European Journal of Scientific Research
ISSN 1450-216X Vol.27 No.3(2009), pp.431-453
© EuroJournals Publishing, Inc. 2009
http://www.eurojournals.com/ejsr.htm

Characteristics of Agent-Based Hierarchical Diff-EDF Schedulability Over Heterogeneous Real-Time Packet Networks

Moutaz Saleh
Department of CSE, Faculty of Engineering, Qatar University, 2713 Doha, Qatar
E-mail: moutaz.saleh@qu.edu.qa

Zulaiha Ali Othman
Department of Science & System Management, UKM, 43600 UKM Bangi, Malaysia
E-mail: zao@ftsm.ukm.my

Mohammed Saleh
Department of CSE, Faculty of Engineering, Qatar University, 2713 Doha, Qatar
E-mail: mohd.saleh@qu.edu.qa

Abstract

Packet networks are currently enabling the integration of heterogeneous traffic with a wide range of characteristics that extend from video traffic with stringent QoS requirements to best-effort traffic requiring no guarantees. QoS guarantees can be provided in packet networks by the use of proper packet scheduling algorithms. In this paper, we propose a new priority assignment scheduling algorithm, Hierarchical Diff-EDF, which can meet the real-time needs while continuing to provide best effort service over heterogeneous network traffic environment. The Hierarchical Diff-EDF service meets the flow miss rate requirements through the combination of single step hierarchal scheduling for the different network flows and the admission control mechanism that detects the overload conditions to adjust packets' priorities. To examine the proposed scheduler, we introduced an attempt to provide an exact analytical solution. The attempt showed that the solution was apparently very complicated due to the high interdependences between the system queues' service. Hence, the use of simulation seems inevitable. A multi-agent simulation that takes the inspiration from object-oriented programming is adopted. The simulation itself is aimed to the construction of a set of elements which, when fully elaborated, define an agent system specification. When evaluating our proposed scheduler, it was extremely obvious that the Hierarchical Diff-EDF scheduler performs over both of the EDF and Diff-EDF schedulers.

Keywords: Real Time, Packet Networks, Schedulability, Hierarchical, EDF, QoS, Analytical Model, Agent-based Simulation.

1. Introduction

Recently, many applications of computer networks rely on the ability of the network to provide Quality of Service (QoS) guarantees. These guarantees are usually bounded in the form of delay, bandwidth, packet loss rate, and buffer utilization or a combination of these parameters. Furthermore, packet networks are currently enabling the integration of traffic with a wide range of characteristics that extend from video traffic with stringent QoS requirements to best-effort traffic requiring no guarantees. QoS guarantees can be provided in packet networks by the use of proper packet scheduling algorithms. The function of a scheduling algorithm is to select the packet to be transmitted in the next cycle from the available arrived packets.

Network traffic can be categorized into two types: real-time traffic, such as video and audio, and non-real-time traffic such as http data. Recently, there has been a significant increase in the amount of multimedia services transmitted over networks. These multimedia applications, due to the stringent delay constraints, have to meet certain QoS guarantees. Since scheduling has a direct impact on the system capacity and delay as well as throughput, it is therefore necessary to investigate the suitable scheduling algorithms for such traffic. The distinguishing characteristic of real-time traffic is that it requires bounded delay while it can tolerates some packet losses. The delay can be bounded by associating a deadline for each packet. Once a packet misses its deadline, it will be dropped as it is no longer useful. Therefore the main goal for any scheduling scheme for real-time traffic is to deliver packets in a timely manner.

As a computer revolution, many scheduling algorithms have been proposed to meet this goal. The First-Come-First-Serve (FCFS) scheduling algorithm, which is mostly used in conventional networks, is widely adopted for best-effort traffic. On the other hand, many scheduling algorithms have been proposed to provide different schemes of QoS guarantees, with Earliest Deadline First (EDF) as the most popular one.

2. Real-Time Systems

A real-time system is typically composed of several or sequential tasks with timing constraints. In most real time systems, tasks are invocated repeatedly: each invocation of a task is referred as a job; and the corresponding time of invocation is referred as the job's release time or job's deadline [1]. Thus, the relative deadline parameter is used to specify the timing constraints of the jobs. In addition, real-time systems can be broadly classified as hard or soft depending on the criticality of deadlines [2]. In hard real-time systems, all deadlines must be met; equivalently, a deadline miss results in an incorrect system. On the other hand, in a soft real-time system, timing constraints are less stringent; occasional deadline misses do not affect the correctness of the system.

A real-time system has two notions of correctness: logical and temporal [1]. In particular, in addition to producing correct outputs (logical correctness), such a system needs to ensure that these outputs are produced at the correct time (temporal correctness). However, selecting appropriate methods for scheduling activities is one of the important considerations in the design of a real-time system [3]; such methods are essential to ensure that all activities are able to meet their timing constraints. These timing constraints are usually specified using a deadline.

3. Previous Research

Many real-time systems rely on the earliest deadline first (EDF) scheduling algorithm. This algorithm has been shown to be optimal under many different conditions. For example, for independent, preemptable tasks, on uni-processor EDF is optimal in the sense that if any algorithm can find a schedule where all tasks meet their deadline then EDF can meet the deadline [4]. Also, Jackson's rule [5] says that ordering a set of tasks by deadline will minimize the maximum lateness. Further, it also has been shown that EDF is optimal under certain stochastic conditions [6].

In spite of these advantageous properties, EDF has one major negative aspect. That is, when using EDF in a dynamic system, if overload occurs, tasks may miss deadlines in an unpredictable manner, and in the worst case, the performance of the system can approach zero effective throughput [7]. This is due to the fact that EDF gives highest priority to those processes that are close to missing their deadlines. In such situations, EDF does not provide any type of guarantee on which tasks will meet their timing constrains. This is a very undesirable behavior in practical systems, since in real-world applications intermittent overloads may occur due to exceptional situations, such as modifications in the environment, arrival of a burst of tasks, or cascades of system failures. In that case, matters may be improved by introducing some congestion control mechanism.

A robust earliest deadline scheduling algorithm for dealing with sporadic tasks under overloads in hard real-time environment was proposed by [8]. The algorithm synergistically combines many features including a very minimum level of guarantee, dynamic guarantees, graceful degradation in overloads, deadline tolerance, and resource reclaiming. Also, in 1995, [9] presented a comparative study among scheduling algorithms which use different priority assignments and different guarantee mechanisms to improve the performance of a real-time system during overload conditions. Their results showed that EDF scheduling performs best if admission control is used along with a reclaiming mechanism that’s takes advantage of early completions. In 1997, [10] introduced algorithms for flow admission control at an EDF link scheduler. Their results showed that these algorithms have very low computational complexity and are easily applicable in practice.

While real-time system designers try to design the system with sufficient resources, because of cost and highly unpredictable environments, it is sometimes impossible to guarantee that the system resources are sufficient; in this case EDF's performance degrades rapidly in overload situations. However, it is worthy to say that in the year of 1998, EDF was a major paradigm for real-time scheduling [11].

EDF is a widely used algorithm for online deadline scheduling. It has been known for long that EDF is optimal for scheduling an under-loaded, single processor system; recent results on the extra-resource analysis of EDF further revealed that EDF when using moderately faster processors can achieve optimal performance in the under loaded, multi-processor setting. [12], initiated the extra resource analysis of EDF for overloaded systems, showing that EDF supplemented with simple form of admission control can provide a similar performance guarantee in both single and multi-processor settings.

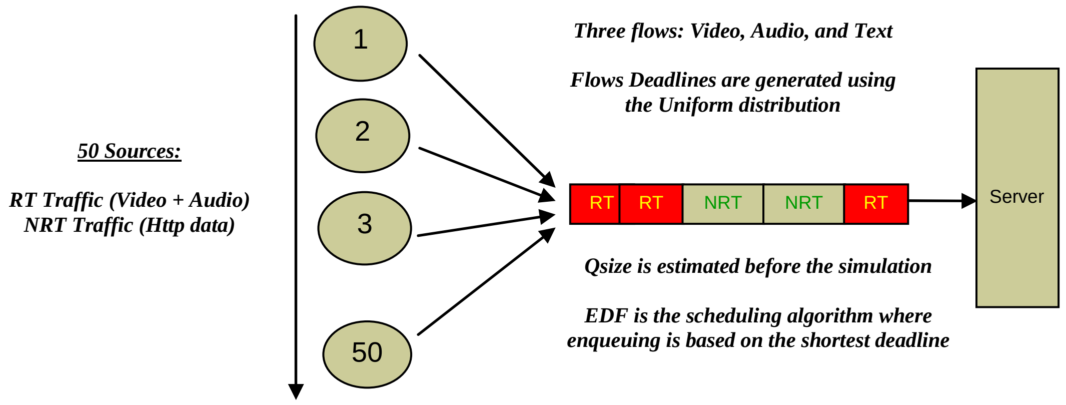
Also, EDF is widely used in scheduling real-time database transactions [13]. By using EDF, database transactions are classified into two categories, those that have missed their deadlines and those that have not. The latter category can be scheduled using the EDF algorithm, while the former can be kept in background and executed whenever there are no transactions that have not missed their deadlines awaiting services.

Recently, a major problem with EDF when scheduling network traffic, is that all flows receive the same miss rate regardless of their traffic characteristics and deadlines [14]. 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 requirements.

4. System Structure of the Hierarchical Diff-EDF

The goal of the Hierarchical Diff-EDF scheduling algorithm is to guarantee that a flow’s deadline miss rates meet its QoS requirements and ensure that real-time applications and best effort traffic can coexist on the same network structure. In EDF scheduler, low priority flows, such as Non-Real-Time traffic, can starve as it is characterized with long lead-times. Despite EDF provides stable QoS guarantees to high priority flows, such as Real-Time traffic, the deadline miss rates of the low priority flows can be unacceptably high. Figure 1 below models the EDF scheduler.

