Multimedia Systems 8: 83–91 (2000)

Multimedia Systems

Dynamic quota-based admission control with sub-rating in multimedia servers

Sheng-Tzong Cheng[1], **Chi-Ming Chen**[1], **Ing-Ray Chen**[2]

[1] Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan, R.O.C.; e-mail: stcheng@mail.ncku.edu.tw
[2] Department of Computer Science, Virginia Polytechnic Institute and State University, Northern Virginia Graduate Center, 7054 Haycock Road, Falls Church, VA 22043, USA

Abstract. An admission control algorithm for a multimedia server is responsible for determining if a new request can be accepted without violating the Quality of Service (QoS) requirements of the existing requests in the system. A novel quota-based admission control algorithm with sub-rating for two priority classes of requests is proposed in this study. The server capacity is divided into three partitions based on the quota values: one for each class of requests and one common pool shared by two classes of requests. Reward and penalty are adopted in the proposed system model. High-priority requests are associated with higher values of reward as well as penalty than low-priority ones. Given the characteristics of the system workload, the proposed algorithm finds the best partitions, optimizing the system performance based on the objective function of the total reward minus the total penalty. The sub-rating mechanism will reduce the QoS requirements of several low- priority clients, by cutting out a small fraction of the assigned server capacity, to accept a new high-priority client and to achieve a higher net earning value. A stochastic Petri-Net model is used to find the optimal quota values and two approximation approaches are developed to find sub-optimal settings. The experiment results show that the proposed algorithm performs better than one without sub-rating mechanism, and that the sub-optimal solutions found by the proposed approximation approaches are very close to optimal ones. The approximation approaches enable the algorithm to dynamically adjust the quota values, based on the characteristics of the system workload, to achieve higher system performance.

1 Introduction

Delivering multimedia streams with QoS requirements to viewers is one crucial issue in designing a multimedia system. The server of such a system requires admission control policy to guarantee the delivery of on-demand multimedia streams. Upon the arrival of a new request, the server decides if the request can be admitted based on the availability of the server capacity. QoS guarantee of continuous multimedia stream delivery is met once it is admitted. One mechanism to admission control is based on the *reservation scheme* [6,16]. The reservation scheme allocates a fraction of the server capacity (e.g. CPU time and network bandwidth) for a new request based on certain criteria. The reserved capacity is used to retrieve a specified number of disk blocks, to perform multimedia data processing, and to transmit data on the allocated channel. The allocated server capacity is reserved for the specific request until it leaves the system. A new request may be rejected if no available resource is left to serve the request. In such a case, the system incurs a loss due to the rejected requests. As described above, an efficient admission control policy is essential to maximize the system performance and to reduce the loss rate.

In the literature, various admission control algorithms have been proposed. The deterministic approach derives a formula of the maximum number of admitted requests under the worst case load [1]. The requests are assured of their QoS requirements throughout their existence in the system. A system using such a deterministic approach might be under-utilized, since the admission control policy is based on the pre-determined scenarios. This approach deterministically reserves the most amount of resources that one client will use in the worst case, though this client might consume a small part of resources reserved in most of the time. The deterministic approach represents one extreme of the spectrum in the admission control algorithms, while the observation-based approach stands for the other extreme [5]. The latter approach is based on the prediction from the measurements of the resource usage status [2–5, 7, 10, 12, 13] and provides predictive service guarantee to clients, not absolute guarantee. The basic idea of such an observation-based algorithm is to aggressively accept a request as long as the acceptance of the request does not violate the service guarantee of the existent requests. The statistical approach [5] assumes that the average data access time does not change significantly, and it admits new clients as long as the server can meet the statistical estimation of the total data rate. In paper [2], the proposed adaptive admission control algorithm admits new clients based on the extrapolation from the past measurements of the storage server performance.

Correspondence to: S.-T. Cheng

The above research does not consider different priorities of client requests. Most research tries to admit as many requests as possible without considering the importance of each request. We observe that, in some systems, clients might offer high value of reward and should be given priority services. Similarly, the system pays a high penalty if it rejects a high-priority request. Different priorities are associated with different values of reward and penalty. The admission control policy for such a system attempts to maximize the net earning (the total reward minus the total penalty) in order to optimize the system performance.

A class of quota-based admission control algorithms, based on the above cost model, was proposed in our previous study [6]. The server capacity is partitioned into several partitions based on their quota values: one for each class of requests and possibly one common pool shared by all classes. Requests of a specific priority are granted as long as the current load for the priority class is below the corresponding quota. The server capacity from the common pool can only be used if the priority class requests have used up all the corresponding reserved partition of the server capacity. The admission control algorithm reaches an optimal objective value by dynamically adjusting the quota values, or server partitions, based on the characteristics of the system workload.

We further observe that the system could reach a higher objective value by lowering the service quality of admitted low-priority clients, so as to make room for new arrival of high-priority clients. Such an observation is exploiting the human perceptual tolerance [5], in which few media blocks may be discarded or delayed in a continuous playback process without significantly affecting the perceived quality. In this paper, we propose the dynamic quota-based algorithm with *sub-rating* mechanism. The sub-rating mechanism will reduce the QoS of several low-priority clients by cutting out a small fraction of the assigned server capacity, to accept a new high-priority client and to achieve a higher net earning value. We derive the optimal quota values based on the stochastic Petri-Net (SPN) model. Due to the time complexity of solving an SPN model, we propose two approximation methods to find sub-optimal settings. The multimedia server can dynamically adjust quota values, by one of the approximation methods, based on the characteristics of the current workload to achieve higher performance.

The rest of the paper is organized as follows. In Sect. 2, the system model is presented to characterize the system features and the workload. The objective function is defined based on the system model. In Sects. 3 and 4, the SPN model and the approximation method are developed, respectively. Sect. 5 describes the numerical experiment results. Finally, conclusions are drawn in Sect. 6.

2 System model

The server prioritizes client requests into different priority classes according to their importance to the system. The server adopts the reservation scheme in which a fraction of server capacity is allocated to a client throughout the existence of the request. Throughout the paper, we use the terms of "client" and "request" inter-changeably. The server capacity is divided into several partitions. One partition for each class of requests and possibly one common pool shared by multiple classes. Upon the arrival of a new client, the server checks the remaining capacity for the specific priority class of clients. If the remaining capacity is enough to serve a new request, it will be accepted; otherwise, a sub-rating process may take place to determine if it can be accepted.

