THRESHOLD-BASED ADMISSION CONTROL FOR MULTI-CLASS VIDEO-ON-DEMAND SYSTEMS

Padmavathi Mundur
Arun Sood
Robert Simon
Department of Computer Science
George Mason University
Fairfax, VA 22030

Abstract

The next generation of distributed digital Video-on-Demand (VoD) systems will use admission control tests to ensure that users receive predictable and guaranteed service. If the system cannot support a new request then that request blocks. *This paper presents an analytical model to evaluate unified admission control strategies for distributed VoD systems with multiple video classes. We prove that there exists a computationally efficient technique to determine blocking probabilities for newly arriving requests. Through numerical analysis we show the effect of different admission control policies on overall system blocking rates.*

Keywords: Distributed Multimedia, Video-on-Demand Systems, Threshold-based Analysis, Admission Control, Blocking Probability.

1 INTRODUCTION

Distributed Video-on-Demand systems are expected to be one of the most important services supported by the next generation of high-speed networks, video servers, and distributed multimedia systems. Upon request, VoD systems deliver high quality digitized video directly to users. VoD systems achieve this performance and flexibility by the tight coordination of all system components, from file servers to networks to end-user presentation devices. Such coordination is needed because of the stringent real-time performance and synchronization requirements of video applications and the enormous amount of data present in video objects.

Previous research into designing distributed VoD systems include media server design [7] [4], disk storage hierarchies and retrieval policies [2] [3], and communication bandwidth management policies such as statistical multiplexing [12]. Previous work on modeling admission control for VoD systems has adopted a predictive approach based on observed behavior [10], modeled interactive VoD systems with browsing [6], or analyzed blocking probabilities under the assumption of a hierarchical storage system, based in part upon movie popularity [11].

The above mentioned work typically focuses on specific architectural aspects of VoD design. In this paper we discuss a unified model for distributed VoD systems that incorporates all of the underlying subsystems. The successful development of this model simplifies admission control and provides a basis for optimizing VoD resources. Our admission control model groups newly arriving requests into *priority classes*. A threshold-based admission control procedure is used to decide whether to accept or reject new requests, depending upon the class of the request and current system conditions. We show that the blocking probability for this highly flexible admission control procedure can be efficiently determined using a recursive computational technique. Our approach is similar to methods developed to calculate the performance of multi-rate circuit switched communication networks [9]. We present a numerical performance analysis for a representative VoD system to show that the method presented in this paper can be used to judge the effectiveness of a variety of different threshold-based admission control procedures.

The rest of the paper is organized as follows: Section 2 describes the VoD architecture. Section 3 explains the admission control policies and in particular, threshold type admission control policies. In Section 4, we develop our analytical model and the recursive algorithm for computing blocking probabilities. Numerical results are presented in Section 5. We conclude the paper with a note on future work in Section 6.

2 SYSTEM ARCHITECTURE

Figure 1 shows the target VoD system architecture. The main components of our VoD system architecture are video file servers (VFSs), archival servers (ASs) with tertiary storage devices, user workstations (WSs), request handlers, and a high speed network. Each VFS consists of a secondary storage subsystem of magnetic disk drives such as RAID devices, a large memory buffer, and a network interface. The VFSs are connected to one or more dedicated ASs through the network.

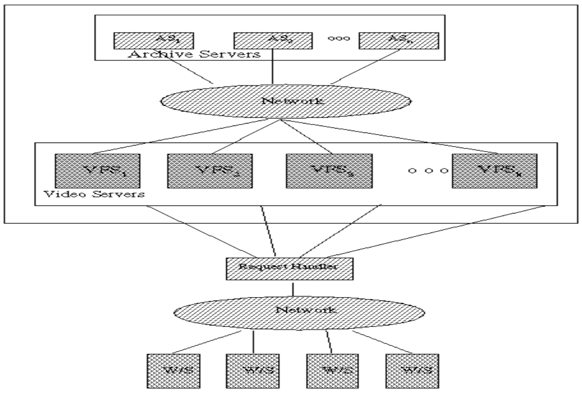

Figure 1. VoD System Architecture

The storage requirements of a VoD system are extremely high; ASs are needed because of the high cost of hard disk storage which precludes storing all movies on the video servers. For example, a 110 minute pre-encoded MPEG movie at 1.5 megabits/sec delivery rate will require hard disk storage of 1.24 Gigabytes/movie, while a delivery rate of 8 megabits/sec will require 6.6 Gigabytes/movie.