Figure 1: EDF scheduler

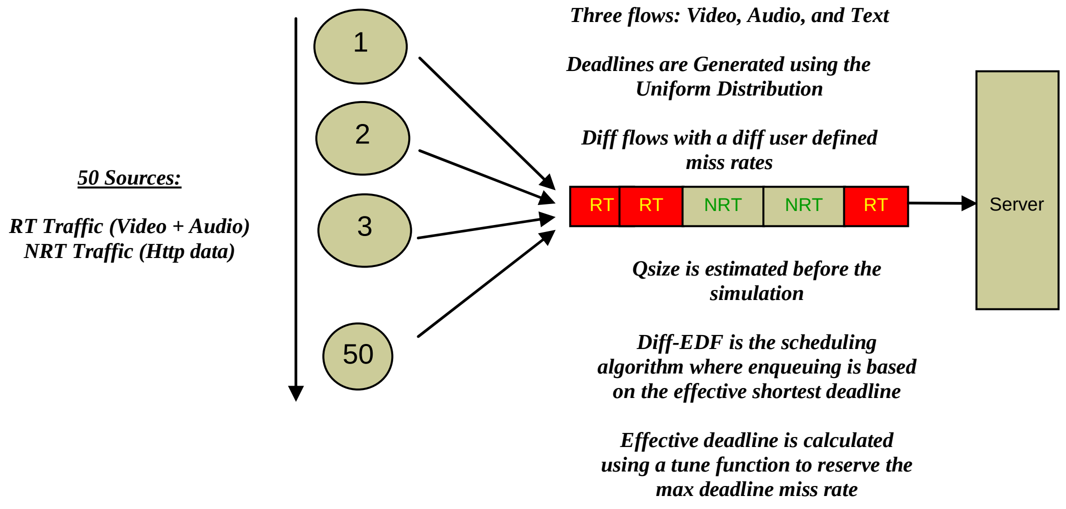

50 Sources:
RT Traffic (Video + Audio)
NRT Traffic (Http data)
1
2
3
50
Three flows: Video, Audio, and Text
Flows Deadlines are generated using the Uniform distribution
RT
RT
NRT
NRT
RT
Server
Qsize is estimated before the simulation
EDF is the scheduling algorithm where enqueuing is based on the shortest deadline

By analyzing the previous figure, three main drawbacks were discovered in using the EDF to schedule real-time packet network traffic:

- 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 requirements.
- Packet Starving for the Non-Real-Time traffic. Since Real-Time Traffic is characterized with short lead-times (time until their deadline expires), then it receives high priority comparing to the Non-Real-Time Traffic which leads to packet starving.
- A random Assignment of Network traffic (Real-Time and Non-Real-Time). As mentioned before, the FCFS scheduling algorithm is widely adopted for best-effort traffic. Having only one service discipline forces all traffic, regardless of their characteristics, to follow the same scheduling algorithm, in our case the EDF.

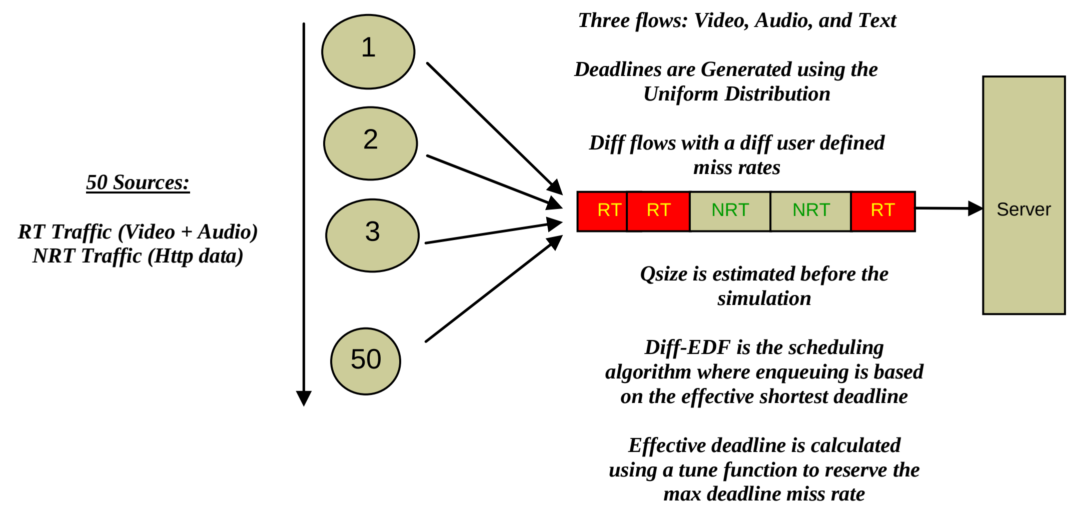
To overcome the first drawback of the EDF, a new Diff-EDF priority assignment algorithm is proposed [14]. The Diff-EDF scheduling algorithm considers each flow as having stochastic traffic characteristic, a stochastic deadline and a maximum allowable miss rate. Figure 2 shows a representative model for the Diff-EDF scheduler.

Figure 2: Diff-EDF scheduler


50 Sources:
RT Traffic (Video + Audio)
NRT Traffic (Http data)
1
2
3
50
Three flows: Video, Audio, and Text
Deadlines are Generated using the Uniform Distribution
Diff flows with a diff user defined miss rates
RT
RT
NRT
NRT
RT
Server
Qsize is estimated before the simulation
Diff-EDF is the scheduling algorithm where enqueuing is based on the effective shortest deadline
Effective deadline is calculated using a tune function to reserve the max deadline miss rate

Again by analyzing the above figure, it is obvious that the last two drawbacks are still discovered when Diff-EDF is used to schedule non-real-time network traffic. As a result, 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 Hierarchical Diff-EDF scheme satisfies this objective. It can meet the real-time needs of such applications, by using the Diff-EDF scheduler, while continuing to provide best effort service to non-real time traffic through depending on the strength of the FCFS scheduler. The Hierarchical Diff-EDF features a feedback control mechanism that detects overload conditions and modifies packet priority assignments accordingly. Figure 3 shows a representative model for the Hierarchical Diff-EDF scheduler.

Figure 3: Hierarchical Diff-EDF scheduler


50 Sources:
RT Traffic (Video + Audio)
NRT Traffic (Http data)
1
2
3
50
Three flows: Video, Audio, and Text
Deadlines are Generated using the Uniform Distribution
Diff flows with a diff user defined miss rates
RT
RT
NRT
NRT
RT
Server
Qsize is estimated before the simulation
Diff-EDF is the scheduling algorithm where enqueuing is based on the effective shortest deadline
Effective deadline is calculated using a tune function to reserve the max deadline miss rate

The Hierarchical Diff-EDF scheduler uses a tuning method, or marker, that adjusts the deadlines of the incoming packets by adding a constant to the relative deadlines before the packets are placed into different queues based on the traffic type. Different constants are added to different flows, and the modified deadlines are known as “Effective Deadlines”.

Packet Processing: A Hierarchical Diff-EDF system consists of two queues: A high priority queue with Diff-EDF service for the real-time traffic such as video and voice, and a low priority queue with FCFS service for the non-real-time traffic such as http data. On receiving each packet from a certain packet flow (assume packet flow *j*), Hierarchical Diff-EDF performs the following operations:

- Identify the associated flow for the packet using Packet Type as an ID label (we define 1 for http data, 2 for audio, and 3 for video) and lookup up the adjustment constant B_j. The relative deadline is then changed to the effective deadline according to the following equation: $D_e = D_j + B_j$
- For Diff-EDF queue, perform the ordinary EDF scheduling using the effective deadline. That is, the packet’s absolute deadline is now given by: $D_a = D_e + i_{at}$
  Where i_{at} is the packet’s inter-arrival time, and insert the packet into the Diff-EDF queue in the order of increasing absolute deadline (smallest absolute deadline is at queue head thus served first).
- For FCFS queue, perform the ordinary FCFS scheduling using the packet inter-arrival time (i_{at}), with smallest i_{at} at the head of the queue.

5. Analysis of the Hierarchical Diff-EDF

Traffic Characteristics: Assume we have *K* packet 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 packet inter-arrival distribution (Exponential Distribution), with a mean of $1/\lambda_j$. Let $\Lambda = \sum_{j=1}^{k} \lambda_j$, the total arrival rate.
- A packet service requirement distribution (Exponential Distribution), with a mean of $1/\mu_j$.
- A soft deadline D_j > 0. For each flow j that is randomly drawn from a distribution G_j (in our case the uniform distribution), then D_j represents the mean of G_j. Let $D_j = \sum_{j=1}^{k} \lambda_j D_j / \Lambda$, the mean packet deadline averaged over all flows.
- Define $\rho_j = \lambda_j / \mu_j$ the traffic intensity of flow j, and $\rho = \sum_{j=1}^{k} \rho_j$. In addition, we define $\alpha_j = \rho_j / \rho$, the faction of the traffic intensity that is attributed to flow j.
- A QoS requirement ϕ_j, interpreted as the requirement that the long run average fraction of flow j's packets missing their deadlines D_j must not exceed ϕ_j.
- The Real-Time Queuing Theory (RTQT) analysis used in this work models the workload process as a Brownian motion with drift $-\theta$, where: $\theta = \dfrac{2(1-\rho)}{\sum_{j=1}^{k} \lambda_j (\lambda_j^2 + \mu_j^2)/\mu_j^2}$

Deadline Miss Rate Prediction: Prediction of miss rate, per each flow j, is based on the RTQT analysis of the EDF algorithm. The basic methodology can be found in [15, 16, and 17]. It had been found that when all flows have the same deadline miss rate, then it can be computed by: $\phi_j = e^{-\theta \overline{D}}$

For Hierarchical Diff-EDF, we will adjust the deadlines of each flow j by adding a constant B_j to the deadline. Using the above equation, it is obvious that when using Hierarchical Diff-EDF, the deadline miss rate for each flow can be computed by: $\phi_j = e^{-\theta \overline{D_e}}$, Where: $\overline{D_e} = \sum_{j=1}^{k} \alpha_j (D_j + B_j)$

Determination of B$_j$'s: As we mentioned before, one of the Hierarchical Diff-EDF system components is the Marker. A Marker adjusts the deadlines of incoming packets by adding a constant B_j, different constants are added to different flows, to the relative deadlines before the packets are placed into the queue. To determine the appropriate values of the constant B_j for the different flows, we must ensure that the flows with the shortest lead-time are always scheduled first. Thus, we are looking for a mathematical function which can provide a proportional relation with respect to the distinct flows' deadlines. Accordingly, we use the logarithmic function to compute the constant values of B_j as in the following equation: $B_j = \dfrac{1}{\theta} Log \dfrac{D_j}{D_1}, 1 \le j \le k$, Where $1/\theta$ is the mean of the exponential stationary distribution of the workload process, D_j is the deadline miss rate of the flow j, D_1 is the smallest deadline miss rate among the flows, and k is the number of the flows to be serviced. If we assume B_1 is the constant to be added for the deadlines in the video flow, and that the video flow has the smallest deadline miss rate, then by applying the above equation: $B_1 = 1/\theta Log 1 = 0$. Hence, once the B_j for the high priority flow is determined, the Diff-EDF system will select a much larger values of B_j's for the flows to be run at low priority.