The proposed admission control algorithm is capable of handling multiple priority classes. For a system with n classes of requests, there are C_2^n possible combinations of sharing between any two classes. It will be complicated to specify the detailed sharing pattern then. For the sake of simple illustration, in this paper we consider a system with only two priority classes of requests, in which one queue is allocated for each class and one queue for the sharing between the two classes. Consequently, three queues are sufficient to demonstrate the main idea of the algorithm. Nevertheless, this simplification does not reduce the capability of the algorithm. We assume that each class of requests is characterized by its arrival/departure rate and its reward/penalty value. Requests providing high reward and penalty [14,15] are considered as high-priority ones. Let the inter-arrival times of the high-priority and low-priority clients be exponentially distributed with the average times of $1/\lambda_h$ and $1/\lambda_l$, respectively. The inter-departure times of the high-priority and low-priority clients are exponentially distributed with the same service time of $1/\mu$. The proposed method is capable of handling different service times. However, we use the same service time for simplicity. Let the reward rate of high-priority and low-priority clients be v_h and v_l, respectively, with $v_h \geq v_l$, and the penalties be q_h and q_l, respectively, with $q_h \geq q_l$.

A server contains n capacity slots divided into three partitions: n_h, n_l and n_m, where $n_m = N - n_h - n_l$. A capacity of n_h slots (referred as the high partition hereafter) is reserved for high-priority clients; n_l slots (referred as the low partition hereafter) are reserved for low-priority clients; while n_m slots (referred as the common pool partition hereafter) are shared by all priority classes. We assume that all classes of clients have the same QoS requirements, and hence each capacity slot serves one client request. When a new client enters the system, the server checks the remaining capacity for the specific priority class. It is accepted if one such slot exists. Otherwise, the server checks the common pool. In other word, a new client can be assigned to the common pool only if the corresponding partition of the server capacity has no vacancy. A sub-rating process starts if all slots in the common pool are occupied and the new coming request is a high-priority one.

The sub-rating process reduces the QoS level of the low-priority requests in the common pool so as to make room for new arrivals of high-priority requests. Each time α low-priority clients are chosen for degradation. Each such client is degraded by $1/\alpha$ and contributes $1/\alpha$ capacity slot. As a result, they make up one slot in total. The total reward value of these degraded clients is $(\alpha - 1) \times v_l$, which is v_l less than the original total reward value contributed by them (i.e. $\alpha \times v_l$) before degradation. For the sake of service quality, a low-priority client is only degraded once. The degraded clients can be resumed to the normal QoS level upon the departure of a high-priority client. Note that no performance

Table 1. Notation

λ_h	Arrival rate of high-priority clients
λ_l	Arrival rate of low-priority clients
μ	Departure rate of clients
v_h	Reward of a high-priority client if the client is serviced successfully
v_l	Reward of a low-priority client if the client is serviced successfully
q_h	Penalty of a high-priority client if the client is rejected on admission
q_l	Penalty of a low-priority client if the client is rejected on admission
N	Total number of server capacity slots for servicing clients
n_h	Number of slots reserved for high-priority clients only, $0 \leq n_h \leq N$
n_l	Number of slots reserved for low-priority clients only, $0 \leq n_l \leq N$ and also $n_h + n_l \geq 0$
n_m	Number of slots that can be used to service either types of clients, $n_m = N - n_h - n_l$
N_h	Number of high-priority clients served in the system per time unit
N_l	Number of low-priority clients served in the system per time unit
M_h	Number of high-priority clients rejected by the system per time unit
M_l	Number of low-priority clients rejected by the system per time unit
D_l	Number of degraded low-priority clients per time unit
α	Number of low-priority clients to be degraded to accommodate a new high-priority client

gain can be obtained if the sub-rating process makes room for a new low-priority request. As stated above, the system gains extra value of $v_h - v_l$ from the sub-rating process, for each newly admitted high-priority request.

Our objective function is the same as our performance index – the *total pay-off rate*, which is defined as the average amount of net earning received by the server per time unit. Let the system on average serve, per time unit, N_h high-priority clients, N_l low-priority ones, and D_l degraded low-priority ones, and reject M_h high-priority ones and M_l low-priority ones per time unit. The total pay-off rate can be obtained by the reward rate minus the penalty rate:

$$N_h v_h + N_l v_l + D_l v_l \times (\alpha - 1)/\alpha - M_h q_h - M_l q_l \ . \quad (1)$$

The proposed problem is formalized as finding an optimal set of quota values under which the above objective function is maximized. Table 1 summarizes the notations used in the paper.

3 Related quota-based admission control algorithms

Our previous work [6] proposes three quota-based admission control algorithms: free quota, fixed quota, and dynamic quota. They can be applied to the systems with multiple priority classes of clients. However, we use a system with two priority classes of clients for illustration in this section.

The free-quota scheme is performed in a fashion of first-come-first-served (FCFS). A client is admitted to the system as long as there is an available capacity slot. All clients are treated with equal importance. The scheme implies that

$$n_h = 0 \ , \quad n_l = 0 \ , \quad \text{and } n_m = N \ .$$

In the fixed-quota scheme, n_h slots are reserved exclusively for high-priority clients, while the remaining n_l (= $N - n_h$) slots are reserved exclusively for low-priority clients. n_m is set to 0 in this scheme. Clients of one priority class are not allowed to use the slots reserved for the other class, even though the system contains free slots. Consequently, clients might be rejected when the system has free slots. It can be seen that the setting of the quota values has significant effect on the performance of the system.

In the dynamic quota scheme, the server capacity is divided into three partitions: n_h, n_l and n_m, as defined in our system model. When a new high- (low-) priority client arrives, it is accepted if a free slot in the high (low) partition exists. Otherwise (i.e., in the case of no free slot in the corresponding partition), it will be admitted if a free slot exists in the common pool partition. In the case of fully occupied common pool, it is rejected.

The closed-form equations are derived for the pay-off rates of the above algorithms. The dynamic-quota scheme obtains higher pay-off rate than the other two schemes, as reported in our previous study. Furthermore, the optimal values of n_h and n_l for the dynamic-quota scheme can be obtained by queuing analysis, once the characteristics of system load are given. In this paper, we extend the dynamic-quota scheme to a more comprehensive scheme – one with sub-rating mechanism. We present the former one and our extension by SPN model in the following section.

4 Analyzing the system model

The value of the pay-off rate for a system can be obtained by the SPN package (SPNP) [8], given a set of input parameters. The SPNP is a modeling tool developed in Duke University for solving the SPN models. The SPN model of a system can be described in the C-based SPN language (CSPL) of the SPNP. The steady-state solution of the SPN model can be solved by writing the SPNP output functions. Interested readers are suggested to study the SPNP manual [8] for further details. In Subsect. 4.1, two Petri Net models for analyzing schemes with and without sub-rating mechanism are described. Experimental results from these two Petri Net models are also provided in Subsect. 4.2.

