Stochastic Resource Prediction and Admission for Interactive Sessions on Multimedia Servers

Matthias Friedrich, Silvia Hollfelder, Karl Aberer

March 1999
In der Reihe GMD Report werden Forschungs- und Entwicklungs-
ergebnisse aus der GMD zum wissenschaftlichen, nicht-
kommerziellen Gebrauch veröffentlicht. Jegliche Inhaltsänderung
des Dokuments sowie die entgeltliche Weitergabe sind verboten.

The purpose of the GMD Report is the dissemination of research
work for scientific non-commercial use. The commercial
distribution of this document is prohibited, as is any modification
of its content.

Anschriften der Verfasser/Addresses of the authors:
Matthias Friedrich
Silvia Hollfelder
Dr. Karl Aberer
Institut für Integrierte Publikations- und Informationssysteme
GMD – Forschungszentrum Informationstechnik GmbH
Dolivostraße 15
D-64293 Darmstadt
E-mail: {friedrich,hollfelder,aberer}@darmstadt.gmd.de

ISSN 1435-2702
Abstract

In multimedia systems, session-based admission control strategies are needed to guarantee QoS for a client session as a whole. This requires the prediction of irregular and varying resource demands which result from interactive user behavior. The admission control scheme we present in this paper models the user behavior as Continuous Time Markov Chains (CTMCs). By performing transient analysis on the CTMC, the model is able to predict varying resource demands. This allows the detection of possible overload situations for future time intervals of specific length. Evaluation studies show that our approach obtains a good ratio of Quality of Service and, at the same time, a high server utilization.

Keywords: Admission Control, Interactive Multimedia Applications, Continuous Time Markov Chains, Multimedia DBMS

Zusammenfassung

Um für multimediale Sitzungen Qualitätsgarantien geben zu können, muß sich deren Zulassung auf die gesamte Sitzungsdauer beziehen. Bei stark interaktivem Nutzerverhalten schwanken die Anforderungen an Ressourcen stark und müssen bei der Zulassung abgeschätzt werden. In dieser Arbeit stellen wir ein Zugangskontrollschema vor, in dem das Nutzerverhalten als zeitkontinuierliche Markov-Kette (Continuous Time Markov Chain) modelliert wird. Mit Hilfe transienter Analysen dieser zeitkontinuierlichen Markov-Kette können die schwankenden Ressourcenanforderungen vorausgesagt und potentielle Überlastsituationen in einzelnen Zeitintervallen entdeckt werden. Evaluierungen zeigen, daß unser Ansatz ein gutes Verhältnis zwischen Quality of Service (QoS) und hoher Serverauslastung ermöglicht.

Schlüsselworte: Zugangskontrolle, interaktive multimediale Anwendungen, zeitkontinuierliche Markov-Ketten, multimediale DBMS
1 Introduction

Interactive multimedia presentations are a key element in many advanced application domains, like home entertainment (e.g., news on demand, interactive VoD, action games), home shopping (e.g., product documentation), education and training (e.g., interactive computer-based training - CBT [YYWL95]), tourist information (e.g., video browsing, interactive presentation of composite multimedia documents, interactive VoD), and multimedia production (e.g., video editing). From the viewpoint of multimedia server requirements, interactive multimedia presentations share two important characteristics. First, there exists a demand for large data rates to be served under tight temporal constraints, in particular for the continuous media. Second, the amount of requested resources is not precisely predictable, due to the interactions a user can take.

The multimedia server disposes of only a limited amount of resources (e.g., the available number of disks and processors, the buffer size, the communication bandwidth). Thus, it can serve only a limited number of clients. In order to avoid service degradation, the number of concurrent clients accepted for service is limited by means of an admission control mechanism. The goal of an admission control mechanism is to achieve a high resource utilization while meeting each clients’ Quality of Service (QoS)\(^1\) requirements.

Most classical admission control mechanisms, like those for video-on-demand applications, are designed for fairly predictable resource requirements, typically, for serving requests for the playout of a single media stream. In highly interactive multimedia applications with composite media presentations and relatively small single media elements, requesting for admission for each media element separately may lead to intolerable delays within a multimedia session\(^2\). In such scenarios users expect all media to be available within a limited amount of time [GB96]. In addition, for composite presentations, synchronization requirements are specified that have to be considered.

From our viewpoint, admission for highly interactive applications has to be granted at session granularity, and admitted clients have to be guaranteed QoS, e.g., low startup latency, throughout the session, independent of unpredictable fluctuations in resource requirements due to interactivity. Such a session-oriented admission control schema needs to adequately take into account the highly varying resource requirements, caused by interactive sessions which require media encoded in various formats, various presentation modes (i.e., VCR-functions) and combined media. In addition, the guaranteed (or required) QoS may vary for different requests [SN95].

In general, two different approaches to admission control can be distinguished. The deterministic or pessimistic strategy is based on the worst-case assumption that all clients impose the maximum possible load upon the server. Within the deterministic approach, a newly arriving client is only admitted if the available resources suffice to cover his maximum requirements. The goal of the deterministic approach is to guarantee the compliance with the QoS requirements of all admitted clients. An obvious problem of the described strategy is the typically low utilization of the server resources. For applications with strongly varying resource demands, like interactive multimedia presentations, this approach is not adequate.

The second approach, the stochastic or optimistic strategy is based on the assumption that all clients require a statistically varying amount of resources from the multimedia server. Statistical

---

\(^1\)Parameters determining the QoS are the requested data format, the fragment rate (audio: sample rate, video: frame rate), the maximum values for jitter, delay, latency, etc.

\(^2\)From our point of view, a session is a period during which the user interacts with the multimedia server and retrieves data [VL96].
distribution functions are used to model the factors determining the server load. Based on the statistical distribution functions, the probability of an overload is calculated, i.e., a situation in which the given amount of server resources is not sufficient to answer the requests of all clients according to the QoS requirements. Within the stochastic approach, a newly arriving client is admitted to service if the calculated overload probability is beyond a certain threshold. The stochastic strategy normally leads to a much better utilization of the server resources than the deterministic approach, but it has to deal with the problem of handling overload situations. A typical method for solving this problem is to adapt the presentation quality at the client [HKR97] or at the server [TK96, TKC97].

