/1/N_x-PS queue offered A_x Erlangs of traffic. Based on the independence assumption, we can define $\lambda_{c,s,n} = \lambda_{c,s,n-1} B_z$ where $z = X_{c,s}(n-1)$ and

$$\lambda_{c,s,0} = \frac{\lambda_c}{k_c}\left(1 - \frac{k_c}{X+Y}\right).$$

Summing over all eligible c and s such that $X_{c,s}(n) = x$, we obtain

$$a_{x,n} = \sum_{c:n<k_c} \sum_{s:X_{c,s}(n)=x} a_{c,s,n}.$$

The above equations form a circular relationship that can be solved via iterated substitution. Finally, the blocking probability of each content $c \in \mathbf{C}$ is $\hat{B}_c = \prod_{x \in \mathbf{X}_c} B_x$ and the overall system blocking probability is

$$B = \frac{\sum_{c \in \mathbf{C}} \lambda_c \left(1 - \frac{k_c}{X+Y}\right) \hat{B}_c}{\sum_{c \in \mathbf{C}} \lambda_c \left(1 - \frac{k_c}{X+Y}\right)}. \quad (1)$$

B. PS-OPCA

Under OPCA, each request in the surrogate model is assigned an extra parameter based on the number of failed download attempts made, from zero to k_c, for a request for content c. Requests with more failed attempts are considered *senior* to requests with less failed attempts. A preemptive priority scheme is applied where incoming requests to a peer can preempt the most senior request in service if **(i)** all upload slots of the peer are full, and **(ii)** the incoming request is junior to the most senior request being served. The preempted request may re-attempt to obtain service from any peer it did not attempt before. Below, we present a version of OPCA for systems of PS queues, namely PS-OPCA.

Formally, consider an incoming request to some peer $x \in \mathbf{X}$ which has overflowed from n_1 peers. If x has any upload slots available, the request is served (and the service rates of all other requests at x decrease accordingly). In the case that all upload slots are full, let n_2 be the seniority of the most senior request being serviced by x. If $n_1 \ge n_2$, the incoming request overflows normally with a new seniority of $n_1 + 1$. However,

if $n_1 < n_2$, then the incoming request preempts the senior call, which overflows with a seniority of $n_2 + 1$.

Define:

- $\lambda_{c,s,n}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 0, k_c - 1 \rceil$, as the arrival rate to peer $X_{c,s}(n)$ of requests for content c with seniority n, having overflowed along the path $\mathbf{X}_{c,s}^{(n)}$.
- $a_{c,s,n} = \lambda_{c,s,n} (\mu_{x,c})^{-1}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 0, k_c - 1 \rceil$, as the offered traffic to peer $X_{c,s}(n)$ made up of requests for content c with seniority n, where $x = X_{c,s}(n)$.
- $a_{x,n}$, $x \in \mathbf{X}$, $n \in \lfloor 0, k_c - 1 \rceil$, as the offered load to peer x made up of requests with seniority n.
- $A_{x,n}$, $x \in \mathbf{X}$, $n \in \lfloor 0, k_c - 1 \rceil$, as the offered load to peer x made up of requests with seniority in $\lfloor 0, n \rceil$, namely $A_{x,n} = \sum_{i=0}^{n} a_{x,i}$.
- $B_{x,n}$, $x \in \mathbf{X}$, $n \in \lfloor 0, k_c - 1 \rceil$, as the probability that all upload slots of peer x are serving requests with seniority in $\lfloor 0, n \rceil$.
- $b_{x,n}$, $x \in \mathbf{X}$, $n \in \lfloor 0, k_c - 1 \rceil$, as the probability that a request with seniority in $\lfloor 0, n \rceil$ overflows from x.

Based on the Poisson traffic assumption, we can define

$$B_{x,n} = \frac{A_{x,n}^{N_x}(1 - A_{x,n})}{1 - A_{x,n}^{(N_x+1)}},$$

which is the blocking probability of an M/M/1/N_x-PS queue offered $A_{x,n}$ Erlangs of traffic. Based on the independence assumption, we can define $\lambda_{c,s,n} = \lambda_{c,s,n-1} B_{z,n-1}$, where $z = X_{c,s}(n-1)$ and

$$\lambda_{c,s,0} = \frac{\lambda_c}{k_c}\left(1 - \frac{k_c}{X+Y}\right).$$

Summing over all eligible c and s such that $X_{c,s}(n) = x$, we obtain

$$a_{x,n} = \sum_{c:n<k_c} \sum_{s:X_{c,s}(n)=x} a_{c,s,n},$$

from which $A_{x,n}$ and $B_{x,n}$ can be computed using the equations above. Then,

$$b_{x,n} = \frac{A_{x,n}B_{x,n} - A_{x,n-1}B_{x,n-1}}{a_{x,n}},$$

with base case $b_{x,0} = B_{x,0}$.

The above equations can be solved iteratively for increasing n. Finally, the blocking probability of each content $c \in \mathbf{C}$ is

$$\hat{B}_c = \frac{1}{k_c} \sum_{s=0}^{k_c-1} \prod_{n=0}^{k_c-1} b_{X_{c,s}(n),n}$$

and the overall blocking probability can be found using (1).