Generating Arrival Packets: In order to generate the arrival packets, a number of arguments must be determined as the following:

- Number of sources in the system as s.
- Number of flows in the system as k.
- Total Number of Arrival rate for all flows j as $\lambda_T, 1 \le j \le k$.
- Arrival rate per flow j as $\lambda_j = \lambda_T / k$.
- Inter-arrival mean per flow j as $1/\lambda_j$
- Relative Deadline range per flow j as $(QoS_{\max}, Qos_{\min})$.

It is worthy to mention here that the total number of arrival rate in the system should equal the summation of all the arrival rates per each flow j as in the following equation: $\lambda_T = \sum_{j=1}^{k} \lambda_j$ ..

Now, after determine the arguments we start generating the packets. Two Steps were carried out:

1. Using the Exponential Distribution, with a mean of $1/\lambda_j$, to generate the inter-arrival time of the different packets.
2. Using the Uniform Distribution, with a mean of $(QoS_{\max}, Qos_{\min})$, to generate the relative deadlines for the different packets.

An important issue that been taken into consideration when generating the arrival packets was to ensure that the Traffic Generator always obtain an initial seeds for the different streams. This mechanism will ensure that each flow is following a specified seed, when its packets are generated, to reflect the real-world traffic and leads for high accuracy. To do that, a Seed Initializer method is used to initialize seeds for the variety streams based on the defined mathematical techniques in [18].

Determination of Effective Deadlines Range: As we mentioned earlier, the Uniform Distribution with a mean of $(QoS_{\max}, Qos_{\min})$ is used to generate the relative deadlines for the different packets. $QoS_{\max}$ and $QoS_{\min}$ are the Effective Deadlines range for each flow j. Calyam and Lee in [19] have built a voice and video traffic measurement test-bed to determine their effective deadline ranges as shown in Figures 4 and 5. The collected results from both figures recorded an effective delay lower and upper bound range in the form of (*a, b*). The delay lower bound is denoted by *a* and the delay upper bound is denoted by *b*. By observing the previous figures, it was discovered that the Good range corresponds to delay values of (0-150) ms, the Acceptable range corresponds to delay values of (150-300) ms, while the Poor range corresponds to delay values > 300ms.

Figure 4: Audio Packet Delay

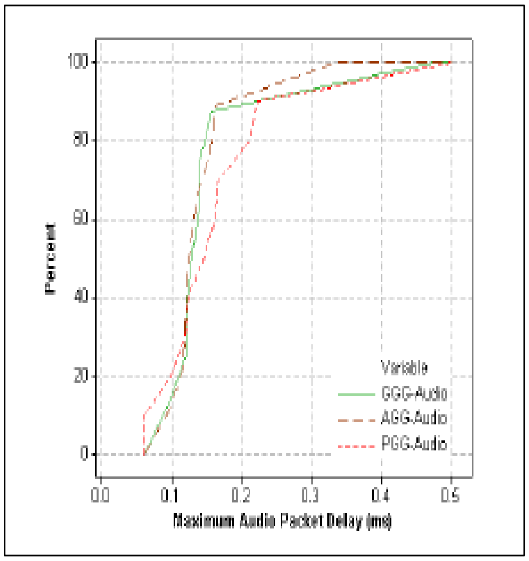
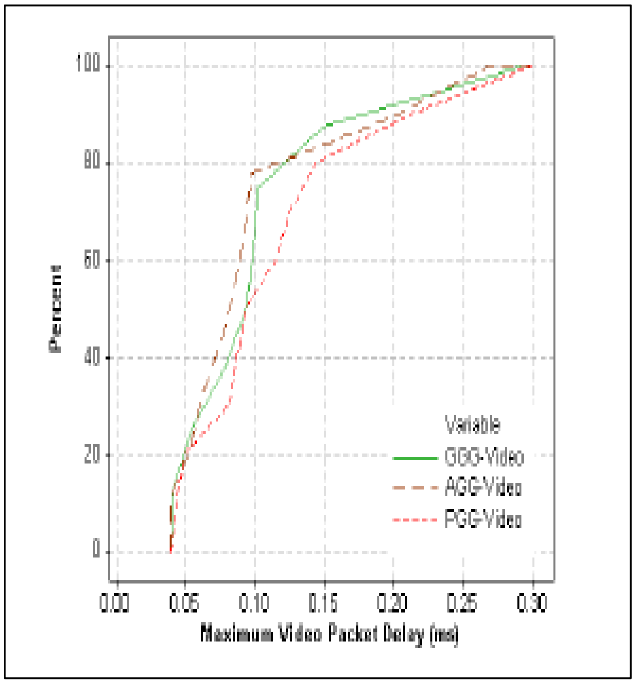


Figure 5: Video Packet Delay



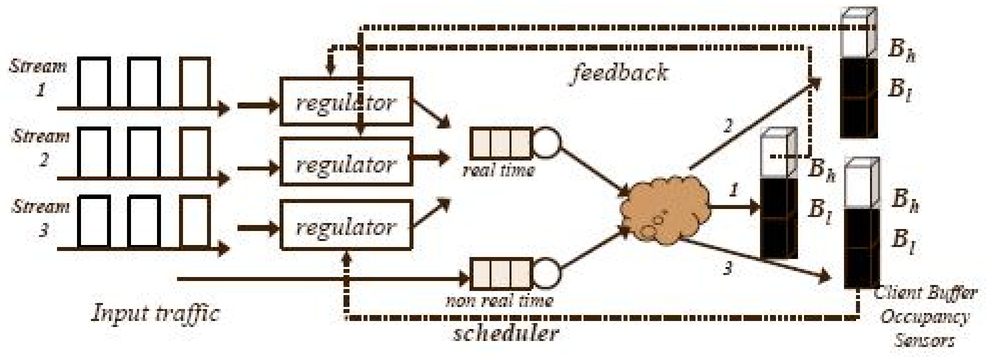
Based on the above real results, collected in large-scale Internet, we choose our effective deadline, to achieve the highest system throughput, in the following manner:

- Effective Video Deadline in the Good range (40, 150).
- Effective Audio Deadline in the Acceptable range (250, 160).
- Effective Text Deadline in the Poor range (assume 400, 300).

The reason to choose the Effective deadline for the text traffic in the poor range is coming from the fact that it is less sensitive to the delay compared with the multimedia traffic.

Feedback Control Mechanism: Despite the significant body of results in real-time scheduling, many real world problems are not easily supported. While EDF can support deadlines and precedence constraints, it is considered as an open loop scheduling algorithms which refers to the fact that once schedules are created they are not adjusted based on continuous feedback and can perform poorly in unpredictable dynamic systems. Hence, the Hierarchical Diff-EDF scheduling algorithm features a feedback control mechanism that detects overload conditions and modifies packet priority assignment accordingly. To do that, the algorithm is implemented with a feedback control mechanism (Threshold limitation). In other words, the server always serves the packets in the Diff-EDF queue (high priority), and serves the FCFS queue (low priority) if either the Diff-EDF queue is empty or the FCFS queue reaches its threshold value. Now, after tuning the system to achieve the highest performance, through meeting all the flows deadline miss rates ϕ_j, it has been found that the threshold value is when $FCFSQsize >= 0.9\lambda_T / k$. On the other hand, the stopping case was found when the value approaches to $0.7\lambda_T / k$. Figure 6 shows a representative model for the feedback control approach over heterogeneous real-time network traffic.

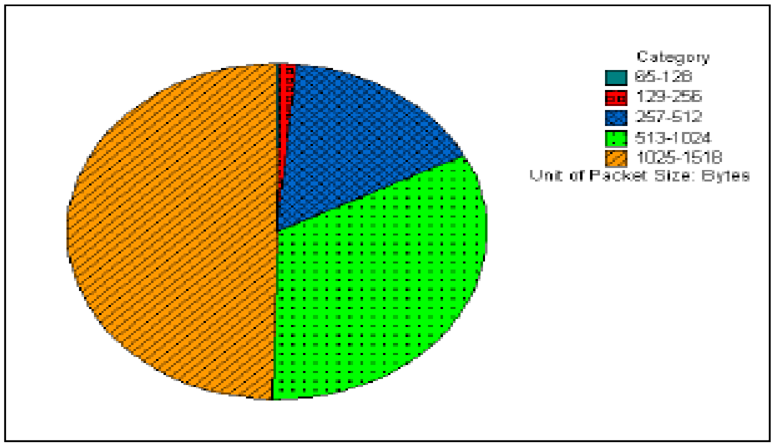
Figure 6: Feedback Control Approach for Real-Time Traffic



System Parameters: For this system a number of parameters were set as the following:

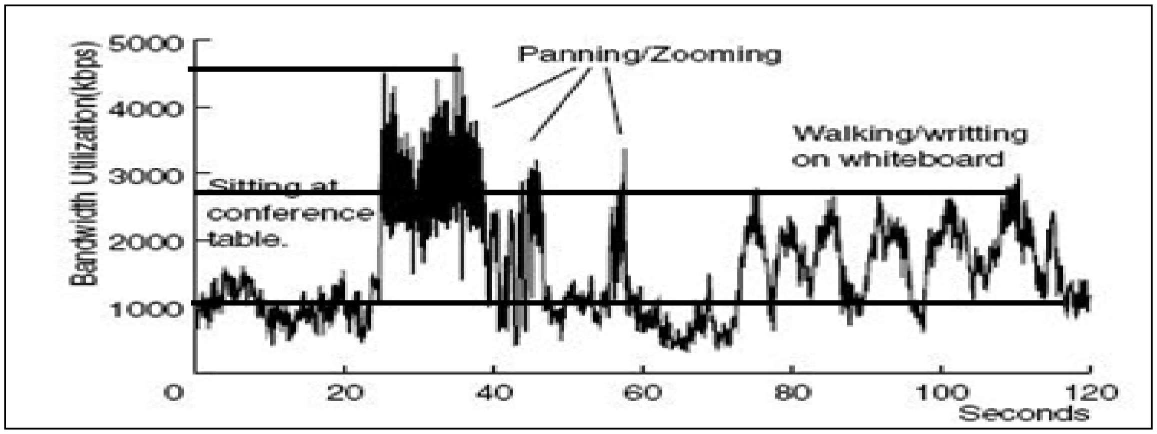
- Packet Size: Understanding traffic characteristics in terms of packet-size distributions is important since it has implications on the end-to-end performance achieved by the traffic streams. For this system, the packet size was chosen to be of 1500 Byte. The reason to choose this value is that almost 50% of the traffic being propagated across the internet has a packet size of 1500 Byte as shown in Figure 7 which was explored by the research work carried out in [19]. Also, the packet size has a direct effect on the packet service rate which is equal to [8 * B_w / Packet Byte Size]. The bigger the packet size, the more the system is overloaded due to having $\lambda > \mu$. Thus, choosing the packet size of 1500 Byte will help us to investigate the overload conditions in the system.