To predict the resource consumption of highly interactive multimedia sessions, we pursue the stochastic approach. The design problem for a stochastic prediction method for resource consumption consists in the kind of application knowledge used to perform the prediction of the resource demands of multimedia sessions.

In the simplest case, the history is observed. Either the past systems’ or past clients’ behavior is inspected and used as an indicator for the future. This approach does not require any understanding of the application semantics and is therefore application-independent.

Another source of information are user profiles (e.g., this user always uses high-quality video). From such user profiles, some estimations on the required resources can be deduced. To improve the estimations, user profiles can be combined with history-based approaches to adapt the profile on the real user behavior.

If the application semantics is known, it can be used to perform predictions on the resource usage directly. One example that has been investigated in [AH99] are browsing scenarios where the multimedia session is the browsing of a ranked hit list of media presentations, e.g., videos. Both the data rates required for the presentation of the media elements and the ranking can be used to perform predictions on the resource requirements. Another example are preorchestrated multimedia documents (e.g., modeled as MHEG [II97] or SMIL documents [HBB98]) in which the temporal order of media and possible interaction points are predetermined. The main difficulty of this approach is that it requires certain heuristic assumptions on how the users behave with regard to their interaction possibilities. This would require extensive observations of behavior, which are nowadays not readily available, such that initial ad-hoc assumptions have to be made.

Which of the three sources of information, namely system and client history, user profiles and application semantics, is adequate is related to the question of which prediction precision is required. Each of the three information sources correspond to a finer level of analysis. For system and client history, statistical usage information is aggregated independently of the type of user and presentation. With user profiles, different classes of users are distinguished, while with application semantics, the analysis is performed down to the level of multimedia document structure and single media elements.

For illustration purposes, we give a simple example. Consider a single-type multimedia presentation that requires from $t_0 = 0$ to $t_1$ (practically) no data and from $t_1$ to $t_2$ the data rate $r$. Depending on the values of $t_1$, $t_2$ and $r$ and the total amount of resources $R$, different admission strategies can be adequate. If $r << R$, an admission based on the average consumption $\frac{t_2 - t_1}{t_2 - t_1}$ will work, since, for a large number of clients, the peak loads from $t_1$ to $t_2$ will be evenly distributed. If $R$ is almost of the same order as $r$ (e.g., $R < 2r$), admission must be based on the worst-case assumption that resource requirements of each client are $r$. However, if $t_2 - t_1 << t_1$, we can make use of the application semantics by lining up the clients appropriately as illustrated in
Figure 1, and thus admit many more clients than with the worst-case assumption. This can be done by an admission strategy that knows the presentation structure and admits only one client at the same time. An admission based on the average consumption may not work in general, since several clients may be admitted at the same time and may create an overlap situation later on.

![Diagram](image)

Figure 1: Admitting Clients by Exploiting Application Characteristics

Thus, the question of what predictions are useful depends on different factors, like the ratio of total system resources and clients’ resource requirements, the distribution of the resource requirements within a session, and the stability of the deviation patterns. For a large number of clients with uniform behavior, a simple method based on average resource consumption may suffice while, for fewer clients with large deviations in consumptions, the use of application characteristics can be beneficial or even crucial. For the typically more complex methods exploiting application knowledge, the implementation and runtime overhead has also to be taken into account.

In a previous paper, we exploited browsing-specific application knowledge. We demonstrated how browsing scenarios can be modeled as a set of states corresponding with different data rates and we showed how user interactions can be modeled as (probabilistic) transitions between these states. This resulted in a Continuous Time Markov Chain (CTMC) model [Tij94]. We heuristically deduced the parameters from a query result list and employed equilibrium analysis for a concrete browsing scenario [AH99]. However, equilibrium analysis can only be applied to specifically structured, so-called closed, CTMCs and is therefore more useful for long-lasting sessions.

In this paper, we focus on a different stochastic admission control strategy. We perform a transient analysis on the CTMC when admitting a new client, in order to predict whether the server load is beyond a given threshold. For short term predictions, the transient analysis is much more precise than the equilibrium analysis, and it is applicable to all types of CTMCs. As an additional advantage, the method is also flexibly adaptable to different types of applications by adjusting the time frequency in which the analysis is performed. Large timesteps smooth out the data rates and are adequate in scenarios with more uniform data rate requirements, while decreasing the timesteps “zooms into” the temporal structure of the multimedia presentations, thus allowing more detailed predictions on resource consumption. The main difficulty that will be solved in this paper is that the method is mathematically harder to treat.

The further content of the paper is structured as follows: In Section 2, we give a survey of the related work. We present our approach in detail, in Section 3, and show simulation results obtained by using our admission control strategy in Section 4. In section 5, we conclude the paper with a result summary and some open issues.
2 Related Work

Most approaches to admission control consider the requests of single media streams. The resource requirements are prespecified by the media request in terms of constant rate or little rate deviations [RVT96] and calculated by stochastic [NMW97], [VGGG94b] or deterministic approaches [VGG95], [ORSN96]. Most concepts providing stochastic service guarantees assume stochastic retrieval time from storage systems which we do not consider. For example, [VGGG94b] exploit the variation in access times from disk.

In the following, we analyse strategies that support interactions and evaluate them with respect to interactive multimedia sessions.

A priori reservation. To guarantee a given QoS, worst-case assumptions about the required data rate can be made. Obviously, in case of reservation of this high data rate, server resources are wasted, and the number of clients that can be served in parallel decreases. Dey-Sircar et al. [DSSKT94] give stochastical guarantees by means of reserving separate server bandwidth for VCR-interactions. The drawback of their work is that they assume interactions to occur rarely.

