RC 20411 (03/18/96)
Computer Science/Mathematics

IBM Research Report

Centralized vs. distributed multimedia servers

Dimitrios N. Serpanos

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, New York

Tasos Bouloutas

Bank of Boston
Treasury Department
Boston, Massachusetts

IBM Research Division
Almaden · T.J. Watson · Tokyo · Zurich

Centralized vs. distributed multimedia servers*

Dimitrios N. Serpanos
IBM Research Division
T. J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Tasos Bouloutas
Bank of Boston
Treasury Department
Mail-Stop: 01-12-08
100 Federal St.
P.O. Box 2016
Boston, MA 02110

Abstract

The organization of multimedia servers is important in the design of low-cost high-performance multimedia application environments. Considering video services as highly demanding applications in a multimedia environment, we analyze and compare centralized and distributed architectures for multimedia video servers. Comparisons are mede in terms of the blocking probability of a video client's request, considering as important parameters the input/output capacity of the system and the amount of storage. Through a combination of analytical results and simulations, we conclude that in general a centralized architecture is preferable. The results indicate however that, in a distributed architecture containing a large number of powerful servers, performance is similar to the centralized architecture under high load conditions. The results also indicate that under light load conditions, the blocking probabilities are quite small. Furthermore, we conclude that centralized and distributed server architectures become equivalent when large amounts of storage are added to the latter or when their input/output capacity is significantly increased.

*This paper is an extended version of one that was presented at the International Conference on Multimedia Communications (MEDIACOMM'95)[2].

1 Introduction

Multimedia servers are one of the fundamental components in environments providing multimedia applications, since they are responsible not only for the storing of large amounts of data, but for the management and accounting of digital libraries that include text, voice and most importantly video.

Multimedia applications can range from education, real-estate and industry in-house training to conferencing and video applications such as video-on-demand, near-video-on demand, etc. In a typical environment providing such applications, a set of servers is connected to a number of clients through interconnections that may range from wide and local area networks to input/output (I/O) channels. Clients request services from the servers that store the multimedia data objects. In response to a request, a server provides the required service at the appropriate quality (Quality of Service, QoS) to the user if it is feasible; otherwise, the client's request is blocked.

Although the current deployment of multimedia services is limited mostly to a few small environments, multimedia applications are expected to be widespread in the future, considering the progress in decreasing the cost of storage and increasing aggregate I/O bandwidth, communication throughput, and processing speed.

A significant amount of effort has been devoted to the architecture and design of distributed servers for various multimedia applications. The reasons for focusing on distributed servers are their scalability and reasonable cost. Many issues have been considered and analyzed, especially for video applications: the placement of data and scheduling [18], the assignment of movies to storage on the basis of their popularity [12], the batching of video streams [6], the interactive control (i.e., stop/resume operations) of video playout [8] [6], the caching of streams for high performance [5], and application routing [7].

As centralized, high-end systems are scalable in many dimensions, they would appear to be an appealing alternative for multimedia servers as well. High-end systems, such as mainframes, are typically multiprocessor (although non-symmetric) systems which can be scaled easily in terms of processing power, memory size as well as I/O throughput, while their cost continues to decrease. Their structure is such that many dedicated processors can off-load functionality from the central units, allowing the systems to scale in many directions to significantly large sizes —for example, aggregate I/O capacity of several Gbps is available today through ESCON channels [15]. In addition to the scaling ability, management of high-end centralized systems is significantly advanced and lower-cost than management of distributed systems, which is quite costly. Therefore, high-end systems should provide a low-cost solution, competitive if not advantageous alternative to distributed servers.

The organization and structure of multimedia servers is a fundamental question, when considering the options, which clearly show that the functions required for multimedia applications can be provided from the whole range of computing systems: from PC and workstation clusters to mainframes and multiprocessors. All of these systems though can be classified in two categories: centralized high-end systems, or distributed systems (clusters of low-end workstations or PCs). It is very important to identify which types of systems should be used as servers in the various environments of interest. Clearly, the metric directing this decision is *cost*, i.e. the cost of providing a user with a required service. In order to calculate cost for a certain environment given its requirements, i.e. total number of customers and QoS required, the question is the cost of the various server configurations in terms of memory size, I/O bandwidth, processing power, network bandwidth and most importantly, management of the various resources. The final answer to the question of cost is far from clear, because of differences between centralized and distributed servers in terms of both functions required and performance; in a distributed server, for example, directory services are necessary to provide a complete solution that is equivalent to the single server solution. Also, the I/O capacity of conventional workstations is very limited when compared to the aggregate channel capacity (e.g., ESCON [4] [15]) of a high-end system. Although the total aggregate I/O capacity of a centralized server may be equal to the aggregate I/O capacity of a distributed one, it is not true that the systems can provide the same service to clients. This occurs because the *Effective I/O capacity* of a distributed server, i.e. the total number of video streams served simultaneously, is affected not only by its aggregate I/O capacity, but by the placement and replication of data as well.

In this paper, we compare centralized and distributed servers using the blocking probability of a client as the meaningful measurement, since the blocking probability indicates the number of clients a system can support. The application considered is video service —due to its demanding requirements— and the system parameters analyzed in this comparison are: *effective I/O capacity* and *storage size*. The reason for choosing these parameters is that, in many multimedia applications the required computational, or processing, power of the server is modest. Only a moderate amount of processing is necessary for such functions as directory lookups, updates, management and similar services when serving video, for example. The fundamental issue in such (video) servers is that the offered I/O capacity is the most important system parameter, because it dictates the number of video streams (VS) that can be simultaneously delivered to clients [1].

We make the comparison for two main configurations. In the first one, we assume that the servers have the same aggregate I/O capacity but the amount of their storage is different. We demonstrate that a centralized server in general provides better performance than a distributed server for the same aggregate I/O capacity. The second configuration assumes that the servers have the same storage, but different I/O capacities.

The paper is organized as follows. Section 2 presents the two main configurations and describes their differences in flexibility and functionality for a complete end-to-end multimedia application. In Section 3 we analyze their performance in terms of blocking probability, which is the relevant metric in this case. We compare the configurations using both analytical methods and simulations for cases in which the distributed servers are equipped with increased amount of storage. Increased storage is used to hold duplicates of video streams in different stations. In Section 4 we analyze system performance for configurations with the same aggregate storage, but different aggregate I/O capacities. The analysis is based on an assumed Poisson arrival process for the arrival of client requests and an exponential service time distribution for the video streams[2]. Although the assumption of Poisson arrivals is realistic, it can be argued that the service time is not exponential in many environments, e.g. in the case of cable TV. Despite that, we can consider other application environments, in which the same server is used for both short video streams (e.g., in real estate and advertising), and longer ones (e.g., teaching and in-house training). Since no specific data exist for application environments, we consider exponential service time as a reasonable first approximation. In Section 5 we describe system performance for applications in which video streams require a uniform (in a certain range) distribution of the service time rather than an exponential one. Such a distribution more realistically models environments in which video streams are more uniform in terms of duration, as in the entertainment field.

[1]The I/O capacity, as defined, is the effective I/O or network link bandwidth offered by the communication subsystems of the server to the application; use of high speed networks coupled with sophisticated architectures for communication subsystems [14] can provide a high throughput, i.e. I/O capacity.

[2]In distributed servers we only consider a homogeneous environment where all workstations provide the same I/O capacity; the analysis can be easily extended to include heterogeneous environments as well, with a different arrival rates and/or where systems with different I/O capacities are clustered.