Figure 7: Packet Sizes Distribution



- Bandwidth B_w: In order to determine the bandwidth, we refer to the work explored in [14] where Figure 8 illustrates the aggregate bandwidth consumption of a video conference flow.

Figure 8: Bandwidth Consumption of a Video Conference Flow



By analyzing the above figure, we can observe that video traffic consumes a highest bandwidth with a value of 4.6 Mbps, median traffic consumes 2.8 Mbps, and Low traffic consumes about 1 Mbps. Considering that the three network flows to be experimented are of same packet generation rate λ_j, then the Aggregate Average Bandwidth in Mbps will equal to (4.6 Mbps + 2.8 Mbps + 1 Mbps) / 3 = 2.8 Mbps. Thus, to ensure a guaranteed bandwidth for the overall network flows we will adjust the aggregate bandwidth to be set for 3 Mbps.

- Total Arrival Rate λ_T: the simulation was carried out for λ_T start at 5000 packet up to 60000 packets with a 5000 packet simulation step. The reason to choose such packet rates is to utilize the existences of the lightly-loaded $\lambda < \mu$, moderately-loaded $\lambda = \mu$, and over-loaded $\lambda > \mu$ conditions during the simulation experiments.
- Mean Service μ: the mean service was calculated with the following equation:
  Mean Service (μ) = 8 * PacketSize / B_w
- Number of Sources s: the simulation was carried out with a 50 generated sources.
- The Experiments were carried out with three different flows; two of them are real-time traffic (video and voice) while the third flow is non real-time traffic (http data or text).

6. Scheduler Work Complexity

Consider an execution of the Hierarchical Diff-EDF scheduling discipline over *K* packet flows. We define the scheduler work complexity as the order of the time complexity with respect to *K* of enqueuing and then dequeuing *n* packets for service.

Theorem 1. The work complexity of the Hierarchical Diff-EDF is less than or equal, in the worst-case, the work complexity of either EDF or Diff-EDF, i.e. $\leq O(n^2)$.

Proof 1. We prove theorem 1 by showing that the total enqueuing and dequeuing operations are each of time complexity $\leq O(n^2)$. We start by finding the time complexity of the enqueuing operation and then the dequeuing operation. Let *s* is the non real-time input data traffic, and *r is* the real-time input data traffic. The time complexity of the enqueuing operation for the *s* traffic is of constant order O(1) since the *s* traffic is enqueued by FCFS algorithm. On the other hand, the time complexity of the enqueuing operation for the *r* traffic is of quadratic order $O(r^2)$ since the *r* traffic is enqueued by EDF algorithm which sorts the incoming traffic with the shortest deadline packet at the head of the queue.

Consequently, this result in a lower enqueuing total time complexity compare to the standard EDF as *r* < *n*. Moreover, if all of the sources' packets have the same deadline, then EDF operates as FCFS with a total enqueuing time complexity of constant order O(1). For the worst-case, however, the total time complexity of the enqueuing operation is defined by a quadratic order of O(n^2). This due to the fact that the worst-case might occur when all sources' traffic is of *r* type, and then we have *r=n* and *s=0*. To find the total time complexity of the dequeuing operation, we calculate the time complexity of the dequeuing operation for both *s* and *r* traffic. The time complexity of the dequeuing operation for the *s* traffic is of constant order O(1) since the *s* traffic is dequeued by FCFS algorithm. Also, the time complexity of the dequeuing operation for the *r* traffic is of constant order O(1) since the *r* traffic is again dequeued by FCFS algorithm as the EDF queue is sorted with smallest deadline at the head of the queue. Moreover, the operation of removing the selected packet for service from the head of the EDF queue is still can be executed in a constant order O(1) time; since the queue is implemented with Linked List data structure. Consequently, the dequeuing total time complexity is still of constant order O(1).

Theorem 2. The work complexity of the Hierarchical Diff-EDF can be significantly improved to a worst-case linear order of O(n) by replacing the sorting step in the enqueuing operation with searching step in the dequeuing operation.
Proof 2. We prove theorem 2 by showing that the total enqueuing and dequeuing operations are each of time complexity ≤ O(n). We start by finding the time complexity of the enqueuing operation and then the dequeuing operation. The time complexity of the enqueuing operation for the *s* traffic is of constant order O(1) since the *s* traffic is enqueued by FCFS algorithm. Also, the time complexity of the enqueuing operation for the *r* traffic is of constant order O(1) since the *s* traffic is again enqueued by FCFS algorithm as the sorting step is removed at this stage. Consequently, the enqueuing total time complexity is of constant order O(1). In addition, to calculate the total time complexity of the dequeuing operation, we calculate time complexity of the dequeuing operation for both *s* and *r* traffic. The time complexity of the dequeuing operation for the *s* traffic is of constant order O(1) since the *s* traffic is dequeued by FCFS algorithm. On the other hand, the time complexity of the dequeuing operation for the *r* traffic is of linear order O(r) since the *r* traffic is dequeued by EDF algorithm which searches the unsorted list for the packet with the shortest deadline to be served first. Moreover, the operation of removing the selected packet for service, associated with smallest deadline, is still can be executed in a constant order O(1) time as defined by the computational complexity for the actual removal in a Linked List data structure. For the worst-case, however, the total time complexity of the dequeuing operation is defined by a linear order of O(n). This due to the fact that the worst case might occur when all sources' traffic is of *r* type, and then we have *r=n* and *s=0.*

7. Model Formal Description

Let $P_{n_1,n_2,n_3}(t)$ be the probability that there are n_1, n_2 and n_3 customers of types Text (T), Audio (A) and Video (V) respectively in the system at time t. Let δt be some small number, and then the probability $P_{n_1,n_2,n_3}(t+\delta t)$ can be expressed as: $P_{n_1,n_2,n_3}(t+\delta t)$ = P(There are $(n_1 - 1)$ customers of type T, n_2 of type A, n_3 of type V in the system at time t and a T arrival in $(t, t+\delta t)$ and the service is for the (T) queue) + P(There are $(n_1 + 1)$ customers of type T, n_2 of type A, n_3 of type V in the system at time t and a T customer is served in $(t, t+\delta t)$ and the service is for the (T) queue) + P(There are (n_1) customers of type T, $(n_2 - 1)$ of type A, n_3 of type V in the system at time t and an A arrival in $(t, t+\delta t)$ and the service is for the (A,V) queue) + P(There are (n_1) customers of type T, (n_2) of type A, $n_3 - 1$ of type V in the system at time t and a V arrival in $(t, t+\delta t)$ and the service is for the (A,V) queue) + P(There are (n_1) customers of type T, (n_2) of type A, n_3 of type V in the system at time t and no arrivals or departures in $(t, t+\delta t)$ and the service is for the (A,V) queue) + P(There are $(n_1 - 1)$ customers of type T,

(n_2) of type A, n_3 of type V in the system at time t and an A arrival in $(t, t+\delta t)$ and the service is for the (A,V) queue) + P(There are (n_1) customers of type T, (n_2-1) of type A, n_3 of type V in the system at time t and an A arrival in $(t, t+\delta t)$ and the service is for the (T) queue) + P(There are (n_1) customers of type T, (n_2+1) of type A, n_3 of type V in the system at time t and the priority is for A customers and an A customer is served in $(t, t+\delta t)$ and the service is for the (A,V) queue) + P(There are (n_1) customers of type T, (n_2) of type A, (n_3-1) of type V in the system at time t and a V arrival in $(t, t+\delta t)$ and the service is for the (T) queue) + P(There are (n_1) customers of type T, (n_2) of type A, (n_3+1) of type V in the system at time t and the priority Is for V customers and a V customer is served in $(t, t+\delta t)$ and the service is for the (A,V) queue) + P(There are (n_1) customers of type T, (n_2) of type A, n_3 of type V in the system at time t and no arrivals or departures in $(t, t+\delta t)$ and the service is for the (T) queue).

Now, knowing that the DiffEDF Queue will serve the Video and Audio traffic while the FCFS Queue will serve the Text traffic, the following conditions should be satisfied:

```
1. If (!EmptyDiffEdfQueue && !thresholdOn)
       Serve the Diff-EDF Queue
2. Else If (!EmptyDiffEdfQueue && thresholdOn)
       If (FCFSQueue > (0.7*pgr/3))
               Serve the FCFS Queue
       Else    Serve the Diff-EDF Queue
3. Else        Serve the FCFS Queue
```

Again, rewriting the above event probabilities symbolically we obtain four boundary conditions:

$$P_{n_1,n_2,n_3}(t+\delta t)=\theta\left[\begin{array}{l} P_{n_1-1,n_2,n_3}(t)\ \lambda\delta t + P_{n_1+1,n_2,n_3}(t)\ \mu\delta t + P_{n_1,n_2-1,n_3}(t)\ \lambda\delta t + P_{n_1,n_2,n_3-1}(t)\ \lambda\delta t + \\ \quad P_{n_1,n_2,n_3}(t)(1-3\lambda\delta t-\mu\delta t)+o(\delta t) \end{array}\right]$$
$$+(1-\theta)\left[\begin{array}{l} P_{n_1-1,n_2,n_3}(t)\ \lambda\delta t + P_{n_1,n_2-1,n_3}(t)\ \lambda\delta t + P_{n_1,n_2+1,n_3}(t)\ \mu\delta t\ \beta + P_{n_1,n_2,n_3-1}(t)\ \lambda\delta t + \\ \quad P_{n_1,n_2,n_3+1}(t)\ \mu\delta t\ (1-\beta) + P_{n_1,n_2,n_3}(t)(1-3\lambda\delta t-\mu\delta t)+o(\delta t) \end{array}\right] \quad (1)$$

$$P_{n_1,0,0}(t+\delta t)=P_{n_1-1,0,0}(t)\ \lambda\delta t + P_{n_1+1,0,0}(t)\ \mu\delta t + P_{n_1,0,0}(t)(1-\lambda\delta t-\mu\delta t)+o(\delta t) \quad (2)$$

$$P_{k,n_2,n_3}(t+\delta t)=P_{k-1,n_2,n_3}(t)\ \lambda\delta t + P_{k,n_2-1,n_3}(t)\ \lambda\delta t + P_{k,n_2,n_3-1}(t)\ \lambda\delta t + P_{k,n_2,n_3}(t)(1-3\lambda\delta t)+o(\delta t) \quad (3)$$

$$P_{s,n_2,n_3}(t+\delta t)=P_{s,n_2-1,n_3}(t)\ \lambda\delta t + P_{s,n_2+1,n_3}(t)\ \mu\delta t\ \beta + P_{s,n_2,n_3-1}(t)\ \lambda\delta t + P_{s,n_2,n_3+1}(t)\ \mu\delta t\ (1-\beta) +$$
$$P_{s,n_2,n_3}(t)(1-2\lambda\delta t-\mu\delta t)+o(\delta t) \quad (4)$$

Where θ is the probability that the Text queue has reached the threshold *k*. In the following section, we will show our attempt to solve the equations (1) through (4) as an analytical solution for our mathematical model.