The tertiary storage devices available to each AS, such as magnetic tapes and optical disks provide a cost effective way of storing terabytes of data for a VoD system. However, the long access times of the tertiary devices preclude serving users directly from ASs. All user interaction, therefore must go through the VFSs. The movie collection should be judiciously distributed between the secondary and tertiary devices based on a measurable criterion, such as popularity. More popular movies will reside on the video server where as unpopular ones are loaded from the archive servers as needed. We classify movies into classes based on their popularity. If the VoD system has two classes, popular (class 1) and unpopular (class 2), class 1 movies reside on VFSs and class 2 movies reside on ASs.

Requests for new movies are sent to the request handler, which runs an admission control procedure. We describe that procedure in the next section.

3 ADMISSION CONTROL

When a request is received, the request handler determines whether the requested movie resides on the video server disk. If it does, the request is accepted subject to the available capacity on the server and the network. If the requested movie resides on the archive server, the request handler first checks the availability of network capacity. If there is enough network capacity for data transfer, the request handler checks for video server availability and the necessary network capacity to the user before accepting the request. The admission control procedure is illustrated in Figure 2.

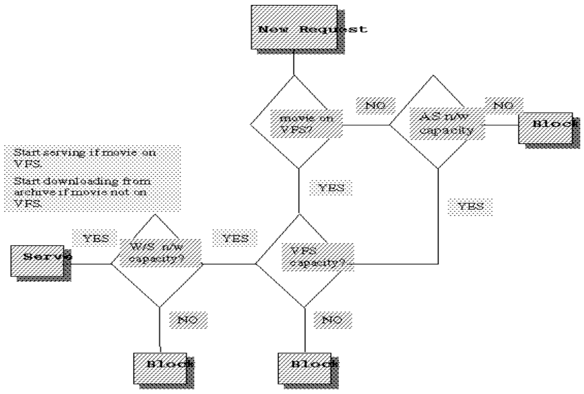

Figure 2. Admission Control Procedure

An admission control *policy* determines whether an arriving request for a certain class of movie should be admitted given the current state of the VoD system. Without loss of generality, suppose there are two classes of requests and the vector (n_1, n_2) describes the state of the system in terms of the number of class 1 requests, n_1 and the number of class 2 requests, n_2. The state of the VoD system is describable as a state space constrained by capacity. A capacity constraint is defined by $\sum_{k=1}^{2} n_k \leq C$ where C is the maximum number of simultaneous requests that a VFS can handle. The state space can further be constrained by other restrictions, such as when the admission control policy favors one class more than the other by enforcing a threshold on one of the classes.

We define three types of class-based admission control: *complete partitioning*, *complete sharing*, and *threshold type*. A complete partitioning policy means that server capacity is completely partitioned and every class of request has access only to a dedicated set of resources. A complete sharing policy is one where requests of all classes are always accepted if there is available capacity. A threshold type policy allows requests of classes up to the thresholds assigned to those classes. A threshold *class-2* policy limits the class 2 requests up to a specified threshold whereas always accepting class 1 requests if there is available capacity. Threshold type policies are useful because it may not be desirable to share all of the server capacity equally among classes. For instance, requests for certain classes of movies might bring in more revenue.

Given a state space of the system based on the admission control policy, we can identify those states where an arrival will be blocked either because of the capacity constraint or the threshold. We compute blocking probability using this information. An example follows.

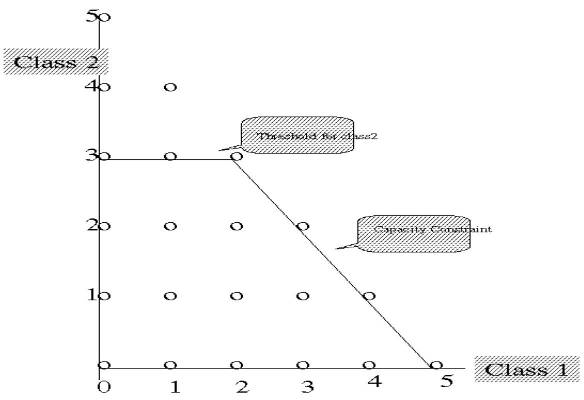

Figure 3. State Space for Threshold Class-2 Policy

Figure 3 illustrates the situation for two classes of requests and maximum capacity of five requests at the video server. The policy illustrated in the figure is to allocate no more than 60% of the resources at the video server to class 2 requests. Only a maximum of three class 2 requests are allowed in the system. There are no restrictions on class 1 requests other than the capacity of the system. Because of capacity and policy restrictions, arrivals of the pertinent classes are rejected when the VoD system is in one of the blocking states.