2 Video server configurations

As already indicated, our analysis focuses on two configurations: *centralized* and *distributed.*

In the *centralized* server configuration, shown in Figure 1, a single server manages the storage containing L video streams (VS), and its I/O capacity is equal to K_C video streams. We measure the I/O capacity using VS as a unit in order to compare the configurations independently of the video compression technology used, since employment of different compression algorithms, e.g. DVI technology [13] or MPEG [11], may result in different bandwidth requirements per VS. By measuring I/O capacity in video streams, our analysis is independent of the specifics of video compression technology.

In the centralized server configuration the clients attach to the server through a network which may be either digital, analog, or a combination of the two. The exact configuration depends on the network technology used. In our model we assume that the clients are capable of sending requests to the server which in turn responds with the establishment of a connection delivering the video to the client, or by denying the service in case it cannot accommodate it. The centralized configuration is quite flexible, since it can provide *any-to-any* service, i.e. the server operates as a switch that implements any-to-any connectivity in terms of providing any video stored on any disk to any of the clients.

Alternatively, the server might be a distributed one, as depicted in Figure 2, where N workstations are clustered to provide a server with a total aggregate I/O capacity of $K_D = (K_1 + K_2 + \ldots + K_N)$ video streams. Incoming requests can be served by any station that manages the storage in which the requested video stream is stored. The requests may be handled by either a distributed manager or by a centralized controller, depending on the exact implementation of the configuration.

2.1 Flexibility

An important advantage of the centralized server is its *flexibility*. When considering such a server with I/O capacity of K_C video streams and a distributed server with I/O capacity $K_D = N \times K_W$, with $K_D = K_C$, the following theorem applies.

Theorem 2.1 *In the distributed server, any-to-any connectivity can be achieved if and only if each workstation has a full copy of all video streams.*

Proof:
If: if we assume that each workstation has a full copy of the VS, then each of the K_D connections established can deliver any video stream, since all VS are available in each station. Thus, any combination of VS served by a centralized server can be also achieved by the distributed one.
Only If: in this case, we assume that workstation $WS[i]$ has all video streams in its local storage except $VS[j]$. If the server is at a state where all workstations $WS[m]$, $m \neq i$ are already serving at their full I/O capacity, and a request for $VS[j]$ arrives, then it will be blocked, while in the single server configuration it would be served. QED. □

The above theorem holds when the system is not able to reconfigure, i.e. established connections cannot change. In a reconfigurable system in which an established connection by a workstation $WS[i]$ can be switched to another one $WS[j]$, $i \neq j$, in order to allow $WS[i]$ to serve another request, the theorem holds provided $\frac{L}{N} \geq K_D$, where L is the total number of video streams, N the number of workstations and all workstations have the same I/O capacity $\frac{K_D}{N}$.

2.2 Management

Large differences exist among the centralized and distributed servers with regard to management of the system. These differences are quite important, and it is expected that they will play a very important role in the final acceptance of either architecture. There are four main areas in management where differences exist:

1. scheduling;
2. fault tolerance and management;
3. security;
4. day-to-day maintenance, billing and accounting.

The centralized architecture has simple performance management requirements. When a new request arrives, the server determines whether there is enough I/O capacity to satisfy it. Thus the server is able to easily perform the service scheduling of new video requests. In contrast, the management requirements of the distributed server architecture are more complex. Scheduling of the new requests is a formidable task by itself.

Many approaches can be followed to determine whether a station is able to serve a client's request and further study is necessary to examine in detail the merits of each one. Examples include:

- A central manager receiving the client's request will route it to the appropriate station. If the station is unable to provide the service, the manager will be notified and the request will be rerouted to a station which holds a second copy of the requested video stream (if such a copy exists). This may occur repeatedly, if many copies are available. If no station can provide the video stream, the client will be blocked and notified that the request cannot be served.

- A manager will broadcast the newly arrived request to all stations and a capable one will serve the request. This approach requires an arbitration scheme that elects a single station to serve the placed request.

- By using the appropriate addressing scheme, a newly arrived request will be routed directly to the station that has a copy of the requested video stream. If this station's I/O capacity is fully used, this station will provide the required rerouting information and eventually the client will either receive the requested video stream or it will be informed that the service request is denied.

Many similar scheduling schemes based on known scheduling or routing algorithms [1] may be used in the distributed server but certainly all such schemes require a larger amount of complexity when compared to that required by the centralized server configuration.

A centralized server suffers from low fault tolerance. Since there is a single server having access to the video streams there is a single point of failure. Special resource consuming approaches need to be employed to ensure a long mean time between failures (MTBF). In contrast, the distributed server inherently provides a good platform for a fault tolerant system, since it allows for easy replication of data and there is no single point of failure. Special consideration must be taken in data placement so that station failures result is graceful degradation of quality of service. However, fault management is more difficult in distributed environments, since faults can be traced only with complex algorithms as well as substantial special infrastructure [3].

In many environments in which security is an issue, the centralized server provides a more manageable solution. Since there is a single point of entry to the system, authentication and security are simpler to implement and manage. Although security is an important issue in distributed systems, known approaches (pp. 430 – 436 in [17], [10]) can be employed to provide a solution in this environment. These solutions are more complex, when compared to the centralized configuration, have increased system cost and the cost of managing them is higher.

A management entity in a server will also be responsible for day-to-day maintenance, billing and accounting as necessary. It is clearly easier to implement and manage a central controller which will be adding and deleting subscribers, customers and video streams as required. In centralized servers billing and accounting is simpler as well, since all service requests are dealt with by a single manager.

Distributed database techniques can be used for maintenance, billing and accounting in distributed servers. In addition to maintaining clients and video streams, distributed managers need to account for stations as well, i.e. for station addition and/or deletion. As in the case of the security issue though, such solutions require a higher complexity, increasing system cost significantly.

Management of multimedia servers is a large topic which is beyond the scope of this paper. We have only mentioned some of the issues here in order to stress that a comparison of various multimedia architectures can not be complete without taking these issues into account.

3 Equal I/O capacity configurations

In this section we examine the case in which the distributed and the centralized architectures have the same aggregate number of input/output channels but with aggregate storage space of variable size. In the next section we examine the complementary case in which the two architectures have the same amount of storage space but different aggregate I/O capacity.

Theorem 2.1 states that a distributed server is equivalent to a centralized one —for the same number of input/output channels— if each station in the distributed system has the same amount of storage as the centralized system. Thus, the two architectures are not equivalent unless the distributed system replicates all video stream data in all stations. This solution is quite expensive considering that video streams consume a large amount of storage and the total amount of storage required by the distributed system would be $N \times L$, where L is the storage size of the central server. Thus, it is interesting to compare the performance of the two server architectures for moderate degrees of data replication. The blocking probability of a video stream client request, P_B, is the most appropriate metric in this comparison.

In the following, we measure P_B using both analytical and simulation methods. In our model we examine the effect of video stream replication on the performance of distributed servers. It is valuable to assess the degree of replication needed to achieve a low blocking probability, since a large degree of replication implies high cost. In the model we use the variable *copy factor* CF to indicate the total number of copies of a video stream in the server. CF is, in general, a real number such that $CF \in [1, +\infty)$; for example, $CF = 1.5$ implies that half of the video streams are duplicated (the copies exist in two different workstations, while half exist in only one copy in the disk space of the server).