8. Attempting Analytical Solution

Once the model formal description is defined and represented in a symbolic model then it must be solved with the proper performance measures of interest. However, the solution itself requires both of the joint Probability Density Function (PDF) for the two real-time flows, video and audio and then the integration of some complicated functions over some restricted regions. Furthermore, some manipulations with the transformation techniques of random variables should be used before the final analytical solution relies on solving some derived deferential equations.

Let $Y_i = IAT_i + QoS_i$, $i = 1,\ldots,n_2$, where IAT_i is $Exp(\lambda)$, and QoS_i is $U(a_A, b_A)$

Also, let $Z_j = IAT_j + QoS_j,\ \ j = 1,\ldots,n_3$ where IAT_j is $Exp(\lambda)$, and QoS_j is $U(a_V, b_V)$

Let $R = \begin{cases} 1, & \text{if } \min\limits_{\text{i,j}}(Y_i, Z_j) \text{is a Y} \\ 0, & \text{Otherwise} \end{cases}$

Hence, $\beta = P(R = 1)$

Where $\beta = P(R = 1) = P\left(\min\limits_i Y_i \le \min\limits_j Z_j\right)$

Now we want to find the PDF of $MY = \min\limits_i Y_i$ and $MZ = \min\limits_j Z_j$

$P(MY \le u) = 1 - P(MY > u) = 1 - P(Y > u)^{n_1}$

Now let $Y = IAT + QoS$, where IAT is $Exp(\lambda)$, QoS is $U(a,b)$

The Probability Density Function (PDF) of Y can be found using the distribution function technique in [20] as follows:

Let $G(y) = P(Y \le y)$

Substituting $Y = IAT + QoS$, then $G(y) = P(IAT + QoS \le y)$

The sample space here is a strip in the Cartesian plane where IAT extends horizontally from 0 to ∞ and QoS extends vertically from a to b. This leads us to consider two cases: Case 1: $a \le y \le b$, Case 2: $y > b$

For Case 1:

$$G(y) = \int_a^b \int_0^{y-v} \frac{1}{b-a} \lambda e^{-\lambda u} du dv$$
$$= 1 - \frac{\mathrm{e}^{-\lambda(\mathrm{y}-\mathrm{b})} - e^{-\lambda(y-a)}}{\lambda(b-a)}$$

For Case 2:

$$G(y) = \int_a^y \int_0^{y-v} \frac{1}{b-a} \lambda e^{-\lambda u} du dv$$
$$= \frac{y-a}{b-a} - \frac{1 - e^{-\lambda(y-a)}}{\lambda(b-a)}$$

Now, the PDF of Y can be found by differentiating $G(y)$, this gives:

$$g(y) = \begin{cases} \dfrac{e^{-\lambda(b-a)} - e^{-\lambda(y-a)}}{(b-a)}, & a \le y \le b \\[2ex] \dfrac{1 - e^{-\lambda(y-a)}}{(b-a)} & , y > b \end{cases}$$

Hence, the PDF of $MY = \min\limits_i Y_i$ can be found using the distribution function technique as follows:

Let $F(t)$ be the distribution function of MY, then

$$F(t) = P(MY \le t) = P\left(\min_i Y_i \le t\right) = 1 - P\left(\min_i Y_i > t\right) = 1 - (P(Y > t))^{n_2}$$
$$= 1 - (1 - G(t))^{n_2}$$

Therefore, the PDF of MY is given by

$$f(t) = n_2(1 - G_Y(t))^{n_2 - 1} g_Y(t) \qquad (5)$$

Where

$$g_Y(t)=\begin{cases}\dfrac{e^{-\lambda(b_A-a_A)}-e^{-\lambda(t-a_A)}}{(b_A-a_A)}, a_A \le t \le b_A\\ \dfrac{1-e^{-\lambda(t-a_A)}}{(b_A-a_A)} \qquad ,t > b_A\end{cases}$$

Similarly,
Let $H(t)$ be the distribution function of MZ ,then

$$H(t)=P(MZ \le t)=P\left(\min_i Z_i \le t\right)=1-P\left(\min_i Z_i > t\right)=1-(P(Z>t))^{n_3}$$
$$=1-(1-H(t))^{n_3}$$

Hence, the PDF of MZ is given by

$$h(t)=n_3(1-G_Z(t))^{n_2-1}g_Z(t) \qquad (6)$$

Where

$$g_Z(t)=\begin{cases}\dfrac{e^{-\lambda(b_V-a_V)}-e^{-\lambda(t-a_V)}}{(b_V-a_V)}, a_V \le t \le b_V\\ \dfrac{1-e^{-\lambda(t-a_V)}}{(b_V-a_V)} \qquad ,t > b_V\end{cases}$$

The probability we need now is $P(MY < MZ)$ where the PDF of MY and MZ is given in equations 5 and 6 respectively. Now, the analytical evaluation of this probability requires the joint PDF of MY and MZ , the integration of some complicated functions over some restricted regions, and the manipulations with transformation techniques of random variables. Accordingly, rearranging the terms of equations 1 through 4 and taking the limit as $\delta t \to 0$ we get:

$$\frac{d}{dt}P_{n_1,n_2,n_3}(t)=\theta\left[P_{n_1-1,n_2,n_3}(t)\,\lambda+P_{n_1+1,n_2,n_3}(t)\,\mu+P_{n_1,n_2-1,n_3}(t)\,\lambda+P_{n_1,n_2,n_3-1}(t)\,\lambda+P_{n_1,n_2,n_3}(t)(-3\lambda-\mu)\right]+(1-\theta)$$
$$\left[P_{n_1-1,n_2,n_3}(t)\,\lambda+P_{n_1,n_2-1,n_3}(t)\,\lambda+P_{n_1,n_2+1,n_3}(t)\,\mu\beta+P_{n_1,n_2,n_3-1}(t)\,\lambda+P_{n_1,n_2,n_3+1}(t)\,\mu(1-\beta)\;P_{n_1,n_2,n_3}(t)(-3\lambda-2\mu)\right]$$

$$\frac{d}{dt}P_{n_1,0,0}(t)=P_{n_1-1,0,0}(t)\,\lambda+P_{n_1+1,0,0}(t)\,\mu+\;P_{n_1,0,0}(t)(-\lambda-\mu)$$

$$\frac{d}{dt}P_{k,n_2,n_3}(t)=P_{k-1,n_2,n_3}(t)\,\lambda\text{ - }P_{k,n_2,n_3}(t)\lambda$$

$$\frac{d}{dt}P_{s,n_2,n_3}(t)=P_{s,n_2-1,n_3}(t)\,\lambda+P_{s,n_2+1,n_3}(t)\,\mu\beta+P_{s,n_2,n_3-1}(t)\,\lambda+P_{s,n_2,n_3+1}(t)\,\mu(1-\beta)+\;P_{s,n_2,n_3}(t)(-2\lambda-\mu)$$

In the steady state, all derivatives are zero and the probabilities are independent of time. So, we obtain:

$$0=\theta\left[P_{n_1-1,n_2,n_3}\,\lambda+P_{n_1+1,n_2,n_3}\,\mu+P_{n_1,n_2-1,n_3}\,\lambda+P_{n_1,n_2,n_3-1}\,\lambda+P_{n_1,n_2,n_3}(-3\lambda-\mu)\right]+(1-\theta)\left[P_{n_1-1,n_2,n_3}\,\lambda+\right.$$
$$\left.P_{n_1,n_2-1,n_3}\,\lambda+P_{n_1,n_2+1,n_3}\,\mu\beta+P_{n_1,n_2,n_3-1}\,\lambda+P_{n_1,n_2,n_3+1}\,\mu(1-\beta)+P_{n_1,n_2,n_3}(-3\lambda-2\mu)\right] \qquad (7)$$

$$0=P_{n_1-1,0,0}\,\lambda+P_{n_1+1,0,0}\,\mu+P_{n_1,0,0}(-\lambda-\mu) \qquad (8)$$

$$0=P_{k-1,n_2,n_3}\,\lambda\text{ - }P_{k,n_2,n_3}\lambda \qquad (9)$$

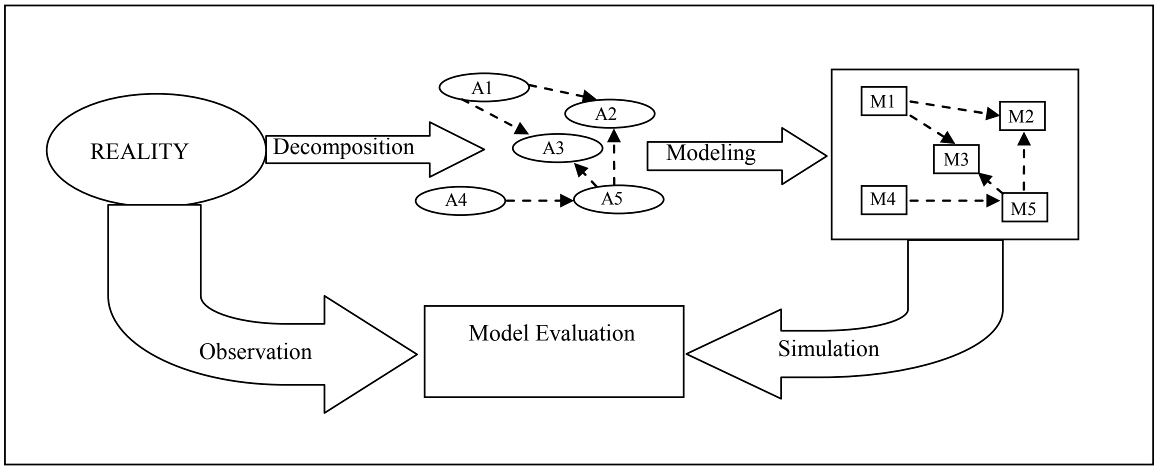
$$0=P_{s,n_2-1,n_3}\,\lambda+P_{s,n_2+1,n_3}\,\mu\beta+P_{s,n_2,n_3-1}\,\lambda+P_{s,n_2,n_3+1}\,\mu(1-\beta)+P_{s,n_2,n_3}(-2\lambda-\mu) \qquad (10)$$

The evaluation of performance measures for the queuing system under consideration depends on the solution to the above system of difference equations (7, 8, 9, and 10). The above system of equations is apparently very complicated. What makes things worse is the interdependence between the different queues as the service for one of them begins after certain conditions concerning the other queues are satisfied. This interdependence presents great difficulties in obtaining a closed form analytical solution. In fact, even for two-queue systems with such interdependence the analytical solution is not available. The use of simulation techniques seems inevitable.