4.1 SPN models

The SPN model of the dynamic-quota scheme without sub-rating (NoSUB) is plotted in Fig. 1. The places RH, RL, and RS indicate the available capacity slots in the three partitions, the high, low, and common pool partitions, and have initially n_h, n_l, and n_m tokens, respectively. In this model, one token represents one capacity slot and there are N tokens in the system. H and L represent the number of the high- and low-priority clients served by the high and low partition, respectively. SH and SL denote the number of the high- and low-priority clients served by the common pool partition, respectively. H, L, SH, and SL are set to zero initially. The interpretation of places and transitions in Fig. 1 is as follows.

The SPN model of the dynamic-quota scheme with sub-rating (SUB) is shown in Fig. 2. The notations and their initial values are the same as those in Fig. 1, except that RS is initialized to $\alpha \times n_m$. The new place, SLL, indicates the number of degraded low-priority clients and is initialized to 0. One token in the high and low partitions represent one capacity slot in the system, while α tokens in the common pool partition represents one slot. Therefore, a client served

Places:

(In the high partition)

RH: *mark* (RH) indicates the number of available slots for high-priority clients

H: *mark* (H) indicates the number of high-priority clients being served
(*mark* (RH)+ *mark* (H) = n_h)

(In the low partition)

RL: *mark* (RL) indicates the number of available slots for low-priority clients

L: *mark* (L) indicates the number of low-priority clients being served
(*mark* (RL)+ *mark* (L) = n_l)

(In the common pool partition)

RS: *mark* (RS) indicates the number of available slots

SH: *mark* (SH) indicates the number of high-priority clients using the common pool part

SL: *mark* (SL) indicates the number of low-priority clients using the common pool part
(*mark* (RS) + *mark* (SH) + *mark* (SL) = n_m)

Transition:	*Rate Function:*	*Enabling function:*
T1:	λ_h	*mark* (RH) > 0
T2:	*mark* (H) $\times\mu$	*mark* (H) > 0
T3:	λ_l	*mark* (RL) > 0
T4:	*mark* (L) $\times\mu$	*mark* (L) > 0
T5:	λ_h	*mark* (RH) = 0
T6:	*mark* (SH) $\times\mu$	*mark* (SH) > 0
T7:	λ_l	*mark* (RL) = 0
T8:	*mark* (SL) $\times\mu$	*mark* (SL) > 0

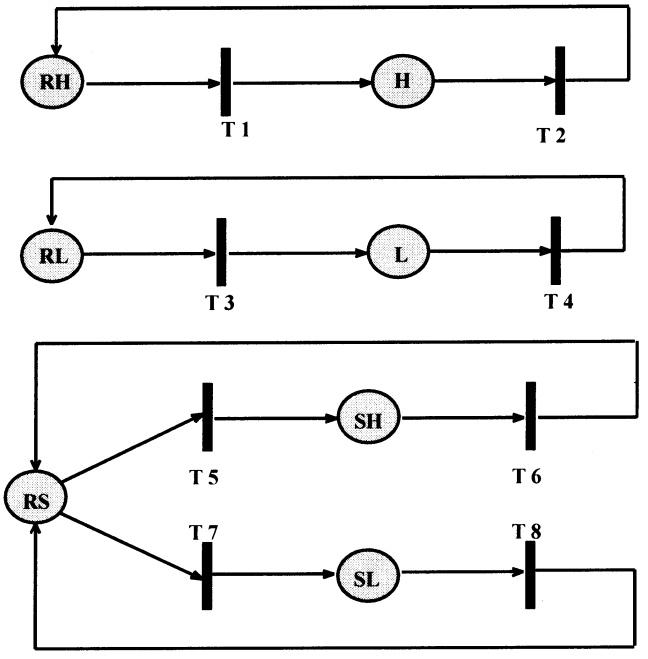


Fig. 1. SPN model for quota-based admission control with no sub-rating (NoSUB)

by the common pool partition consumes α tokens by the transition T1 or T2, and returns α tokens to RS by T3 or T4 when leaving. When a sub-rating process occurs, α low-priority clients (totally α^2 tokens) from SL are degraded. Each loses a token and they contribute α tokens in total. A new high-priority client is then able to be admitted and enters the place SH by the transition T6. The degraded low-priority clients (with a total of $\alpha \times (\alpha - 1)$ tokens) enter the place SLL by T6. A degraded client may leave the system and returns its tokens by T5. Degraded clients are resumed to the normal service level by T7, if RS contains free resources (i.e., tokens released by other clients). The interpretation of the places, arcs, and transitions in Fig. 2 is as follows.

Places:

(In the high partition)

RH: *mark*(RH) indicates the number of available tokens for high-priority clients.

H: *mark*(H) indicates the number of high-priority clients being served
(*mark*(RH)+ *mark*(H) = n_h)

(In the low partition)

RL: *mark*(RL) indicates the number of available tokens for low-priority clients.

L: *mark*(L) indicates the number of low-priority clients being served
(*mark*(RL)+ *mark*(L) = n_l)

(In the common pool partition:)

RS: *mark*(RS) indicates the number of tokens available for high- and low-priority clients.

SH: *mark*(SH) indicates the number of tokens held by high-priority clients.

SL: *mark*(SL) indicates the number of tokens held by low-priority clients.

SLL: *mark*(SLL) indicates the number of tokens held by degraded low-priority clients.
(*mark*(RS) + *mark*(SH) + *mark*(SL) + *mark*(SLL) = $\alpha \times n_m$)

Transition:	*Rate Function:*	*Enabling function:*
T1:	λ_h	*mark*(RH) = 0
T2:	λ_l	*mark*(RL) = 0
T3:	*mark*(SH) $\times\mu/\alpha$	*mark*(SH) $\geq \alpha$
T4:	*mark*(SL) $\times\mu/\alpha$	*mark*(SL) $\geq \alpha$
T5:	*mark*(SLL) $\times\mu/(\alpha - 1)$	*mark*(SLL) $\geq \alpha - 1$
T6:	λ_h	*mark*(RS) = 0 & *mark*(RH) = 0
T7:	(immediate transition)	*mark*(RS) > 0 & *mark*(SLL) $\geq \alpha - 1$
T8:	λ_h	*mark*(RH) > 0
T9:	*mark*(H) $\times\mu$	*mark*(H) > 0
T10:	λ_l	*mark*(RL) > 0
T11:	*mark*(L) $\times\mu$	*mark*(L) > 0

