Multimed Tools Appl (2018) 77:12725–12748
DOI 10.1007/s11042-017-4912-6

A QoS-enhanced intelligent stochastic real-time packet scheduler for multimedia IP traffic

Suman Paul[1,2] · **Malay Kumar Pandit**[1]

Received: 19 October 2016 / Revised: 29 March 2017 / Accepted: 5 June 2017 /
Published online: 22 June 2017

Abstract A re-configurable, QoS-enhanced intelligent stochastic real-time optimal fair packet scheduler, QUEST, for IP routers is proposed and investigated. The objective is to maximize the system QoS subject to the constraint that the processor utilization is kept at 100%. All past work on router schedulers for multimedia traffic were of earlier generation, in that they focused on maximizing utilization whereas being QoS-aware but without explicitly maximizing the QoS. Keeping utilization fixed at nearly 100%, QoS is dynamically maximized, thus moving to the next generation. QUEST's other unique advantages are three-fold. First, it solves the challenging problem of starvation for low priority processes; second, it solves the major bottleneck of Earliest Deadline First scheduler's failure at heavy traffic loads. Finally, QUEST offers the benefit of arbitrarily pre-programming the process utilization ratio. Three classes of multimedia IP traffic, namely, VoIP, IPTV and HTTP have been considered. Two most important QoS metrics, namely, packet loss rate (PLR) and mean waiting time, are addressed. All claims are supported by discrete event and Monte Carlo simulations. The proposed scheduler outperforms benchmark schedulers and offers 37% improvement in packet loss rate and 23% improvement in mean waiting time over the best competing current scheduler Accuracy-aware EDF. The proposed scheduler was validated in a test-bed platform of a NetFPGA® router and results were observed with Paessler® PRTG network monitor.

Keywords Cache and deadline misses · Hidden Markov model · Packet loss rate (PLR) · Quality of service (QoS) · Scheduling algorithm

✉ Suman Paul
paulsuman999@gmail.com

Malay Kumar Pandit
mkp10011@yahoo.com

[1] Department of Electronics and Communication Engineering, Haldia Institute of Technology 721657, West Bengal University of Technology (Maulana Abul Kalam Azad University of Technology West Bengal), Kolkata 700064, India

[2] School of Engineering and Technology, West Bengal University of Technology (Maulana Abul Kalam Azad University of Technology West Bengal), Kolkata 700064, India

1 Introduction

Quality of Service (QoS) [41] in telecommunication systems is directly related to the network performance of the underlying routing systems. QoS is defined as the collective effect of service performance which determines the degree of satisfaction of a user of the service. In quest for quality, current researchers are trying to maximize the quality of service of real-time embedded systems including IP (internet protocol) routers. A router is a specific case of soft-real time embedded systems. Scheduling is a *crucial* integral part of modern IP routers. Optimally scheduling the different tasks in a multitasking computing system is vitally important. Optimizing the system performance critically depends on appropriate processor usage time allocated to the processes for guaranteeing high system QoS. The latter is of prime concern in designing state-of-the-art real-time embedded systems e.g., routers as it addresses key attributes (parameters) like sources of errors, packet loss rate (PLR), latencies (sum of mean waiting time and service time), resource availabilities, end-to-end delay, jitter (delay variation), throughput, fair bandwidth allocation etc. A rigorous probabilistic *framework* for a novel *optimal intelligent* embedded computing scheduler, QUEST (quality-of-service enhanced stochastic), for IP routers is presented here for the first time. Two major gaps in scheduler research have been identified. One is the starvation of low priority processes. The other is the poor performance of the premier EDF scheduler at heavy traffic loads. Addressing these two problems *motivated* the authors to undertake the present research. In EDF scheduler and its variants, the rise of the mean waiting time to an unacceptably high level at heavy loads, is a *long-standing* problem which has been successfully solved in this work by explicitly focusing on the heavy-load zone (utilization close to 100%).

1.1 Scheduling attributes

The proposed QoS-enhanced intelligent stochastic packet scheduler, QUEST, for IP routers is based on pre-emptive scheduling but it differs from the conventional schedulers in that it is probabilistic in nature in order to keep the utilization fixed in a fair way. The scheduler offers the following unique advantages:

(i) Higher priority processes cannot monopolize the processor and the lower priority processes do not starve. Lower priority processes acquire a guaranteed minimum amount of processor time due to the pre-designed distribution of individual process utilization. This justifies that the scheduler is fair in nature and eliminates the problem of priority starvation.
(ii) The scheduler is an adaptive and re-configurable one. A machine-learning feedback controller is used to implement this adaptability and re-configurability. This feedback-controller with the help of run-time cache-miss and deadline-miss error feedbacks learns and takes corrective decisions to maximize the system QoS.
(iii) The objective is to maximize the system QoS, subject to the constraint that utilization is kept at 100%. An optimum utilization close to 100% is *enforced*. In this scheduling scheme, process utilization, U_i for a process P_i, is expressed as,

$$U_i = \frac{T_i}{D_i} \tag{1}$$

where T_i is the fraction of time spent for execution of process P_i.D_i is denoted as the deadline of the process P_i. The state probability vector of process utilization ratio of n processes running in a system can be expressed as, $\prod$

$$\prod = [U_1 : U_2 : ..U_{n-1} : U_n] \tag{2}$$

The proposed scheduler is dynamic priority based. In Section 6.3, it is demonstrated that $\sum U_i = 1$, which indicates that the processor utilization is 100%. Hence, the scheduler is *optimally* schedulable [22].