The state space, Ω, is defined by

$\Omega = \{(n_1, n_2) : 0 \le n_1 \le 5$ and $0 \le n_2 \le 3$ and $\sum_{k=1}^{2} n_k \le 5\}$

Blocking states for class 1 requests, $\Omega_1 = \{(2,3)(3,2)(4,1)(5,0)\}$ and blocking states for class 2 requests, $\Omega_2 = \{(0,3)(1,3)(2,3)(3,2)(4,1)(5,0)\}$.

4 VOD ANALYTICAL MODEL

To begin the discussion of our analytical model, we notice that the VoD architecture from Section 2 can be described as a a single video server connected over network to a pool of archive servers. The justification for the simplified system comes from assuming a well balanced VoD system in which each video server is equally likely to serve the incoming request. With only two classes of requests, with class 1 movies residing on all video servers and class 2 movies in the archive, any of the video servers can serve both class 1 and class 2 requests. The single video server is now being viewed as an accumulation of individual video servers with a combined maximum capacity of handling C simultaneous requests. The single video server acts as C virtual servers, each of which is serving a request. The VoD system may now be modeled as a single node, multi-server, 2-class queueing network model. We assume that the requests arrive according to a Poisson distribution and the the service times are exponentially distributed. Based on a birth-death model, the state of the system is discribable by a Markov chain with state space Ω. By using the Markov chain model, the three types of admission control policies presented in Section 3 fall into a class of policies known as *coordinate convex* [1]. Arrivals or departures are never blocked, except for the boundary states, where any further arrivals are blocked. The advantage of coordinate convex policies is that they give rise to product form solutions to the equilibrium state probabilities using which all other performance measures can be determined.

We assume two classes of movies, *popular* and *unpopular*. All movies of the unpopular class reside in the archive servers and will be transferred to the video server over the network on demand. The network element between the archives and the video server and the workstations is ignored in this model. The archive servers are assumed to have infinite capacity; we do not assume any blocking at that level. All incoming requests are blocked if the video server capacity is not available or if there are requests up to the threshold for that class. The admission control policy used is threshold type. Blocking probability for class 1 and class 2 requests may now be computed using the approach below.

Model Parameters and Definitions:

- λ_1 - arrival rate of request for movies on disk at the video server (class 1 - popular)
- λ_2 - arrival rate of request for movies in the archives (class 2 - unpopular)
- $1/\mu_1$ - service time for popular movies
- $1/\mu_2$ - service time for unpopular movies (includes additional loading time from the archive over the network to the video server)
- C - Maximum number of requests that the video server can handle simultaneously
- l_k - threshold for class k, $k = 1, 2$
- P_{b_1} - blocking probability for class 1 requests
- P_{b_2} - blocking probability for class 2 requests
- $P(n_1, n_2)$ - probability of n_1 class-1 requests and n_2 class-2 requests, $\sum_{k=1}^{2} n_k \le C$

- Ω - set of admissible states under a given policy

  for a system with threshold on both classes,

  $\Omega = \{(n_1, n_2) : 0 \leq n_1 \leq l_1$ and $0 \leq n_2 \leq l_2$ and $\sum_{k=1}^{2} n_k \leq C\}$

  for a threshold class-2 system,

  $\Omega = \{(n_1, n_2) : 0 \leq n_1 \leq C$ and $0 \leq n_2 \leq l_2$ and $\sum_{k=1}^{2} n_k \leq C\}$

