Diff-EDF: A Simple Mechanism for Differentiated EDF Service *

HaiFeng Zhu[1], John P. Lehoczky[2], Jeffery P. Hansen[3], Ragunathan Rajkumar[1]
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Many existing and emerging network applications such as voice-over-ip, videoconferencing and online gaming have end-to-end timing requirements. Despite the real-time demands of these applications, they are usually deployed on best-effort networks such as the Internet. This results in unpredictable and often unsatisfactory performance. In this paper we propose a simple and novel task (or packet) scheduling algorithm Diff-EDF (Differentiated Earliest Deadline First) which can meet the real-time needs of these applications while continuing to provide best effort service to non-real time traffic. In our system we consider each flow as having stochastic traffic characteristics, a stochastic deadline and a maximum allowable miss rate. The Diff-EDF service meets the flow miss rate requirements through the combination of an admission control test and a scheduling algorithm similar to EDF (Earliest Deadline First). However, unlike standard EDF scheduling each flow receives a deadline bias based on the flow's miss rate requirement. Applying this bias allows the miss rate to be controlled on a flow-by-flow basis. Both the admission control test and the bias selection algorithms can be computed as a linear function of the flow traffic parameters and the logarithms of the miss rate requirements resulting in an efficient implementation. In this paper, we presented the proposed system structure, protocols, algorithms, analysis and experiments. Experiments with randomly generated and real-life data closely match values predicted by the theory.

1 Introduction

Applications such as VoIP (Voice over IP), online gaming, streaming media, and virtual-presence video services generate some of the most demanding network traffic with stringent end-to-end timing needs. Other applications such as online shopping must provide acceptable response times to users even under high loads. Service agreements between the network provider and the customer typically quantify the Quality of Service parameters that must be satisfied for these applications. These QoS parameters are often satisfied by over-provisioning network resources based on absolute worst-case traffic conditions. While end-to-end timing constraints can be satisfied by such over-provisioning, this leads to inefficient, inflexible and pessimistic use of network capacity.

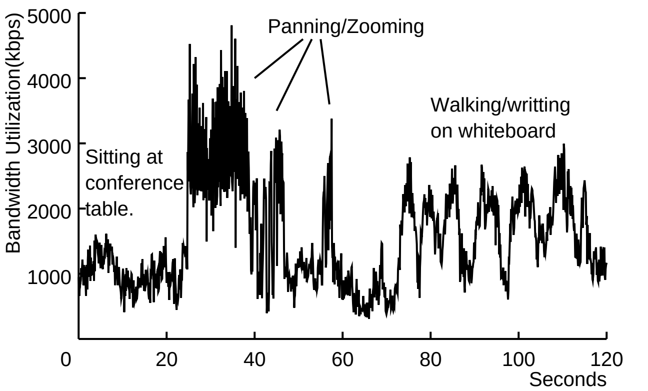

Figure 1. Bandwidth consumption of a video-conference flow

In some cases using over provisioning to meet worst case requirements can drastically increase the resource needs compared to what is needed for meeting average case requirements. As an example taken from [8], consider a set of video-conference applications. Video-conference applications are generally soft-real-time (some packets can be late or even lost without a significant reduction in QoS) and bandwidth consumption is highly dynamic (Figure 1). Suppose that all connections pass through two routers, a high speed router with uniformly distributed service times between 0.1ms and 3.5ms and a low speed router with uniformly distributed service times between 0.57ms and 20ms. Now suppose each flow has a packet inter-arrival time that is uniformly distributed between 20 ms and 180 ms.

Figure 2 shows the number of flows that can be admitted using a hard deadline requirement (the curve labeled Abs.QoS) compared with the number that can be accepted with a 10^{-7} miss rate (the curve labeled Stat.QoS) plotted as a function of the end-to-end deadline. This example shows that by tolerating just a minimal packet miss rate, an order of magnitude increase in the number of flows that can be accepted can be achieved.

*This work was supported in part by the DoD MURI program administered by the Office of Naval Research under Grant N00014-01-0576. 1. Department of Electrical and Computer Engineering, {*haifengz,raj*}*@ece.cmu.edu*, 2. Department of Statistics *jpl@stat.cmu.edu.*, 3. Institute for Complex Engineered Systems *hansen@cmu.edu.*

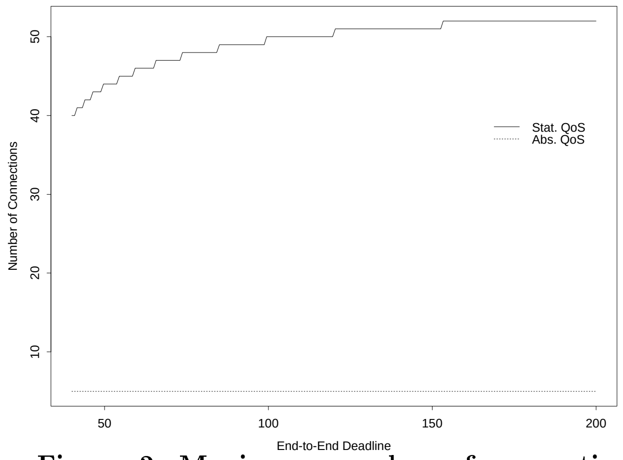

Figure 2. Maximum number of connections (Deadline Miss Rate is bounded by 10^{-7})

Considerable research effort on resource and QoS management has been conducted to address the efficient use of resources ([2] [4] [5] [10] [12]). Of those that have looked at network resources, most of these consider only the bandwidth issues while timing issues have for the most part been treated as a second order effect. In this paper, we focus primarily on meeting application timing requirements. Since most real-time network applications can tolerate a small fraction of their packets arriving late or being lost, we employ a soft real-time model and focus on controlling the lateness rather than forcing an inefficient hard real-time solution. Since many real-time applications have dynamic resource needs, our traffic model allows each flow's packet inter-arrival time, packet size (service time) and deadline to be stochastic, characterized by some distribution (or possibly a constant). We also assume each flow has its own miss rate (fraction of packets that are late or lost) requirement.

One scheduling algorithm that is frequently proposed for meeting real-time deadlines is EDF (Earliest Deadline First). In an operating system, each task has its own deadline by which it must be completed. A task can be preempted by other tasks. In a network, each packet has a deadline by which it must arrive to the destination. A priority queue is used to service packets/task in the order of their absolute deadlines. Packets with short lead-times (time until their deadline expires) receive higher priority, but unlike a static priority scheme even tasks that were low priority on arrival will eventually have short lead-times and thus are guaranteed not to starve.