(iv) Last, the QUEST is strongly immune from hacking because the scheduler is random in nature and therefore the next process to be executed cannot be predicted apriori.

In practice, for an end-to-end QoS sensitive multimedia traffic, which has a commitment to deliver on time, the process utilization for different classes of multimedia traffic is tailored in such a manner that a guaranteed minimum amount of processor attention for each traffic is maintained. For multimedia embedded (router) applications considered in this paper, Voice over Internet Protocol (VoIP), Internet Protocol Television (IPTV) which are real-time traffic and web browsing using Hyper Text Transfer Protocol (HTTP) which is the best effort network traffic processes follow a long-tailed *Pareto* distribution of process utilization ratio. In this proposed service-differentiated scheduling model, a *target* process utilization ratio is achieved and maintained as per designer's requirement. A practical case of process utilization ratio, U_i, for three processes has been provisioned in the ratio of 80:16:4.

1.2 System quality of service (QoS)

Delivering QoS means guaranteeing given service parameters within certain bounds for-connections made over a network [5]. The most dominant QoS parameter in a router is the packet loss rate (PLR) [36] encountered in system activities that may arise due to different errors like deadline miss, L1 and L2 cache misses [28], page fault, etc. Overall, two most important QoS's metrics, namely, PLR and mean waiting time (related to system latency) are focused on in this paper. Practical cache miss error probabilities come in the range of $[10^{-2}–10^{-1}]$ [32]. Practical deadline miss error probabilities come in the range of [0.013–0.12] [18]. For practical real-time tasks, the deadline varies in the range of 10–300 ms [2, 30].

2 Related work

Several distinguished studies deal with QoS metrics for scheduling multimedia traffic in routers. In routers, the simplest First-come first-served (FCFS) scheduler receives packets from all input traffic classes. Packets are assigned to a single queue upon arrival and are serviced on a first-come, first-served basis. An FCFS scheduler cannot differentiate multimedia traffic classes. Packets may be dropped if the queue is full. Cristofaro et al. [8] have presented a detailed comparative analysis of QoS attributes for the VoIP and video conferencing traffic with different queueing policies. However, the study has no focus on PLR. By using First-Come-First-Served (FCFS) and Earliest Deadline First (EDF) schedulers, Saleh and Dong [29]

have studied three QoS metrics, namely, miss ratio, delay, and average size of the buffer. The authors have demonstrated the efficiency by using the EDF scheduler in a hybrid network to provide QoS guarantees. But the authors have shown that the FCFS schedulers are more efficient for serving best-effort data traffic than the EDF. But, the research has no specific theme on the priority starvation of lower priority traffic class, re-configurability of the scheduler and process utilization. In [15], the authors have proposed an analytical model for priority queueing systems in a heterogeneous long range dependent self-similar and short range dependent Poisson traffic. The proposed model cannot guarantee a steady state process utilization ratio.

Toral-Cruz et al. [33] have analyzed QoS parameters, namely, jitter and packet loss rate of VoIP traffic. The studies have revealed that VoIP jitter can be modeled by self-similar processes with short or long range dependences. However, the work does not concentrate on maximizing the QoS metrics. Rikli et al. [27] have evaluated various queueing disciplines, such as, fair queueing (FQ), priority queueing (PQ), custom queueing (CQ), low-latency queueing (LLQ) in IP routers to provide the end-to-end QoS requirements for various traffic classes. In case of increasing high priority traffic sources, for target QoS requirements, the authors have proposed solution either by changing the prioritization scheme at the switching routers in favour of priority classes or by allocating more bandwidth. However, the scheme cannot eliminate the problem of priority starvation for low priority best effort traffic classes and allocation of bandwidth is not a dynamic one.

Ghazela and Saïdaneb [13] have proposed a queuing delay control and adjustment method, which guarantees the required QoS in terms of per-service traffic flow authorized for the real-time multi-service traffic. This method deals how to control the queuing delay value at the specified waiting delay by adjusting the arrival probability, so that the QoS delay for real-time services may be guaranteed. However, the scheme has no provision of reconfiguring the scheduler. The proposed method does not deal with the dominant QoS metric PLR.

In [21], the authors have demonstrated a reconfiguration-aware real-time scheduling mechanism under QoS constraints where only VoIP traffic has been considered. Further, no explicit mechanism to enhance the system QoS and supporting queueing theory are not mentioned. Greco et al. [14] have contributed on a multitasking, pre-emptive RTOS environment in a stochastic scheduling domain. Although the model is based on Markov chain, it provides no focus on state estimation by machine learning. Further, the scheduler is not a re-configurable one.