- Ω_1 - set of blocking states for class 1 requests

- Ω_2 - set of blocking states for class 2 requests

As shown in [1] and [5], the equilibrium probability, $P(n_1, n_2)$, of finding the server in the state (n_1, n_2) has the following product form:

$$P(n_1, n_2) = \frac{1}{G} \prod_{k=1}^{2} \rho_k^{n_k}/n_k!, \quad \text{all } (n_1, n_2) \in \Omega \tag{1}$$

where

$$G = \sum_{(n_1,n_2)\in\Omega} \prod_{k=1}^{2} \rho_k^{n_k}/n_k!$$

and

$$\rho_k = \lambda_k/\mu_k$$

The blocking probability, P_{b_k}, is given by

$$P_{b_k} = \sum_{(n_1,n_2)\in\Omega_k} P(n_1, n_2) = G(\Omega_k)/G \tag{2}$$

where
$\Omega_k = \{(n_1, n_2) | (n_1, n_2) \in \Omega,\ (n_1, n_2) + \overline{e_k} \notin \Omega\}$ where $\overline{e_k}$ is a vector equal to either $(0, 1)$ or $(1, 0)$, where the 1 is in the Kth position. Ω_1 and Ω_2 are the set of blocking states for class 1 and class 2 requests respectively.

The cardinality of Ω grows roughly as C^k which makes exhaustive enumeration techniques computationally too expensive. Hardware limitations rule out computing G for even moderate sized problems. Some previous studies in finding computationally tractable solutions for computing G were done in the context of multi-rate circuit switched networks. For instance, Kaufman [5] and Roberts[8] independently developed recursive relations to compute the G factor for complete sharing policies. Threshold type policies introduce additional blocking states that should be included in the computation of blocking probabilities. Also, the state space Ω is now truncated according to the threshold values. Kaufman/Roberts recursion was generalized to compute blocking probability for a multi-rate tree networks in [9]. Here, we develop a recursion for single node, multi-server, 2 class threshold type system that is similar to the recursion presented in [9]. The recursion is applicable to either or both classes having a threshold restriction.

The basic idea behind the recursion is summing $P(n_1, n_2)$ over Ω recursively along disjoint parallel planes on the diagonal, as shown in Figure 4. From this summing technique, we get a distribution in terms of the total number of servers being used. Denote the total number of servers used as $I = \sum_{k=1}^{2} n_k$, and define a state space,

$$\Theta(i) = \{(n_1, n_2) \in \Omega : \sum_{k=1}^{2} n_k = i\}$$

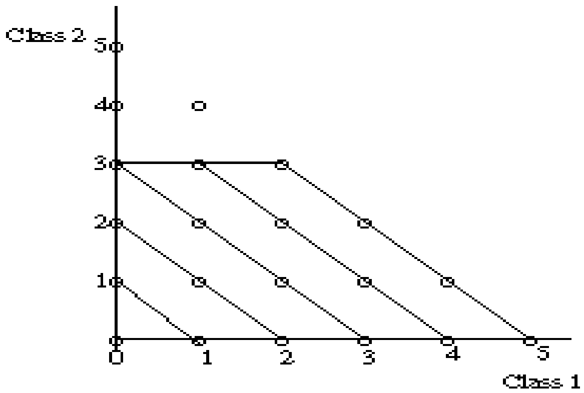

Figure 4. Planar States for Distribution q_i

The distribution q_i, indicates the number of servers being used.

$$q_i = P(I = i) = \sum_{(n_1,n_2)\in\Theta(i)} P(n_1, n_2), \quad i = 0, 1, \ldots, C \tag{3}$$

Define,

$$G(i) = \begin{cases} \sum_{(n_1,n_2)\in\Theta(i)} \prod_{k=1}^{2} \rho_k^{n_k}/n_k! & i = 0, 1, \ldots, C \\ 0 & \text{otherwise} \end{cases}$$

Multiplying (3) with G and substituting for $P(n_1, n_2)$ from (1) we get the expression above for $G(i)$,

$$G(i) = q_i G \tag{4}$$

To handle threshold type policies, we define a threshold value which is less than C to limit the number of requests for those classes. Define $l_k < C$ as the threshold for class k requests.
Define for $k = 1, 2$,

$$B_k(i) = \begin{cases} P(I = i, n_k = l_k)G & \text{if } l_k \leq i \leq C - 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$B_k(i)$ is the unnormalized probability of being in one of the additional blocking states due to the threshold. If no threshold is defined for a particular class, $B_k(i)$ for that class is 0.
Blocking probability,

$$P_{b_k} = P(I = C) + P(I \leq C-1, n_k = l_k)$$

$$P_{b_k} = \frac{G(C) + \sum_{i=l_k}^{C-1} B_k(i)}{\sum_{i=0}^{C} G(i)} \tag{6}$$

To compute blocking probability, we need only calculate the G factor and $B_k(i)$, $i = l_k, \ldots, C-1$, for $k = 1, 2$. We use the following recursion to accomplish that.