In [9], a soft-real-time EDF algorithm is designed for four classes of tasks. However, none of these four classes allow arbitrary task execution times. In particular, when the task arrivals are random as is specified in the fourth class called ASCU(Aperiodic/Sporadic Constant Utilization), the approach presented requires the execution times to be a constant fraction of the inter-arrival time. This is a very stringent constraint that limits the use of these methods for many practical scenarios.

One problem with EDF scheduling is that all flows receive the same miss rate regardless of their traffic characteristics and deadlines. This makes the standard EDF algorithm unsuitable for situations in which the different flows have different miss rate requirements since in order to meet all miss rate requirements it is necessary to limit admissions so as to satisfy the flow with the most stringent miss rate requirement.

In order to solve this problem, we propose a Differentiated EDF service, Diff-EDF, that allows the flow miss rate requirements to be independently specified. The service is comprised of an admission control test, and a scheduling algorithm. The approach used in our Diff-EDF algorithm is to use a *shadow deadline* for each flow by adding or subtracting a bias based on each flow's traffic characteristics and miss rate requirements. Packets or tasks are then scheduled using EDF as if the shadow deadline were the real deadline. Flows with a low or negative bias relative to other flows will have lower miss rates, and flows with higher biases relative to other flows will have higher miss rates.

In order to demonstrate the effectiveness of Diff-EDF we present an analysis of its performance and validate the analysis through simulation studies. We also discuss issues relevant to a system implementation. The analysis is based on Real-Time Queueing Theory (RTQT) [6]. This is a relatively new analytic methodology that assumes that tasks have timing requirements and focuses on the deadline miss rate as its performance measure. The analysis is done assuming the system is heavily loaded, that is that the system utilization is between 90% and 100%.

The rest of the sections will be arranged as follows: Section 2 introduces the system structure, Section 3 presents the Diff-EDF algorithm, Section 4 gives the analysis of the algorithm, section 5 presents the supporting experiment results, and finally, Section 6 presents our conclusion.

2 System Structure of Diff-EDF

If a system is heavily loaded, and task arrivals and service requirements are random, then there can be transient intervals of overload, and some tasks (or packets) will miss their deadlines. The goal of the Diff-EDF scheduling algorithm is to guarantee that a flow's deadline miss-rates meets its pre-specified QoS requirements. In static priority schemes, low-priority flows can starve. While this provides stable QoS guarantees to high-priority flows, the deadline miss-rates of low-priority flows can be unacceptably high. It would be desirable to have a scheme which allows the individual deadline miss-rates of different flows to be distinct and controllable. Our proposed Diff-EDF scheme satisfies this objective.

A Diff-EDF system is the same as a normal EDF queue except that it has a *marker* that adjusts the deadlines of incoming tasks (or packets) by adding a constant to the relative deadlines before the tasks are placed into the EDF queue. Different constants are added to different flows. We

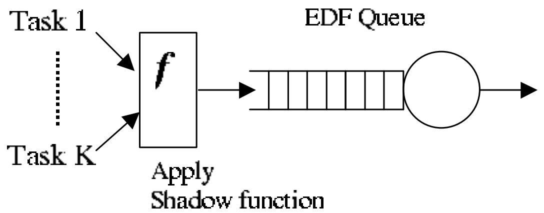

Figure 3. Diff-EDF Scheduler

refer to these modified deadlines as "effective deadlines". Diff-EDF maintains a single EDF priority queue, enqueueing tasks in ascending order of their effective deadlines. Deadline miss-rates will still be computed on the basis of whether a task/packet met its original deadline (as opposed to its effective deadline).

We assume there are K distinct task flows where flow j requires a deadline miss rate $\leq \hat{\phi}_j$ (for flow j). Each task's utilization is ρ_j, and the total system utilization is $\rho = \sum_{j=1}^{K} \rho_j, j = 1, \ldots, K$.

Figure 3 shows Diff-EDF structure. The marker applies adjustments to the deadlines. The adjustment is made using a *deadline adjustment function* (or *shadow function*) which adds or subtracts a constant, specific to the flow, to the task's deadline. This effective deadline is used by the Diff-EDF scheduler. (The scheduler need not know the true deadlines. However, it must be noted that the deadline miss-rates are based on the true deadlines and not the effective deadlines). Specifically, suppose D_j is the deadline of a task from the j^{th} flow. Then the shadow deadline is given by $D_{s,j} = D_j + B_j$ for some constant B_j. This adjustment is easy to implement, and the idea can also be applied to other scheduling algorithms such as deadline-monotonic or weighted fair queueing.

Diff-EDF essentially uses EDF scheduling, thus allows deadline misses while accepting high intensity traffic. Since we do not perform admissions control through worst case analysis, as shown in Section 1, high utilization (for example, over 90%) can be achieved by Diff-EDF with negligible deadline misses.

Diff-EDF can be implemented in the operating system or for Internet-based applications. Figure 4 shows the Differentiated Service (Diffserv) architecture of the Internet [1]. This architecture was proposed to allow the support to per-flow or per-flow aggregate quality of service in a scalable manner. It is usually realized through different treatments to different classes of traffic in the high speed core of the network and perform per-flow treatment only at the edge which are usually connected to slow access links. For Diff-EDF, the marker resides in the edge router that maintains a flow table that keeps the constants for the various flows, B_j, $j = 1, \ldots, K$. According to Diffserv, the edge router sits on the edge links that are usually much slower, thus is able to perform per-flow operations. Each packet has a relative deadline that can encoded in a field of the IP packet header by the user host before being sent into the network. In practice there may be a limited number of bits available to encode this deadline. While this appears to be an impediment to a practical EDF implementation, it was shown in [13] that only three bits (such as the 6-bit DS/ToS field) of deadline information are required to achieve nearly the same real-time performance as unlimited granularity EDF. The limited priority scheme is referred to as Q-EDF (Quantized EDF) and is easily implementable within the DiffServ framework. Whenever a packet comes in through the edge router, the marker looks in the flow table for the appropriate B_j for that flow and adds it on the packet's relative deadline to produce the shadow deadline. This deadline is then written back to the deadline field (such as the ToS/DS field) of the IP packet. Within the network, due to the high speed in the backbone, the core routers cannot maintain a flow table. However, it can simply read the deadline field of each packet which now contains the shadow deadline and schedule it using ordinary EDF. This avoids the use of flow tables in the core, and fully conforms to the Diffserv Internet standard.