Based on literature survey it is observed that in a multitasking scheduler in IP routers, dynamically optimizing the system QoS based on Markov chain model has not been specifically focused. The novelty of search technique to find the global minimum value of PLR in the search space is novel in this work. Several approaches have been proposed based on real-time pre-emptive scheduling algorithms, for example static priority scheduling: rate monotonic (RM), dynamic priority scheduling: earliest deadline first (EDF) and its variants. In these scheduling mechanisms, lower priority processes get over penalized because of suspension of execution by the higher priority processes. Using EDF in a dynamic environment of real-world applications for an overloaded system processes miss deadlines frequently resulting in very low value of throughput. EDF is unsuitable in real-time packet network traffic as all traffic classes receive the same miss rate irrespective of deadline requirements and traffic characteristics. Further, EDF does

not honour class differentiation for traffic and therefore fails to comply with the service level agreements (SLAs) with client processes. Last, EDF and its variant A-EDF are deadline driven, where process utilization has no explicit focus. The *root* of the problem can be traced to its deterministic and deadline-driven mode of operation. Taking a *novel* alternative *route* here, namely, non-deterministic stochastic and utilization (load)-driven operation, the bottleneck has been circumvented.

These problems have been *solved* through the proposed scheduling *framework*. Here, a non-deterministic *optimal* scheduler, QUEST, which is random in nature has been implemented so that the highest priority process does not dominate the processor execution time and the problem of starvation of the low priority process never occurs. QUEST is strictly traffic class-sensitive and fully conforms to SLAs. Additionally, QUEST is a *deadline-aware utilization-driven* scheduling scheme.

The rest of this paper is organized as follows. Section 3 and 4 discuss proposed system model and formulate the scheduling mechanism and queue management, respectively. Section 5 presents simulation methodology, followed by simulation results in Section 6. Dynamic global optimization and re-configurability of the scheduler are described in Section 7. Section 8 reports run-time estimation of transition probability matrix (TPM) by machine learning. Stability and accuracy of run-time TPM estimation is provided in the same Section. A comparative performance analysis of QUEST is evaluated in Section 9. Test-bed implementation for QUEST is presented in Section 10. Finally, conclusion is stated in Section 11.

3 Proposed system model

The design has been implemented for three classes (multimedia traffic flows) - VoIP, IPTV and HTTP. A Finite-state machine (FSM) based on Markov chain model for the scheduler is reported in this paper. Markov model is a stochastic model in which the probability that a random variable, X, takes on the value x_{n+1} at time step (n + 1) is entirely determined by its state value in the previous time step n and it is independent of its state values in earlier time steps: n-1, n-2, etc. Each process in this scheme modelled as a particular Markov state. The processes are characterized by their state probabilities (p_{ij})s which are defined as probabilities of processes to be in their own states ($p_{ij,i=j}$) or to make transitions to other states ($p_{ij,i\neq j}$). In this scheme, the class processes settle to a steady state probability distribution according to time evolution.

The underlying model behind this scheduling framework is a Hidden Markov Model (HMM). To find the *most likely* (ML) path of reaching the desired final steady state probability vector (string) is a *heuristic* process. Therefore, HMM is an NP-Hard problem. Since HMM is an NP-Hard problem [23], Markov initial TPM parameters (matrix elements) are calculated *apriori* using *machine learning* Metropolis-Hastings algorithm: stated in algorithm 1 [7]. Metropolis-Hastings algorithm is a special class of Markov Chain Monte Carlo (MCMC) method, with constraints like the diagonal elements of the TPM are in the range: [0.4–0.9] and the non-diagonal elements are in the range: [0.01–0.6] [34]. It has been observed that a faster convergence is achieved [34] in such cases. Because of Markovian property, target steady state probability distribution can be generated. The corresponding TPM is estimated by maximum likelihood. To support the above proposition in an embedded computing environment in a router, the

desired steady state probability distribution, *ξ:φ:ï (where ξ + φ + ï = 1)* for three processes representing their corresponding three classes have been considered.

Algorithm 1 Metropolis-Hastings algorithm

1: Initialize: $y^{(0)}$: $p(y)$

2: for iteration i= 1,2… do

3: Propose: y^{sample}: $p(y^{(i)}|y^{(i-1)})$

4: Acceptance Probability:

5: $\alpha(y^{sample}|y^{(i-1)}) = min\left\{1, \frac{p(y^{(i-1)}|y^{sample})\pi(y^{sample})}{p(y^{sample}|y^{(i-1)})\pi(y^{(i-1)})}\right\}$

6: u: Uniform$(u; 0,1)$

7: **if** $u < \alpha$ then

8: Accept proposal: $y^{(i)} \leftarrow y^{sample}$

9: **else**

10: Reject proposal: $y^{(i)} \leftarrow y^{(i-1)}$

11: **end if**

12: **end for**

The first step is to initialize the sample value for each random variable. The algorithm consists of three steps: First, a proposal sample y^{sample} is generated from the proposal distribution $p(y^{(i)}|y^{(i-1)})$; second, based upon the proposal distribution and the full joint density $\pi(\cdot)$, the acceptance probability is computed using acceptance function $\alpha(y^{sample}|\ y^{(i-1)})$; third, the candidate sample is accepted with probability α, or rejected with probability $(1-\alpha)$. For multimedia IP traffic considered in this work, the desired (fractal Pareto type) steady-state distributions are of the order of 0.80: 0.16: 0.04 as justified later in Section 4 with Table 1. So ξ = 0.8, φ = 0.16 and ï = 0.04 are considered. An initial approximate estimate for the 3 × 3 Transition Probability Matrix (TPM), *'T'* is estimated by using the *machine learning* Metropolis-Hastings