Theorem 1

$$G(i) = \frac{1}{i} \sum_{k=1}^{2} \rho_k [G(i-1) - B_k(i-1)], \quad i = 1, \ldots, C \tag{7}$$

The proof of Theorem(1) is contained in the Appendix.

We now consider computing $B_k(i)$ for a threshold class-2 system. A threshold class-2 system allows class-2 requests up to a threshold and always accepts class 1 requests if capacity is available. In our model, it is reasonable to limit class 2 (unpopular) requests and allow more of class 1 (popular) requests to be accepted into the system. Therefore, we do not assume a threshold restriction for class 1. Since we do not have a threshold restriction for class 1, by definition we have, $B_1(i) = 0$.
Define,

$$r(i) = \frac{1}{i}\rho_1[r(i-1)], \quad i = 0, \ldots, C-1-l_2 \text{ and } r(0) = 1$$

and compute

$$B_2(i) = (\rho_2^{l_2}/l_2!)\, r(i-l_2), \quad i = l_2, \ldots, C-1 \tag{8}$$

Stripping off the recursion in (8) we have

$$B_2(i) = (\rho_2^{l_2}/l_2!)(\rho_1^{(i-l_2)}/(i-l_2)!)$$

which can be proved from equilibrium conditions, the product form expression in (1), and the definition of $B_2(i)$.

5 NUMERICAL RESULTS

By using the results of the previous section a number of different types of admission policies may be analyzed. In this section we present numerical results for a hypothetical VoD system with two classes of requests, class 1 (popular) and class 2 (unpopular), and an overall capacity of $C = 500$. We assume that there are no threshold restrictions on class 1 except the capacity of the system, but there is a threshold on class 2. The recursion $r(i) = \frac{1}{i}\rho_1[r(i-1)]$ is used to compute all $r(i)$s and (8) is used to compute $B_2(i)$. Once all $B_2(i)$ are determined, we use Theorem(1) to compute $G(i)$.

Blocking probability for both classes are calculated as follows:

$$P_{b_1} = \frac{G(C)}{\sum_{i=0}^{C} G(i)} \tag{9}$$

$$P_{b_2} = \frac{G(C) + \sum_{i=l_2}^{C-1} B_2(i)}{\sum_{i=0}^{C} G(i)} \tag{10}$$

A computer program was written to compute blocking probabilities using (9) and (10). We used standard 15 digit double precision using GNU C compiler on a Sparc workstation to compute our results. The results of the first numerical analysis shown in Figure 5, shows the effect of varying the admission threshold for class 2 while ρ_1 and ρ_2 are held constant. In this analysis the capacity C is set to 500 (simultaneous requests), the threshold for class 2 ranges from 50 to 140, ρ_1 is set to 550 and ρ_2 is set to 150. As the threshold for class 2 approaches the value of ρ_2 the blocking probability for class 1 increases as a result of increased sharing of resources. Eventually we have a situation similar to complete sharing policy, where each class has the same blocking probability.

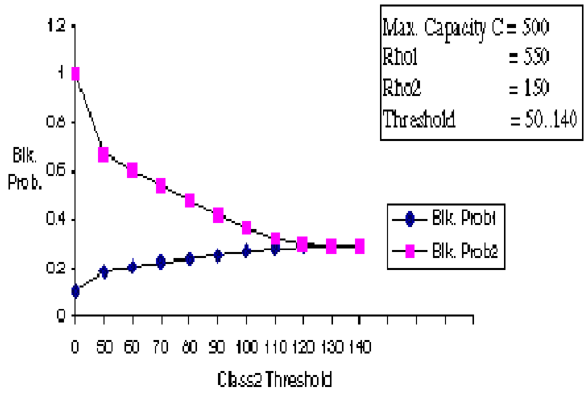

Figure 5. Blocking Probabilities vs. Class 2 Threshold

Figure 6 shows the effect of increasing the arrival rate of class 2 while maintaining a fixed threshold. The blocking probability rapidly increases for class 2, while the blocking rate for class 1 barely varies.
Figure 7 shows the effect of increasing the arrival rate for