Re-admission at interaction points. A straightforward way to support interactive client sessions is to initiate for each user interaction a new admission request at the server, as described in Gollapudi and Zhang [GZ96]. This leads to high startup latency in case of high system load. Reddy [Red97] and [DSKS97] aims to reduce this problem but neglect varying consumption rates caused by various formats and media combinations. Gopal et al. [GB96] observe the system performance of running hypermedia sessions consisting of discrete and continuous data requests. The user interaction process is modeled as a queueing system (M/G/1). For simulation purposes the user behavior is represented as a hyperstory time-based Petri net (TPN*). The approach works well for low workload (e.g., 50 percent), but startup latency gets intolerable for higher workload.

Smoothing of data rates for VCR-interactions. Some approaches to admission control for interactive applications propose to “smooth” the data rate deviations to achieve a relatively constant workload. Shenoy and Vin [SV96] propose a data placement and encoding method employing human perceptual tolerances to support multi-resolution streams for interactive scan operations, like fast forward and fast rewind, without additional system overhead. Video streams are encoded in base sub-streams and enhancement sub-streams. For fast forward, only the base sub-stream is used. Chen, Kandlur, and Yu [CKY94] suggest segment skipping for fast forward or fast rewind operations, where a segment can be a set of Group of Pictures (GoP) of an MPEG-video. Chen et al. [CKLV95] change the order of MPEG-frames to a priority sequence. For fast forward and fast rewind, only the most important frames (I- and P-Frames) are pushed to the client. The higher data rate is reduced by quality adaptation on the temporal dimension of other requests by a dynamic resource reservation. The smoothing approach is, however, restricted to relatively simple interactive scenarios where interactions take place within the presentation of one single media stream.

Inspection of the past system behavior. Cherkasova and Phaal [LP98] propose an observation based method that gives admission to web sessions, for example, an e-business application, which consist of a sequence of web server accesses. The goal is to achieve a fair completion guarantee for any accepted session. This approach does not adequately reflect time-constraints of multimedia sessions. Agrawal et. al. propose to stochastically predict network delays by recording historical information in order to reduce buffering delays [ACS98]. Vin et al. [VGG94a] give
predictive service guarantees to increase server utilization for statistical variations in the access times of media blocks from disk. In earlier work, we presented a general admission control mechanism which is applicable for varying resource requirements of highly interactive sessions [HA98]. It consists of (1) the admission of new clients when server resources are available and (2) the scheduling and adaptation of requests of admitted clients. For the admission of new clients, we inspect the past system behavior. For a large number of parallel sessions, the average client consumption is a good estimate for prediction. Data rate variations are accounted for by introducing a safety margin. In the worst case, quality adaptations are required to enable guaranteed continuous delivery.

**Usage of application semantics.** Zhao and Tripathi [ZT98] propose a session-based reservation approach for multimedia applications with varying resource requirements. A multimedia session consists of the presentation of multiple multimedia objects that have to be synchronized in temporal order. The temporal order of the presentation is known at admission time. They propose an “advanced resource reservation” mechanism, i.e., to reserve resources for time intervals in the future. The goal of the approach is to determine a starting point for the presentation for which all required resources (i.e., network and end system) are available. The basic reservation model does not consider user interactions, but following extensions for interactions are suggested: (1) the specification of a minimum upper bound which is not economical and (2) re-admission at interaction point as discussed earlier in this section. In [AH99], we show how Continuous Time Markov Chains can be used to model interactive video browsing applications. We heuristically deduced the parameters specifying a CTMC by using application semantics of a previous, content-based query to the system, and used it to predict the future resource requirements.

The use of Continuous-Time Markov Chains for modeling the access behavior in a multimedia database system to support the efficient vertical data migration between the tertiary and secondary storage has been devised in [KW97]. This shows that the application of the CTMCs to model resource usage in multimedia databases is not only limited to admission control but is applicable to other aspects of resource management as well.

### 3 Approach

In the following we describe a stochastic admission control strategy for multimedia presentation scenarios characterized by high variations in the data rates caused by interactive access. Our approach, called *Admission Control Based On Stochastic Prediction (ACSP)*, rests upon the prediction of the overload probability at the server resulting from the admission of an additional client. The precision of the prediction can be flexibly adapted by choosing the length of the *rounds* which will be statistically analyzed one by one. In this section, we describe our model of interactive client sessions, the admission control system architecture, and the statistical approach to resource usage prediction.

#### 3.1 Modelling of interactive client sessions

For session modelling, we assign to each client an application class \( c (c = 0, \ldots, C - 1) \) which is further differentiated into a finite number of states \( i (i = 0, \ldots, S_c - 1) \) representing the different presentation modes. Each client is allowed to switch between the states of his class. At each
point in time, he is assigned to a unique current state. In our context a transition from one state to another is interpreted as an interaction of the client leading to another presentation mode.

Each class $c$ models a set of clients with similar behaviors. This means that the duration times in the states, the data rates requested on average and the transition behavior of the clients belonging to class $c$ are either identical or similar. The model can be flexibly used for different desired levels of accuracy, by combining subsets of clients according to different degrees of similarity in their behaviors.

For illustration purposes, we give the concrete example for a preorchestrated multimedia presentation. Within a preorchestrated multimedia presentation, the temporal and spatial relationships of components within a multimedia document presentation are specified. In addition, interactions are specified that enable a user to influence the future behavior of the presentation. Preorchestrated multimedia presentations are used in the implementation of various applications, like product documentation or computer based training.

We give an example of a product documentation application. It is modelled as a class consisting of 6 different states. In Figure 2 the nodes relate to states and the arrows represent the possible user interactions, i.e., presentation mode changes. State 0 represents the start of the presentation, state 1 the presentation of a synchronized audio and video shot. Apparently, during processing, the user has no option to skip the multimedia document combination represented by state 1. The states 3, 4 and 5 can only be reached from state 2, the presentation of a text fragment. State 3 represents the presentations of an audio and a video in parallel, state 4 corresponds with the parallel presentations of an image and an audio, and state 5 with the presentations of two videos and an audio in parallel. In the given scenario, the user always has the possibility to directly reach the end state 6 except from state 0.