C. PS-IESA

OPCA has been shown to be accurate when each request can access a high percentage of the system resources, but less so when the availability is not high [16]. IESA improves accuracy when approximating low-availability systems by using an alternate surrogate model based on *information exchange* rather than preemption. In this paper, we will show this is also true of PS-IESA, where the queue discipline is changed from FCFS to PS.

In addition to a count of failed download attempts, each request in the surrogate model of IESA also stores a *congestion estimate* of the number of peers for which all upload slots are occupied. When a request is declined by a peer with no upload slots available, the incoming request exchanges congestion estimates with the request in service possessing the highest congestion estimate, if the incoming request is junior to that request. After each failed download attempt (and possible information exchange), a request may immediately give up based on the probability that all remaining peers are full.

Formally, consider an incoming request for content x with n_1 overflows and a seniority of j_1, which we denote as an (n_1, j_1)-request. If x has upload slots available, the request is served. In the case that all upload slots are full, select an (n_2, j_2)-request from the list of requests currently being serviced by x such that j_2 is the maximum of all the congestion estimates of the requests. If $j_1 \geq j_2$, the incoming request overflows normally as an $(n_1 + 1, j_1 + 1)$-request and the request in service remains an (n_2, j_2)-request. However, if $j_1 < j_2$, then the incoming request overflows as an (n_1+1, j_2+1)-request while the call in service becomes an (n_2, j_1)-request. Thus, $j \geq n$ for any incoming (n, j)-request, although this may be false for requests in service.

The congestion estimate is used to control the overflow of requests as follows. Consider an (n, j)-request for content c. We define

$$P_{c,n,j} = \frac{\binom{j-n}{k_c-n}}{\binom{X-n}{k_c-n}},$$

which is the probability that the remaining $k_c - n$ unvisited peers in $\mathbf{X}_c$ are all full. With probability $P_{c,n,j}$, the request immediately gives up without attempting any additional peers. Otherwise, the process repeats until $j = X$ or $n = k_c$, upon which $P_{c,n,j} = 1$ and the request is automatically blocked.

Define:

- $\lambda_{c,s,n,j}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 0, \min(j, k_c - 1) \rceil$, $j \in \lfloor 0, X - 1 \rceil$, as the arrival rate to peer $X_{c,s}(n)$ of (n, j)-requests for content c, having overflowed along the path $\mathbf{X}_{c,s}^{(n)}$.
- $a_{c,s,n,j}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 0, \min(j, k_c - 1) \rceil$, $j \in \lfloor 0, X - 1 \rceil$, as the offered load to peer $X_{c,s}(n)$ made up of (n, j)-requests for content c, where $x = X_{c,s}(n)$.
- $a_{x,n,j}$, $x \in \mathbf{X}$, $n \in \lfloor 0, \min(j, k_x^\star - 1) \rceil$, $j \in \lfloor 0, X - 1 \rceil$, as the offered load of all (n, j)-requests to peer x for all contents $c \in \mathbf{C}_x$.
- $\tilde{a}_{c,s,n,j}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 0, \min(j, k_c - 1) \rceil$, $j \in \lfloor 0, X - 1 \rceil$, as the offered load to $X_{c,s}(n)$ made up of all (n, Ω)-requests for content c, $\Omega \in \lfloor 0, j \rceil$.
- $\tilde{a}_{x,n,j}$, $x \in \mathbf{X}$, $n \in \lfloor 0, \min(j, k_x^\star - 1) \rceil$, $j \in \lfloor 0, X - 1 \rceil$, as the offered load of all (n, Ω)-requests to peer x for all contents $c \in \mathbf{C}_x$, $\Omega \in \lfloor 0, j \rceil$.
- $v_{c,s,n,j}$, $c \in \mathbf{C}$, $s \in \lfloor 0, k_c - 1 \rceil$, $n \in \lfloor 1, \min(j, k_c) \rceil$, $j \in \lfloor 1, X \rceil$, as the *overflow traffic* from peer $X_{c,s}(n)$ made up of (n, j)-requests for content c.

By definition, any of the above values are zero for $j < n$. For recursion purposes, we shall also set the above variables to be zero for negative n or j. We further define:

- $A_{x,j}$, $x \in \mathbf{X}$, $j \in \lfloor 0, X-1 \rceil$, as the total offered load to peer x of all requests with seniority $\Omega \in \lfloor 0, j \rceil$.
- $B_{x,j}$, $x \in \mathbf{X}$, $j \in \lfloor 0, X-1 \rceil$, as the blocking probability of peer x at level j.