Table 1 Service models parameters

Traffic class	Service type	QCI value	Deadline (ms)	Arrival feature
P_1 (VoIP)	RT-GBR	1	(ITU G.711) 20	MMPP
P_2 (IPTV)	non-GBR	6	(ITU G.114) 100	MMPP
P_3 (HTTP)	Best effort non-GBR	9	400	MMPP

algorithm to provision a steady state distribution of process utilization ratio 0.80: 0.16: 0.04. '*T*' is stated in Eq. (3).

$$T = \begin{pmatrix} 0.90 & 0.08 & 0.02 \\ 0.39 & 0.56 & 0.05 \\ 0.42 & 0.18 & 0.40 \end{pmatrix} \tag{3}$$

The $\prod$, the state probability vector, is treated as process utilization ratio as discussed earlier. Ignoring apriori information, an initial unbiased state probability vector, $\prod_0 = 1/3[1\ 1\ 1]$ is applied and the *estimated* final state probability vector, $\prod_f$ is obtained as, *[0.79829:0.16154: 0.040071], as* illustrated in Fig. 1a. Figure 1b indicates that, although an initial biased state probability vector, $\prod_0 = [0.1\ 0.5\ 0.4]$ is applied, the *estimated* final state probability vector, $\prod_f$ is obtained as, $\prod_f = [0.79837:0.16155:0.040075]$, approximately same as in Fig. 1a.

The result *confirms* that a final practical process utilization ratio, $\prod_u = [U_1: U_2: U_3]$ distribution i.e. [0.80: 0.16: 0.04] for three processes of corresponding classes, has been achieved, irrespective of the initial distribution. It is to be noted that a specific value of U_i, achieved here, is under the control of designer's choice. In general, any target v*alues* of $\prod_f$, namely, *[0.81 0.13 0.06], [0.65 0.25 0.10]*, etc. can be achieved as per designer's requirement because Metropolis-Hastings algorithm can generate any arbitrary desired steady state distribution [7].

4 Scheduling mechanism and queue management

The multi-service packet scheduling (PS) scheme, QUEST, as shown in Fig. 2 accepts three different classes of incoming multimedia traffic - VoIP, IPTV and HTTP. Traffic streams are classified by a classifier and fed to three distributed FIFO queues: Q_1, Q_2 and Q_3 for VoIP, IPTV and HTTP, respectively. Migration of traffics among the queues are not allowed. The proposed model is defined as M/BP/1/./QUEST. In this underlying model, 'M' denotes traffic arrivals which are of Markovian type modulated by Poisson process (MMPP). In real world applications, this scheme is a fair estimation of large number of independent memory-less events [20]. Further, according to recent approaches [1], for a settled system, incoming traffic streams defined by different distributions converge to a Poisson distribution as time evolves. 'BP' refers to the service time distribution which is of Bounded Pareto type. '1' indicates single processor. The incoming processes are being scheduled and executed according to the QUEST scheduler in a preemptive-resume manner. Service of each traffic is related with the

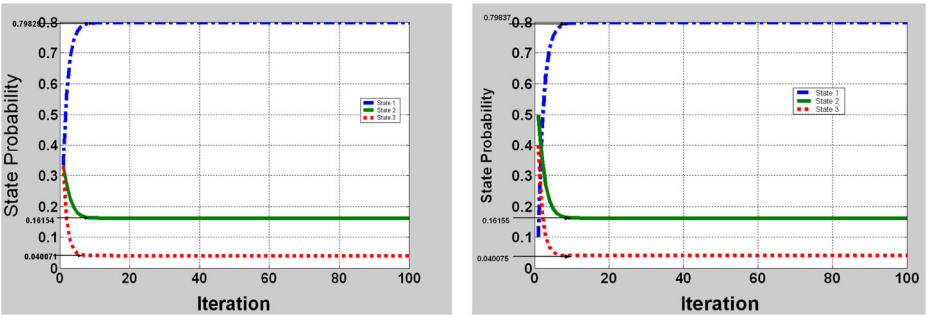

Fig. 1 Confirmation of convergence of three states. **a** $\prod_0 = 1/3[1\ 1\ 1]$, **b** $\prod_0 = [0.1\ 0.5\ 0.4]$

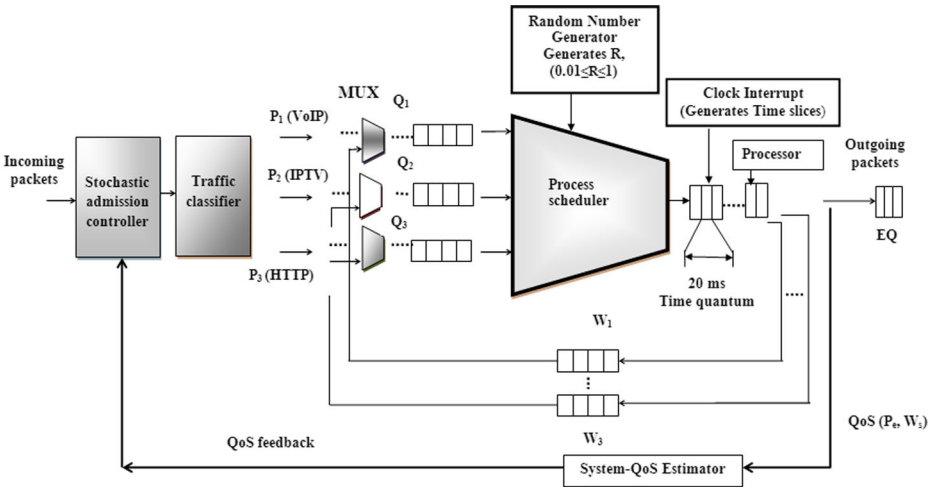

Fig. 2 Illustration of M/BP/1/./QUEST model. Q_i: Ready queues, W_i: Waiting queues, EQ: Expired queue

defined value of QoS Class Identifiers (QCI). QCI defines its performance objective and a lower value QCI denotes more restrictive services in terms of performance. The deadlines for VoIP, IPTV, HTTP are set as stated in Table 1. These values have been taken considering acceptable practical deadline [6, 31] in real world applications.