In our model we make the following assumptions: The servers store a total of L different video streams. The centralized server has an aggregate I/O capacity of K_C channels, and the distributed one has K_D channels, with $K_C = K_D = K$. The distributed server consists of N stations, and without loss of generality we assume that the video streams are symmetrically distributed among the N stations. So, each station's I/O capacity is equal to $\frac{K}{N}$ channels; furthermore, each station stores $\frac{L}{N}$ different video streams. Considering the copy factor CF, each station has $(CF-1) \times \frac{L}{N}$ video streams replicated from the other stations. The replicated copies of a station's video stream are uniformly distributed among the remaining $(N-1)$ stations.

Hence, the system is modelled as follows: A client's request for a video stream VS_i is first directed to a station WS_k storing VS_i. If WS_k is blocked, i.e., it does not have any free channels, the request may be served by another station if VS_i is stored in additional copies. If $CF < 2$ and VS_i cannot be served by the first attempted workstation WS_k, the probability is $(CF-1)$ that a second copy of VS_i exists in another workstation WS_m, while no second copy of VS_i exists in the server with a probability of $(2-CF)$. If a second copy exists, then it is stored with an equal probability, $\frac{1}{N-1}$, on any of the remaining $(N-1)$ stations. Assuming that VS_m is the station with the second copy, if WS_m is also blocked, the client is blocked. If $CF > 2$, there may be more than two attempts in order to locate an unblocked workstation for VS_i. In general there are at most $\lceil CF \rceil$ and at least $\lfloor CF \rfloor$ attempts (redirections) to locate a specific stream before a client is blocked. The last effort occurs with a probability of $P = CF - \lfloor CF \rfloor$.

3.1 Analytical modelling

The analytical model for the centralized architecture can be solved if we assume Poisson arrivals and exponentially distributed service (channel holding) times.

If P_j represents the steady state probability that there are j clients (customers) being served, then

$$P_j = P_0 \frac{1}{j!} \frac{\lambda^j}{\mu^j} \tag{1}$$

$$where \quad P_0 = \{\sum_{i=0}^{k} \frac{1}{i!} \frac{\lambda^i}{\mu^i}\}^{-1} \tag{2}$$

and where λ is the arrival rate and μ is the service rate ($\frac{1}{\mu}$ is the average service time). The blocking probability P_B is then P_K, the probability that all the K channels are busy.

The analytical model for the distributed architecture is more complex. We introduce this model with an example, for which we assume the presence of 2 stations each having 3 channels I/O capacity, and a copy factor $CF \leq 2$. As indicated previously, if a client's request for a video stream is first directed to a blocked workstation, the probability is $(CF - 1)$ that this video stream is located at the second server and the probability is $(2 - CF)$ that the stream is not replicated. Thus, a first request to a blocked workstation is rerouted to the other workstation with a probability of $(CF - 1)$ or it is immediately blocked with a probability of $(2 - CF)$. If the second workstation is also blocked, the client's request is denied (blocked).

Denoting a as the probability of rerouting, i.e. $a = CF - 1$, the Markov Chain representing the server is shown in Figure 3. The associated balance equations are:

$$(\lambda_1 + \lambda_2)P_0 = \mu P_1 + \mu P_3 \tag{3}$$

$$(\mu + \lambda_1 + \lambda_2)P_1 = \lambda_1 P_0 + 2\mu P_2 + \mu P_4 \tag{4}$$

$$(2\mu + a\lambda_1 + \lambda_2)P_2 = P_1\lambda_1 + \mu P_5 \tag{5}$$

$$(\mu + \lambda_1 + \lambda_2)P_3 = \lambda_2 P_0 + 2\mu P_6 + \mu P_4 \tag{6}$$

$$(2\mu + \lambda_1 + \lambda_2)P_4 = \lambda_2 P_1 + \lambda_1 P_3 + 2\mu P_5 + 2\mu P_7 \tag{7}$$

$$(3\mu + \lambda_2 + \lambda_1 a)P_5 = (\lambda_2 + \lambda_1 a)P_2 + \lambda_1 P_4 + 2\mu P_8 \tag{8}$$

$$(2\mu + \lambda_1 + a\lambda_2)P_6 = P_3\lambda_2 + P_7\mu \tag{9}$$

$$(3\mu + \lambda_1 + a\lambda_2)P_7 = \lambda_2 P_4 + 2\mu P_8 + (\lambda_1 + a\lambda_2)P_6 \tag{10}$$

$$4\mu P_8 = (\lambda_1 + a\lambda_2)P_7 + (\lambda_2 + a\lambda_1)P_5 \tag{11}$$

$$1 = P_0 + P_1 + P_2 + P_3 + P_4 + P_5 + P_6 + P_7 + P_8 \tag{12}$$

The above ten equations yield nine independent equations. Since the first nine equations above are dependent, the last equation replaces the first one, and with rearrangement we obtain the following system of independent equations:

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \frac{\lambda_1}{A} & -1 & \frac{2\mu}{A} & 0 & \frac{\mu}{A} & 0 & 0 & 0 & 0 \\ 0 & \frac{\lambda_1}{B} & -1 & 0 & 0 & \frac{\mu}{B} & 0 & 0 & 0 \\ \frac{\lambda_2}{A} & 0 & 0 & -1 & \frac{\mu}{A} & 0 & \frac{2\mu}{A} & 0 & 0 \\ 0 & \frac{\lambda_2}{C} & 0 & \frac{\lambda_1}{C} & -1 & \frac{2\mu}{C} & 0 & \frac{2\mu}{C} & 0 \\ 0 & 0 & \frac{\lambda_2+a\lambda_1}{D} & 0 & \frac{\lambda_1}{D} & -1 & 0 & 0 & \frac{2\mu}{D} \\ 0 & 0 & 0 & \frac{\lambda_2}{F} & 0 & 0 & -1 & \frac{\mu}{F} & 0 \\ 0 & 0 & 0 & 0 & \frac{\lambda_2}{E} & 0 & \frac{\lambda_1+a\lambda_2}{E} & -1 & \frac{2\mu}{E} \\ 0 & 0 & 0 & 0 & 0 & \frac{\lambda_2+a\lambda_1}{4\mu} & 0 & \frac{\lambda_1+a\lambda_2}{4\mu} & -1 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \\ P_7 \\ P_8 \end{bmatrix}$$

where:

$$A = \mu + \lambda_1 + \lambda_2 \tag{13}$$

$$B = 2\mu + a\lambda_1 + \lambda_2 \tag{14}$$

$$C = 2\mu + \lambda_1 + \lambda_2 \tag{15}$$

$$D = 3\mu + \lambda_2 + a\lambda_1 \tag{16}$$

$$E = 3\mu + \lambda_1 + a\lambda_2 \tag{17}$$

$$F = 2\mu + \lambda_1 + a\lambda_2 \tag{18}$$

Solving the above system of equations, we obtain the values of P_i, $i = 0, \ldots, 8$, which are used to calculate the blocking probability of the server. Since the arrival rate in each state is $\lambda_1 + \lambda_2$, the clients that are served in each state can be found by summing all of the outflow due to λ_1 or λ_2 from the state. For example, in state P_2 the outflow due to λ_1 or λ_2 is $a\lambda_1 + \lambda_2$, while in state P_4 is $\lambda_1 + \lambda_2$. Thus the total blocking probability, P_B, is given by:

$$P_B = \sum_{i=0}^{8} P_i \times \left\{ 1 - \frac{Rate\ of\ customers\ served}{\lambda_1 + \lambda_2} \right\}$$

3.2 General Case Solution

The example presented in the previous subsection gives an outline of the methodology that can be followed to solve the general case problem analytically. In order to solve the problem, we express the balance equations in matrix form and replace its first row with the equation indicating that the sum of the steady state probabilities is equal to 1, as above. We thus obtain a set of independent equations which, when solved, provide the steady state probabilities of all states.