By definition,

$$A_{x,j} = \sum_{n=0}^{\min(j,k_x^\star-1)} \tilde{a}_{x,n,j},$$

$$\tilde{a}_{x,n,j} = \sum_{i=n}^{j} a_{x,n,i},$$

$$a_{x,n,j} = \sum_{c:n<k_c} \sum_{s:X_{c,s}(n)=x} a_{c,s,n,j},$$

and

$$\tilde{a}_{c,s,n,j} = \sum_{i=n}^{j} a_{c,s,n,i}.$$

Based on the Poisson assumption, we can define

$$B_{x,j} = \frac{A_{x,j}^{N_x}(1-A_{x,j})}{1-A_{x,j}^{(N_x+1)}}$$

which is the blocking probability of an M/M/1/N_x-PS queue offered $A_{x,j}$ Erlangs of traffic.

There are two ways to generate an (n,j)-request. In the first case, an $(n-1, j-1)$-request overflows without any information exchange. In the second case, an $(n-1, \Omega)$-request, $\Omega \leq j-2$, exchanges its congestion estimate with a request in service, the request in service having a seniority of $j-1$. Combining the two, we obtain

$$\begin{aligned} v_{c,s,n,j} &= a_{c,s,n-1,j-1} B_{z,j-1} \\ &\quad + \tilde{a}_{c,s,n-1,j-2} \left(B_{z,j-1} - B_{z,j-2} \right), \end{aligned}$$

where $z = X_{c,s}(n-1)$. From this we obtain $\lambda_{c,s,n,j} = v_{c,s,n,j}\mu_{z,c}(1-P_{c,n,j})$ for $n \in \lfloor 1, j \rceil$, with base cases $\lambda_{c,s,0,0} = \frac{\lambda_c}{k_c}\left(1 - \frac{k_c}{X+Y}\right)$ and $\lambda_{c,s,0,j} = 0$ for $j \neq 0$.

The above equations can be solved iteratively for increasing j. Finally, the blocking probability for each content $c \in \mathbf{C}$ is

$$\hat{B}_c = \left(\frac{1}{\lambda_c}\right) \sum_{n=1}^{k_c} \sum_{s=0}^{k_c-1} \sum_{j=k_c}^{X} P_{c,n,j} v_{c,s,n,j},$$

and the overall blocking probability is obtained using (1).

V. Numerical Results

In all experiments, we obtain the blocking probability of the given system by discrete event simulation. In each run of the simulation, we have ten million arrival events, with enough runs to ensure a 95% confidence interval within 1% of the simulation mean, as computed using Student's t-distribution, with a minimum of five runs and a maximum of fifteen runs. Each data point in each plot shows the mean and standard deviation of twenty random content allocations; however, the standard deviations may be too small to be clearly visible. We compute the error between the approximation and the simulation in terms of the *logarithmic error*. Given an approximation result x and a simulation result y, the logarithmic error is $\log_{10} x - \log_{10} y$.

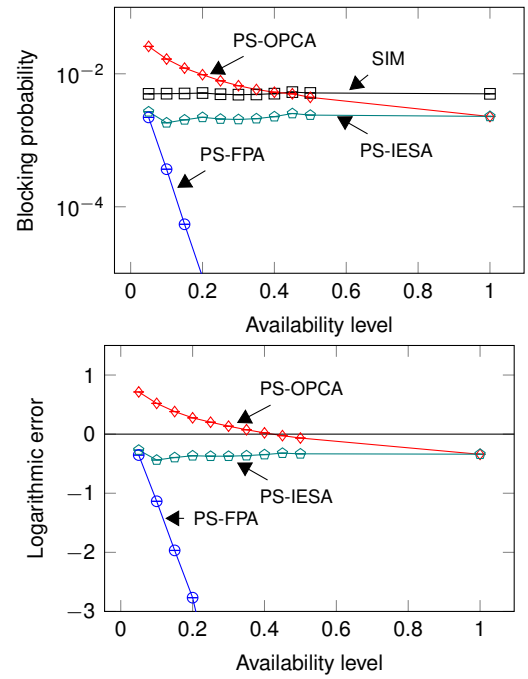

Fig. 1. Blocking probabilities and logarithmic errors for the systems described in Section V-A.

We shall define the *availability level* of a P2P system as

$$\frac{\text{total number of storage slots in system}}{\text{number of contents} \times \text{number of peers}};$$

thus an availability level of 1.0 implies that each peer holds a copy of every content in the system. Note that the availability level of a system is an average value and an availability level of x does not mean each content is hosted on roughly x of all peers; in this paper the number of copies of each content is made proportional to that content's popularity. Copies are distributed randomly and as evenly as possible such that all peers hold roughly the same number of contents.

A. Varying the per-peer storage capacity

In this configuration, there are $X = 200$ peers with five upload slots each, with the number of leeches Y set so that the blocking probability is approximately 0.5% in all cases. The arrival rate of all requests is $\sum_c \lambda_c = X + Y$ requests per hour. There are 2000 contents, with a Zipf(0.271) popularity distribution, i.e. $\lambda_c \propto c^{-0.271}$, a commonly used distribution in VoD systems [30]–[32]. The mean service time of each content is $(\mu_{x,c})^{-1} = 0.2$ hours. We examine different availability levels from 0.05 to 0.5. We shall also examine the limiting case of full availability. The results are shown in Fig. 1.