Let, P_i denotes the representative process for the corresponding class C_i. The priorities assigned to processes are inversely proportional to their deadline [19]. Therefore, the priority of execution of processes are kept in the order of, $P_1 > P_2 > P_3$ and the process utilization ratio is provisioned as [0.8:0.16:0.04]. In this scheduling policy, a clock interrupt generates the timing slices or *quanta*. After each slice, the next process is picked up from the ready queue. The scheduler runs through the ready queue, selects a process from a queue of processes to execute depending on the outcome of a random number generator, runs through the time slice, eventually placing the finished process in an expired queue. For practical real-time tasks, deadlines are in the range of 10–300 ms [2, 30]. Considering uniform burst time which is made possible by traffic conditioning algorithms like token bucket, leaky bucket, etc., the process utilization (U_i) [22, 37] of the system is expressed in Eq. (4).

$$\sum_{1}^{3} U_i = T_B.\left(\frac{1}{D_1}+\frac{1}{D_2}+\frac{1}{D_3}\right)\leq 1 \tag{4}$$

In this scheme, T_B denotes the burst time (service time) and the deadlines of processes are denoted by D_i. In case, $D_1 = 20\ ms$, $D_2 = 100\ ms$, $D_3 = 400\ ms$ the value of burst time is calculated as, $T_B \leq 16\ ms$. Allowing *4 ms* timing jitter (T_J) provides the required value of time quantum (T_Q). Thus, $T_Q = T_B + T_J = 20\ ms$. In this framework, the time quantum, T_Q, is set at 20 ms so that pre-emption does not result in deadline misses. In practical case, this value of time quantum 20 ms is acceptable because it is at least equal to the minimum process deadline 20 ms, which is required for highest priority VoIP (process P_1) traffic to avoid context switching. Thus, designing the value of burst time as 16 ms *concretely justifies* its use to keep the system utilization 100%. Although, for demonstrating the concept, the authors have considered three traffic classes, the framework is general and can be expected to any number of processes because it is based on Markov model.

4.1 QUEST scheduling algorithm

Algorithm 2 states formal description of the proposed scheduling algorithm.

Algorithm 2 QUEST scheduling algorithm

1: Generate random number R,0.01≤R≤1;

2: Set: Time quantum T_Q: 20 ms

3: Initialize: timer, t=0

4: for t=1,2...20 ms do

5: **if** (0.01≤R≤0.8) then

6: execute P_1;

7: **else if** (0.81≤R≤0.96) then

8: execute P_2;

9: **else** execute P_3;

10: **end if**;

11: **end for**;

12: Place P_i in expired queue.

Algorithm 2 clearly indicates that QUEST is a *true dynamic-priority* scheduler because the next process to be executed depends purely on the outcome of the random number generator decided at run-time and *may not* have the highest priority among the pending processes.

4.2 Mean waiting time

Let, a random variable X taking value x in the interval *[l, q]*. The probability density function of Bounded Pareto distribution of queue service time is given by

$$f_x(x) = \frac{\theta . l^{\theta} . x^{-(\theta+1)}}{1-\left(\frac{l}{q}\right)^{\theta}}, \quad l \le x \le q \tag{5}$$

where θ is the shape parameter, l and q denote minimum and maximum IP data file sizes, respectively.

The second moment of this distribution is calculated as,

$$E_x\left(x^2\right) = \int_l^q x^2 . f_x(x) dx = \frac{\theta . l^{\theta}}{1-\left(\frac{l}{q}\right)^{\theta}} \cdot \frac{\theta}{(\theta-2)} \cdot \left(l^{2-\theta} - q^{2-\theta}\right) \tag{6}$$

The second moment of the service time distribution, $E[X^2]$ is calculated as,

$$E\left[X^2\right] = \frac{E_X(x^2)}{L_C^2} \tag{7}$$

where L_c, is the link capacity of the system.

From queueing theory, mean waiting time, W_s without using a stochastic admission controller can be expressed as

$$W_S = \frac{\lambda E\left[X^2\right]}{2(1-\rho)} \tag{8}$$

where ρ, is normalized load of the system and the traffic arrival rate is denoted by λ. The arrival rate is expressed in terms of number of incoming packets per second.