The distributed server can be in one of $n = (\frac{K}{N}+1)^N$ states. Let $P = [P_0 \cdots P_{n-1}]^T$ be the $n \times 1$ matrix representing the state probabilities at steady state. Then:

$$A = M \times P$$

where $A = [1\ 0\ 0 \cdots 0]^T$ and M is an $(n \times n)$ matrix with:

$$\begin{aligned}
M[0][j] &= 1 \quad for\ j = 0, ..., n-1 && (19)\\
M[i][i] &= -1 \quad for\ i = 1, ..., n-1 && (20)\\
M[i][j] &= \frac{f_{ji}}{\sum \forall_j f_{ij}}\ for\ \forall i, j,\ i \neq 0,\ i \neq j && (21)
\end{aligned}$$

where f_{ij} represents the flow from state i to state j. This flow is calculated by examining the state of the system.

Assume that we are modelling a system with N servers and aggregate I/O capacity of K channels, i.e., each server has I/O capacity $\frac{K}{N}$ channels. We represent each state with an N-tuple, where the t-th number indicates the I/O state of the t-th station. Thus, if i represents state $(c_1^i,\ c_2^i,\ \ldots,\ c_N^i)$, there are c_1^i clients being served by the first station, c_2^i clients by the second station, ..., and c_N^i clients by the N-th station. Considering two states i and j with one allocated (used) channel difference between them, f_{ij} is calculated as

$$\begin{aligned}
f_{ij} &= \lambda && (22)\\
&\quad for\ i = (c_1^i, \ldots, c_r^i, \ldots, c_s^i),\ j = (c_1^i, \ldots, c_r^i + 1, \ldots, c_s^i),\ and\\
&\quad c_1^i, \ldots, c_r^i, \ldots, c_s^i < \frac{K}{N}\\
f_{ij} &= c_r^i \mu && (23)\\
&\quad for\ i = (c_1^i, \ldots, c_r^i, \ldots, c_s^i),\ j = (c_1^i, \ldots, c_r^i - 1, \ldots, c_s^i)\ and\ c_r^i > 0\\
f_{ij} &= \lambda + \frac{b\lambda}{s-1} && (24)\\
&\quad for\ i = (c_1^i, \ldots, c_r^i, \ldots, c_s^i),\ j = (c_1^i, \ldots, c_r^i + 1, \ldots, c_s^i),\ c_r^i < \frac{K}{N},\\
&\quad \text{and there are } b \text{ servers in maximum capacity in state } i. && (25)
\end{aligned}$$

When the matrix M is constructed, P is obtained as

$$M^{-1} \times A = P \tag{26}$$

Thus, P is the first column of M^{-1}.

We have developed a C program using ESSL [9] mathematical routines to solve the above equations and calculate P_B in various system configurations. As an example we provide the results obtained by solving analytically a system with a total I/O capacity of $K = 20$ channels. Specifically we examine the case of a single server with 20 channels, and the distributed configurations of 2 servers with 10 channels each, 4 servers with 5 channels each, and 5 servers with 4 channels each.

Figure 4(a) shows the blocking probability P_B for the examined configurations when the copy factor is $CF = 1$, i.e. when there is no data replication in the system, while Figure 4(b) shows P_B for configurations of more than 1 server when there is exactly 1 extra copy of each video stream, i.e. $CF = 2$. As the results demonstrate, P_B increases as the number of stations increases. The same behavior is exchibited in case of $CF = 2$ as well as in larger CF cases we have solved. For all cases $\mu = 1$.

As the number of stations increases, the I/O capacity per station decreases. This increases the probability that a station will block. In a centralized system, any available I/O capacity can be used to serve any request, while in a distributed server there may be I/O available but the stations storing the requested VS may be busy, resulting in call blocking. Thus, as expected, Figure 4 shows that P_B increases as the number of stations increases. As intuitively expected, Figure 5 demonstrates that if data is replicated in a fixed configuration, P_B decreases. An important observation is that as CF approaches N, the number of stations, the blocking probability decreases, approaching the blocking probability of the centralized server, a result which is consistent with Theorem 2.1.

As Equation 26 shows, analytical solution of the distributed server configuration requires inversion of the M matrix, which has dimensions $((\frac{K}{N}+1)^N \times (\frac{K}{N}+1)^N)$. This is an appreciable problem even for moderate server configurations; e.g., for the configuration of 5 stations with 4 channels I/O capacity per station, matrix M has dimensions 3125×3125. Inversion of such a large matrix requires significant computing power and large system memory configurations, which are not easily available. To overcome these limitations and facilitate study of system behavior for large configurations we have developed a simulator for the system.

3.3 Simulation

As described above, the analytical model has its limitations. The solution of the Markov chain can be obtained only for small system configurations because the number of its states is exponential in the number of channels, i.e. is $(\frac{K}{N}+1)^N$. Use of a simulation model has made it possible to calculate P_B in larger configurations. The simulator is coded using CSIM [16] and has been verified against the results we obtained analytically for small server configurations.

The simulated configuration presented in this section is for an aggregate I/O capacity of $K_C = 200$ streams. We compare 7 different configurations: the centralized server with an I/O capacity of 200 streams, and 6 distributed servers composed of 2, 4, 5, 8, 10 and 20 clustered systems each providing I/O capacity of 100, 50, 40, 25, 20 and 10 video streams, respectively.

Figure 6(a) shows the simulation results for the examined configurations when the copy factor is $CF = 1$, while Figure 6(b) shows the simulation results for the examined configurations when the copy factor is $CF = 4$. The figures verify the intuitively expected result that for a fixed value of λ, the blocking probability increases as the number of stations in the server increases.

The results for the various copy factors follow a similar behavior: for low load (under 100) or for high load (over 500) conditions, the blocking probabilities for the alternative architectures are almost identical; thus they are equivalent in terms of blocking probability under such load conditions. The blocking probability of the centralized architecture becomes very low for light loads, and it is always smaller than the blocking probability of the distributed architectures. The results for the centralized server are more favorable for medium loads when no replication is employed, as indicated in Figure 6 where the significantly large P_B of the centralized server (approximately 0.2) is less than 50% of P_B for the distributed server. Thus, in case of average loads the results indicate that the centralized server should display a lower blocking probability (by 50%); for low and high load conditions the centralized architecture should not offer significant advantages. As Figure 6(b) shows when compared with Figure 6(a), replication results in significantly lower blocking probabilities for the distributed server, and thus can improve system performance significantly.

Figure 7(a) shows the simulation results for a 5-station server configuration with I/O capacity of 40 channels per station and for different values of CF, while Figure 7(b) shows the simulation results for the 8-station server configuration with I/O capacity of

25 channels per station, to show the effects of replication within a configuration.

The effects of increasing the storage size are clearly shown in Figure 7. The curves indicate that high redundancy in video streams, increasing storage size, should not provide any advantage under conditions of low and high loads. As system load increases the advantage for having replicated video streams disappears. This occurs because the traffic for the blocked stations generates extra traffic for the unblocked ones. The advantage of replication disappears under light load conditions (less than 100) as well. This occurs because under light loading it is very rare for a station to be blocked, and thus replicated copies are not used. In the load spectrum, storage replication is advantageous in the range of average laod, where it improves the server's blocking probability by approximately 50%. Also, based on Figures 5 and 7, we see that there should be a consistent improvement in the examined configurations as CF increases. This is consistent with Theorem 2.1. As the CF increases to reach N, the number of stations of the distributed system, the blocking probability approaches that of the centralized system.