![Figure 2: Simple Example of a Preorchestrated Multimedia Document](image)

### 3.2 State Transitions and Data Rates

The transitions between the states of a client class are modelled as a stochastic process $X^c(t)$. $X^c(t)$ indicates the state of a client of class $c$ at time $t$. For statistically modelling the state transition system, we use the Continuous Time Markov Chain model (CTMC). Such a process is stationary, time-continuous, and has the Markovian Property, i.e., it is memory-less. For our model, the Markovian property means the following: If a client of class $c$ leaves state $i$, the probability of his moving to state $j$ is always $p^c_{ij}$, no matter how he attained state $i$. For the $p^c_{ij}$, which will subsequently be called *time-independent transition probabilities*, the following
equations hold:

\[ \sum_{j=0, j \neq i}^{S_c-1} p_{ij}^c = 1. \tag{1} \]

If a client moves to state \( j \) of class \( c \), he stays there for a time interval being exponentially distributed with parameter \( \nu^c_j \), independent of how he reached state \( j \). We will subsequently call the \( \nu^c_j \) leaving rates.

The implementation of the ACSP requires the identification of the classes and their corresponding states, as well as the determination of the parameters \( p_{ij}^c \) respectively \( \nu^c_j \). The latter could possibly be carried out by means of observation. In the subsequent part of this paper, we therefore assume that the time-independent transition probabilities as well as the leaving rates are known.

We distinguish two different types of states the client may reside in, namely active states, in which they request data at a specific data rate, and idle states, in which they are inactive. In active states, the requested average data amount per round for a client of class \( c \) is described by the random variable \( N_{ci} \). We assume \( N_{ci} \) to be rectangularly distributed, with minimum \( u_{ci} \) and maximum \( v_{ci} \).

We consider the rectangular distribution for modelling the active states to be appropriate as it describes a corridor which the requested data amounts fall into. The width of this corridor \( v_{ci} - u_{ci} \) captures the variability of the data rates requested by the clients. In the idle states, the clients request with probability 1 no data.

### 3.3 Implementation of the admission control module

Within the admission control module, we predict the probability of a situation in which the given server resources are smaller than the amount of resources requested by the clients. Such a situation is called overload. Our prediction is performed for a time window \( W \) starting at time \( t_{start} \). We further divide \( W \) into rounds which all have the same length \( l_{round} \) (see Figure 3). The statistical analysis in the prediction is performed for each round separately.

![Figure 3: Time Window for Resource Prediction](image-url)

In this analysis, the random variable \( T^d_{service} \) is the time the server needs in the round starting at time \( t_{start} + t \) for serving the data requested by all admitted clients on average. Then an overload is the probability that \( T^d_{service} \) is greater than \( l_{round} \). Thus, for each of the rounds in \( W \), we calculate an upper bound \( ub(t) \) for the probability that, within this round, an overload situation occurs. For \( ub(t) \) the following holds:

\[ ub(t) \geq P(T^d_{service} > l_{round}) \tag{2} \]
The calculation of $ub(t)$ is the essential part in the ACSP. It is performed in two separate steps. First, in a prognosis module 1, a matrix $M(t)$ is calculated which describes the predicted number of clients in each state of each class for the round beginning at time $t$ including the client to be admitted. Second, in a prognosis module 2, $M(t)$ is used to calculate $ub(t)$.

![Figure 4: Flow Diagram of the ACSP Mechanism](image)

Figure 4 shows how the admission control within the ACSP proceeds on the arrival of a new client. At the beginning of the ACSP process the time parameter $t$ is set to the value $t_{\text{start}}$, i.e., to the beginning of the first round within $W$. Next, within the prognosis modules 1 and 2, the upper bound $ub(t)$ for that first round is calculated. This value is then compared with a value $p_{\text{lim}}$ representing the maximal overload probability the system is willing to accept without rejecting the client. The value $p_{\text{lim}}$ is a configuration parameter showing how optimistic or pessimistic the admission control proceeds. High values for $p_{\text{lim}}$ lead to a larger number of admitted clients and therefore to more frequent overload situations. If $ub(t)$ is greater than $p_{\text{lim}}$, the client is rejected, otherwise the time parameter $t$ is increased by $l_{\text{round}}$, the matrix $M(t + l_{\text{round}})$ and the upper bound $ub(t + l_{\text{round}})$ are calculated, and the algorithm proceeds as described before. If, for none of the rounds within $W$, the calculated upper bound of the overload probability is greater than $p_{\text{lim}}$, the new client is accepted to service. Otherwise, if $ub(t)$ exceeds $p_{\text{lim}}$ for a single round, the client is rejected.

3.4 The determination of $M(t)$ in prognosis module 1

The objective of the calculations within prognosis module 1 is to determine the matrix $M(t)$. $M(t)$ is a $C \times S$ matrix with non-negative integers. $S$ is the maximum number of states a class contains. An element $m_{ci}^t$ of $M(t)$ represents the estimated number of clients being in state $i$ of class $c$ at the round beginning at time $t$. For the calculation of $M(t_{\text{start}})$, all admitted clients as well as the actual client requesting for admission are taken into account. We assume for the calculation that none of the clients will leave the system during the time window $W$, i.e., the following equations must hold:

$$
\sum_{i=0}^{S_c-1} m_{ci}^t = \sum_{i=0}^{S_c-1} m_{ci}^{t_{\text{start}}} \quad (3)
$$

If, contrary to our assumption, one client stops requesting the server during the time window $W$, the predictions do not become invalid but only more conservative.