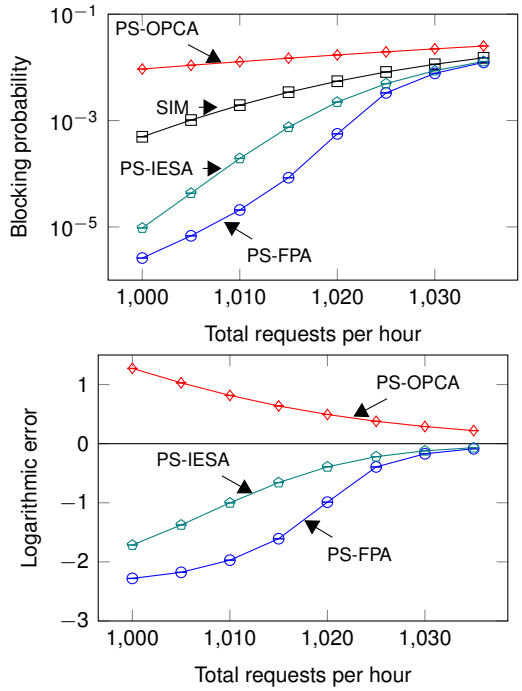

Fig. 2. Blocking probabilities and logarithmic errors for the systems described in Section V-B.

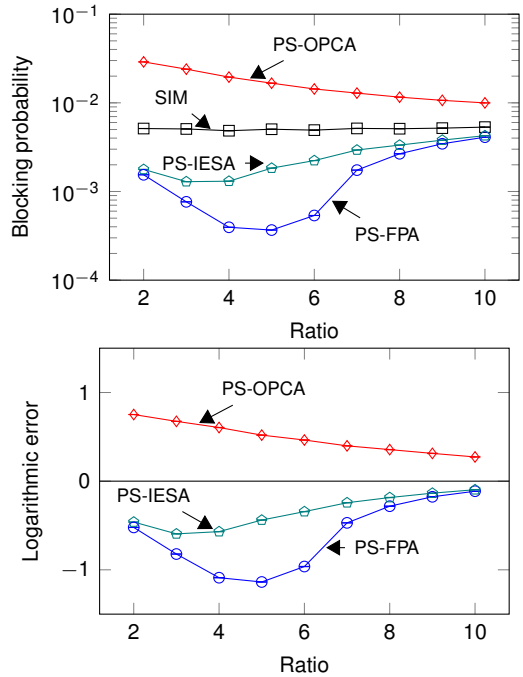

Fig. 3. Blocking probabilities and logarithmic errors for the systems described in Section V-C with constant blocking probability. "Ratio" denotes the ratio of the upload bandwidth to the video bitrate.

It can be seen that the accuracy of PS-FPA quickly deteriorates as the availability level increases, while the accuracy of PS-IESA is not very sensitive to the availability level in this case. Furthermore, PS-OPCA tends to overestimate blocking probability under low availability, converging with PS-IESA as the availability increases. This means that PS-OPCA transitions from overestimating for low availabilities to underestimating for high availabilities. From the bottom plot, we can see that while PS-OPCA is in general more accurate than PS-IESA for a wider range of availability levels, PS-IESA does well when the availability is low. Our results in this subsection match those in [16, Figs. 1b-c] for systems of FCFS queues.

B. Varying the offered load

In this configuration, there are $X = 200$ peers with five upload slots each, with the number of leeches Y varying from 800 to 835. The arrival rate of all requests is $\sum_c \lambda_c = X + Y$ requests per hour. There are 2000 contents, with a Zipf(0.271) popularity distribution and an availability level of 0.1. The mean service time of each content is $(\mu_{x,c})^{-1} = 0.2$ hours. The results are shown in Fig. 2.

It can be seen that all approximations converge towards the simulation value when the offered load and thus the blocking probability are high, however PS-FPA is the most inaccurate when the offered load is not high.

C. Varying the per-peer upload bandwidth

In this subsection, we will vary the upload bandwidth of each peer, from twice d, the minimum required rate, to ten times. We shall set the arrival rate of requests in two ways. In the first method, we shall keep a target blocking probability of 0.5%. In the second method, we shall set the total request arrival rate proportional to the upload bandwidth.

1) Constant blocking probability: In this configuration, there are $X = 200$ peers with n upload slots each, n varying from 2 to 10. The number of leeches Y is set so that the blocking probably is roughly 0.5% in all cases. The arrival rate of all requests is $\sum_c \lambda_c = X + Y$ requests per hour. There are 2000 contents, with a Zipf(0.271) popularity distribution and an availability level of 0.1. The mean service time of each content is $(\mu_{x,c})^{-1} = n^{-1}$ hours. The results are shown in Fig. 3. While all three approximations converge to the simulation result as the upload bandwidth of each peer increases, PS-FPA and PS-IESA also become more accurate when bandwidth is low, for example in the case with two upload slots per peer. Also, PS-OPCA is less accurate than PS-FPA in some cases, demonstrating that PS-OPCA is not robust.