4 Equal storage configuration

In the previous section we compared various server architectures assuming the same aggregate I/O capacity for all configurations. Our analysis indicated that both server configurations should have a similar performance if we provide increased storage to the distributed server. In this section, we compare the servers assuming the *same aggregate storage capacity* but different I/O capacities, and no replication of video streams.

We assume Poisson arrivals for client requests and exponentially distributed service times. If we assume that the video streams are uniformly distributed on the N workstations $WS[i]$, $0 \leq i \leq N-1$, the average arrival rate per station is $\frac{\lambda}{N}$, where λ is the average arrival rate in the centralized server. Denoting as $P_{WS[i]}$ the probability that station $WS[i]$ is blocked, and since K_C denotes the central server's I/O capacity, K_D the distributed server's I/O capacity, and K_W a workstation's I/O capacity (with $K_W = \frac{K_D}{N}$), the blocking probability of the distributed server is given by:

$$\begin{aligned} P_B &= \sum_{i=0}^{N-1} P_{WS[i]} \times \ Prob(new\ arrival\ goes\ to\ WS[i]) && (27) \\ &= P_{WS[i]} \times \sum_{i=0}^{N-1} Prob(new\ arrival\ goes\ to\ WS[i]) && (28) \\ &= P_{WS[i]} && (29) \end{aligned}$$

The probability that a station $WS[j]$ is blocked is given by:

$$P_{WS[j]} = P_0 \frac{1}{K_W!} \frac{\left(\frac{\lambda}{N}\right)^{K_W}}{\mu^{K_W}} \quad (30)$$

$$where \quad P_0 = \{\sum_{i=0}^{K_W} \frac{1}{i!} \frac{\left(\frac{\lambda}{N}\right)^{i}}{\mu^{i}}\}^{-1}. \quad (31)$$

A direct comparison of these equations with Equations 1 and 2 provides the optimal for specific values of λ, μ, K_W, K_C and N.

Figure 8 shows the *Channel Expansion Factor* $CEF = \frac{K_D}{K_C} = \frac{N \times K_W}{K_C}$, for various servers. Each plot corresponds to a fixed K_W, while K_C is a variable. Thus, each plot shows the necessary increase in the aggregate I/O channel capacity of a distributed system relatively to the total I/O capacity of a centralized server, in order to achieve the same system performance in terms of blocking probability P_B. The results are parameterized by the system load L, which is a function of the arrival rate: $L = \frac{\lambda}{K_C}$. Although the figures show the *Channel Expansion Factor* when the distributed server's P_B is the same with the centralized server's, similar results have been obtained when the distributed server's P_B is within 10% of the one for the centralized server. As a reference, shown in Figure 8(d) is the blocking probability of the centralized server for all the loads considered.

As Figures 8(a), 8(b) and 8(c) show, the Channel Expansion Factor, CEF, increases dramatically when the load is low, while it is small when the load is high in all configurations. An important observation is that the increase of CEF is smaller as the I/O capacity of each individual station increases, and for large K_C. This effect becomes clear in Figure 9 in which CEF is plotted for different server loads with each line in a plot corresponding to a different K_W. The ragged curves occur because as K_C increases, the distributed system introduces new stations —each with K_W streams I/O capacity— in order to achieve the same or lower P_B. At each point at which a new station is introduced, CEF decreases, since $CEF = \frac{K_D}{K_C} = \frac{K_W \times N}{K_C}$. This can be better understood via an example. As Figure 8(c) shows, for a centralized server with 100 channels, at least 3 servers with 50 channels each are needed to match the performance; thus, $CEF = 150/100 = 1.5$. If the number of channels of the centralized server is increased to 150, 4 distributed servers with aggregate capacity 200 must be introduced in order to reach the same performance for load equal to 1; then $CEF = 200/150 = 4/3$. However in order to match the performance for load 0.7, 5 distributed servers with aggregate capacity 250 must be introduced, resulting in $CEF = 250/150 = 5/3$.

Taking into account the blocking probability P_B, shown in Figure 8(d), we see that the large CEF (larger than 10) occurs when P_B is very small, i.e. when it is less than 0.01 for load 0.7 and $K_C > 30$. For large values of P_B, i.e. over 0.2 for load 1.3, and over 0.3 for load 1.5, CEF is much smaller, less than 2 in most cases. Despite this relatively small factor, a significant increase in total I/O capacity has to be provided when small stations are used to create distributed servers when considering the cost (shown by values for small K_D and light load).

5 Uniform service time distribution

The results presented thus far assume Poisson arrivals and exponential service time distributions. However other service time distributions may be more representative of certain application environments; since there is no significant experience with video-on-demand services, we can not reliably characterize associated service times of a realistic environment. One could argue though that if multimedia services are to be used for entertainment video distribution, a uniform distribution may be a better approximation for the service time. In this section we present simulation results for various servers assuming uniform service time distribution. We simulate the configurations of Subsection 3.3 with total aggregate capacity $K_C = N \times K_W = 200$.

Figure 10(a) shows the simulation results thus obtained for the examined configurations when the copy factor is $CF = 1$, while Figure 10(b) shows the simulation results obtained for the examined configurations when the copy factor is $CF = 4$. Figure 11(a) shows the simulation results obtained for a 5-station configuration (40 channels I/O capacity per station) for variable copy factor, while Figure 11(b) shows the simulation results obtained for the 8-station configuration (with 25 channels I/O capacity per station). The results indicate that, the behavior of the various server configurations should be similar to those obtained for the exponential service time distribution.

Figure 12 demonstrates that all of the server configurations should behave similarly in either service time distribution environment. Figure 12(a) shows the simulation results for a 5-server configuration (40 channels I/O capacity per server) for variable copy factor for exponential and uniform service time distributions, while Figure 12(b) shows the simulation results for the 10-server and 4-server configurations.

6 Summary and conclusions

We have introduced a model for comparing centralized and distributed server architectures for video service distribution, assuming Poisson arrivals and exponential as well as uniform service time distributions. We argue that the I/O capacity of servers, and not their CPU speed, is the most important parameter. The relevant comparison metric is the blocking probability of the system, i.e., the probability that a client's request for a video service is denied. Two basic configurations are examined and compared: one in which the two server architectures (centralized and distributed) are designed to have the same aggregate I/O capacity but different amounts of storage, and one in which the two architectures have the same amount of storage but different I/O capacities. For the first configuration we conclude that the centralized architecture should be equivalent to the distributed one if each station of the distributed architecture has a copy of all the video streams available to the centralized architecture. If less memory is assigned to each station of the distributed architecture, the centralized server should always provide lower blocking probability, as indicated through analysis and simulations. The difference between the blocking probabilities of the centralized and the distributed server architectures should be more significant for small configurations. When the two architectures have the same amount of storage but different I/O capacities, the centralized architecture continues to be preferable. Under low load conditions, the difference between the two architectures should be more evident. Finally we examined the case in which the distribution of the service times of video streams is not exponential but uniform. The simulation results indicated that system behavior should be similar to that obtained assuming exponential service time distributions.
[2]

References

[1] D. Bertsekas and R. Gallager. *Data Networks.* Prentice Hall, 1987.

[2] A. Bouloutas and D. N. Serpanos. A Comparison of Multimedia Server architecture. In *Proceedings of the International Conference on Multimedia Communications (MEDIACOMM'95), Southampton, U.K.*, pages 136–138, April 11-12 1995.