4.3 Packet loss rate (PLR)

PLR is expressed as, PLR = $(N_s - N_r)/N_s$, where N_s and N_r are denoted as number of packets sent and number of packets received, respectively. In this work, the packet loss rate (PLR) is expressed as the root mean squared error, $P_{e,rms,}$ of L1, L2 cache miss and deadline miss errors of the system. $P_{e,rms}$ is stated in Eq. (9). L1 cache miss error, L2 cache miss error and the deadline miss error are denoted by C_{L1}, C_{L2} and D_e respectively.

$$P_{e,rms} = \sqrt{C_{L1}^2 + C_{L2}^2 + D_e^2} \tag{9}$$

For each of the three processes: VoIP, IPTV, HTTP, the above r.m.s error is calculated from Eq. (9) and substituted in the second row of error probability matrix, E, given in Eq. (10).

5 Simulation methodology

For simulation, an initial model is characterized by two matrices, i) the TPM, *'T'* stated in Eq. (3) for the Markov model considered (here three-state model) and ii) *'E'*, an error (vector) probability matrix in (10). Practical values of cache miss errors [32] and deadline miss error [18] rates have been taken.

$$E = \begin{pmatrix} 0.98 & 0.9 & 0.8 \\ 0.02 & 0.1 & 0.2 \end{pmatrix} \tag{10}$$

The three elements in the second row in Eq. (10) represent error probabilities of the processes of corresponding classes and the elements in first row indicate the probabilities of correctness. The simulation framework has been developed using a discrete event simulator, DEVS suite [9] and MATLAB R 2015 b (version 8.6) in a computer having specification of Intel i3 CPU 2.5 GHz, 4GB RAM, Windows 7 platform. Monte Carlo method has been applied for confirmation. Following (Table 2) system environment for simulation was used:

Table 2 Simulation parameters

Parameter	Conditions
Arrival rate:	50 packets/s
Data file size:	20–400 KB
Burst time:	16 ms
Shape parameter (θ):	0.14
Service Discipline:	QUEST
Link Capacity (L_c):	10 Mbps
Packet size:	1 KB
Simulation time:	289 s

6 Simulation results

In this section simulation results are presented.

6.1 Waiting time of individual process

Waiting time for each class of traffic in this simulation are plotted with respect to increasing normalized load as shown in Fig. 3.

6.2 Comparative performance analysis of mean waiting time

In this subsection, a comparative performance analysis in terms of mean waiting time, for QUEST, with current state-of-the-art scheduling algorithms - deferred pre-emption (DP) [4], earliest deadline first (EDF) [37] and accuracy-aware EDF (A-EDF) [24] has been illustrated in Fig. 4.

Figure 4 shows that, QUEST experiences significantly the lowest value of mean waiting time with higher normalized load and it exhibits 23% improvement with respect to best competing A-EDF. Usage of a stochastic admission controller [12] which is permissible in QUEST, keeps the mean waiting time low even at high traffic loads close to 100%. On the other hand, EDF and its variant A-EDF are not stochastic, avoiding usage of such admission controllers. Therefore, for EDF, mean waiting time can be low only for loads below about 80% [12], which contradicts our original problem objective of 100% utilization. If stochastic admission controller is not used, in high load condition, the mean waiting time rise-rate would be steeper as happens with EDF and A-EDF

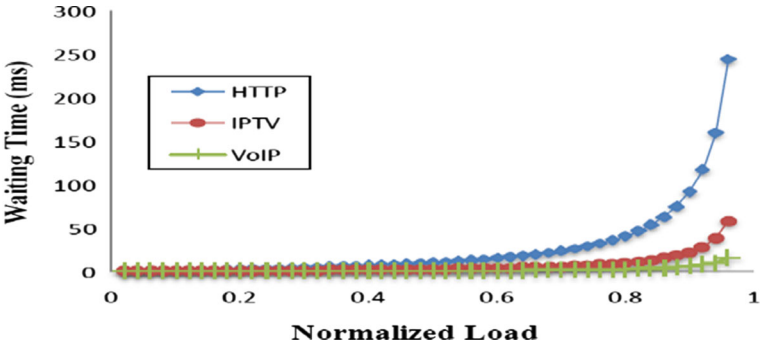

Fig. 3 Waiting time comparison for different processes for QUEST

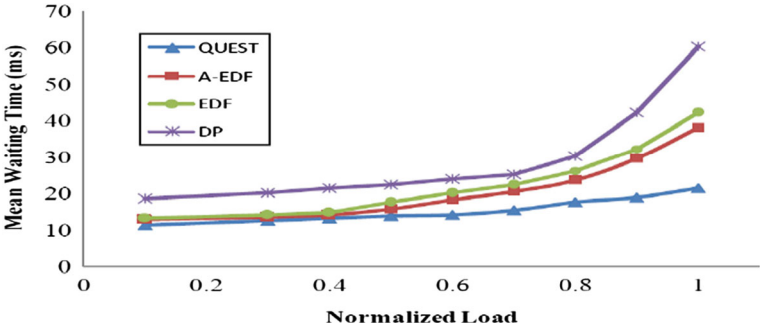

Fig. 4 Mean waiting time with increasing load

depicted in Fig. 4. Furthermore, Rate Monotonic (RM) as well as DP (Fig. 4) are static priority scheduling algorithms and therefore, experiences significant rise of mean waiting time with increasing normalized traffic load.

6.3 Steady state probability analysis and system stability

Simulations were performed considering random arrival of processes with the given error vector. The error vector provides error positions in 2000 sequences (iterations). The probability of finding the processor in a given state is calculated from *'T'* and the error probability is obtained from *'E'*.