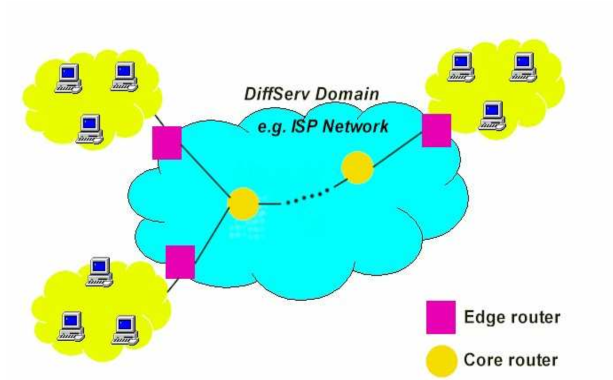

Figure 4. Internet Differentiated Service Architecture

Simple versions of Diff-EDF may also be of practical value. For example, one can simply divide network traffic or stochastic tasks into two classes, real-time and non-real-time (also known as best effort). For real-time flows, we need to guarantee that the maximum deadline miss-rate is not greater than $\hat{\phi}$, a user-specified parameter. For best-effort flows, using Diff-EDF, the deadline miss-rates will become whatever rates can be achieved by designating their maximum tolerable deadline miss-rate to be 1. In other words, best-effort flows need not be starved and their deadline miss-rates will improve to the point where any further improvements will begin to affect the real-time flows.

3 Diff-EDF Protocol and Algorithm

3.1 Admission Control

Each user's request is processed dynamically by the Admission Control (AC) system component *Diff-EDF-AC*. One approach is to have this component reside in the same place as the marker in the edge router. A user then submits a request for a task, either through a system call, a user-level call or a through socket level connections, along with the required parameters including the maximum tolerable deadline miss rate, the task deadlines, the mean and variance of the task inter-arrival and service distributions.

For simplicity, we assume deadlines are constant, although the theory allows for random deadlines. In this case, only the mean deadline needs to be specified. The Diff-EDF-AC maintains the *task list table* (or *flow table*) that keeps all flow information. This table can reside in cache, main memory, or vitural memory, depending on how fast the system requires the requests to be processed. After receiving such a request, the Diff-EDF-AC checks the task list table, applies the Diff-EDF admissions control algorithm, and notifies the user whether or not the request can be satisfied. The notification can be done through return values in function calls or socket communications.

On receiving a QoS request (for timing guarantee) for a flow, the Diff-EDF-AC does the following things:

1. Computes the first two moments (mean and variance) of the inter-arrival time and the service (execution) times of the tasks in this flow, from the user's request.

2. Computes the current system's total utilization ρ (including the flow to be admitted). If $\rho > 1$, reject the request.

3. Computes a workload traffic parameter θ and applies the admissions test of Theorem 2. Details of the computation of θ and the admissions test will be given in Section 4. The request is accepted if and only if the admissions test is passed.

4. If the request is accepted, use Equation 5 to to compute the B_js and add them to the actual deadlines for scheduling.

An important thing to notice here is, our admissions control procedure does not require complex information or overly restrictive assumptions. It simply requires the engineers to obtain the mean and variance of the inter-arrival time and service time of the tasks in the flow. For example, if this flow is a movie stream, the engineer can simply stream out this movie from the computer over the network, and use simple tools like `tcpdump` to monitor the packets' arrival time and sizes. By computing the mean and variance of the time between packets (i.e. inter-arrival time) and the packet sizes divided by bandwidth (i.e. service time), the first two moments in item 1 above can be obtained. Note that the service time on different links depends on the link bandwidth which should also be obtained by the engineer.

3.2 Diff-EDF Task Processing

On receiving each task from a task flow (say, task flow $_j$), Diff-EDF performs the following operations:

1. Identify the associated flow for the task using the task ID label and look up the adjustment constant B_j. The relative deadline is changed to the shadow deadline $D_{s,j} = D_j + B_j$.

2. Perform the ordinary EDF scheduling using the shadow deadline. That is, the task's absolute deadline is now given by $d_{s,j} = D_{s,j} + t_a$, where t_a is the task's arrival time, and insert the task into the EDF queue in the order of increasing absolute deadline (smallest absolute deadline is at queue head thus served the first)

3.3 Task Completion

When all tasks of a particular task are completed, the user sends the Diff-EDF-AC service completion request for $Task_j$. The Diff-EDF-AC erases the record associated with this task and deletes its related parameters from the task list table. Then it modifies the θ traffic parameter and the total utilization ρ, and updates the B_j values in the task list table according to Equation 5.

It is also possible that the service completion request is then further sent to an accounting module for billing purposes which we will not discuss in detail here.

4 Analysis of the Diff-EDF Algorithm

4.1 Assumptions and Notation

We begin by introducing some assumptions and notation.

Assume we have K task flows and we want to determine whether they can be scheduled so that their QoS requirements are met. Each flow j is characterized by:

- A task inter-arrival distribution given by a c.d.f. F_{aj} with mean $1/\lambda_j$ and standard deviation σ_{aj}. Let $C_{aj}^2 = \sigma_{aj}^2 \lambda_j^2$, the squared coefficient of variation of F_{aj}. Let $\Lambda = \sum_{i=1}^{K} \lambda_i$, the total arrival rate.

- A task service requirement distribution given by a c.d.f. F_{sj} with mean $1/\mu_j$ and standard deviation σ_{sj}. Let $C_{sj}^2 = \sigma_{sj}^2 \mu_j^2$, the squared coefficient of variation of F_{sj}.

- A soft deadline $D_j > 0$. Note, if the deadlines for the tasks of a particular flow are random drawn from a distribution, G_j, then D_j represents the mean of G_j. Let $\overline{D} = \sum_{i=1}^{K} \lambda_i D_i / \Lambda$, the mean task deadline averaged over all flows.

- Define $\rho_j = \lambda_j / \mu_j$ the traffic intensity of flow j, and $\rho = \sum_{j=1}^{K} \rho_j$, the total traffic intensity. In addition, we define $\alpha_j = \rho_j / \rho$, the fraction of the traffic intensity that is attributed to flow j.

- A QoS requirement, $\hat{\phi}_j$, interpreted as the requirement that the long run average fraction of flow j's tasks missing their deadline D_j must not exceed $\hat{\phi}_j$.

The RTQT analysis used in this paper models the workload process as a Brownian motion with drift $-\theta$ where $\theta = \frac{2(1-\rho)}{\sum_{i=1}^{K} \lambda_j (C_{aj}^2 + C_{sj}^2)/\mu_j^2}$. The stationary distribution of the workload process is exponential with mean $1/\theta$.

4.2 Deadline Miss Rate Prediction

The following is based on an RTQT analysis of the EDF algorithm. The basic methodology is not presented here, but the interested reader may refer to [6], [7], [8], or [13]. For EDF scheduling, [13] showed that all flows have the same deadline miss rate in this case given by $e^{-\theta\overline{D}}$. Using Diff-EDF, we will adjust the deadlines of each flow j by adding a constant B_j to the deadline (note B_j can be negative). If we adjust all deadline D_j to become $D_j + B_j$, what will happen to the deadline miss rates ϕ_j, $j = 1, \cdots, K$?