2) Proportional arrival rate: In this configuration, there are $X = 200$ peers with n upload slots each, n varying from 2 to 10. The number of leeches Y is set such that $X + Y = 200n$. The arrival rate of all requests is $\sum_c \lambda_c = X + Y$ requests per hour. There are 2000 contents, with a Zipf(0.271) popularity distribution and an availability level of 0.1. The mean service time of each content is $(\mu_{x,c})^{-1} = n^{-1}$ hours. The results are shown in Fig. 4.

Based on the results, we see that all three approximations become less accurate as n increases, with PS-FPA the most

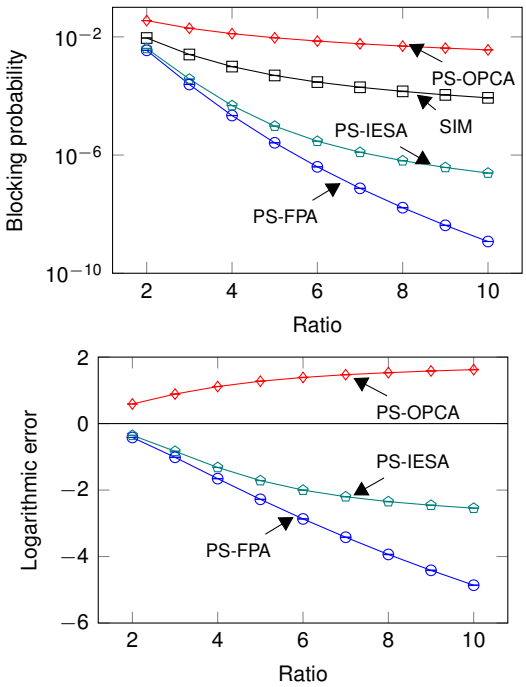

Fig. 4. Blocking probabilities and logarithmic errors for the systems described in Section V-C with proportional arrival rate. "Ratio" denotes the ratio of the upload bandwidth to the video bitrate.

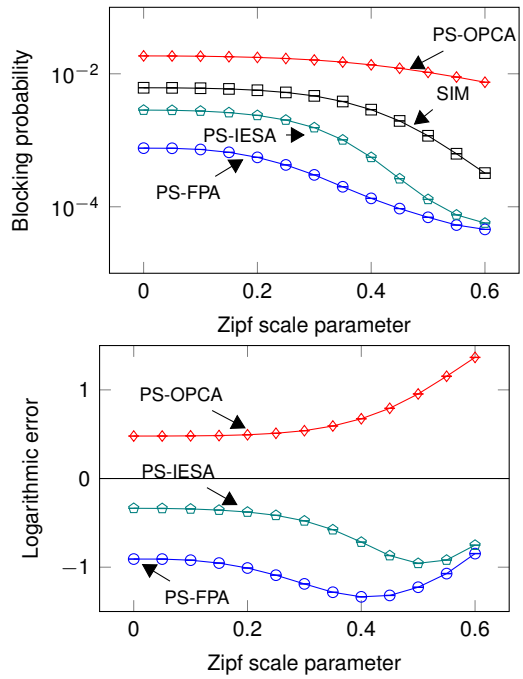

Fig. 5. Blocking probabilities and logarithmic errors for the systems described in Section V-D with an availability ratio of 0.1.

inaccurate. This is in contrast with Fig. 3, in which PS-FPA and PS-OPCA are accurate for large n. This reflects the effect of the arrival rate on the accuracy of each approximation as shown in Fig. 2.

D. Varying the content popularity distribution

In this configuration, there are $X = 200$ peers with five upload slots each, with the number of leeches $Y = 819$ so that the blocking probability is roughly 0.5% when the Zipf scale parameter is set to 0.271. The arrival rate of all requests is $\sum_c \lambda_c = X + Y = 1019$ requests per hour. There are 2000 contents, with availability levels of 0.1, 0.2, and 0.6, and a Zipfian popularity distribution, with a scale parameter of 0 to 0.6. The mean service time of each content is $(\mu_{x,c})^{-1} = 0.2$ hours. The results are shown in figures 5–7.

Based on the results, we see that as the content popularity distribution becomes more skewed, the blocking probability of the system decreases. Also, PS-FPA is shown to perform poorly except for low availability levels, as also demonstrated in Section V-A, while PS-OPCA and PS-IESA both perform quite well. However, Fig. 5 shows a case where PS-OPCA is less accurate than PS-FPA, once again demonstrating that PS-OPCA is not robust.

E. Blocking probability of individual contents

In this configuration, there are $X = 200$ peers with five upload slots each, with the number of leeches Y set so that the blocking probability is roughly 0.5% when the Zipf scale parameter is set to 0.271. The arrival rate of all requests is $\sum_c \lambda_c = X + Y = 1019$ requests per hour. There are 2000 contents, with a Zipf(0.271) distribution. The mean service time of each content is $(\mu_{x,c})^{-1} = 0.2$ hours.