As shown in Fig. 5, Process P_1 (VoIP), Process P_2 (IPTV), and Process P_3 (HTTP) achieve steady state probabilities of 0.796, 0.161 and 0.043, respectively. The PLR (denoted as P_e) thus obtained is 0.0045 (Fig. 6), which is acceptable because it falls within the standard PLR threshold of 1% [17].

Thus, the lowest priority process traffic HTTP secures a *guaranteed* 4.3% process utilization which validates authors' claim that *low-priority process starvation* is *eliminated*. Simulations were performed to calculate the packet loss rate (PLR) which is denoted as P_e. Results show that with the increasing count of sequences (iterations), P_e settles to a steady state value (shown in Fig. 6). This validates consideration of the processes as *stable* Markov states, and establishes *system stability.*

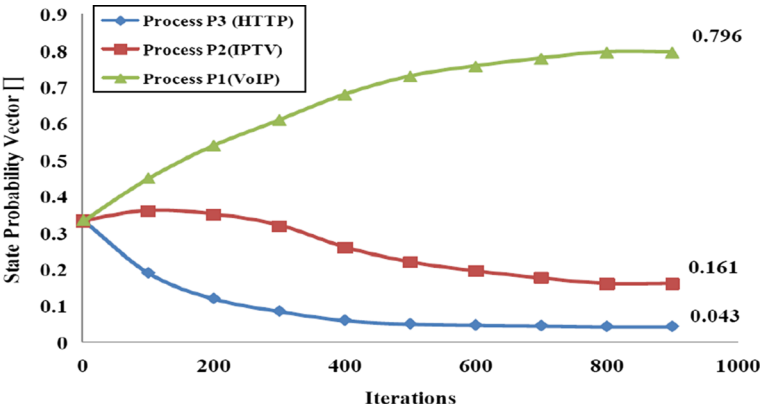

Fig. 5 Convergence of State Probability Vector Π

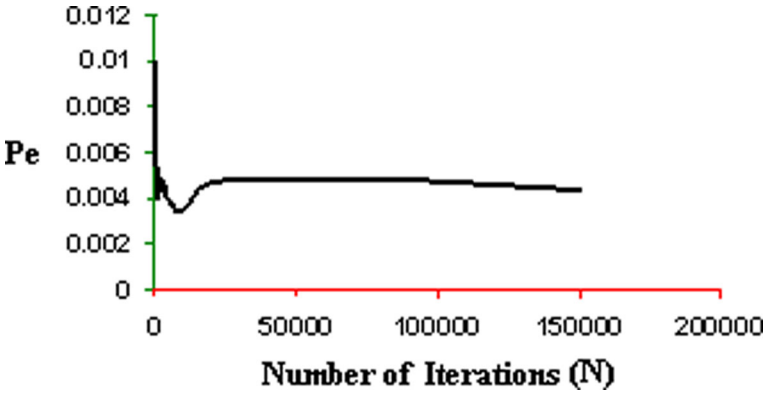

Fig. 6 P_e converges to a steady state with number of increasing iterations

7 Dynamic global optimization and re-configurability of QUEST

PLR is to be minimized to optimize system performance. Due to the varying nature of load, the pre-allocated state transition probabilities of matrix '*T*' are unfit to provision the QoS at its *maximum*. This problem is *solved* in a unique, ingenious way by re-configuring the matrix '*T*' using reconfiguration (*tuning*) parameters, Δ_1, Δ_2 and Δ_3 as stated in Eq. (11).

$$T_{recon} = \begin{pmatrix} 0.90-2\Delta_1 & 0.08+\Delta_1 & 0.02+\Delta_1 \\ 0.39+\Delta_2 & 0.56-2\Delta_2 & 0.05+\Delta_2 \\ 0.42+\Delta_3 & 0.18+\Delta_3 & 0.40-2\Delta_3 \end{pmatrix} \tag{11}$$

These reconfiguration parameters drive the PLR to a *minimum* value and hence QoS back to maximum value by the feedback controller shown in Fig. 7.

In reality, the processor usage allotment to all processes is dynamic over time and event-driven. The system QoS is dynamically monitored by the scheduler using a feedback controller with the help of decision making unit (DMU) and necessary corrective actions are implemented.

Use of feedback controller in the proposed QUEST is of twofold. Feedback controller increases performance of QUEST irrespective of internal and external uncertainties. Further, it automatically reconfigures the scheduler to run within user defined range on-the-fly.

The error feedback controller is used to reconfigure the QUEST by suitably tuning Δ_i s. The 3D-contour plot of PLR (denoted as P_e) as function of Δ_1 and Δ_2 with $\Delta_3 = 0$) is shown in Fig. 8. Similarly, P_e can be plotted as function of Δ_2, Δ_3 and Δ_1, Δ_3. It has been noted that P_e is *globally minimum* at 0.001 if values of Δ_1, Δ_2, Δ_3 are kept at 0.025, −0.09 and 0, respectively.