Suppose that only task j's deadline is changed, and the others remain unchanged. This will change the average deadline to $\overline{D'} = \overline{D} + \alpha_j B_j$. If $B_j > 0$, then $\overline{D'} > \overline{D}$, and the deadline miss rate for all tasks other than j will be given by $e^{-\theta\overline{D'}}$ which is smaller than its previous value. Thus all of the deadline miss rates for the tasks other than j will decrease. Now consider task j. Its deadline has been artificially increased by B_j, but its true deadline is unchanged. Consequently, since the scheduling is being done using the deadline $D_j + B_j$, it must finish with a lead-time of B_j or greater in order to actually meet its D_j deadline requirement. This means that its deadline miss rate will be given by $e^{-\theta(\overline{D'}-B_j)} = e^{-\theta(\overline{D}+\alpha_j B_j - B_j)}$. This is greater than its former value by a factor $e^{-\theta(1-\alpha_j)B_j}$.

Conversely, had B_j been negative, then the deadline miss rates of all tasks other than j would increase, while the deadline miss rate of task j would decrease. The results are intuitively clear. If we add $B_j > 0$ to flow j and leave the other flows untouched, then flow j has been made relatively less urgent than the other flows. In the Diff-EDF queue, task j will be somewhat less important. As a consequence, its actual miss rate will increase, while the other miss rates will decrease.

The above analysis is for a change in only one flow. The argument can be easily generalized. Suppose that we add B_k to the deadlines of flow k. This changes the mean deadline from $\overline{D}$ to $\overline{D'} = \overline{D} + \sum_{k=1}^{K} \alpha_k B_k$. Now, without any adjustments task j would be late if its exiting lead-time was smaller than 0. With the adjustments, it will be late if and only if its exiting lead-time is smaller than B_j. The probability that upon exiting flow j has a lead-time smaller than B_j is given by $e^{-\theta(\overline{D'}-B_j)}$.

It is important to realize that the above analysis breaks down when one of the parameter B_j grows large relative to the others. When this happens, the flow with the large B_j will rarely, if ever, have lead-times that are less than those of the other flows. In this case, flow j has, in effect, become a low priority flow running in background relative to the other flows. When this happens, the RTQT formula referred to above is no longer correct and it must be replaced by a formula that this priority queueing effect into consideration. This will be discussed in the next section. We conclude this section with a theorem that characterizes the deadline miss rates for the Diff-EDF algorithm when the B_js are not extreme so no flow has low priority status.

Theorem 1 (Diff-EDF Deadline Miss Rate) *In Diff-EDF, if adding a constant B_j is added to flow j's relative deadline to produce a shadow deadline for scheduling, the resulted RTQT approximation of this flow's deadline miss rate is:*

$$\phi_j = e^{-\theta(\overline{D'}-B_j)} \tag{1}$$

where $\overline{D'} = \sum_{j=1}^{K} \alpha_j (D_j + B_j)$ □

It remains to determine the feasible region for the user-specified QoS levels, $(\hat{\phi_1}, \hat{\phi_2}, \cdots, \hat{\phi_K})$, and the set of values of the B_js that will achieve a set of feasible QoS levels.

4.3 Feasible Region

Now we move to the determination of the feasible region and the calculation of the B_j values that will achieve a particular QoS level.

We will illustrate this problem in the two flow case for easy understanding and then extend it to K flow case. According to Theorem 1,

$$\phi_1 = e^{-\theta(\overline{D}-\alpha_2(B_1-B_2))} \tag{2}$$

and

$$\phi_2 = e^{-\theta(\overline{D}+\alpha_1(B_1-B_2))} \tag{3}$$

We can see that the deadline miss rate depends only on the difference $B_1 - B_2$, not on the individual values. This means one of them is redundant. It is useful to have all initial deadlines be positive. This can be achieved by identifying the flow with the smallest QoS specification and set its B value to 0. The B value for the other flow will be positive. Assume we want $\phi_1 \le \phi_2$ and set $B_1 = 0$. Taking logarithms linearizes the problem. We find:

$$(-\log(\phi_1), -\log(\phi_2)) = (\theta(\overline{D} + \alpha_2 B_2), \theta(\overline{D} - \alpha_1 B_2)).$$

These points fall on a line given by

$$\alpha_1(-\log(\phi_1)) + \alpha_2(-\log(\phi_2)) = \theta\overline{D}.$$

This line has slope $-\alpha_1/\alpha_2$ and goes through the point $(\theta\overline{D}, \theta\overline{D})$ when $B_2 = 0$. Exponentiation allows us to write an equivalent expression, $\Pi_{i=1}^{2} \phi_i^{\alpha_i} = e^{-\theta\overline{D}}$.

If the desired values $\hat{\phi_1}$ and $\hat{\phi_2}$ satisfy this equation, then there is a unique value of B_2 that will give these two values exactly. If the desired values leave some slack, then one or both of the deadline miss rates will be smaller than that specified by the user, i.e. its QoS will be improved. An additional criterion is needed to determine how this "bonus" QoS should be allocated to the two users. We will shortly see this analysis easily generalizes to the case of K flows.

Now, the analysis presented always uses the RTQT result for EDF scheduling; however, this is only an approximation. When B_2 converges to $\pm\infty$, the system behaves as a true static priority system in which one flow always has higher priority than the other. Moreover, as B_2 moves away from 0, the scheduling algorithm begins to deviate from EDF and slowly moves toward static priorities.

To see this, consider the following simple illustration. Suppose we have two flows, each with deadlines of 30. Now suppose we add 30 to flow 2 (i.e. set $B_2 = 30$), so its shadow deadline is 60. The two flows are processed using EDF. Each task from flow 2 will enter the system with a deadline of 60, and it will have lower priority than any task from flow 1 (which enter with deadline 30), until it has resided in the system for at least 30 time units. But after 30 time units, any task from flow 2 has missed its actual deadline. Consequently, it is behaving exactly like a static priority system while its lead-time is positive. Tasks from flow 2 can finish before their actual deadlines only if they were lower priority than all tasks from flow 1 in the system while they were active. The resulting system is, however, not strictly static priority either, since the tasks from flow 2 that are late eventually have short enough lead-times that they will have higher priority than flow 1 tasks. Consequently the system is not completely equivalent to a static priority system. The overall effect is to create a total deadline miss rate that is higher than it would have been if one used pure EDF or pure static priorities. While this is an important effect, if deadline miss rates are small, then the increase is very small. Moreover, Diff-EDF has the advantage that it can meet individual users' QoS requirements.