Arc:	*Multiplicity function:*
RS $\rightarrow$ T1	α
T1 $\rightarrow$ SH	α
RS $\rightarrow$ T2	α
T2 $\rightarrow$ SL	α
SH $\rightarrow$ T3	α
T3 $\rightarrow$ RS	α
SL $\rightarrow$ T4	α
T4 $\rightarrow$ RS	α
SLL $\rightarrow$ T5	$\alpha - 1$
T5 $\rightarrow$ RS	$\alpha - 1$
SL $\rightarrow$ T6	$\alpha \times \alpha$
T6 $\rightarrow$ SH	α
T6 $\rightarrow$ SLL	$\alpha \times (\alpha - 1)$
SLL $\rightarrow$ T7	$\alpha - 1$
T7 $\rightarrow$ SL	α

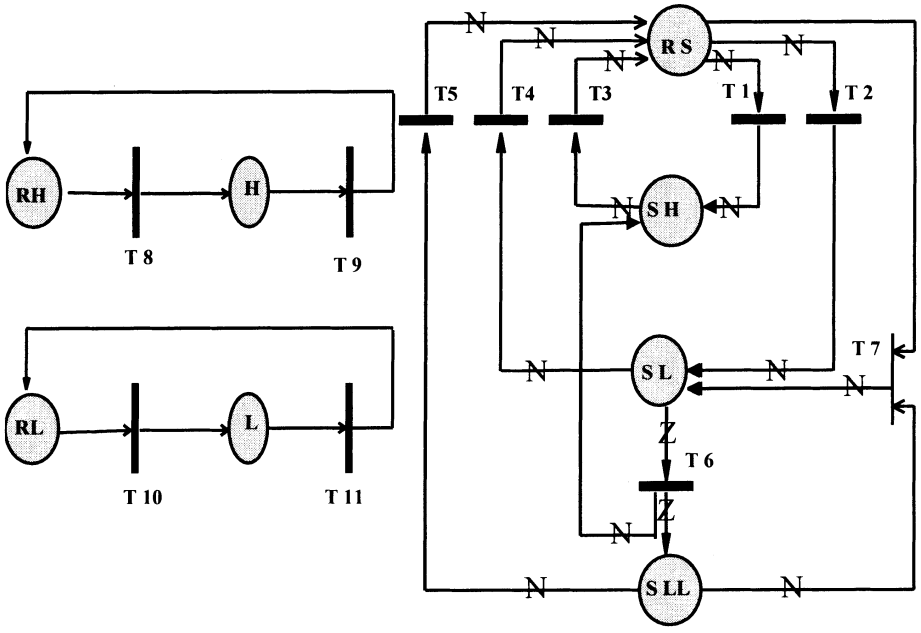


Fig. 2. SPN model for quota-based admission control with sub-rating (SUB)

4.2 Experimental results

To evaluate the performance of the two algorithms, we define two comparison measurements: the best case and average case *gain ratios*. The best case gain ratio indicates the maximal ratio difference that the sub-rating mechanism can get. Note that the best case does not always correspond to the optimal case in which the pay-off rate of system reaches the maximal gain value. To fairly demonstrate the significance of the proposed sub-rating scheme, the gain ratio rather than the gain value is considered. On the other hand, the average-case gain ratio shows the average gain ratio of applying the sub-rating mechanism over all possible combinations of quota values.

The best case gain ratio is defined as

$$\max\left(\frac{\text{SUB}(x)-\text{NoSUB}(x)}{\text{NoSUB}(x)},\ \forall x\right),$$

where x is one of the possible partitioning (i.e., the combinations of the quota values) of the server capacity, and SUB(x) and NoSUB(x) indicate the pay-off rates, given the partition configuration x, obtained by the SUB and NoSUB algorithms, respectively. For N = 16, there are 153 (i.e., $C(16+2,2)$, where $C(x,y) = x!/(y!(x-y)!)$) possible ways of dividing 16 slots into three groups (i.e., n_h, n_l, and n_m slots). In general, there are C(N+2,2) possible combinations of the quota values for the server with N capacity slots.

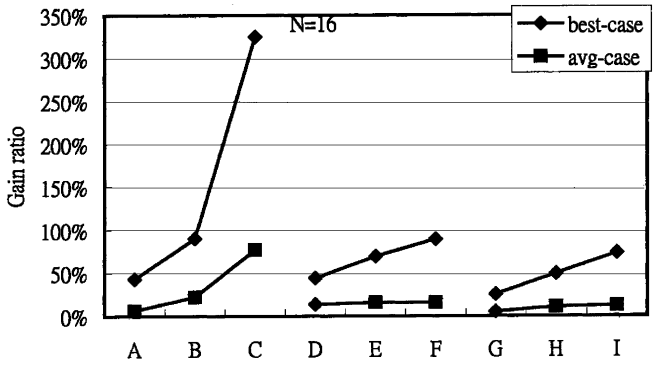
The average-case gain ratio is defined as

$$\frac{\sum_x \frac{\text{SUB}(x)-\text{NoSUB}(x)}{\text{NoSUB}(x)}}{C(N+2,2)}.$$

The input parameters to the SPN models of the SUB and NoSUB algorithms are the arrival and departure rates, i.e., λ_h, λ_l, and μ, and the reward and penalty parameters, i.e., v_h, v_l, q_h, and q_l. That is, a workload is characterized by these input parameters. The pay-off rate with the quota values (n_h, n_l, n_m) can be obtained by the following steps: (1) model the system based on the SPN model; (2) calculate the values of N_h, N_l, D_l, M_h, and M_l by the SPNP; and (3) compute the pay-off rate by Eq. 1.

Figures 3 and 4 show the comparison results for N = 16 and 32 under various workload conditions. Two curves are plotted: one for the best case gain ratio and the other for the average-case one. Nine workload cases are considered in each figure, grouping as three sets. From the first set to third set, v_h varies from 2 to 10. The workloads in one set have the same reward and penalty values, but they have different utilization rates (i.e., $(\lambda_h + \lambda_l)/\mu$) varying from 15/16 to 25/16. A system is nearly saturated when the utilization value is around 15/16, and is over-saturated when it is greater than 1. These workload situations are investigated, since they are the cases to show that an effective admission control algorithm is essential. The average-case and best case gain ratios rise as the workload increases, as shown in each set. It illustrates that (1) the sub-rating mechanism allows a system to achieve a higher pay-off rate under heavy and over-loaded situations and (2) it is beneficial to apply sub-rating in heavy and over-loaded systems, especially when the arrival rate of high-priority clients is large. The experimental results follow our intuition. When the system workload is low, it is unnecessary to use any sub-rating (or degrading) mechanism because all requests will be met. Note that we fix the service rate of a system and change the arrival rates of requests. Therefore, when the arrival rate of high-priority clients is large (i.e., high system workload), the effect of sub-rating shows.