In Fig. 8, we group the contents by the number of copies of each content and show the mean blocking probability for each group. The top graph shows an availability level of 0.1 and the bottom graph 0.4. The results show that PS-IESA performs reasonably well at estimating the blocking probability of individual contents, whereas PS-OPCA does not perform so well in the 0.1 availability case, especially for the most popular contents. On the other hand, PS-FPA grossly underestimates the blocking probability in both cases, with the worse performance for the 0.4 availability case, consistent with the results of the mean blocking probability in Fig. 1 (note the different y-scale for the two graphs).

F. Sensitivity to service time distribution

In this configuration, there are $X = 200$ peers with five upload slots each, and $Y = 819$ leeches (which gives a blocking probability of roughly 0.5% for exponentially distributed service times). The arrival rate of all requests is $\sum_c \lambda_c = X + Y = 1019$ requests per hour. There are 2000 contents, with a Zipf(0.271) distribution and an availability index of 0.1. The mean service time of each content is $(\mu_{x,c})^{-1} = 0.2$ hours.

For each system, we calculate via simulation the request blocking probability of 20 random replica assignments for several service time distributions: Dirac delta (standard deviation of zero), exponential (standard deviation of 0.2), and lognormal with standard deviations of 0.04 and 0.09. The results are shown in Fig. 9. Box plot whiskers represent the

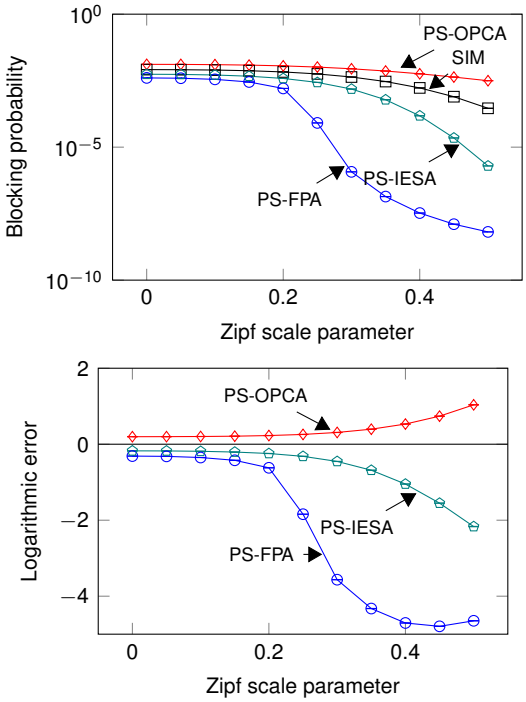

Fig. 6. Blocking probabilities and logarithmic errors for the systems described in Section V-D with an availability ratio of 0.2.

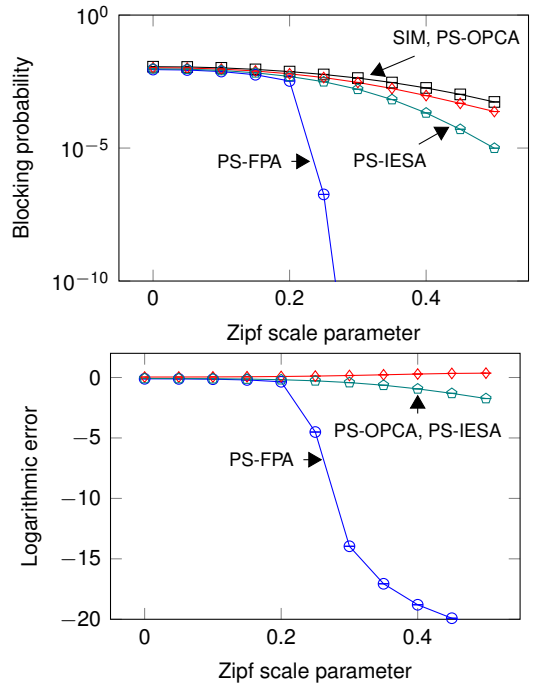

Fig. 7. Blocking probabilities and logarithmic errors for the systems described in Section V-D with an availability ratio of 0.6.

TABLE I
RUNNING TIMES IN SECONDS FOR SECTION V-A

Availability level		SIM	PS-FPA	PS-OPCA	PS-IESA
0.1	(Mean)	4906	1.4897	0.0218	16.75
	(St. dev.)	901.9	0.01672	0.0079	0.039
0.2	(Mean)	6077	10.647	0.0891	37.47
	(St. dev.)	1805	0.1752	0.0077	0.705
0.6	(Mean)	8071	88.582	0.7801	128.78
	(St. dev.)	2538	0.0696	0.0179	0.066

smallest and largest values within 1.5 times the inter-quartile range. The results show that the model considered in this paper is not very sensitive to the service time distribution and that assuming an exponential distribution is adequate for approximation of the general case.

G. Running time comparison

The running times for the configurations in Section V-A were analyzed for availability levels of 0.1, 0.2, and 0.6. For each availability level, ten random content allocations were chosen. The mean running time in seconds for each configuration is given in Table I.