One can provide an RTQT-based analysis that will include this priority queueing effect. As we let B_2 become extremely large, the system converges to a true static priority queue with flow 2 having low priority. In this case, flow 1 will have a deadline miss rate that can be calculated as if it were the only flow. That will be given by $e^{-\theta_1 D_1}$ where θ_1 is the theta value considering flow 1 as if it were the only flow. The resulting deadline miss rate is usually essentially 0, since in most cases $\theta_1 >> \theta$. On the other hand, the deadline miss rate for flow 2 running as a low priority flow can be computed using RTQT to be given by $\phi_2^* = e^{-\theta\alpha_2 D_2}$. A similar analysis gives the deadline miss rate for flow 1 when B_2 is very large negatively, $\phi_1^* = e^{-\theta\alpha_1 D_1}$. Using these two static priority values, we can construct a complete feasibility analysis.

For notational simplicity let $x_i = -\log(\phi_i), C_i = \theta\alpha_i\overline{D_i}$ $i = 1, 2$. Theorem 1 based on standard RTQT analysis for EDF gives the line $\alpha_1 x_1 + \alpha_2 x_2 = \theta\overline{D}$. The static priority analysis for each flow gives $x_i = C_i$. boundary values $-\log(\phi_i) = \theta\alpha_i\overline{D_i},\ i = 1, 2$.

Consider the figure defined by the horizontal line connecting the points $(0, C_2)$ and point $1 = (\frac{\theta\overline{D}-\alpha_2 C_2}{\alpha_1}, C_2)$, the vertical line connecting the points $(C_1, 0)$ and point $2 = (C_1, \frac{\theta\overline{D}-\alpha_1 C_1}{\alpha_2})$, and the sloped line connecting point 1 and point 2. Any pair of points below this boundary is feasible. Note that this feasibility region is defined in terms of x_1, x_2 which we want to be able to make as large as possible. If we convert to a region defined in terms of ϕ_1, ϕ_2 then the feasible region would lie above the curve.

Figure 5 presents the results of an experiment with two

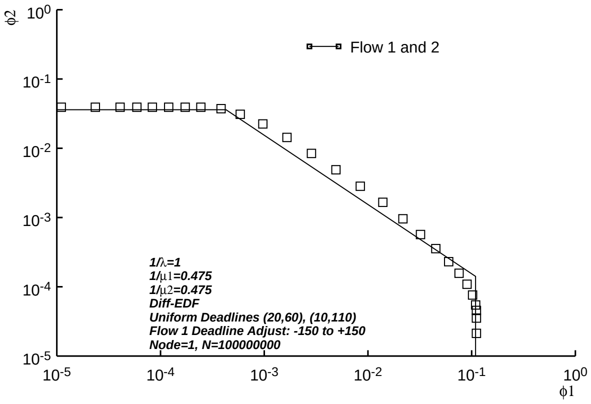

Figure 5. Feasible region of a 2-flow Diff-EDF system

flows, with their miss rates plotted as X and Y-axis on a log scale by varying B_1 from -150 to 150. It is important to remember that in this figure, the region above the curve is the feasible region, whereas the region to the right including the origin are pairs of QoS points that are not feasible. This experiment was done assuming the tasks from each flow have random deadlines to further illustrate that even though we have assumed constant deadlines for each flow, the methodology is more general. The boundary curve can be viewed as having three segments: the horizontal segment, vertical segment and "Regular Segment". Equations 2 and 3 actually predicts the regular segment. The theoretical slope of this segment is $-\alpha_1/\alpha_2$.

4.4 Diff-EDF Admission Test

We now present the admissions test and the construction of the B_i values for the case of K flows. We assume that user j, $1 \leq j \leq K$ has specified the following parameter values: $\lambda_j, \mu_j, \sigma_{aj}, \sigma_{sj}, \overline{D}_j$, and $\hat{\phi}_j$. θ_j, the traffic intensity and the theta parameter for the flow assuming it is the only flow, $\overline{D}_j$, the mean deadline for the tasks in the flow, and $\hat{\phi}_j$, the required deadline miss rate (i.e. the QoS requirement). The system takes this information from each of the flows and computes α_i, $1 \leq i \leq K$, θ, the theta parameter for the system, and $\overline{D}$, the average deadline across all flows. Define $x_i = -\log(\phi_i)$ and $C_i = \theta\alpha_i\overline{D_i}$.

Now, if we apply the usual RTQT analysis and ignore the priority queueing effects described in the previous subsection, then the x_i lie in the first octant of R^K on the hyperplane specified by the equation $\sum_{j=1}^{K} \alpha_j x_j = \theta\overline{D}$. Now, if we let $B_j \to \infty$ while keeping all the other $B_i = 0$, then flow j tasks will always have the lowest priority. In this case, we can estimate ϕ_j by $e^{-\theta\alpha_j\overline{D}_j}$. If $\hat{\phi}_j \geq e^{-\theta\alpha_j\overline{D}_j}$, then flow j is feasible even if it is scheduled at the lowest priority. Consequently the region above the hyperplane $\sum_{i=1}^{K} \alpha_i(-\log(\phi_i)) \leq \theta\overline{D}$ can be augmented by the set of points that is generated when any subset of tasks is given lower priority than the other flows, i.e. their B_j values are made very large.

The admissions test is most easily defined sequentially.

We begin by assuming there are K flows. Using that information specified by those K users, we calculate θ and $(\alpha_1, \cdots, \alpha_K)$. We first determine if any of the flows' QoS requirements can be met if the flow runs at the lowest priority. This is done by checking if any of the inequalities $\hat{\phi}_j \geq e^{-\theta\alpha_j \overline{D}_j}$, $1 \leq j \leq K$ is satisfied. If one or more of these inequalities are satisfied, we select the flow with the largest slack and assign it the lowest priority by choosing a very large value for its B. This flow is then removed from the set, reducing the set to $K-1$ flows. We repeat this sequence of steps for the reduced set of flows, first recalculating the θ and α_js, then finding any flows that can meet their QoS requirements running at lowest priority. Eventually, we will reach of set of flows that must be scheduled using EDF.

For ease of notation, suppose that none of the K flows can meet its QoS requirements running at low priority and that θ and the α_js have been determined. The resulting feasibility test is whether the user-specified QoS levels satisfy the inequality $\sum_{j=1}^{K} \alpha_j(-\log(\hat{\phi}_j)) \leq \theta\overline{D}$. We summarize this final step in the following theorem covering the case in which all flows are scheduling using EDF.