9. Agent-Based Simulation Stages

In a discrete event simulation the time and nature of future events is computed in a predetermined fashion from the list of past events which have occurred. Thus the designer of a simulation will typically pre-specify all possible transitions, and will not leave the definition of state transitions to entities which are not fully pre-determined. Thus it would be very useful to introduce agents in a simulation whose behavior is determined by specific circumstances and through adaptive behavior. The alternative we propose is that, in addition to the usual attributes of a discrete event simulation (such as event lists and possibly random number generator driven events), a simulation should contain a certain number of agents. These agents store information during a simulation, and use it to modify their behavior during that same simulation or during distinct simulation runs. From the point of view of GAIA agent methodology [21], we break up our simulation into three stages as shown in Figure 9:

Figure 9: Agent-Based Simulation Stages



- First stage: includes the decomposition of the real phenomenon into a set of autonomous elements that interacts between each other, and whose interactions reproduce the real phenomenon.
- Second stage: includes modeling each element by an agent, and definition of its knowledge, its functional capacities, its behaviors and its interaction modes with other agents.
- Third stage: includes the description of possible actions between agents, by defining the environment in which these agents evolve and the rules which control them.

By taking again the three stages described in Figure 9, we place ourselves in an agent context. Thus, we follow the steps of a multi-agents simulation process:

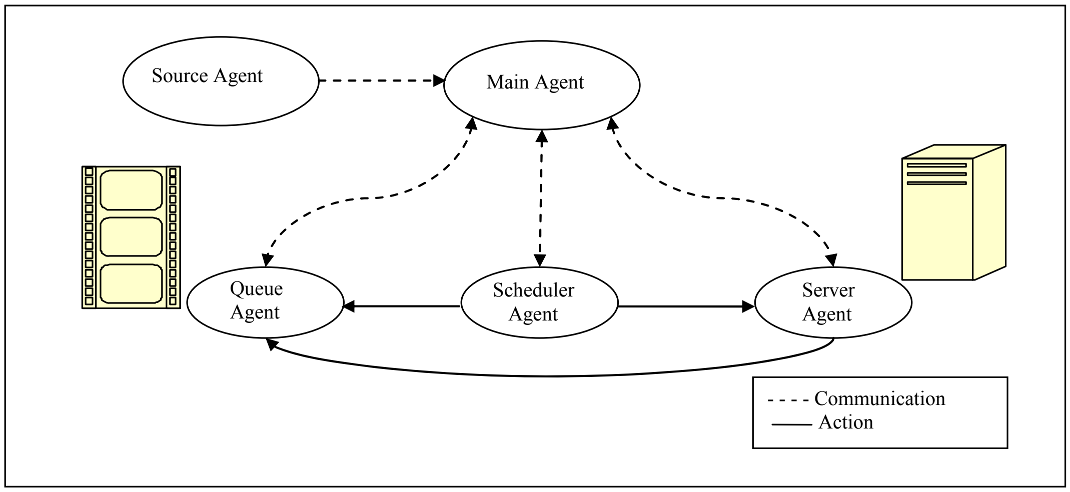
- System Decomposition: a queuing model is an infrastructure that gathers six types of elements: server, scheduler, queue, source, packet, and clock. In addition to the model itself, we have to create a new element (Main) that will be the principle actor of the system behavior.
- Modeling each element by an agent: with regard to the elements of the environment, we consider two types of elements: static (passive) agents and dynamic (active) agents. To bring great dynamicity to the system and preserve computer resources we decide to model server, scheduler, queue, and main as active agents while source, packet, and clock as passive agents.

- Description of possible actions between the agents: there are many possible actions between the agents, i.e. messages exchange between the agents to take certain action, change their policy, or update their information.

10. System Agent Model

Our Real Time Agent-based Simulation model allows for the expression and enforcement of timing constrains on real-time agent interactions. This model is based on the assumption that agents may able to perform their tasks in variety ways. The model itself is made up of five real-time agents, and a set of communications among these agents. Figure 10 displays the active agents of the model.

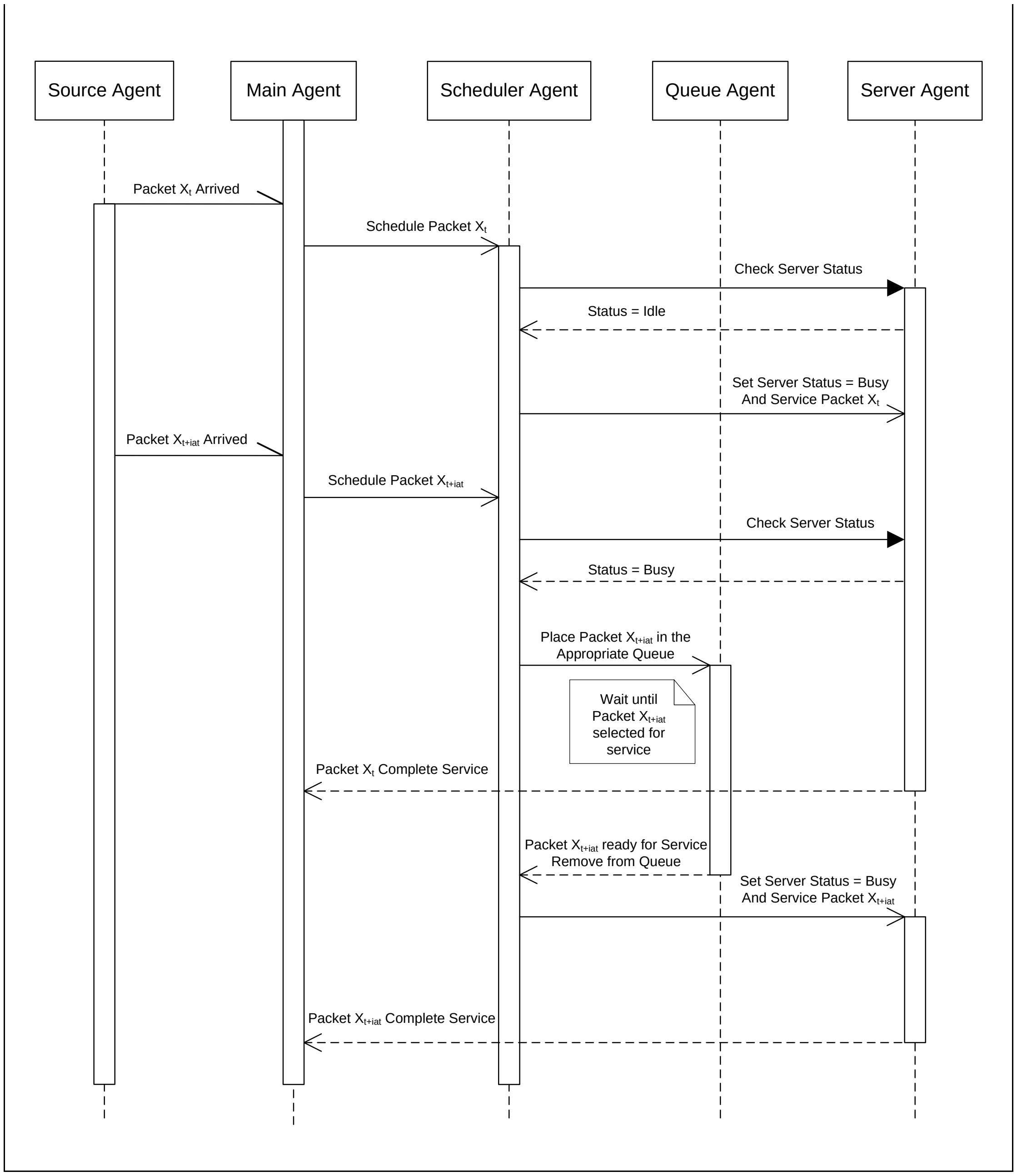
Figure 10: System Agent Model



The interactions between these agents, as shown in Figure 11, started when a packet X_t generated by the source agent requests a service. This request is controlled by the Main agent until its service is completed. Once the Main agent receives the request, then it will interact with the scheduler agent to handle this request. Accordingly, to determine whether the request currently arriving can begin service, the scheduler agent will send an asynchronous message to the server agent to check whether its status is idle or busy. Initially, the server agent is idle, and thus it will reply to the scheduler with idle status. Consequently, the scheduler agent will request the server agent to change its status to busy and start serving the packet request X_t.

At some point in the future, a new packet request X_{t+iat} will be generated by the source agent. This packet will arrive at a time equal to the time the last packet arrived plus an inter-arrival time determined by the source agent. Again, to determine whether this new request can begin service, the scheduler agent will made a check on server status. However, since the server is busy with the packet X_t, the scheduler agent will interact with the queue agent to handle the enqueuing process of the packet X_{t+iat}. Consequently, the queue agent will place the new packet in the appropriate hierarchy queue based on the packet QoS characteristics. At some point in the future, when the packet X_t completes its service, the server agent will change its status to idle and report the packet service completion to the Main agent. Meanwhile, the queue agent will acknowledge the scheduler agent that packet X_{t+iat} is ready for service. Thus, the scheduler agent will request the server agent to change its status to busy and start serving the packet request X_{t+iat} after it is removed from the queue. Once this packet completes its service, the server agent will report its service completion to the Main agent. Indeed, these two scenarios, with different packet requests, will repeat themselves until the whole source generated packets are completed.