The above running times are for an Intel® i7-3770 processor with 8 GB of RAM. The results show that the speed of PS-IESA is comparable with PS-FPA, and roughly two orders of magnitude faster than simulation.

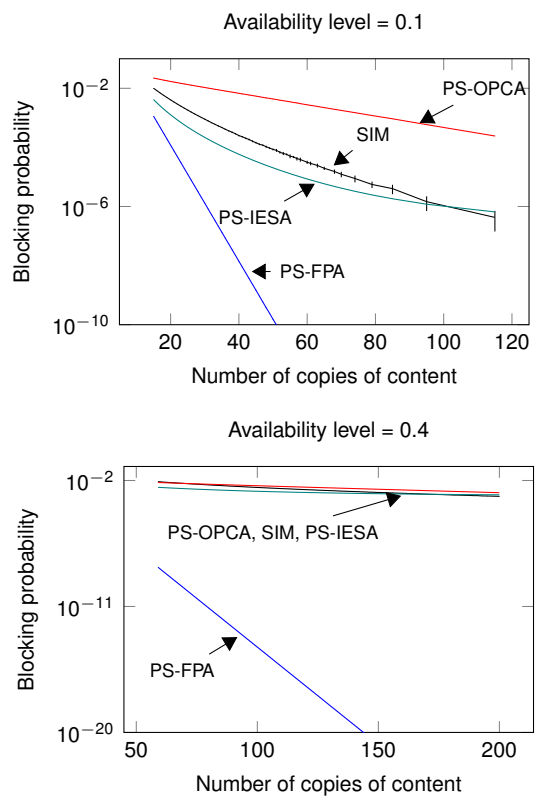

Fig. 8. Blocking probabilities for the systems described in Section V-E.

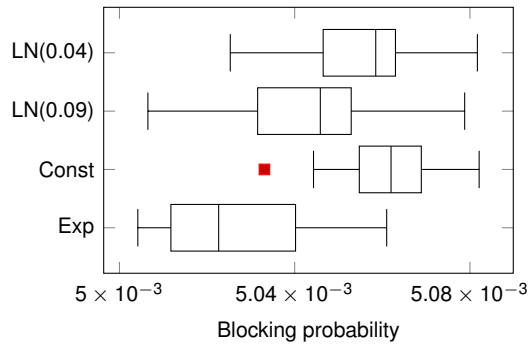

Fig. 9. Blocking probability distributions for the service time distributions described in Section V-F. LN(x) means lognormal with a standard deviation of x.

VI. Concluding Remarks

Considering an overflow loss system of multiple finite-buffer PS queues, we have shown that such a model is nearly insensitive to the service time distribution. We have compared three approximations for evaluation of the blocking probability, namely PS-FPA, PS-OPCA and PS-IESA. Both PS-OPCA and PS-IESA show improvement over PS-FPA in terms of accuracy, and are on par with or faster than PS-FPA in terms of computation time. However, PS-OPCA is not robust: there are certain cases where PS-OPCA can be less accurate than PS-FPA, especially when the availability level is low. In contrast, PS-IESA is not very sensitive to the availability level. The results also demonstrate that PS-IESA is more robust compared to PS-FPA and PS-OPCA for estimating the blocking probability of individual contents.

Although PS-IESA does much to approximate the dependency effects between different nodes in the system, further reduction of the approximation error may be possible by taking higher moments of the overflow traffic into account. However, unlike systems of FCFS queues for which established moment matching techniques exist, more research is required to study the overflow processes of systems with multiple PS queues.

References

[1] A. Lotze, "History and development of grading theory," in *Proc. ITC 5*, 1967.

[2] J. Guo, E. W. M. Wong, S. Chan, P. Taylor, M. Zukerman, and K.-S. Tang, "Performance analysis of resource selection schemes for a large scale video-on-demand system," *IEEE Trans. Multimedia*, vol. 10, no. 1, pp. 153–159, Jan. 2008.

[3] J. M. Chaiken and E. Ignall, "An extension of Erlang's formulas which distinguishes individual servers," *Journal of Applied Probability*, vol. 9, no. 1, pp. 192–197, Mar. 1972.

[4] M. Restrepo, S. G. Henderson, and H. Topaloglu, "Erlang loss models for the static deployment of ambulances," *Health Care Management Science*, vol. 12, no. 1, pp. 67–79, Mar. 2009.

[5] G. Koole and J. Talim, "Exponential approximation of multi-skill call centers architecture," in *Proc. QNETs '00*, Jul. 2000.

[6] N. Litvak, M. van Rijsbergen, R. J. Boucherie, and M. van Houdenhoven, "Managing the overflow of intensive care patients," *European Journal of Operational Research*, vol. 185, no. 3, pp. 998–1010, Mar. 2008.

[7] J. P. Jarvis, "Approximating the equilibrium behavior of multi-server loss systems," *Management Science*, vol. 31, no. 2, pp. 235–239, Feb. 1985.

[8] D. C. Walden and T. Van Vleck, Eds., *The Compatible Time Sharing System (1961-1973): Fiftieth Anniversary Commemorative Overview*. IEEE Computer Society, 2011.