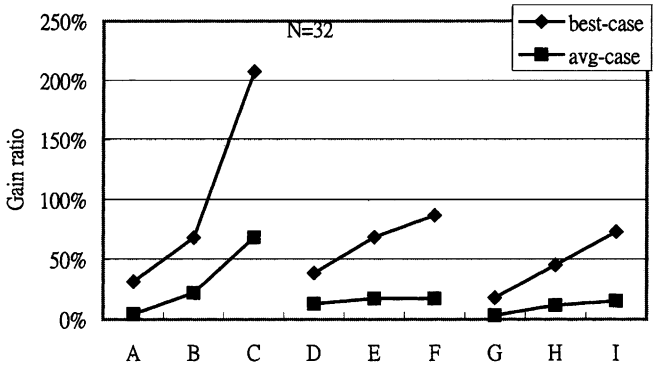
The optimal quota value setting of (n_h, n_l, n_m) for the NoSUB and SUB algorithms can be exploited to maximize the pay-off rate for a system, given the workload characteristics. The NoSUB and SUB algorithms find out the optimal setting by enumerating all possible combinations of the quota value setting, calculating the pay-off rate for each combina-



System Parameters $(\lambda_h, \lambda_l, \mu, v_h, v_l, q_h, q_l, \alpha)$

A = (5,10,1,2,1,2,1,2)	D = (5,10,1,5,1,2,1,2)	G = (5,10,1,10,1,2,1,2)
B = (10,10,1,2,1,2,1,2)	E = (10,10,1,5,1,2,1,2)	H = (10,10,1,10,1,2,1,2)
C = (15,10,1,2,1,2,1,2)	F = (15,10,1,5,1,2,1,2)	I = (15,10,1,10,1,2,1,2)

Fig. 3. Performance comparison for N = 16



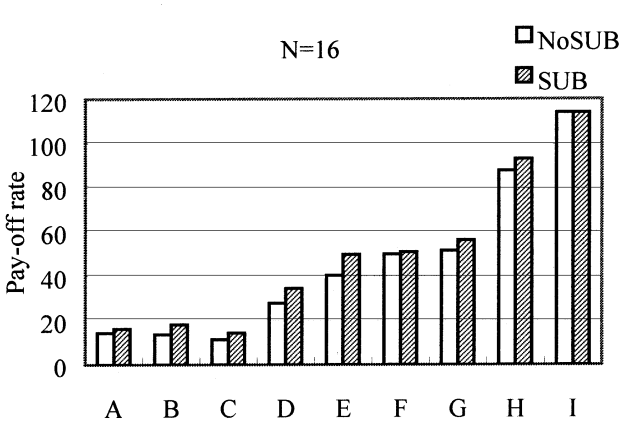
System Parameters $(\lambda_h, \lambda_l, \mu, v_h, v_l, q_h, q_l, \alpha)$

A = (10,20,1,2,1,2,1,2)	D = (10,20,1,5,1,2,1,2)	G = (10,20,1,10,1,2,1,2)
B = (20,20,1,2,1,2,1,2)	E = (20,20,1,5,1,2,1,2)	H = (20,20,1,10,1,2,1,2)
C = (30,20,1,2,1,2,1,2)	F = (30,20,1,5,1,2,1,2)	I = (30,20,1,10,1,2,1,2)

Fig. 4. Performance comparison for N = 32

tion, and selecting the combination with the maximum pay-off rate. Figure 5 and 6 illustrate the optimal settings for the NoSUB and SUB algorithms under various workloads.

Note that the gain ratios, as shown in Figs. 3 and 4, are calculated with respect to the identical quota values over $C(N + 2, 2)$ combinations. On the other hand, the optimal pay-off rates for the SUB and NoSUB, as illustrated in Tables 2 and 3, are obtained from different quota values over $C(N + 2, 2)$ combinations. As shown above, optimal quota values can be found based on the proposed SPN model. However, analyzing a SPN model is very time consuming. The admission control algorithm cannot timely adjust the quota values, based on the current workload, without approximating the proposed model. Therefore, we propose two approximation methods based on queuing analysis. The goal of the approximation methods is to enable the server adaptively configure the resource capacity according to the run-time workload.



System Parameters (for N = 16)	No Sub-rating (by SPNP)		Sub-rating (by SPNP)	
$(\lambda_h, \lambda_l, \mu, v_h, v_l, q_h, q_l, \alpha)$	Quota (n_h, n_m, n_l)	Optimal pay-off rate	Quota (n_h, n_m, n_l)	Optimal pay-off rate
A=(5,10,1,2,1,2,1,2)	2,14,0	14.25	0,16,0	16.07
B=(10,10,1,2,1,2,1,2)	9,7,0	13.59	0,16,0	18.14
C=(15,10,1,2,1,2,1,2)	16,0,0	11.32	0,16,0	14.34
D=(5,10,1,5,1,2,1,2)	5,11,0	27.77	8,8,0	34.32
E=(10,10,1,5,1,2,1,2)	13,3,0	40.26	8,8,0	49.64
F=(15,10,1,5,1,2,1,2)	16,0,0	49.82	10,6,0	50.68
G=(5,10,1,10,1,2,1,2)	7,9,0	51.28	0,16,0	56.01
H=(10,10,1,10,1,2,1,2)	15,1,0	87.67	0,16,0	92.90
I=(15,10,1,10,1,2,1,2)	16,0,0	113.97	16,0,0	113.97

Fig. 5. Optimal pay-off rates and quota values for N = 16

5 Approximation methods

Consider a system with the quota values (n_h, n_m, n_l). The arrival-departure process of high-priority clients served by the high partition of the n_h slots can be modeled as a M/M/n_h/n_h queuing system. Similarly, The process of low-priority clients using the low partition of the n_l slots can be modeled as a M/M/n_l/n_l queuing system. Therefore, the reward rates of the high- and low-priority clients served by the high and low partition are