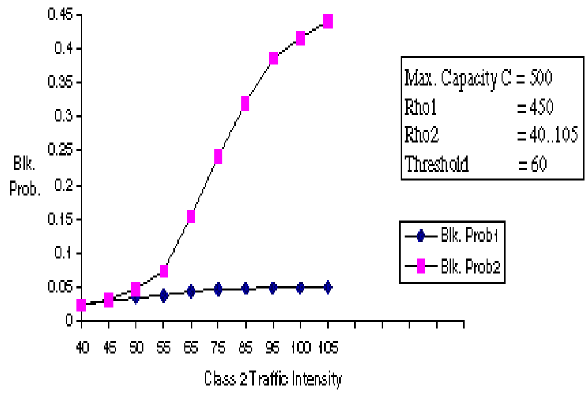

Figure 6. Blocking Probabilities vs. Class 2 Traffic Intensity

class 1, while maintaining a fixed threshold and arrival rate for class 2. Blocking for both classes remains constant until the value of ρ_1 approaches capacity, at which point, blocking for both classes increases rapidly, as expected.

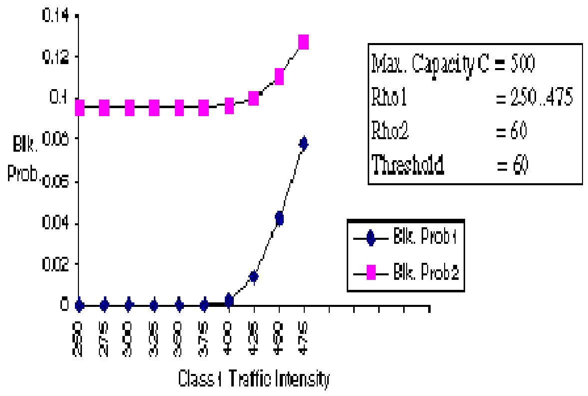

Figure 7. Blocking Probabilities vs. Class 1 Traffic Intensity

6 CONCLUSION AND FUTURE WORK

In this paper, we have developed an analytical model for evaluating admission control policies for a VoD system. The model takes into consideration complex issues in a VoD system such as different classes of requests each with its own arrival and service rates. The recursive algorithm has given us an efficient way of evaluating performance measures; blocking probability being the most important. Although we have modeled a two-class system here, the model can easily be extended to K classes. We showed through numerical analysis that a variety of admission control procedures can be analyzed.

We are currently working on three different areas. First, we will increase the architectural complexity by including individual video servers each with a different capacity and a different repertoire of movies. This relaxes the assumption in Section 4 about a well balanced system with ideal data distribution, thus making load balancing and data management issues important. Second, we will incorporate a more sophisticated network element. Third, we will develop a precise optimization model for different optimization functions and classes of requests.

References

[1] J. M. Aein. A multi-user-class, blocked-calls-cleared demand access mode. *IEEE Transactions on Communications*, COM-26(3):378–385, March 1978.

[2] D. Brubeck and L. Rowe. Hierarchical storage management in a distributed vod system. *IEEE Multimedia*, pages 37–47, Fall 1996.

[3] C. Federighi and L. Rowe. A distributed hierarchical storage manager for a video-on-demand system. In *SPIE Proceedings*, pages 185–197, 1994.

[4] D. Gemmell, H. Vin, D. Kandlur, P. V. Rangan, and L. Rowe. Multimedia storage servers: A tutorial. *Computer*, 28(5):40–49, May 1995.

[5] J. S. Kaufman. Blocking in a shared resource environment. *IEEE Transactions on Communications*, COM-29(10):1474–1481, October 1981.

[6] V. O. K. Li and E. W. M Wong W. Liao, X. Qiu. Performance model of interactive video-on-demand systems. *IEEE Journal on Selected Areas in Communications*, 14(6):1099–1109, August 1996.

[7] P. V. Rangan, H. Vin, and S. Ramanathan. Designing an on-demand multimedia service. *IEEE Communications*, 30(7):56–64, July 1992.