[9] J. Cao, M. Andersson, C. Nyberg, and M. Kihl, "Web server performance modeling using an M/G/1/K* PS queue," in *Proc. ICT '03*, vol. 2, 2003, pp. 1501–1506.

[10] J. P. Muñoz-Gea, S. Traverso, and E. Leonardi, "Modeling and evaluation of multisource streaming strategies in P2P VoD systems," *IEEE Trans. Consum. Electron.*, vol. 58, no. 4, pp. 1202–1210, Nov. 2012.

[11] C. Knessl, "On finite capacity processor-shared queues," *SIAM Journal on Applied Mathematics*, vol. 50, no. 1, pp. 264–287, Feb. 1990.

[12] M. Zukerman. Introduction to queueing theory and stochastic teletraffic models. [Online]. Available: http://www.ee.cityu.edu.hk/~zukerman/classnotes.pdf

[13] R. B. Cooper and S. S. Katz, "Analysis of alternate routing networks with account taken of nonrandomness of overflow traffic," Bell Telephone Lab., Memo. 64-3122-2, Nov. 1964.

[14] F. P. Kelly, "Blocking probabilities in large circuit-switched networks," *Advances in Applied Probability*, vol. 18, no. 2, pp. 473–505, Jun. 1986.

[15] E. W. M. Wong, A. Zalesky, Z. Rosberg, and M. Zukerman, "A new method for approximating blocking probability in overflow loss networks," *Computer Networks*, vol. 51, no. 11, pp. 2958–2975, Aug. 2007.

[16] E. W. M. Wong, J. Guo, B. Moran, and M. Zukerman, "Information exchange surrogates for approximation of blocking probabilities in overflow loss systems," in *Proc. ITC 25*, Sep. 2013.

[17] L. Kleinrock, "Time-shared systems: A theoretical treatment," *Journal of the ACM*, vol. 14, no. 2, pp. 242–261, Apr. 1967.

[18] T. Bonald and A. Proutiere, "Insensitivity in processor-sharing networks," *Performance Evaluation*, vol. 49, no. 1–4, pp. 193–209, Sep. 2002.

[19] G. Yamazaki and H. Sakasegawa, "An optimal design problem for limited processor sharing systems," *Management Science*, vol. 33, no. 8, pp. 1010–1019, Aug. 1987.

[20] Y.-C. Huang, K.-T. Ko, and M. Zukerman, "A generalized quasi-stationary approximation for analysis of an integrated service system," *IEEE Commun. Lett.*, vol. 16, no. 11, pp. 1884–1887, Nov. 2012.

[21] V. B. Iversen, *Teletraffic Engineering Handbook*. Tech. Univ. Denmark, 2010.

[22] A. Girard, *Routing and Dimensioning in Circuit-Switched Networks*, ser. Addison-Wesley Series in Electrical And Computer Engineering. Addison-Wesley, Jun. 1990.

[23] R. C. McNamara, "Applications of spanning trees to continuous-time Markov processes, with emphasis on loss systems," Ph.D. dissertation, The University of Colorado, 2004.

[24] R. I. Wilkinson, "Theories for toll traffic engineering in the U.S.A." *Bell System Tech. Journal*, vol. 35, no. 2, pp. 421–514, Mar. 1956.

[25] P. Chevalier and N. Tabordon, "Overflow analysis and cross-trained servers," *International Journal of Production Economics*, vol. 85, no. 1, pp. 47–60, Jul. 2003.

[26] G. J. Franx, G. Koole, and A. Pot, "Approximating multi-skill blocking systems by hyperexponential decomposition," *Performance Evaluation*, vol. 63, no. 8, pp. 799–824, Aug. 2006.

[27] Q. Huang, K.-T. Ko, and V. B. Iversen, "Approximation of loss calculation for hierarchical networks with multiservice overflows," *IEEE Trans. Commun.*, vol. 56, no. 3, pp. 466–473, Mar. 2008.

[28] E. W. M. Wong, M. Y. M. Chiu, M. Zukerman, Z. Rosberg, S. Chan, and A. Zalesky, "A novel method for modeling and analysis of distributed video on demand systems," in *Proc. IEEE ICC '05*, vol. 1, May 2005, pp. 88–92.

[29] Y. Qiao, D. Lu, F. E. Bustamante, and P. A. Dinda, "Looking at the server side of peer-to-peer systems," in *Proc. ACM LCR '04*, Oct. 2004, pp. 1–8.

[30] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," in *Proc. ACM MULTIMEDIA '94*, 1994, pp. 15–23.

[31] J. Guo, E. W. M. Wong, S. Chan, P. Taylor, M. Zukerman, and K.-S. Tang, "Combination load balancing for video-on-demand systems," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 7, pp. 937–948, Jul. 2008.

[32] M. Ryu, H. Kim, and U. Ramachandran, "Why are state-of-the-art flash-based multi-tiered storage systems performing poorly for HTTP video streaming?" in *Proc. ACM NOSSDAV '12*, Jun. 2012, pp. 3–8.