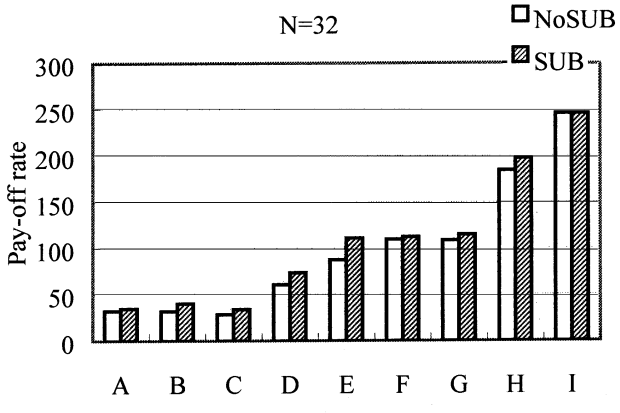
$$\sum_{i=1}^{n_h} i\mu v_h \times \frac{\frac{1}{i!}\left(\frac{\lambda_h}{\mu}\right)^i}{\sum_{j=0}^{n_h}\frac{1}{j!}\left(\frac{\lambda_h}{\mu}\right)^j} \tag{2}$$

and

$$\sum_{i=1}^{n_l} i\mu v_l \times \frac{\frac{1}{i!}\left(\frac{\lambda_l}{\mu}\right)^i}{\sum_{j=0}^{n_l}\frac{1}{j!}\left(\frac{\lambda_l}{\mu}\right)^j}. \tag{3}$$

Clients enter the common pool partition, only when there is no vacant slot in the corresponding partition. Therefore, the arrival rate of high- (low-) priority clients entering the common pool partition can be approximated as φ_h (φ_l). Namely,

$\varphi_h = \lambda_h \times$ *probability* of having n_h clients



System Parameters (for N = 32)	No Sub-rating (by SPNP)		Sub-rating (by SPNP)	
$(\lambda_h, \lambda_l, \mu, v_h, v_l, q_h, q_l, \alpha)$	Quota (n_h, n_m, n_l)	Optimal pay-off rate	Quota (n_h, n_m, n_l)	Optimal pay-off rate
A=(10,20,1,2,1,2,1,2)	5,27,0	32.41	0,32,0	34.67
B=(20,20,1,2,1,2,1,2)	19,13,0	32.00	0,32,0	40.26
C=(30,20,1,2,1,2,1,2)	31,1,10	28.46	0,32,0	34.17
D=(10,20,1,5,1,2,1,2)	10,22,0	60.58	15,13,4	73.46
E=(20,20,1,5,1,2,1,2)	24,8,0	87.42	17,15,0	111.14
F=(30,20,1,5,1,2,1,2)	32,0,0	109.78	24,8,0	112.34
G=(10,20,1,10,1,2,1,2)	12,20,0	108.69	0,32,0	114.67
H=(20,20,1,10,1,2,1,2)	27,5,0	183.86	0,32,0	197.30
I=(30,20,1,10,1,2,1,2)	32,0,0	245.34	32,0,0,	245.34

Fig. 6. Optimal pay-off rates and quota values for N = 32

$$= \lambda_h \times \frac{\frac{1}{n_h!}\left(\frac{\lambda_h}{\mu}\right)^{n_h}}{\sum_{j=0}^{n_h}\frac{1}{j!}\left(\frac{\lambda_h}{\mu}\right)^{j}}. \tag{4}$$

φ_l = $\lambda_l \times$ *probability* of having n_l clients

$$= \lambda_l \times \frac{\frac{1}{n_l!}\left(\frac{\lambda_l}{\mu}\right)^{n_l}}{\sum_{j=0}^{n_l}\frac{1}{j!}\left(\frac{\lambda_l}{\mu}\right)^{j}}. \tag{5}$$

Let the probability that there are i high-priority clients and j low-priority clients served by the common pool partition be $P(i,j)$. The probability distribution of $P(i,j)$ can be approximated by the technique of reduced Markov chain [9]. In Eq. 6, the first term at the right-hand side indicates the probability of having i high-priority clients, and the second term indicates the probability of having j low-priority clients, given i high-priority clients in the common pool partition.

$$P(i,j) = \frac{\frac{1}{i!}\left(\frac{\varphi_h}{\mu}\right)^{i}}{\sum_{k=0}^{n_m}\frac{1}{k!}\left(\frac{\varphi_h}{\mu}\right)^{k}} \times \frac{\frac{1}{j!}\left(\frac{\varphi_l}{\mu}\right)^{j}}{\sum_{k=0}^{n_m-i}\frac{1}{k!}\left(\frac{\varphi_l}{\mu}\right)^{k}}. \tag{6}$$

Hence, the reward rate of the common pool partition is approximated as

$$\sum_{i=0}^{n_m}\sum_{h=0}^{n_m-i} P(i,j) \times (i\mu v_h + j\mu v_l). \tag{7}$$

Consider a state (i,j) in which $i + j = n_m$. Upon arrival of a new high-priority client, the sub-rating process takes place to degrade the j low-priority clients to make room for the new high-priority arrival. It can be seen that at most $\Omega(i)$ (= $\lfloor (n_m - i)/\alpha \rfloor$) slots can be squeezed out for the new high-priority clients. Two methods are developed in the following to approximate the pay-off rate obtained by sub-rating.

Method I

The arrival-departure process of high-priority clients, under a sub-rating process, can be modeled as a M/M/Ω(i)/Ω(i) queuing system. The arrival rate is $\Lambda(i) = \varphi_h \times P(i, n_m - i)$, where i is the number of high-priority clients in the common pool partition before sub-rating is performed. Each time a new high-priority client is admitted, α low-priority clients are degraded and the penalty for the degradation is v_l. The penalty rates of high-priority and low-priority clients are

$$\sum_{i=0}^{n_m} \Lambda(i) \times q_h \times \frac{\frac{1}{\Omega(i)!}\left(\frac{\Lambda(i)}{\mu}\right)^{\Omega(i)}}{\sum_{j=0}^{\Omega(i)}\frac{1}{j!}\left(\frac{\Lambda(i)}{\mu}\right)^{j}} \tag{8}$$

and

$$\sum_{i=0}^{n_m} P(i, n_m - i) \times \varphi_l \times q_l. \tag{9}$$

The reward rate of sub-rating is

$$\sum_{i=0}^{n_m}\left[\sum_{k=0}^{\Omega(i)} k\mu \times (v_h - v_l)\frac{\frac{1}{k!}\left(\frac{\Lambda(i)}{\mu}\right)^{k}}{\sum_{j=0}^{\Omega(i)}\frac{1}{j!}\left(\frac{\Lambda(i)}{\mu}\right)^{j}}\right]. \tag{10}$$

The overall pay-off rate can be approximated as (2) + (3) + (7) + (10) − (8) − (9).