To calculate $M(t)$, we first determine the time-dependent transition probabilities $p_{ci}^{c',t}(t)$. 

12
\[ p_{ij}(t) = P(X^c(t) = j | X^c(0) = i) \] (4)

As the transition process \( X^c(t) \) is stationary and has the Markovian Property, the \( p_{ij}(t) \) can be interpreted as the probability that the client moves from state \( i \) to state \( j \) within \( t \) time units.

The \( p_{ij}(t) \) must be clearly distinguished from the known \( p_{ij} \). Whereas \( p_{ij} \) indicates the probability that a client of class \( c \) will eventually move from state \( i \) to state \( j \), \( p_{ij}(t) \) indicates the probability that a client of class \( c \) will move from state \( i \) to state \( j \) within time \( t \).

The \( p_{ij}(t) \) can be calculated via the uniformization method [Tij94] using the given \( p_{ij} \) as well as the given leaving rates \( \nu^c_j \) as follows:

\[
p_{ij}(t) = \sum_{k=0}^{\infty} e^{-\nu^c_i t} \frac{(\nu^c t)^k}{k!} p_{ij}^{(k)}
\]
with \( p_{ij}^{(k)} = \sum_{s=0}^{S_c-1} p_{is}^{(k-1)} p_{sj} \)
\[
\begin{align*}
p_{ij}^{(0)} &= 1 \\
p_{sj}^{(0)} &= 0 \text{ for } j \neq s \\
\nu^c_i &\geq \nu^c_j \forall i = 0, \ldots, S_c - 1
\end{align*}
\]

Next, we use the \( p_{ij}(t) \) to calculate the matrix \( \overline{M}(t) \) giving the expected values for the numbers of clients in the individual states of the different classes.

\[
\overline{M}(t) = \begin{pmatrix}
m^t_{0,0} & \cdots & m^t_{0,S-1} \\
\vdots & \ddots & \vdots \\
m^t_{C-1,0} & \cdots & m^t_{C-1,S-1}
\end{pmatrix}
\]
with \( m^t_{cl} = \sum_{k=0}^{S_c-1} m^{t\text{start}}_{ck} p_{kl}(t) \)
\( m^{t\text{start}}_{cl} \in M(t_{\text{start}}) \)
\( t \in \{0, t_{\text{round}}, 2t_{\text{round}}, \ldots, (W-1)t_{\text{round}}\} \)

Finally, we obtain \( M(t) \) by rounding the values of the elements of \( \overline{M}(t) \).

### 3.5 The determination of \( ub(t) \) in prognosis module 2

Based on the matrix \( M(t) \), we can calculate the upper bound \( ub(t) \) according to equation (2). To achieve this, we first introduce a random variable \( N^t \) representing the amount of data the admitted clients and the new client request on average in the round starting at time \( t_{\text{start}} + t \). We further assume that the following relationship between \( N^t \) and \( T^t_{\text{service}} \) holds:

\[
T^t_{\text{service}} = \frac{N^t}{\text{capacity}}
\]

13
Equation (8) expresses a linear relationship between the amount of data the server has to transmit to the client and the time it takes the server to execute this transmission. The parameter capacity is a constant that gives the data amount the server is able to deliver in a single round. To calculate $ub(t)$, we use the Chernov Inequality [Kle75] which has the form

$$ p(Y \geq x) \leq \inf_{\theta \geq 0} e^{\theta x} G_Y(\theta) \tag{9} $$

In this inequality, $G_Y(\theta)$ is the so-called Moment Generating Function of $Y$ which is defined as [Kle75]:

$$ G_Y(\theta) = \int_{-\infty}^{\infty} e^{\theta y} f_Y(y) dy \tag{10} $$

with $f_Y(y)$: density function of random variable $Y$

Between the Moment Generating Function $G_Y$ and the so-called Laplace Transformation $F_Y^l(\theta)$ the following relationship exists [Kle75]:

$$ G_Y(\theta) = F_Y^l(-\theta) \tag{11} $$

Inequality (10) applied to equation (2), leads to

$$ ub(t) = \inf_{\theta \geq 0} e^{-\theta\cdot round} G_{T^{l}_{\text{service}}}(\theta) \leq P(T^{l}_{\text{service}} > l_{\text{round}}) \tag{12} $$

Thus, to find $ub(t)$, we first have to solve the problem of how to determine $G_{T^{l}_{\text{service}}}(\theta)$, i.e., the Moment Generating Function of $T^{l}_{\text{service}}$. To achieve this, we proceed as follows: First, we express $T^{l}_{\text{service}}$ in terms of known random variables. Then, we use this expression to calculate the Laplace Transformation of $T^{l}_{\text{service}}$ and derive from this Laplace Transformation the desired Moment Generating Function using equation (11).

For the first step, we observe that the random variable $N^{t}$ for the total data consumption can be computed from the matrix $M(t)$ and the known random variables $N_{ci}$ for the amount of data a client on average requests in one round if he is in state $i$ of class $c$, as follows:

$$ N^{t} = \sum_{c=0}^{C-1} \sum_{i=0}^{S_c-1} \sum_{k=1}^{m_{ci}} N_{ci} \tag{13} $$

Thus, according to equation (8) $T^{l}_{\text{service}}$ can be rewritten as follows:

$$ T^{l}_{\text{service}} = \frac{1}{\text{capacity}} \sum_{c=0}^{C-1} \sum_{i=0}^{S_c-1} \sum_{k=1}^{m_{ci}} N_{ci} \tag{14} $$