Theorem 2 (Diff-EDF Admissions Test: EDF Tasks) *A set of user-specified maximum deadline miss rates, $\hat{\phi}_j$, $1 \leq j \leq K$ can be feasibly met using Diff-EDF if*

$$\sum_{j=1}^{K} \alpha_j(-\log(\hat{\phi}_j)) \leq \theta\overline{D}.$$

Having given the feasibility test, we next turn to the determination of the B_j values that will achieve the users' QoS specifications.

4.5 Determination of the B_js

Once feasibility has been determined using Theorem 2, we must determine appropriate values for the B_j for those flows. We must solve a system of linear equation given by:

$$\sum_{i=1}^{K} \alpha_i B_i - B_j = \frac{-\log(\hat{\phi}_j)}{\theta} - \overline{D},\ 1 \leq j \leq K.$$

It turns out that there is a one-parameter family of solutions. If we difference the first and the jth equation, we find the general solution

$$B_j = B_1 + \frac{1}{\theta}\log\frac{\hat{\phi}_j}{\hat{\phi}_1},\ 1 \leq j \leq K. \tag{4}$$

It is convenient to set $B_1 = 0$ and then set

$$B_j = \frac{1}{\theta}\log\frac{\hat{\phi}_j}{\hat{\phi}_1},\ 1 \leq j \leq K. \tag{5}$$

If we number the flows so that flow 1 has the smallest $\hat{\phi}$ value, this will ensure that $B_j \geq 0$, $1 \leq j \leq K$.

Once the Bs are determined for the EDF flows, we must select B values much larger than $\max B_j$ for the flows that were selected to run at low priority.

4.6 Extension to Multi-Node

In this paper we have focused exclusively on the single node case. We briefly outline how the Diff-EDF algorithm and the associated analysis developed for the single node case can be extended to the multi-node case. In the multi-node case there are several choices for implementation of EDF depending upon whether the number of hops on the route is considered and whether the amount of time taken at previous nodes is used to alter the time remaining. The easiest to implement is one that takes into account the number of hops but does not consider the actual amount of time spent at the prior nodes. For example, suppose that a task has an end-to-end deadline of 200ms and it must traverse 5 hops. We assign a deadline of 200ms/5 = 40ms to the task. At each node, it's initial deadline is assumed to be 40ms, and it joins an EDF queue with that initial priority. We refer to this network scheduling algorithm as "simple network EDF" (also called "EDF Static" scheme [14]). We assume the routes of our traffic flows are known, on which each hop's bandwidth is also known. In practice, this can be provided by static routing techniques, or more sophisticated traffic engineering methods.

The papers [14, 15] provide an analysis of simple network EDF and a tool, RETINA, that implements that analysis. In particular, given a set of network flows over fixed, known roots with known traffic parameters, it is possible, using RTQT, to compute the individual flow lateness probabilities in the heavy traffic case. Now suppose that one adjusts the deadline of any single task. Using the available analysis, one can compute the new lateness probabilities. By adjusting the deadlines of the various tasks, one can sweep out the feasible set of lateness probabilities for all the tasks, just as was done in the single node case in this paper. Note that this implementation sets the deadline at the source, and it remains unchanged at every node on the route, and there is no need to keep track and use the actual timing at any node along the route.

5 Experiments

In this section, we will conduct several experiments to validate our analytical results to show that flows with different deadline miss-rate requirements can be supported by the appropriate use of shadow deadlines. We will use both artificially generated data and traffic patterns from real-life network experiments. Our experiments are conducted in two phases. In the first phase, we use simulation experiments with two task flows. In the second phase, we use multiple flows, some of which use real traffic pattern.

5.1 Experiments with Two Flows

In this first set of experiments, we consider two flows whose inter-arrival and service (or execution) times are exponentially distributed. All the inter-arrivals have the mean value as 1 time unit and the service times of the two flows were designed to be changed from experiment to experiment, to create different utilizations (or traffic intensities). The deadlines have uniform distributions (thus constant deadlines become a special case). By varying B_1 from -150 to 150 time-units and fixing $B_2 = 0$, we obtain a sequence of experimental deadline miss rates of $< \phi_1, \phi_2 >$ and plot them in log scale. The X-axis and Y-axis are $\log(\phi_1), \log(\phi_2)$.

The first experiment is shown in Figure 6 where the two flows are identical. The service times of both flows are 0.475 time-units leading to a total utilization ρ=95%. The deadlines are also identical, with a uniform distribution between 20 to 60 time units (a generalization of the common case of constant deadlines). As a convention in all the experiments, the dot curves represent simulation results, and the solid lines represent our theoretically predicted values. In the two-flow case, the theoretical predictions are given by Equations 2 and 3. We can see that the experimental deadline miss-rates and theoretically predicted feasible regions match very well.

Now, fixing the inter-arrival time and varying the service times and the deadlines, we created three more scenarios:

- Two flows with identical traffic and different deadlines (Figure 5)
- Two flows with different traffic and identical deadlines (Figure 7), and
- Two flows with different traffic and different deadlines (Figure 8).

In all of these experiments, we found surprisingly accurate matches between the experimental miss-rates and our theoretical predictions.

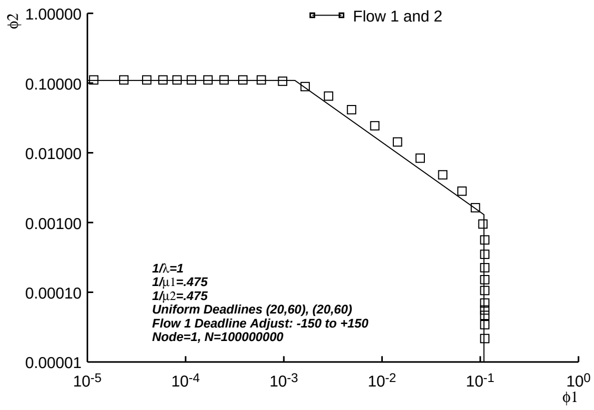

Figure 6. Feasible region of a 2-flow Diff-EDF system: Identical traffic identical deadlines

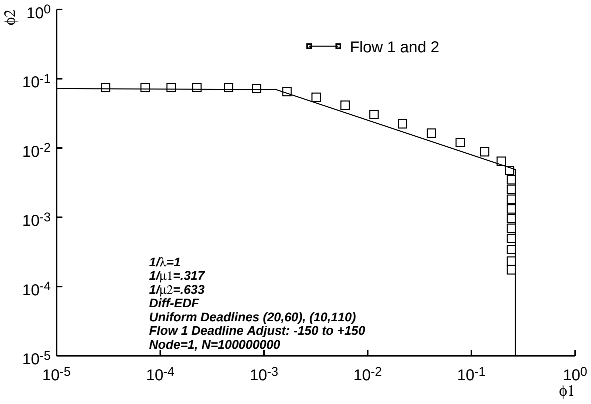

Figure 7. Feasible region of a 2-flow Diff-EDF system: Different traffic identical deadlines