Method II

Another approach to modeling the sub-rating process is to calculate the pay-off rate for each $P(i, n_m - i)$. The sub-rating process is modeled as a M/M/$\Omega(i)$/$\Omega(i)$ queuing system with the arrival rate of φ_h. The penalty rate of high-priority clients can be expressed by Eq. 8, where $\Lambda(i)$ is replaced with φ_h. Similarly, the reward rate of high-priority clients can be expressed by Eq. 10, where $\Lambda(i)$ is replaced with φ_h. The pay-off rate of high-priority clients in the system with sub-rating is

$$\sum_{i=0}^{n_m} P(i, n_m - i)\left[\sum_{k=0}^{\Omega(i)} k\mu \times (v_h - v_l)\frac{\frac{1}{k!}\left(\frac{\varphi_h}{\mu}\right)^{k}}{\sum_{j=0}^{\Omega(i)}\frac{1}{j!}\left(\frac{\varphi_h}{\mu}\right)^{j}} - \varphi_h \times q_h \times \frac{\frac{1}{\Omega(i)!}\left(\frac{\varphi_h}{\mu}\right)^{\Omega(i)}}{\sum_{j=0}^{\Omega(i)}\frac{1}{j!}\left(\frac{\varphi_h}{\mu}\right)^{j}}\right]. \tag{11}$$

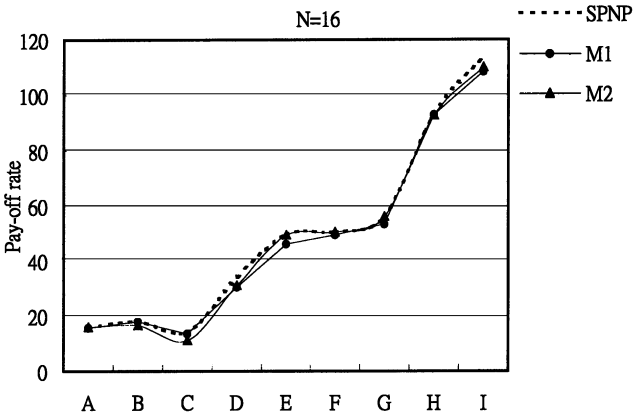
Combining Eqs. 2, 3, 7, 9, and 11, the overall pay-off rate of a system using the dynamic quota-based admission control with sub-rating can be approximated as (2) + (3) + (7) + (11) − (9).

6 Numerical experiments

Consider an example of building an on-demand multimedia system on CATV network [17]. The system delivers the multimedia services via the hybrid fiber coaxial (HFC) access network. Clients with different priority levels enter the system via a QoS manager. The main responsibility of the QoS manager is admission control and dynamic resource (i.e., network bandwidth and/or server storage) allocation. An array of 64 disks each with a storage capacity of 1 GB can be implemented in the server, as indicated from the experiment results [6]. Continuous media blocks with 512 KB each of a video stream are randomly stored on the disk. The playback rate is assumed to be 30 frames/s per client request. For such a physical configuration, the maximum number of clients that concurrently exist in the system was found to be near 16 if strict deterministic admission control is performed. The number of clients could be up to 32 if video compression is applied. According to our cost model, high-priority clients contribute a higher reward and incur a higher penalty if rejected. Workload characteristics of such a system are changeable such that a static admission control algorithm is infeasible and unable to adapt to run-time changes.

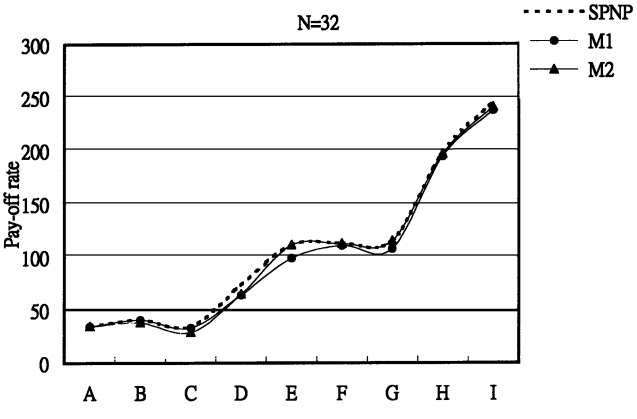
Admission control with sub-rating (SUB) can be implemented in a multimedia system as stated above. One challenge facing the SUB algorithm is dynamic partitioning of the system resource as workload changes. An optimal setting of the quota values for any workload conditions shall be found so as to maximize the system pay-off rate. One way to dynamic partitioning is to identify the possible workload conditions before the system is up for service. Time complexity is the main concern of solving the SPN model by the SPNP. The experiments are run on a SUN Ultra-1 model 140 machine equipped with a 143-MHz UltraSPARC processor, 32 MB memory, and 2.1 GB FAST SCSI-2 hard disk. On average, it takes 94 and 6678 s (i.e., approximately 1 h 50 min) to find out the optimal settings, for N = 16 and N = 32 respectively. For such a reason, the optimal quota values are obtained from the SPNP tool before run time, for each identified workload. The optimal settings are maintained in a table such that the QoS manager is capable of looking up the table to accordingly re-configure the resource partition at run time, upon a workload change. The limitation of such an approach is the contents of the look-up table. In an event of a sudden change that was not identified beforehand, the SPNP-approach is unable to respond in real time. Consequently, the SPNP-approach falls apart.

On the other hand, the approximation approaches are capable of finding sub-optimal solutions in real time, as workload changes. The optimal quota values found by Methods I and II could be different from those by the SPNP. Let the optimal settings found by the SPNP, Methods I and II be x_1, x_2, and x_3, respectively, given a workload condition. Note that x_2 (or x_3) being the optimal setting of Method I (or II) means that the pay-off value of x_2 (or x_3) calculated by the method is the maximum. However, it is not true in the real case. The true pay-off rate of x_2 should be the one obtained by solving the SPN model when the partition is specified according to the values in x_2. Therefore, the maximum pay-off values by Methods I and II are calculated by mapping their optimal quota values to the SPNP.