[3] A. T. Bouloutas, S. Calo, and A. Finkel. Fault Identification and Alarm Correlation in Communication Networks. *IEEE Transactions on Communications*, 42(2-4):523–533, Feb.-Apr. 1994.

[4] S. A. Calta, S. A. deVeer, E. Loizides, and R. N. Starangewayes. Enterprise Systems Connection (ESCON) Architecture Systems Overview. *IBM Journal of Research and Development*, 36(4):535–551, July 1992.

[5] A. Dan, D. M. Dias, R. Mukherjee, D. Sitaram, and R. Tewari. Buffering and Caching in Large Scale Video Servers. Technical Report RC 19903, IBM Research, January 1995.

[6] A. Dan, P. Shahabuddin, D. Sitaram, and D. Towsley. Channel Allocation under Batching and VCR Control in Video-on-Demand Systems. *Journal of Parallel and Distributed Computing*, 30(2):168–179, November 1995.

[7] A. Dan and D. Sitaram. Application Routing in Distributed Multimedia Database Systems. Technical Report RC 20284, IBM Research, Yorktown Heights, 1995.

[8] J. K. Dey-Sircar, J. D. Salehi, J. F. Kurose, and D. Towsley. Providing VCR Capabilities in Large-Scale Video Servers. In *Proceedings of ACM Multimedia'94*, pages 25–32, October 1994.

[9] IBM. Engineering and Scientific Subroutine Library.

[10] P. Janson and R. Molva. Security in Open Networks and Distributed Systems. *Computer Networks and ISDN Systems*, 22(5):323–346, October 1991.

[11] D. Le Gall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4), April 1991.

[12] T. D. C. Little and D. Venkatesh. Popularity-Based Assignment of Movies to Storage Devices and Video-on-Deman System. *ACM/Springer Multimedia Systems*, 2(6):280–287, January 1995.

[13] A.C. Luther. *Digital Video in the PC Environment.* McGraw-Hill, 2nd Ed., 1990.

[14] H. E. Meleis and D. N. Serpanos. Designing Communication Subsystems for High-Speed Networks. *IEEE Network Magazine*, 6(4):40–46, July 1992.

[15] M. W. Sachs and J. C. Elliot. The IBM Enterprise Systems Connection (ESCON) Architecture. *IBM Journal of Research and Development*, 36(4):577–591, July 1992.

[16] H. Schwetman. CSIM User's Guide. Technical report, Microelectronics and Computer Technology Corporation, March 1990.

[17] W. R. Stevens. *UNIX Network Programming*. Prentice Hall, 1990.

[18] R. Tewari, R. Mukherjee, D. M. Dias, and H. M. Vin. Real-Time Issues for Clustered Multimedia Servers. Technical Report RC 20020, IBM Research, April 1995.

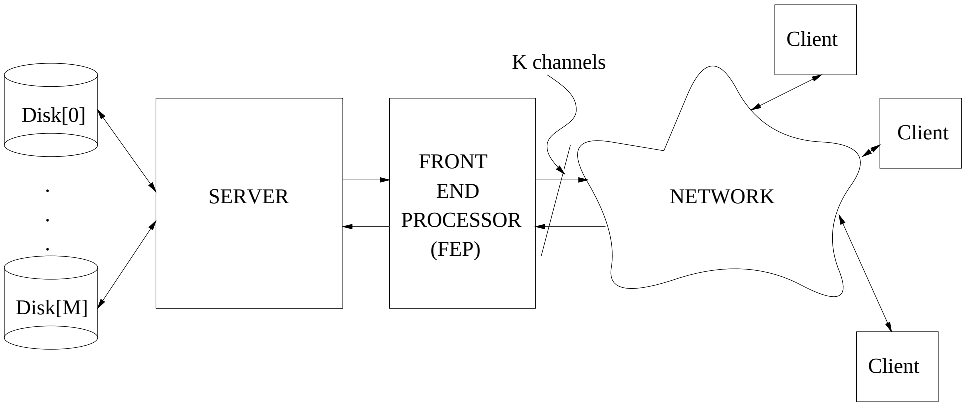


Figure 1: Centralized server configuration

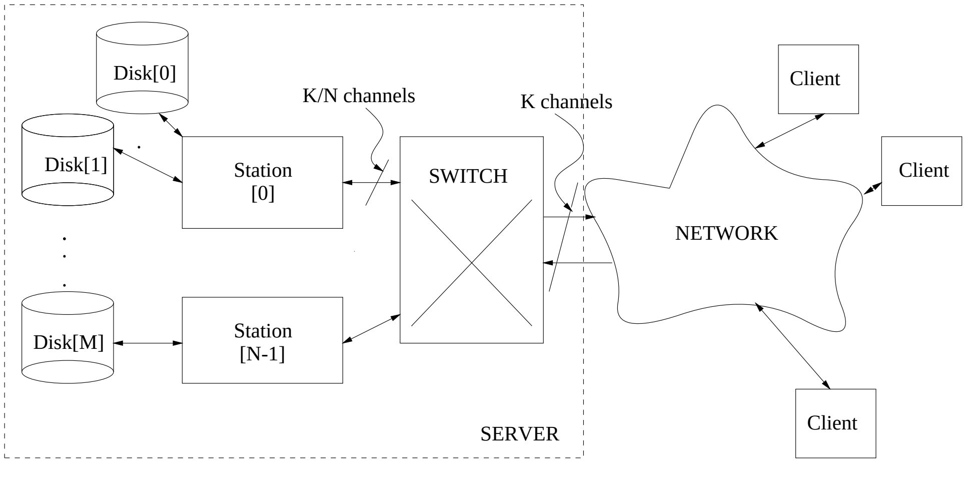


Figure 2: Distributed server configuration

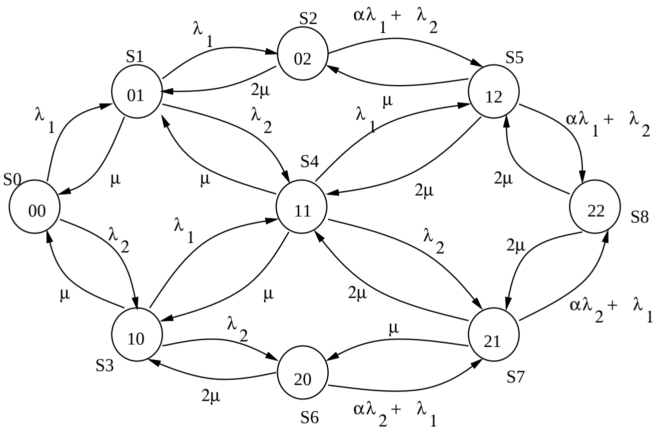
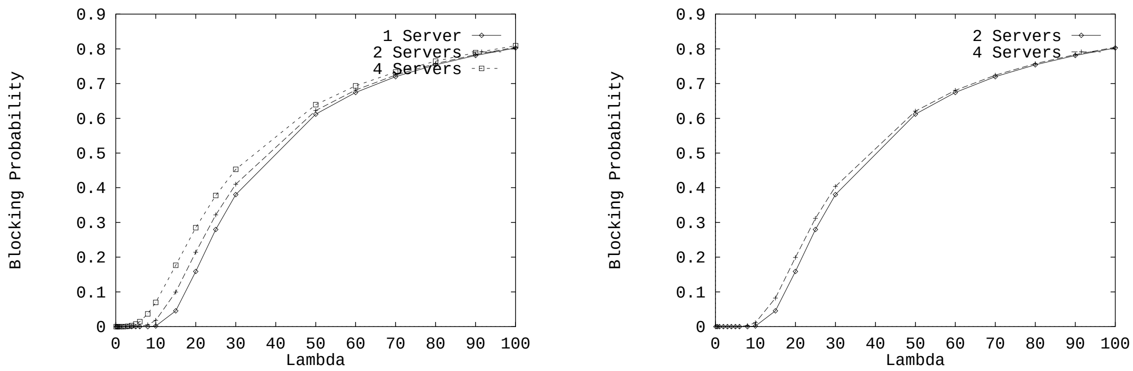