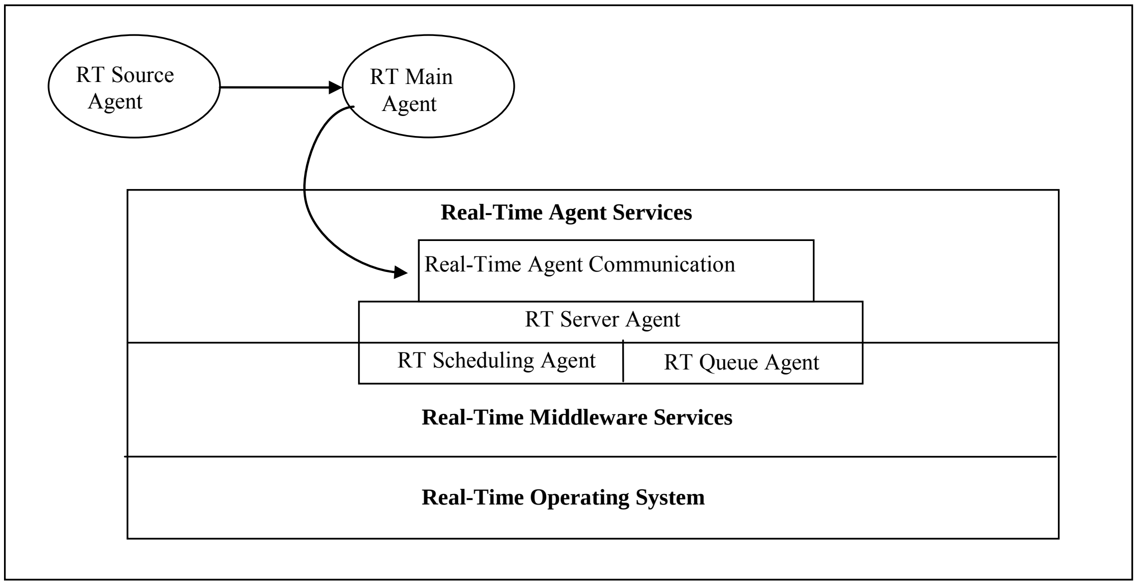
Figure 11: System Agents' Interactions


Source Agent
Main Agent
Scheduler Agent
Queue Agent
Server Agent
Packet X_t Arrived
Schedule Packet X_t
Check Server Status
Status = Idle
Set Server Status = Busy And Service Packet X_t
Packet X_{t+iat} Arrived
Schedule Packet X_{t+iat}
Check Server Status
Status = Busy
Place Packet X_{t+iat} in the Appropriate Queue
Wait until Packet X_{t+iat} selected for service
Packet X_t Complete Service
Packet X_{t+iat} ready for Service Remove from Queue
Set Server Status = Busy And Service Packet X_{t+iat}
Packet X_{t+iat} Complete Service

11. ARTAS Architecture

Our Adaptive Real-Time Agent Scheduling (ARTAS) architecture is a three-layered architecture as depicted in Figure 12. At the lowest layer, we assume having a real-time operating system (RTOS). Above that are the real-time middleware services which consist of the real-time scheduling agent and the real-time queue agent. At this layer, all tasks are scheduled based on the implemented algorithm, i.e. in our case the Hierarchical Diff-EDF scheduler. Finally, the real-time agent services layer extends the scheduling services resulting in task completion. Moreover, the top layer provides a service for real-time agent communications and interactions. In this section we will describe each of the proposed architecture components.

Figure 12: ARTAS Architecture



RT Source Agent: communication among agents in this architecture is performed through a packet request for service from one agent to the others. Each packet request, generated by the source agent, has a formal description of <*A, I, D* >, where *A* represents the source to generate this packet request, *I* is the flow priority to which this packet belongs, and *D* represents the deadline by which the request packet must be completed. If the servicing agent cannot complete this request before its deadline expires then it will be discarded.
RT Main Agent: as we mentioned earlier, the Main agent is the principle actor of the system behavior. Hence, all packet requests should be sent through this agent. It is the agent to control the entire environment through communications with the other agents in the system. The Main agent enforces other agents' policies, disciplines, and actions.
RT Scheduler Agent: our real-time scheduling agent is the key behind enforcing expressed timing and QoS constrains. It is the agent to determine the exact disciplines of how the packets are going to be served. The scheduler agent performs the packets' request depending on their original flow, i.e. RT or Non-RT flow, in order to achieve the best QoS flow's requirements.
RT Queue Agent: it is the agent which treats the incoming packets and place them in the appropriate defined queue based on their flow characteristics requested by the Main agent. The queues have limited size and provided with threshold. The queue agent monitors the queue threshold and interacts with the Main agent to inform the queues' status. It also, has the authority to control the queues' filling rate in real-time and the number of discarded packets according to their class.
RT Server Agent: This agent is responsible for packet request completion. However, it is the Main agent which tells this agent when to change its service discipline. The server agent keeps track of the missed packet and reported them to the Main agent.

In response to the ARTAS architecture management, agent actions are reported to the main agent. The main agent receives these reports as agent events. Possible events are Arrival, Departure, Migration, and Termination. Depending on the agent nature, any other events are also possible such as Buffer Overflow and Threshold. The main agent stores these incoming events which consist of the event type, the emitting source, and some event-specific information such as deadline. Also, we discriminate synchronous events where the event emitting agent waits for the main agent acknowledge, and asynchronous events where the event emitting agent continues its work without acknowledge. Synchronous events will be used for agent control and monitoring while asynchronous events represent simple monitoring information. Events like Migration may be implemented as synchronous events;

control of the agent's moves is necessary if domain administrators want to restrict access to certain agent classes. On the other hand, Events like Termination may be asynchronous since it is sufficient to simply know that this particular agent ceases. Indeed, the main agent handles the list of all agents that are currently in its domain.

12. Comparative Analysis

For evaluating our proposed Hierarchical Diff-EDF scheduler, we present a simulation experiment to study the performance of the Hierarchical Diff-EDF against both EDF and Diff-EDF schedulers. The simulation has been run for arrival rates (λ_T) of 10000 – 60000 packets with an increment step of 5000 packets. The bandwidth is assumed to be 3 Mbps while the packet size 1500 Byte. The analysis elaborates different performance metrics with a focus on the miss rate values per each flow j.

In this section, four graphs were plotted to compare the performance of the three scheduling algorithms for the different flow j. Figure 13 shows the packet miss rate of the video flow when using each of the three scheduling algorithms. The results show that when the system is moderately loaded the three scheduling algorithms give almost the same results. However, when the system is overloaded it is obvious that the EDF performance degrades rapidly while the Hierarchical Diff-EDF scheduler shows the best packet serving with minimum miss rate.

Figure 13: Video Traffic Miss Ratio

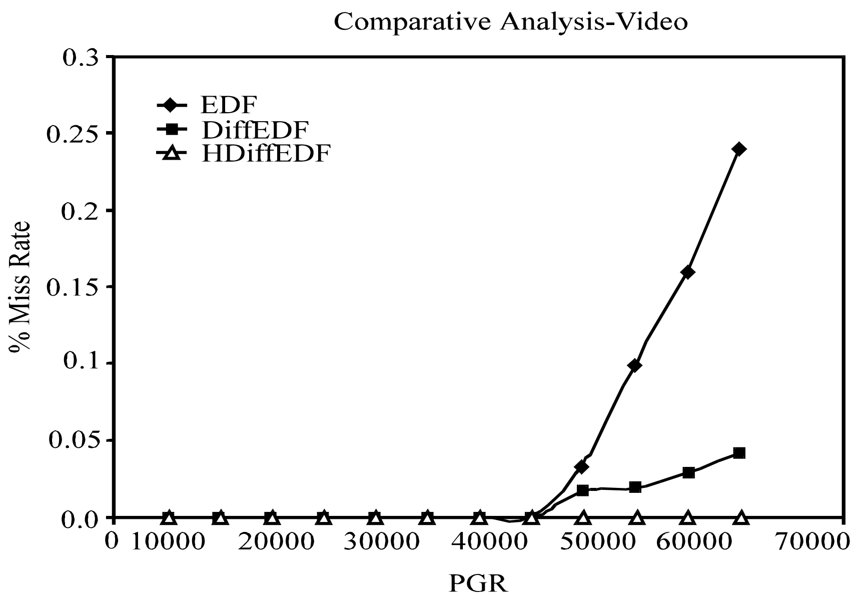
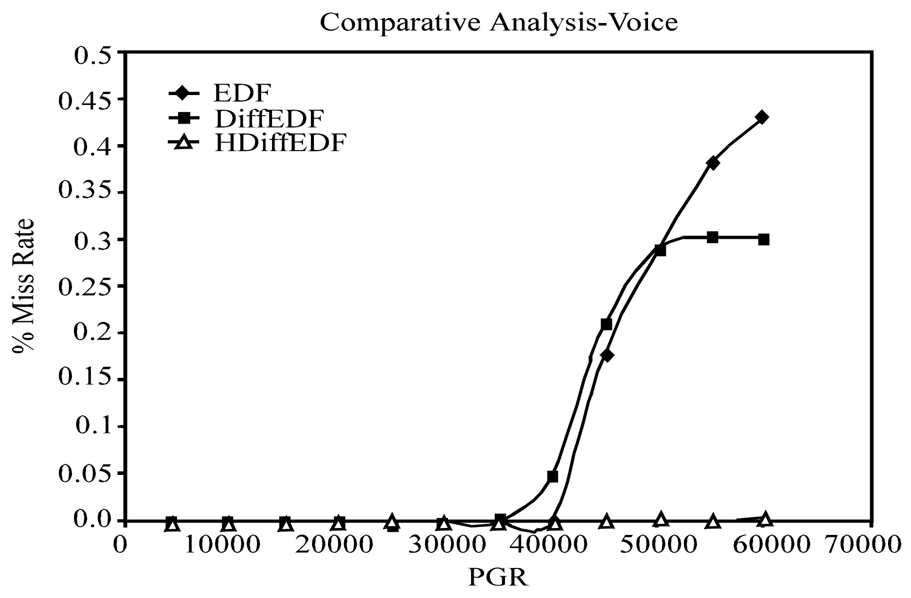


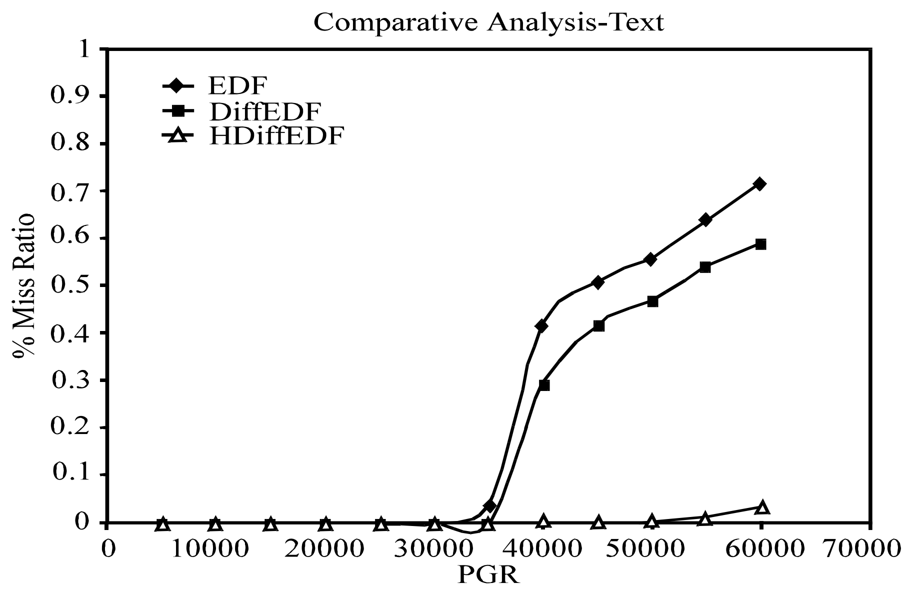
Figure 14 shows that when the system is overloaded the Diff-EDF scheduler gives the lowest miss ratio compare to both EDF and Diff-EDF schedulers. The figure also shows that the EDF performance continues to degrade proportionally with the number of generated packets, while the Diff-EDF degradation settle at a certain point.