A = (5,10,1,2,1,2,1,2) D = (5,10,1,5,1,2,1,2) G = (5,10,1,10,1,2,1,2)
B = (10,10,1,2,1,2,1,2) E = (10,10,1,5,1,2,1,2) H = (10,10,1,10,1,2,1,2)
C = (15,10,1,2,1,2,1,2) F = (15,10,1,5,1,2,1,2) I = (15,10,1,10,1,2,1,2)

Fig. 7. Approximation results for N = 16



A = (10,20,1,2,1,2,1,2) D = (10,20,1,5,1,2,1,2) G = (10,20,1,10,1,2,1,2)
B = (20,20,1,2,1,2,1,2) E = (20,20,1,5,1,2,1,2) H = (20,20,1,10,1,2,1,2)
C = (30,20,1,2,1,2,1,2) F = (30,20,1,5,1,2,1,2) I = (30,20,1,10,1,2,1,2)

Fig. 8. Approximation results for N = 32

Experiment results are illustrated in Figs. 7 and 8. They demonstrate that the system performance (pay-off rate) by the approximation methods is very close to that by the SPNP. Method II (short for M2 in the figures) performs slightly better than Method I (short for M1 in the figures). For N = 16, the average performance difference between the SPNP and M1 is 3.88%, while that between the SPNP and M2 is 2.83%. For N = 32, the difference between the SPNP and M1 is 4.53% on average, while that between the SPNP and M2 is 2.48%. The performance differences between the SPNP and the approximation methods are within a reasonable range.

7 Conclusions

In this paper, we have investigated the admission control problem for systems with two classes of client requests by

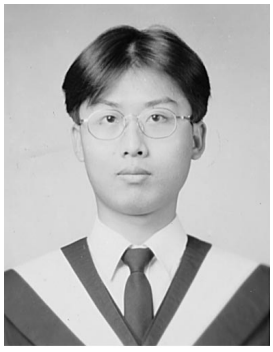
exploiting a cost model. In our cost model, each class of request has its reward and penalty to the system. High-priority requests are associated with high reward and penalty values. We have proposed an admission control algorithm with sub-rating mechanism and investigated its performance. Sub-rating attempts to accept high-priority requests under heavy and over-loaded systems, by lowering the service requirements of some low-priority requests. The experimental results demonstrated that the sub-rating mechanism can significantly improve the system performance. An SPN model is used to find optimal solutions and the approximation approaches are developed to find sub-optimal ones. The results showed that the sub-optimal solutions found by the proposed approximation methods are very close to optimal ones. Therefore, a multimedia server can exploit the approximation methods to dynamically adjust quota values based on the characteristics of the workload in order to achieve high system performance.

Some future research areas include (a) extending sub-rating to a system with multiple priority classes, and (b) changing the mandatory sub-rating mechanism to a voluntary degradation one, in which the low-priority clients have options either to keep their QoS levels or to accept the degradation in an altruistic fashion.

References

1. Ramanathan S, Rangan PV (1994) Architecture for personalized multimedia. IEEE Multimedia, pp 37–46
2. Vin HM, Goyal A, Goyal P (1995) Algorithms for designing multimedia servers. Comput Commun 18: 192–203
3. Chang E, Zakhor A (1996) Cost analysis for VBR video servers. IEEE Multimedia, pp 56–71
4. Rangan P, Vin HM (1991) Designing File Systems for Digital Video and Audio. In: 12th ACM Symposium on Operating Systems, 1991
5. Vin HM, Goyal P, Goyal A (1994) A statistical admission control algorithm for multimedia servers. ACM Int. Conference on Multimedia, 1994, San Francisco, Calif.
6. Chen I, Chen C (1996) Threshold-based admission control policies for multimedia servers. Comput J 39(9): 1–10
7. Vin HM, Goyal A, Goyal P (1995) An observation-based admission control algorithms for multimedia servers. 1st IEEE Int. Conference on Multimedia Computing and Systems, 1995, Boston, Mass.
8. Trivedi KS, Ciardo G, Muppala JK (1991) Manual for the SPNP Package. Dept. of Electrical Engineering, Duke University, Durham, N.C
9. Kleinrock L (1975) Queuing Systems, Vol. 1: Theory. John Wiley & Sons, Chichester
10. Chen M, Hsiao H, Li C, Yu P (1997) Using rotational mirrored declustering for replica placement in a disk-array-based video server. ACM Multimedia Syst 5: 371–379
11. Vin HM, Rangan P (1993) Designing a multi-user HDTV storage server. IEEE J Sel Areas Commun 11(1): 153–164
12. Oomoto E, Tanaka K (1993) OVID: Design and implementation of a video-object database system. IEEE Trans Knowl Data Eng 5: 629–643
13. Oyang Y, Wen C, Cheng C, Lee C, Li J (1995) A multimedia storage system for on-demand playback. IEEE Trans Consum Electron 41: 53–64
14. Bestavros A, Braoudakis S (1995) Value-congnizant speculative concurrency control. In: Int. Conference on Very Large Databases, 1995
15. Locke C (1986) Best effort decision making for real-time scheduling. Ph.D. thesis. Carnegie-Mellon University, Dept. of Computer Science, Philadelphia, Pa.
16. Mercer CW, Savage S, Tokuda H (1994) Processor capacity reserves: Operating system support for multimedia applications. In: 1st IEEE Int. Conference on Multimedia Computing and Systems, 1994, Boston, Mass. pp 90–99
17. Kuo Y, Lee W, Cheng W, Homg M (1998) The Implementation of Intelligent Service Navigator for Virtual Club Multimedia Service System on CATV Network. In: Workshop on Distributed System Technologies & Applications, 1998, Taipeh, Taiwan, pp 393–401

SHENG-TZONG CHENG received the BS (1985) and MS (1987) in Electrical Engineering from the National Taiwan University, Taipei, Taiwan. He received the MS (1993) and PhD (1995) in Computer Science from the University of Maryland, College Park, Md. He was an Assistant Professor of Computer Science and Information Engineering at the National Dong Hwa University, Hualien, Taiwan, in 1995, and became an Associate Professor in 1996. He is currently an associate professor in the department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan. His research interests are in design and performance analysis of mobile computing, wireless communications, multimedia, and real-time systems.



CHI-MING CHEN was born in Taiwan, R.O.C. He received the B.S. degree in computer and information sciences from Tung-Hai University, Taichung, Taiwan, R.O.C., in 1995. He is currently pursuing the Ph.D. degree at the National Cheng Kung University, Tainan, Taiwan. His research interests include mobile computing, wireless communications, and performance evaluation.

ING-RAY CHEN received the BS degree from the National Taiwan University, Taipei, Taiwan, in 1978, and the MS and PhD degrees in computer science from the University of Houston, University Park, Houston, Tex., in 1985 and 1988, respectively. He is currently an associate professor in the Department of Computer Science, Virginia Polytechnic Institute and State University. His research interests are in reliability and performance analysis, mobile computing, multimedia, and distributed systems. Dr. Chen is a member of the IEEE/CS and ACM and a member of the editorial board of The Computer Journal.