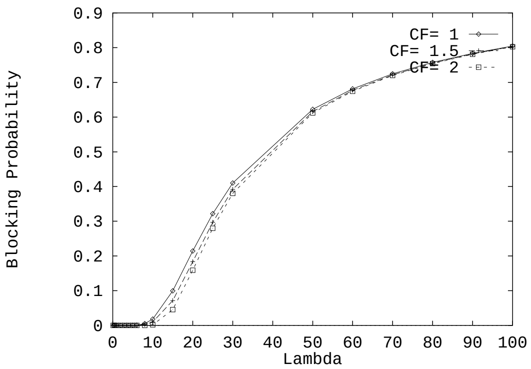
Figure 3: Markov chain



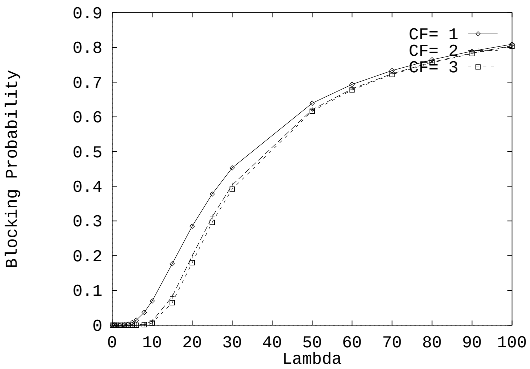
(a) CF= 1

(b) CF= 2

Figure 4: Analytical solution for $K = 20$

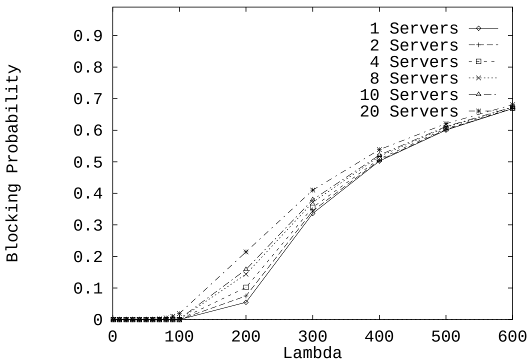


(a) 2 Servers

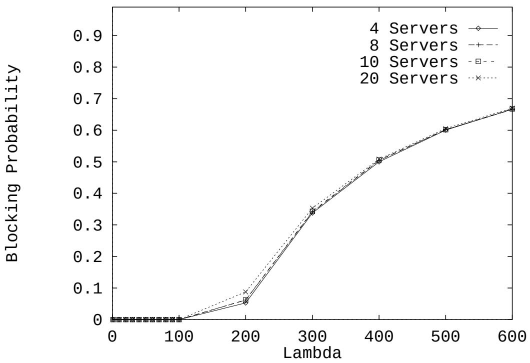


(b) 4 Servers

Figure 5: Analytical solution for $K =\ 20$

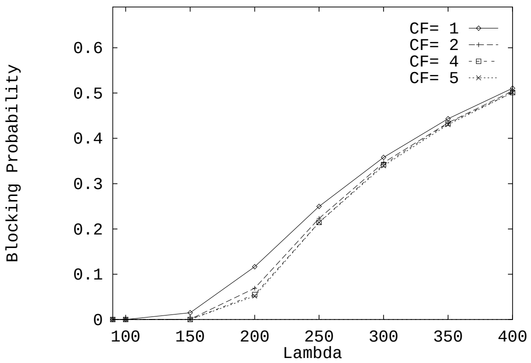


(a) CF= 1

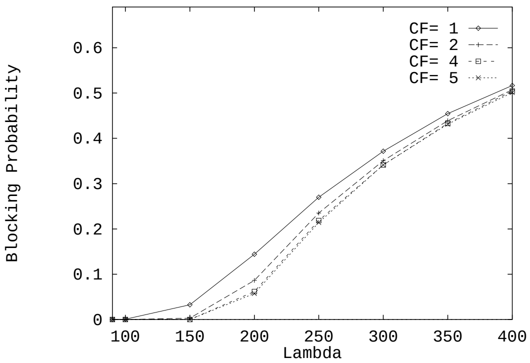


(b) CF= 4

Figure 6: Simulation results for $K =\ 200$

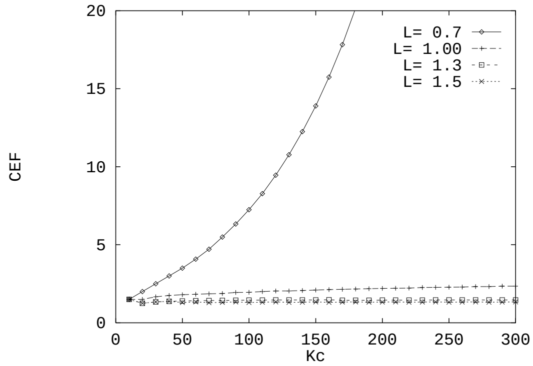


(a) 5 Servers

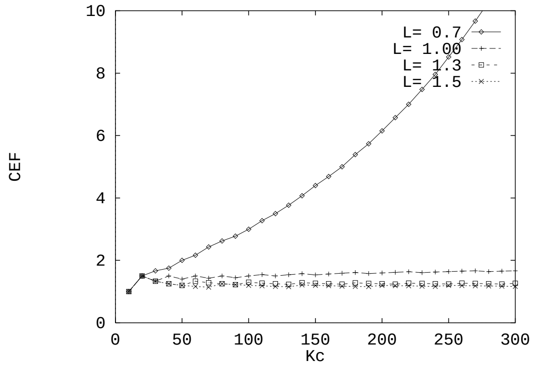


(b) 8 Servers

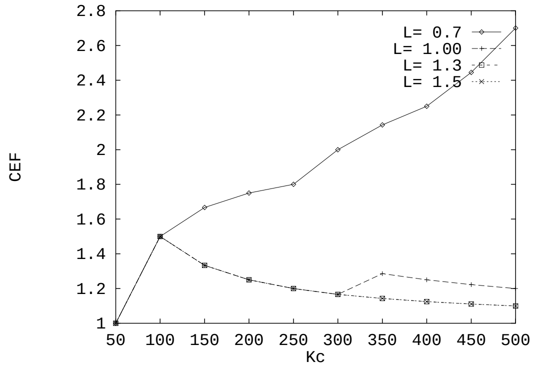
Figure 7: Simulation results for $K = 200$