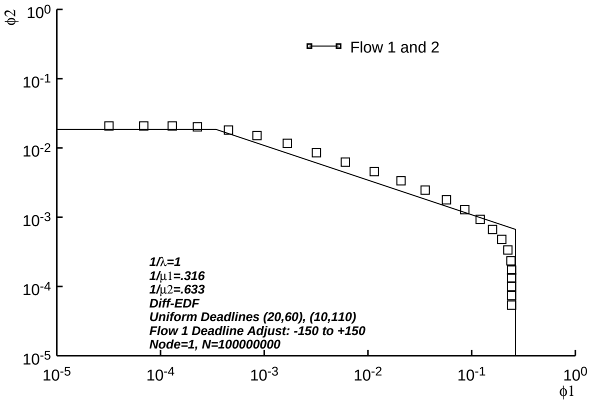

Figure 8. Feasible region of a 2-flow Diff-EDF system: Different traffic different deadlines

5.2 Multi-flow Experiments

We now consider experiments with more than two flows. We will use a data trace produced by using the Berkeley Vic [11] video-conferencing application in a real-life setting. The conference was held among a group of people at CMU. We found the traffic consumed the highest bandwidth (close to 5 Mbps) when the camera was panning or zooming. Median traffic (roughly 3 Mbps) happens when people were walking or writing on whiteboard, and low traffic (about 1 Mbps) appears when people were sitting at the conference table. The aggregate average bandwidth is 3.2 Mbps. We also add Poisson traffic simulating a specified number of pure audio channels or aggregated chunk of audios each with an average of 476 Kbps in Section 5.3 and 340 Kbps in Section 5.4 and 5.5. For the sake of convenience, we use normalized time-units. That is, when we view a flow's mean inter-arrival time of 1 ms as 1 time-unit, if the service time (transmission time in the network case) is 950 microseconds, it will be mapped to 0.95 time-units. In all the experiments, we loaded the system with very high utilization (95%) of the bandwidth and ran the simulation over a long run of at least 100 million packets.

Our goal as before is to show that Diff-EDF can be used to satisfy different deadline miss-rates for different flows.

5.3 Audio Channels

In this setting, we have two experiments each with 21 identical audio channels. After normalization, the inter-arrival becomes 1 time unit for each flow and the service time are the same across all flows, summing up to 0.95. The deadline is 3 after normalization corresponding to $47ms$ x 3 = $141ms$.

In the first experiment whose results are summarized in

Flow	Missed PKTs	Total PKTs	Resulted ϕ	Req. $\hat{\phi}$
1	59	4761045	1.239E-5	0.00001
2	200802	4764182	0.0421	0.05
3	200680	4760015	0.0421	0.05
4	201063	4763375	0.0422	0.05
5	201052	4763560	0.0422	0.05
6	200714	4757397	0.0421	0.05
7	200732	4760399	0.0421	0.05
8	200831	4763220	0.0421	0.05
9	200696	4763965	0.0421	0.05
10	201306	4760275	0.0422	0.05
11	200834	4761591	0.0421	0.05
12	201424	4763918	0.0422	0.05
13	200451	4763400	0.0420	0.05
14	200320	4760866	0.0420	0.05
15	201508	4764203	0.0422	0.05
16	200558	4762959	0.0421	0.05
17	200399	4762056	0.0420	0.05
18	201311	4763067	0.0422	0.05
19	200689	4759730	0.0421	0.05
20	200048	4763247	0.0419	0.05
21	200552	4757499	0.0421	0.05

Table 1. One High Priority Audio and 20 Low Priority Audio

Flow	Missed PKTs	Total PKTs	Resulted ϕ	Req. $\hat{\phi}$
1	76742	4761045	0.0161	0.02
2	76505	4764182	0.0160	0.02
3	76351	4760015	0.0160	0.02
4	76345	4763375	0.0160	0.02
5	75953	4763561	0.0159	0.02
6	76452	4757398	0.0160	0.02
7	76561	4760399	0.0160	0.02
8	76086	4763220	0.0159	0.02
9	76095	4763965	0.0159	0.02
10	76432	4760275	0.0160	0.02
11	357798	4761591	0.0751	0.10
12	358215	4763918	0.0751	0.10
13	357637	4763400	0.0750	0.10
14	357032	4760865	0.0749	0.10
15	359642	4764202	0.0754	0.10
16	358361	4762959	0.0752	0.10
17	358133	4762056	0.0752	0.10
18	359123	4763066	0.0753	0.10
19	358588	4759730	0.0753	0.10
20	358359	4763246	0.0752	0.10
21	358123	4757498	0.0752	0.10

Table 2. 10 High Priority Audio and 11 Low Priority Audio

Table 1, we stipulate that flow 1 is an "important" flow. We therefore want to guarantee a crystal-clear channel for flow 1 with a requested miss-rate equal to or lower than 0.00001. On the other hand, we are willing to relax the required miss-rates of the other flows as 0.05. These requirements are labeled as (Req.$\hat{\phi}$) in Table 1. Equation 5 from Section 4 is used to compute the deadline offsets to be applied to each flow. By convention, the 'important' flow with the lowest desired miss-rate is labeled as Flow 1, and gets a deadline offset B_1 of 0. We have $\theta = 1.163$, and we obtain $B_2 = B_3 = \ldots = B_{21} = 7.32$. Applying the deadline offsets, we find in Table 1 that all the flows achieve the requested delay guarantee (i.e. deadline miss rates). Flow 1 essentially has a miss-rate of close to 0 with only 6 packets missing their deadlines during the total simulation run of at least 100 million packets.

Now, suppose that we want to guarantee the first 10 flows as "important" and provide only best-effort services to the remaining 11 flows. We hope the first 10 flows to have miss rates smaller or equal to 0.02, and the rest can be up to 0.1. By Equation 5, we obtained $B_i = 0$ (for $i = 1, \ldots, 10$) and $B_j = 1.383$ (for $j = 11, \ldots, 21$). The results tabulated in (Table 2) again show that all flows achieved the requested deadline miss-rates.

5.4 High Priority Video and Low Priority Audio Channels

In this experiment, we combine one video flow with 20 audio flows. We set the bounded deadline as 282 ms (which is normalized to 6 time-units). The video flow is again obtained from a Berkeley Vic videoconference. We specify that we want to guarantee extremely high quality video that almost all packets were delivered, and we are willing to sacrifice the quality of the audio channels as necessary. With $B_1 = 0$ and $B_2 = B_3 = \ldots = B_{21} = 3$, the Table 3 again shows differentiated service is obtained. We can see that in order to guarantee the video, it can cause some loss of qualities for other channels, but the degradation is kept to the minimum levels possible.