Next, we define a random variable $M_{ci}$ as follows:

$$ M_{ci} = \frac{1}{\text{capacity}} N_{ci} \tag{15} $$

3 We abstract from resource specific parameters, like buffer size, disk seek and transfer time, and rotational latency.
We have to distinguish between the idle and the active states of the clients. Within the latter, \( M_{ci} \) is rectangularly distributed, and the parameters \( u'_{ci} \) and \( v'_{ci} \) corresponding to the minimum and maximum of the rectangular distribution of \( M_{ci} \) as well as the distribution function can directly be deduced from those of \( N_{ci} \):

\[
P(M_{ci} \leq x) = \begin{cases} 
0, & \text{for } -\infty < x \leq u'_{ci} \\
\frac{x - u'_{ci}}{v'_{ci} - u'_{ci}}, & u'_{ci} < x < v'_{ci} \\
1, & v'_{ci} \leq x
\end{cases}
\]

with \( u'_{ci} = \frac{u_{ci}}{\text{capacity}}, v'_{ci} = \frac{v_{ci}}{\text{capacity}} \) (16)

For the idle states, the random variable \( M_{ci} \) has the value zero. Therefore, we can rewrite \( T^t_{\text{service}} \) in terms of known random variables as follows:

\[
T^t_{\text{service}} = \sum_{c=0}^{C-1} \sum_{i=0}^{S_{ci}-1} \sum_{k=1}^{M_{ci}}
\]

This is the form of \( T^t_{\text{service}} \) that we can use to calculate the Laplace Transformation of \( T^t_{\text{service}} \). This calculation is based on the following observations [Kle75]: The density function of a sum of random variables is equal to the convolution of the density functions of the individual terms of the sum. The Laplace Transformation of a convolution is equal to the product of the Laplace Transformations of the individual terms of the convolution, where the convolution is defined as [Kle75]:

\[
\text{conv}(y) = \int_{-\infty}^{\infty} f_{X_1}(y - x_2 - \cdots - x_n) \ast f_{X_2}(x_2) \ast \cdots \ast f_{X_n}(x_n)dx_n
\]

with \( f_{X_i} \): density function of term \( i \) 
\( n \): number of terms

Using equation (17), we can therefore calculate the Laplace Transformation \( F^t_{T^t_{\text{service}}}(\theta) \) of \( T^t_{\text{service}} \) as follows:

\[
F^t_{T^t_{\text{service}}}(\theta) = F^{t}_{M_{0,0}}(\theta)^{m_{0,0}} \times \cdots \times F^{t}_{M_{0,S_{ci}-1}}(\theta)^{m_{0,S_{ci}-1}} \\
\times \cdots \times F^{t}_{M_{C-1,0}}(\theta)^{m_{C-1,0}} \times \cdots \times F^{t}_{M_{C-1,S_{ci}-1}}(\theta)^{m_{C-1,S_{ci}-1}}
\]

In this context \( F^{t}_{M_{c,i}}(\theta) \) represents the Laplace Transformation of the random variable \( M_{ci} \). As the exponents of the individual factors are the elements of the known matrix \( M(t) \), the search for \( F^t_{T^t_{\text{service}}}(\theta) \) can be reduced to determining \( F^{t}_{M_{c,i}}(\theta) \), which has the following form:

\[
F^{t}_{M_{c,i}}(\theta) = \begin{cases} 
e^{-\theta u'_{ci}} - ne^{-\theta v'_{ci}} \end{cases} (\text{active states})
\]

\[
F^{t}_{M_{c,i}}(\theta) = 1 (\text{idle states})
\]

(20)
Having determined $F_{T_{service}}(\theta)$ the Moment Generating Function $G_{T_{service}}(\theta)$ can be calculated using equation (11). Finally, we compute $u b(t)$ by a polynomial approximation of the term in equation (12).

4 Simulations

4.1 Simulation scenarios

For simulation purposes, we model three preorchestrated interactive multimedia presentations with highly varying resource requirements. As components of the presentations, we use three different media formats, namely the video formats MPEG-1 and MPEG-2 and the audio format MP3. We assume that an MPEG-1 video has a mean data rate of 1.5 Mbit/s. For an MPEG-2 encoded video, we assume a mean data rate of 4.5 Mbit/s. Both video formats are considered to be hardware encoded with a fixed IPB-pattern. Therefore, we assume the rates within a video stream to be rectangularly distributed with maximum or minimum values 20 percent above or below the average data rate (see experiment 3 below). We assume audio streams with CD-quality to be MPEG-1 encoded. For MP3 encoded audio streams, we use a constant data rate of 130 Kbit/s. Since text documents, images and VRML scenes require a relatively low amount of data in comparison with time-dependent media like audio and video, we ignore them within our simulations. The resource requirements of the different media formats are summarized in Table 1.

<table>
<thead>
<tr>
<th>Media Format</th>
<th>Mean Data Rate</th>
<th>Distribution</th>
<th>Rate Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPEG-1</td>
<td>1.5 Mbit/s</td>
<td>rectangular</td>
<td>0.2</td>
</tr>
<tr>
<td>MPEG-2</td>
<td>4.5 Mbit/s</td>
<td>rectangular</td>
<td>0.2</td>
</tr>
<tr>
<td>MP3</td>
<td>130 Kbit/s</td>
<td>rectangular</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 1: Resource Requirements

In our simulations, we consider three different application classes. The first one is adapted from a real-world multimedia presentation of a printing company [Mor98]. It consists of 9 different states (see Figure 5). In the beginning, a general introduction of the company is given (state 0). After this introduction, the user automatically reaches the main menu (state 1) from where he can choose how to proceed. The content of the further presentation can be distinguished into two main parts: a set of audio-video-presentations describing the printing process (states 2, 3, 4 and 5) and a set of video shots and audio sequences in which the service support of the company is explained (states 6 and 7). Table 2 includes the media combinations, the mean data rates and the mean holding times of the users, within the single states of class 1. In Figure 5, the values next to the arrows represent the time-independent transition probabilities. Since the description of the printing process as well as the explanation of the service support are temporally dependent, the probability to visit the states in their temporal order is higher than to visit them, e.g., in backward order.

The second class represents a typical multimedia presentation consisting of 4 different states with extremely high variations in the requested data rates. The mean data rate in state 2, for example, is more than 80 times larger than that of states 0 and 3. The simulation parameters within
the states of class 2 are summarized in Table 3. The time-independent transition probabilities are shown in Figure 6.