[8] J. W. Roberts. A service system with heterogenous user requirements - application to multi-services telecommunications systems. In *Performance Data Communications Systems Conference Proceedings*, pages 423–431. North-Holland Publishing Co., 1981.

[9] D. H. K. Tsang and K. W. Ross. Algorithms to determine exact blocking probabilities for multirate tree networks. *IEEE Transactions on Communications*, 38(8):1266–1271, August 1990.

[10] H. Vin, P. Goyal, A. Goyal, and A. Goyal. An observation-based admission control algorithm for multimedia servers. In *Proceedings of the 1st IEEE International Conference on Multimedia Computing and Systems*, pages 234–243, May 1994.

[11] Y. Won and J. Srivastava. Minimizing blocking probability in a hierarchical storage based vod server. In *Proceedings of the International Workshop on Multimedia Database Management Systems*, pages 12–19, August 1996.

[12] Z. Zhang. *End-to-end Support for Statistical Quality-of-Service Guarantees in Multimedia Networks*. PhD thesis, University of Massachusetts, Amherst, February 1997.

Appendix: Proof of Theorem 1

Let $\overline{e_k}$ be $(0,1)$ or $(1,0)$, where the 1 is in the Kth position. Under equilibrium conditions, we have

$$\sum_{(n_1,n_2)\in\Theta(i)} P((n_1,n_2)-\overline{e_k}) = P(I=i-1) - P(I=i-1, n_k=l_k) \quad (11)$$

Figure 8 illustrates (11) for class 2 requests. The LHS indicates the set of states belonging to $\Theta(5)$ and the state probabilities from which we can get to the states in $\Theta(4)$ and their probabilities by moving one arrival backward for class 2. But this set of states in $\Theta(4)$ does not include the blocking state as indicated on the RHS.

$$\sum_{(n_1,n_2)\in\Theta(5)} P((n_1,n_2)-\overline{e_2}) = P(I=4)-P(I=4,n_2=3)$$

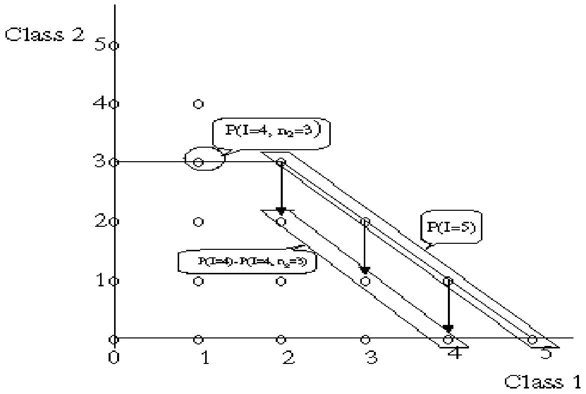

Figure 8. One Arrival Backward in the 2nd Dimension

Consider resources used $I = \sum_{k=1}^{2} n_k$ and multiplying (3) with i and substituting for i and rearranging on the RHS, we have

$$iq_i = \sum_{k=1}^{2} \sum_{(n_1,n_2)\in\Theta(i)} n_k P(n_1,n_2)$$

Substituting from the balance equation, $n_k P(n_1,n_2) = \rho_k P((n_1,n_2)-\overline{e_k})$ we have,

$$iq_i = \sum_{k=1}^{2} \sum_{(n_1,n_2)\in\Theta(i)} \rho_k P((n_1,n_2)-\overline{e_k})$$

$$iq_i = \sum_{k=1}^{2} \rho_k \sum_{(n_1,n_2)\in\Theta(i)} P((n_1,n_2)-\overline{e_k})$$

Substituting from (11) for $\sum_{(n_1,n_2)\in\Theta(i)} P((n_1,n_2)-\overline{e_k})$ we have,

$$iq_i = \sum_{k=1}^{2} \rho_k [P(I=i-1) - P(I=i-1, n_k=l_k)]$$

Multiplying with G and using (4) and moving i to RHS we have the recursion in Theorem 1.

$$G(i) = \frac{1}{i}\sum_{k=1}^{2} \rho_k [G(i-1) - B_k(i-1)] \quad i=1,\ldots,C$$

and $G(i) = 1$ for $i=0$ and 0 for negative i.

QED