Figure 14: Voice Traffic Miss Ratio



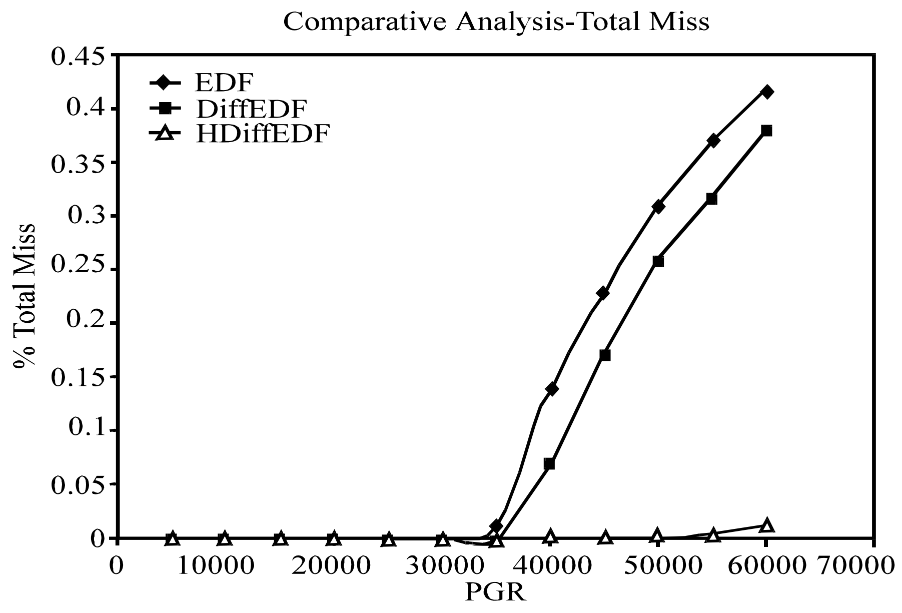
To compare the miss ratio in the case of the text traffic Figure 15 is used. The results show that the Hierarchical Diff-EDF scheduler shows a remarkable performance by achieving a minimum miss ratio compare to both EDF and Diff-EDF schedulers.

Figure 15: Text Traffic Miss Ratio



Finally, the total miss ratio for the different flows j is shown in Figure 16. By analyzing the figure, we can conclude that the Diff-EDF scheduler shows a better performance of packet serving over heterogeneous network traffic through achieving the minimum miss ratio. This improvement is attributed to the use of the QoS priority based packet serving.

Figure 16: Total Traffic Miss Ratio



13. Conclusions

For the past decade, a significant amount of research in data networks and telecommunications has been devoted to providing different levels of service guarantees. In this paper, we have presented the new priority assignment scheduling algorithm Hierarchical Diff-EDF which met the real-time needs while continuing to provide best effort service to the non-real time traffic, over heterogeneous real-time network traffic. The Hierarchical Diff-EDF features a feedback control mechanism that detects overload conditions and modifies packet priority assignments accordingly. Also, our scheduler considers each flow as having stochastic traffic characteristic, a stochastic deadline and a maximum allowable miss rate.

On determining the performance evaluation techniques to be adopted for carrying out this research, we decided to start by attempting toward an analytical solution for the Hierarchical Diff-EDF Scheduler. We started our attempt by defining a formal description for the proposed mathematical model, and then we proceeded with its solution. However, since our queuing system is highly complex, it was difficult to get a closed-form solution, and then the model should be studied by means of simulation.

During the simulation, we adopt a multi agent environment; where a number of agents interact with one another in order to refine system design and management resulting in a higher performance. The multi-agent simulation stages include decomposing the system into a number of elements, modeling each element with agent, and last describing the possible actions between these agents. Also, we introduced our three-layered ARTAS architecture and agent model. The simulation results showed that the Hierarchical Diff-EDF scheduler produces a better performance of packet serving over heterogeneous network traffic through achieving the minimum miss ratio when compared with both EDF and Diff-EDF schedulers. This improvement is attributed to the use of the QoS priority based packet assignment.

Finally, our research paper introduced three contributions. First, we showed that considering a single step hierarchical scheduling allowed for a significant enhance in service guarantees for the different network flows. Second, we proved that deploying a feedback control mechanism to allocate service enforced the desired service guarantees. Last, and third, we demonstrated that our proposed scheduling algorithm can achieve high performance with minimum miss rate.

14. Future Works

Packet-scheduling disciplines are necessary to satisfy the QoS requirements of delay-sensitive applications, and ensure that real-time applications and best effort traffic can coexist on the same network structure. Among packet-scheduling disciplines for providing QoS guarantees to different applications, including real-time services, two classes of algorithms have received particular attention [22, 23]: those based on Generalized Processor Sharing (GPS) and those based on Earliest Deadline First (EDF). As a future work, in situations where the GPS is preferable than EDF, thorough performance evaluation studies between the Hierarchical Diff-EFD and GPS scheduling algorithms can be carried out. Furthermore, the ARTAS architecture and agent model, proposed in this work, can be deployed in such studies to experience its efficiency in refining system's design and management.

References

[1] Srinivasan, A. 2003. Efficient and Flexible Fair Scheduling of Real-time Tasks on Multiprocessors. Chapel Hill.

[2] Zhang, H. & Ferrari, D. 1994. Rate-Controlled Service Disciplines. Journal of High Speed Networks. 3(4): pp. 389-412.

[3] Lu, C., Stankovic, J.A., Tao, G. & Son, S.H. 2002. Feedback Control Real-time Scheduling: Framework, Modeling and Algorithms. Special issue of Real-Time Systems Journal on Control-Theoretic Approaches to Real-Time Computing. 23(1/2): pp. 85-126.

[4] Dertouzos, M. 1974. Control robotics: the procedural control of physical processors. Proceedings of the IFIP Congress, pp. 807–813.

[5] Jackson, J. 1955. Scheduling a Production Line to Minimize Tardiness. Research Report, Management Science Research Project in University of California, 43.

[6] Towsley, D. and Panwar, S.S. 1991. Optimality of the Stochastic Earliest Deadline Policy for the G/M/c Queue Serving Customers with Deadlines. Tech. Report 91-61, Univ. Massachusetts, 1991.

[7] Locke, C.D. 1986. Best Effort Decision Making for Real-Time Scheduling. Ph.D. Thesis, Carnegie Mellon University.

[8] Buttazzo, G.C. & Stankovic, J.A. 1993. RED: Robust Earliest Deadline Scheduling. Proceedings of the 3rd International Workshop on Responsive Computing Systems, Austin, pp. 100-111.

[9] Buttazzo, G., Spuri, M. & Sensini, F. 1995. Values vs. Deadline Scheduling in Overload Conditions. Proceedings of the 16thIEEE Real-Time Systems Symposium, pp. 90-99.

[10] Firoiu, V., Kurose, J. & Towsley, D. 1997. Efficient Admission Control for EDF Schedulers. Proceedings of the Sixteenth Annual Joint Conference of the IEEE Computer and Communications. 1:310-317.

[11] Stankovic, J.A., Spuri, M., Ramamritham, K. & Buttazzo, G.C. 1998. Deadline Scheduling for Real-Time Systems: EDF and Related Algorithms. Boston: Kluwer Academic Publishers.

[12] Lam, T., Ngan, T.J. & TO, K. 2004. Performance Guarantee for EDF under Overload. The ACM Journal of Algorithms. **52**(2): 193-206.

[13] Harista, J.R., Livny, M. & Carey, M.J. 1991. Earliest Deadline Scheduling for Real-Time Database Systems. The IEEE Real-Time Systems Symposium, pp. 232-242.

[14] Zhu, F., Lehoczky, J.P., Hansen J.P. & Rajkumar, R. 2005. Diff-EDF: A Simple Mechanism for Differentiated EDF Services. Proceedings of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 268-277.

[15] Lehoczky, J.P. 1996. Real-Time Queuing Theory. In Proceedings of the 17th IEEE Real-Time Systems Symposium, pp. 186-195.

[16] Lehoczky, J.P. 1998. Scheduling Communication Networks Carrying Real-Time Traffic. In Proceedings of the IEEE Real-Time Systems Symposium, pp. 470-479.

[17] Zhu, H.F., Hansen, J.P., Lehoczky, J.P. & Rajkumar, R. 2002. Optimal Partitioning for Quantized EDF Scheduling. In Proceedings of the 23rd IEEE Real-Time Systems Symposium, pp. 212-222.

[18] Marse, K. & Roberts, S.D. 1983. Implementing a Portable FORTRAN Uniform (0,1) Generator. Simulation. 41: 135-139.

[19] Calyam, P. and Lee, C. 2005. Characterizing Voice and Video Traffic Behavior over the Internet. International Symposium on Computer and Information Sciences (ISCIS).

[20] R.V. Hogg, and A.T. Craig, Introduction to Mathematical Statistics, New York, Macmillan Publishing, 1978.

[21] Wooldridge, M., Jennings, N. R. and Kinny, D. 2000. The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems. 3(3): 285-312.

[22] Fabio, M.C. & Sivaraman, V. 1998. Achieving High Utilization in Guaranteed Services Networks Using Early-Deadline-First Scheduling. The 16th IEEE International Workshop on Quality of Service, pp 209-217.

[23] Zhang, H. 1995. Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks. Proceedings of the IEEE, 83(10): 1374-1396.