Finally the simulation parameters of the third class are summarized in Figure 6 and in Table 4, respectively. In this class which consists of 4 different states, the variations in the requested data rates are not as high as those in class 2.

<table>
<thead>
<tr>
<th>State</th>
<th>Media Combinations</th>
<th>Mean Data Rates</th>
<th>Mean Holding Times</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 MP3 and 2 MPEG-1</td>
<td>3.13 Mbit/s</td>
<td>40</td>
</tr>
<tr>
<td>1</td>
<td>1 MP3</td>
<td>130 Kbit/s</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>1 MP3 and 1 MPEG-1</td>
<td>1.63 Mbit/s</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>1 MP3 and 2 MPEG-1</td>
<td>3.13 Mbit/s</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>2 MPEG-1</td>
<td>3 Mbit/s</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td>1 MP3 and 1 MPEG-2</td>
<td>4.630 Mbit/s</td>
<td>100</td>
</tr>
<tr>
<td>6</td>
<td>1 MP3</td>
<td>130 Kbit/s</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>1 MP3</td>
<td>130 Kbit/s</td>
<td>100</td>
</tr>
<tr>
<td>8</td>
<td>1 MPEG-1</td>
<td>1.5 Mbit/s</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 3: Class 2

### 4.2 Experimental testbed

In our simulations, the time interval between the arrival of two successive clients in a given class is exponentially distributed. For each class, the parameter of the distribution is 0.05, i.e., every 20 seconds a new client asks for admission, on average. The presentation times of the clients
Figure 6: Class 2 (left side) and Class 3 (right side)

<table>
<thead>
<tr>
<th>State</th>
<th>Media Combinations</th>
<th>Mean Data Rates</th>
<th>Mean Holding Times</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 MP3</td>
<td>130 Kbit/s</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>1 MP3 and 1 MPEG-1</td>
<td>1.63 Mbit/s</td>
<td>200</td>
</tr>
<tr>
<td>2</td>
<td>1 MP3 and 1 MPEG-1</td>
<td>1.63 Mbit/s</td>
<td>300</td>
</tr>
<tr>
<td>3</td>
<td>1 MP3 and 1 MPEG-1</td>
<td>1.63 Mbit/s</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 4: Class 3

are also exponentially distributed with a parameter of 0.003, i.e., the average presentation time is 300 seconds for each class. The server resources are set to 100000, that means, the server is able to deliver 100 Mbit/s. In all experiments, 3000 rounds are simulated. The single requests of the clients are scheduled according to the strategy Earliest Deadline First (EDF). Table 5 summarizes the default simulation parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>server resources</td>
<td>100000</td>
</tr>
<tr>
<td>number of rounds</td>
<td>3000</td>
</tr>
<tr>
<td>duration of a round</td>
<td>1 time unit</td>
</tr>
<tr>
<td>average arrival of new clients</td>
<td>20 time units</td>
</tr>
<tr>
<td>distribution of clients arrival</td>
<td>exponential</td>
</tr>
<tr>
<td>average duration of a presentation</td>
<td>300 time units</td>
</tr>
<tr>
<td>distribution of presentation</td>
<td>exponential</td>
</tr>
</tbody>
</table>

Table 5: Default Simulation Parameters

4.3 Experimental results

In this section, we present the simulation results we obtained by using the ACSP. We measure the server utilization and the ratio of requests served within their deadlines (in-time-ratio). Keeping the deadlines corresponds to the temporal QoS parameters, like start-up delay, continuous presentation and synchronizational requirements during a session. Other QoS parameters, like
spatial resolution, are also covered as they influence the requested data rates. Another goal of the simulations is to investigate if the ACSP leads to a stable system behavior, i.e., if the proposed model is able to recover from underload or overload periods. For the implementation of our simulations, we used the CSIM tool [Mes94].

**Experiment 1:** First, we study the system behavior under variations of the time window \( W \) (see Table 6). In all simulations the parameter \( p_{lim} \) is set to a value of 0.05. For time windows of length 1 and 5 we observe that the system behavior is not stable, i.e., the growing number of client requests cannot be handled by the server. For \( W \geq 10 \) the system becomes stable and it can be seen that larger time windows lead to lower server utilizations and to better QoS levels. By using \( W = 20 \) instead of \( W = 10 \) we observe the best improvements of the system behavior. While the in-time-ratio increases by 54.1 per cent the average server utilization decreases by 2.1 per cent only. A time window of length \( W = 80 \) leads to a utilization of about 80 per cent and an in-time-ratio of about 98 per cent. For \( W = 100 \) the additional QoS increase respectively the additional server utilization decrease are only marginal, since for growing \( W \) the transient analysis calculates the equilibrium solution of the underlying CTMC. If not stated otherwise, we use a time window of length 50 in the subsequent simulations.

<table>
<thead>
<tr>
<th>( W )</th>
<th>Average Server Utilization (in per cent)</th>
<th>In-Time-Ratio (in per cent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>96.1</td>
<td>15.9</td>
</tr>
<tr>
<td>20</td>
<td>94.0</td>
<td>70.0</td>
</tr>
<tr>
<td>50</td>
<td>84.3</td>
<td>88.2</td>
</tr>
<tr>
<td>80</td>
<td>81.6</td>
<td>97.9</td>
</tr>
<tr>
<td>100</td>
<td>79.3</td>
<td>98.4</td>
</tr>
</tbody>
</table>

Table 6: System Behavior for Variations of \( W \), with \( p_{lim} = 0.05 \)

**Experiment 2:** Next, we study the effects resulting from variations of the parameter \( p_{lim} \) (see Table 7). As expected, we observe that higher values for \( p_{lim} \) lead to lower server utilizations and to better QoS levels. For \( p_{lim} = 0.01 \) nearly all requests are served within their deadlines. As the average utilization is not much lower than that for \( p_{lim} = 0.05 \), a value of 0.01 for the parameter \( p_{lim} \) is the best choice in the given scenario.