5.5 Low Priority Video and High Priority Audio Channels

Now, suppose that we are willing to sacrifice some video quality to obtain a much better improvement for all the other 20 audio channels. This scenario can happen on voice teleconferences where only occasional image information exchange is needed. With $B_1 = 1$ and $B_2 = B_3 = \ldots = B_{21} = 0$, Table 4 shows differentiated service is again achieved. As a result, the audio has only 2.61% deadline misses.

6 Conclusion

In this paper, we have presented a novel scheduler called Differentiated EDF scheduling (Diff-EDF), which allows each flow to specify its own distinct probabilistic deadline miss-rate to while still achieving high resource utilization. Diff-EDF computes a constant offset that is added to the deadline of a packet belonging to a particular flow and produces a "shadow deadline". The new deadlines are used by a regular EDF scheduler to pick the next packet to

Flow	Missed PKTs	Total PKTs	Resulted ϕ	Req. $\hat{\phi}$
1	35211	4761053	0.007	0.009
2	518021	4764379	0.108	0.109
3	517657	4760333	0.108	0.109
4	518247	4764631	0.108	0.109
5	517440	4759231	0.108	0.109
6	517721	4762879	0.108	0.109
7	515607	4758049	0.108	0.109
8	517773	4766564	0.108	0.109
9	517357	4758200	0.108	0.109
10	515727	4761807	0.108	0.109
11	517918	4761792	0.108	0.109
12	517343	4763277	0.108	0.109
13	517062	4764149	0.108	0.109
14	515710	4757847	0.108	0.109
15	517346	4762719	0.108	0.109
16	517069	4760175	0.108	0.109
17	516827	4764791	0.108	0.109
18	518517	4763536	0.108	0.109
19	517604	4764246	0.108	0.109
20	517125	4758835	0.108	0.109
21	517182	4761486	0.108	0.109

Table 3. High Priority Video and Low Priority Audio Channels

Flow	Missed PKTs	Total PKTs	Resulted ϕ	Req. $\hat{\phi}$
1	464552	4761053	0.0975	0.099
2	124751	4764379	0.0261	0.028
3	124944	4760333	0.0262	0.028
4	125617	4764631	0.0263	0.028
5	125287	4759231	0.0263	0.028
6	124602	4762879	0.0261	0.028
7	124796	4758049	0.0262	0.028
8	124886	4766564	0.0262	0.028
9	124971	4758200	0.0262	0.028
10	124939	4761807	0.0262	0.028
11	124991	4761792	0.0262	0.028
12	125252	4763277	0.0262	0.028
13	124861	4764149	0.0262	0.028
14	124524	4757847	0.0261	0.028
15	125064	4762719	0.0262	0.028
16	124854	4760175	0.0262	0.028
17	124575	4764791	0.0261	0.028
18	125505	4763536	0.0263	0.028
19	124545	4764246	0.0261	0.028
20	124643	4758835	0.0261	0.028
21	124611	4761486	0.0261	0.028

Table 4. High Priority Video and Low Priority Audio Channels

transmit, but deadline checks are made against the original deadline. By artificially manipulating these deadlines, Diff-EDF is able to achieve different deadline miss-rates that satisfy different demands. Simulation experiments with both artificial and real data show very good matches with our theoretical predictions. Our future work on this front include applying this approach to Quantized EDF schemes, and to multi-node end-to-end flows.

References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services," *Internet RFC 2475*, December 1998

[2] P. Chandra, A. Fisher, C. Kosak, T. S. Eugene Ng, P. Steenkiste, E. Takahashi, H. Zhang, "Darwin: Resource Management for Value-Added Customizable Network Service", In *Proc. of the 6th IEEE International Conference on Network Protocols (ICNP'98)*, October 1998

[3] J.P. Hansen, H. Zhu, R. Rajkumar, "Probabilistic Bandwidth Reservation by Resource Priority Multiplexing," In *Proc. of the IEEE Real-Time Applications Symposium*, May 2001

[4] C.L. Hoover, J.P. Hansen, P. Koopman, S. Tamboli, "The Amaranth Framework: Policy-Based Quality of Service Management for High-Assurance Computing", In *International Journal of Reliability, Quality, and Safety Engineering*, vol. 8, no. 4, p. 323-350, December 2001

[5] G. Nutt, S. Brandt, A. Griff, S. Siewert, T. Berk, M. Humphrey, "Dynamically Negotiated Resource Management for Data Intensive Application Suites", In *IEEE Transactions on Knowledge and Data Engineering*, 12(1):78-95, January 2000

[6] J.P. Lehoczky, "Real-Time Queueing Theory," In *Proc. of the IEEE Real-Time Systems Symposium*, December 1996

[7] J.P. Lehoczky, "Scheduling communication networks carrying real-time traffic," In *Proc. of the IEEE Real-Time Systems Symposium*, December 1998

[8] S-N. Yeung, J.P. Lehoczky, "End-to-end Delay Analysis for Real-Time Distributed Networks," In *Proc. of the IEEE Real-Time System Symposium*, December 2001

[9] W. Yuan, K. Nahrstedt, K.Kim, "R-EDF: A Reservation-Based EDF Scheduling Algorithm for Multiple Multimedia Task Classes", In *Proc. of the IEEE Real-Time Applications Symposium*, May 2001

[10] L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zappala, "RSVP: A New Resource ReSerVation Protocol", In *IEEE Network*, vol. 7, no. 5, p. 8-18, September 1993

[11] UCB/LBNL Video Conferencing Tool, http://www-nrg.ee.lbl.org/vic

[12] L.R. Welch, B. Shirazi, B. Ravindran, C. Bruggeman, "DeSiDeRaTa: QoS Management Technology for Dynamic, Scalable, Dependable, Real-time Systems", In Proc. of the 15th IFAC Workshop - Distributed Computer Control Systems (DCCS '98), pp. 7-12, September 1998

[13] H.F. Zhu, J.P. Hansen, J.P. Lehoczky, R. Rajkumar, "Optimal Partitioning for Quantized EDF Scheduling", In *Proc. of the IEEE Real-Time Systems Symposium*, December 2002

[14] H.F. Zhu, J.P. Lehoczky, J.P. Hansen, R. Rajkumar, “Design Trade-Offs for Networks with Soft End-to-end Timing Constraints” In *Proc. of the IEEE Real-Time System and Application Symposium*, May 2004

[15] H.F. Zhu, J.P. Hansen, J.P. Lehoczky, R. Rajkumar, "RETINA: REal-TIme Network Analyzer", Submitted for publication, November 2004