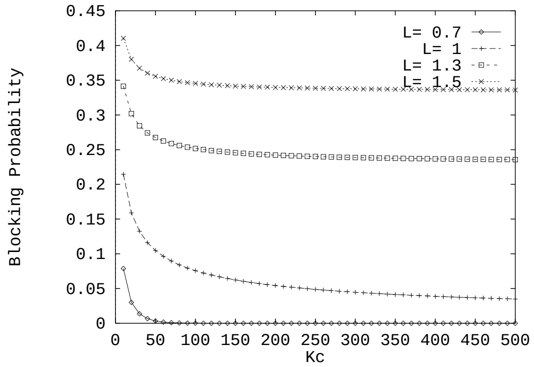
(a) $K_W = 5$



(b) $K_W = 10$

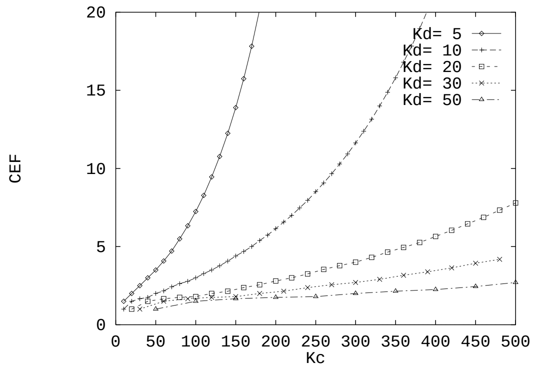


(c) $K_W = 50$

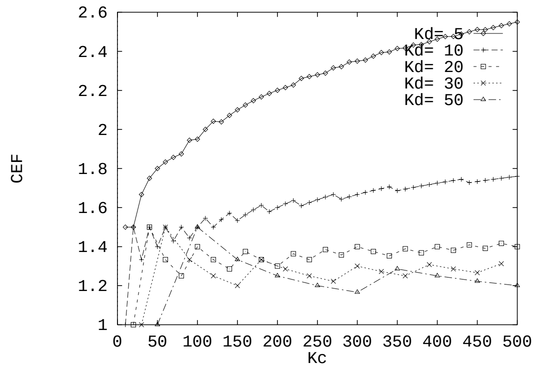


(d) P_B

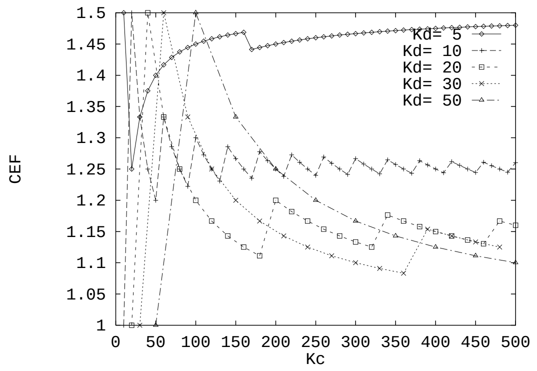
Figure 8: Channel expansion, fixed configurations



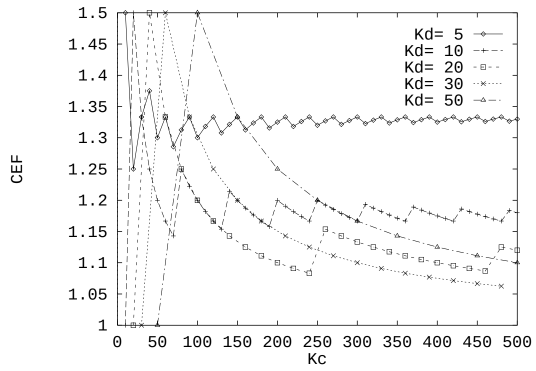
(a) Load= 0.7



(b) Load= 1

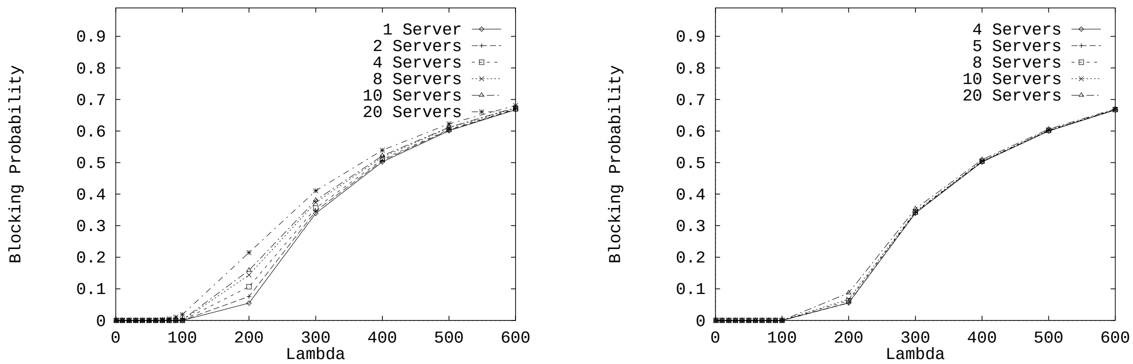


(c) Load= 1.3



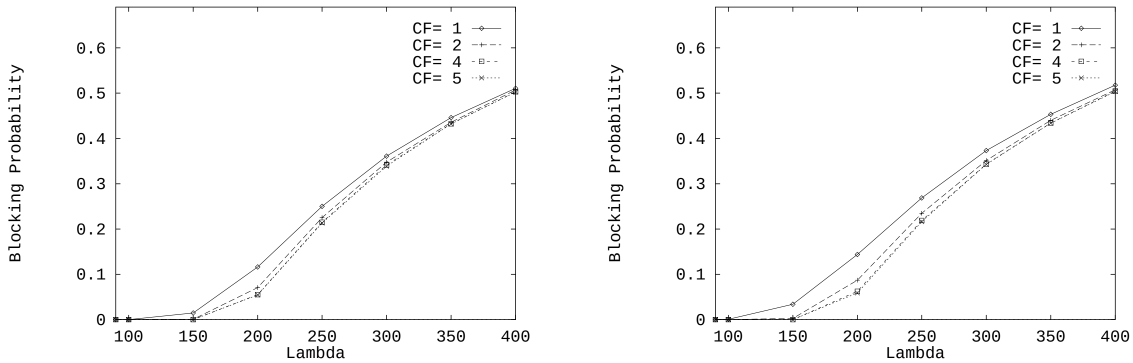
(d) Load= 1.5

Figure 9: Channel expansion, fixed load



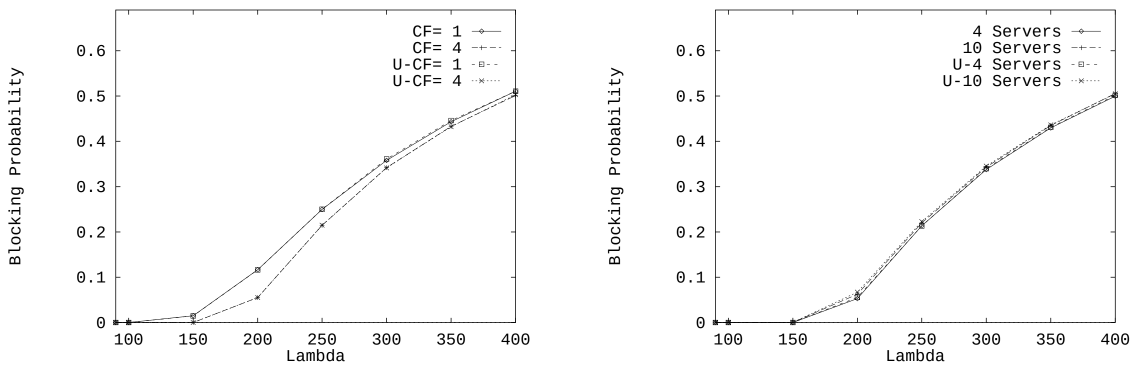
(a) CF= 1 (b) CF= 4

Figure 10: Simulation results, uniform traffic



(a) 5 Servers (b) 8 Servers

Figure 11: Simulation results, uniform traffic



(a) 5 Servers (b) CF= 4

Figure 12: Comparison of exponential and uniform service time distribution