<table>
<thead>
<tr>
<th>( p_{lim} )</th>
<th>Average Server Utilization (in per cent)</th>
<th>In-Time-Ratio (in per cent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>80.2</td>
<td>97.6</td>
</tr>
<tr>
<td>0.05</td>
<td>84.3</td>
<td>88.2</td>
</tr>
<tr>
<td>0.10</td>
<td>94.2</td>
<td>73.8</td>
</tr>
<tr>
<td>0.20</td>
<td>94.8</td>
<td>73.4</td>
</tr>
</tbody>
</table>

Table 7: System Behavior for Variations of \( p_{lim} \) with \( W=50 \)

**Experiment 3:** Now, we evaluate how the system reacts if the data rate variations in the simulation and the data rate variations assumed for the ACSP model differ. We consider the situation
in which the maximum or minimum values are 50 per cent above or below the average data rate. That means that we allow a higher data rate variation in the states of all classes as assumed in the ACSP. The corresponding results for $P_{lim} = 0.05$ in Table 8 show that the higher variation leads to a server utilization and a QoS level which are very similar to the values observed in experiment 1.

<table>
<thead>
<tr>
<th>$P_{lim}$</th>
<th>Average Server Utilization (in per cent)</th>
<th>In-Time-Ratio (in per cent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>78.8</td>
<td>88.6</td>
</tr>
</tbody>
</table>

Table 8: System Behavior if assumed and simulated Consumption Rate differ

**Experiment 4:** Now, we assume that only clients of the second class are asking for admission. That means that the server has to deal with excessively high variations in the requested data rates. The simulation results are summarized in Table 9. It can be seen that the in-time-ratio for $P_{lim} = 0.01$ is very poor. Nevertheless, by choosing a value of 0.005 for $P_{lim}$, we observe that the simulation results become very good again: nearly 99 per cent of all requests can now be served within their deadlines.

<table>
<thead>
<tr>
<th>$P_{lim}$</th>
<th>Average Server Utilization (in per cent)</th>
<th>In-Time-Ratio (in per cent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>85.8</td>
<td>45.9</td>
</tr>
<tr>
<td>0.005</td>
<td>77.8</td>
<td>88.7</td>
</tr>
</tbody>
</table>

Table 9: System Behavior running only Class 2

**Experiment 5:** Next, we evaluate how the system reacts to changes of the parameter $l_{round}$ (see Table 10). In all simulations, $P_{lim}$ had a value of 0.05, and we regarded a future time interval with a length of 80 time units. It can be seen that the average server utilization became better with growing values for $l_{round}$ and that the in-time-ratio decreased dramatically. For $l_{round} = 4$, only one third of all requests could be served in time. For $l_{round} \geq 8$, the system even became instable, i.e., the server could not deal with the growing resource claims. These results show that the granularity of the overload prediction is a critical parameter. Large values for $l_{round}$ mean that the ACSP-predictions are based on the average client behavior. This experiment clearly indicates that less precise prediction is not suitable for scenarios with high data rate variations.

<table>
<thead>
<tr>
<th>$l_{round}$</th>
<th>$W$</th>
<th>Average Server Utilization (in per cent)</th>
<th>In-Time-Ratio (in per cent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>80</td>
<td>81.6</td>
<td>97.9</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>90.1</td>
<td>85.8</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>92.2</td>
<td>37.9</td>
</tr>
</tbody>
</table>

Table 10: System Behavior with Changes in $l_{round}$
Experiment 6: Finally, we consider the system behavior under the assumption that the real data rate distribution is an exponential distribution. As described in Section 3, the ACSP assumes that the amounts of data the clients request are rectangularly distributed. The simulation results shown in Table 11 are very similar to those obtained in experiment 1 and show that our model is very robust against differences in the underlying data rate distributions.

<table>
<thead>
<tr>
<th>$P_{lim}$</th>
<th>Average Server Utilization (in per cent)</th>
<th>In-Time-Ratio (in per cent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>81.1</td>
<td>97.7</td>
</tr>
<tr>
<td>0.05</td>
<td>79.4</td>
<td>80.3</td>
</tr>
</tbody>
</table>

Table 11: System Behavior with exponentially distributed Data Rate

We summarize our simulation results as follows: Instable system behavior could only be observed for very small values for the parameter $W$ and for large values of $l_{round}$. In the given environment, the time window should have a value of $W = 80$. The parameter $l_{round}$ should have a value of 1 to obtain a good system behavior. With $P_{lim}$ set to a value of 0.01, we achieved a very good QoS and a high server utilization, in most cases. We showed that the ACSP mechanism is even applicable if the real requested data rate differs from the assumed one in terms of mean variation and generating distribution function. In scenarios with unexpected behavior, the parameter $P_{lim}$ should be set lower than in scenarios with more predictable behavior.

5 Conclusion and Further Work

In this paper, we presented an admission control scheme that targets at the highly varying resource requirements of multimedia sessions. With our session-based approach, we are able to achieve continuous presentations and to reduce startup latency. This is of special importance for interactive applications with frequent media switches.

We model application classes using Continuous Time Markov Chains (CTMCs) and stochastically predict the resource usage within a future time interval. Simulation results show that a high server utilization as well as a good Quality of Service are achieved.

Future work will be focused on the following aspects: Currently, we do not consider specific user profiles. More precise information on user behavior, like their preferences for a specific content, enable a more precise parameter setting within the ACSP. Another aspect is the integration of discrete data requests (mixed workload). Especially VRML scenes seem to be interesting in our context since, on the one hand, no 'real' time-constraints are given, but, on the other hand, in case of high system load, the delivery of such discrete data may be very slow. This leads to high presentation delays especially at the start of a VRML presentation because at this point in time a high amount of data is typically requested.

21
References