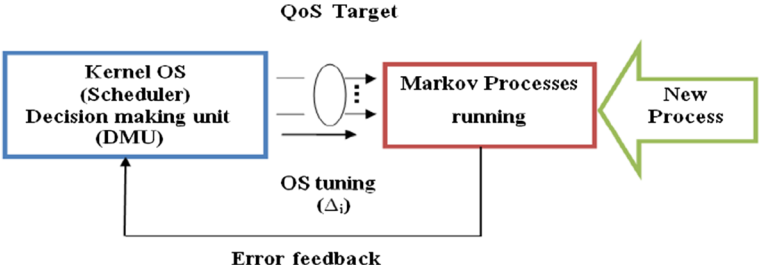

Fig. 7 Feedback control system for re-configuring the QUEST scheduler

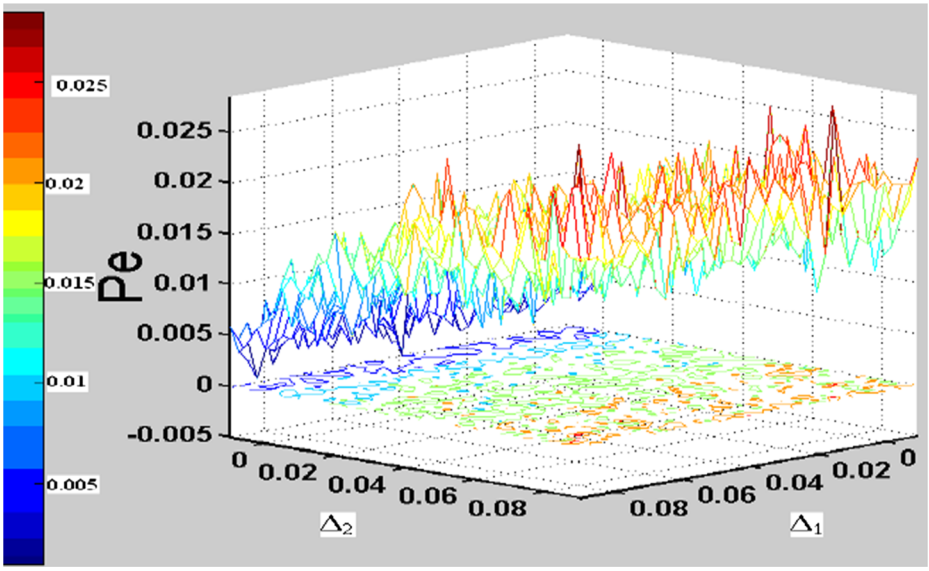

Fig. 8 Re-configuration space of P_e vs. $\Delta_1, \Delta_2, \Delta_3 = 0$

8 Run-time estimation of TPM by machine learning

Machine learning algorithms are used to learn knowledge or properties from the data for optimizing a performance criterion. Recently many state-of-the-art machine learning algorithms have been developed and applied in diversified fields. In [39], the authors have presented an automated and accurate classification method based on eigenbrains and machine learning, in order to detect Alzheimer's disease (AD) subjects and AD-related brain regions using 3D MR images. Zhang, Y. & Wang S.(2015) [38] have proposed a novel AD detection method by displacement field (DF) estimation between a normal brain and an AD brain. The DF was treated as the AD-related features, reduced by principal component analysis (PCA), and finally fed into three classifiers: support vector machine (SVM), generalized eigenvalue proximal SVM (GEPSVM), and twin SVM (TSVM). J. K. Williams [16] have applied random forest algorithm to diagnose aviation turbulence. Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. In [3], the authors have proposed a methodology for multi-label classification via multi-target regression in a streaming setting.

In [10], the authors have studied theoretical and empirical analysis of support vector machine methods for multiple instance classification. Support vector machine is a supervised machine learning algorithm which can be used for both classification or regression challenges. In [11], Elghazel et al. have studied unsupervised feature selection with ensemble learning. Ensemble Learning is a machine learning which uses more than models to make a prediction. The underlying design for this is that collective opinion of many is more likely to be accurate than that of one. A prediction is made based on combined outcomes of each of the models. The outcome can either be combined using average or the outcome occurring the most, or weighted averages. Ensemble Learning attempts to find a trade-off between variance and bias. K-means clustering is an unsupervised Machine Learning algorithm that deals with clustering of data. Using training data, the model finds the best structures and forms clusters. Wang, X. et al. [35]

have modified the MinMax *k*-means algorithm based on PSO to determine the parameters which can subject the algorithm to attain the lowest clustering errors.

Because the QUEST scheduling mechanism is *re-configurable* in nature, specific values of TPM parameters at a given time during system operation are *uncertain*. Therefore, it is essential to dynamically *estimate* the TPM parameters (elements of the matrix *'T'*) during operation. The transition probability matrix (TPM) parameters are estimated by a forward-backward *machine-learning algorithm* which learns during run-time from the observed error patterns (sequences) that serve as *training* data. Here, for a given Δ_i algorithm 3 is applied to estimate the TPM parameters. The flowchart of the algorithm is illustrated in Fig. 9.

In this algorithm, p_{ij}, e_{jk} and n are given by transition probability, probability of error and iteration index, respectively. Let, $\ddot{w}_i(t)$ and $\ddot{w}_i(t+1)$, denote the current state and the next state of the FSM respectively. The *visible* error pattern is presented by $S = [010^{20}..1000^{30}1..0^{44}000001...]$ where elements of this pattern are denoted by S_k and 1 s represent errors.

$$p_{ij} = P\left[\ddot{w}_j(t+1)|\ddot{w}_i(t)\right] \tag{12}$$

and

$$e_{jk} = P\left[S_k(t)|\ddot{w}_j(t)\right] \tag{13}$$

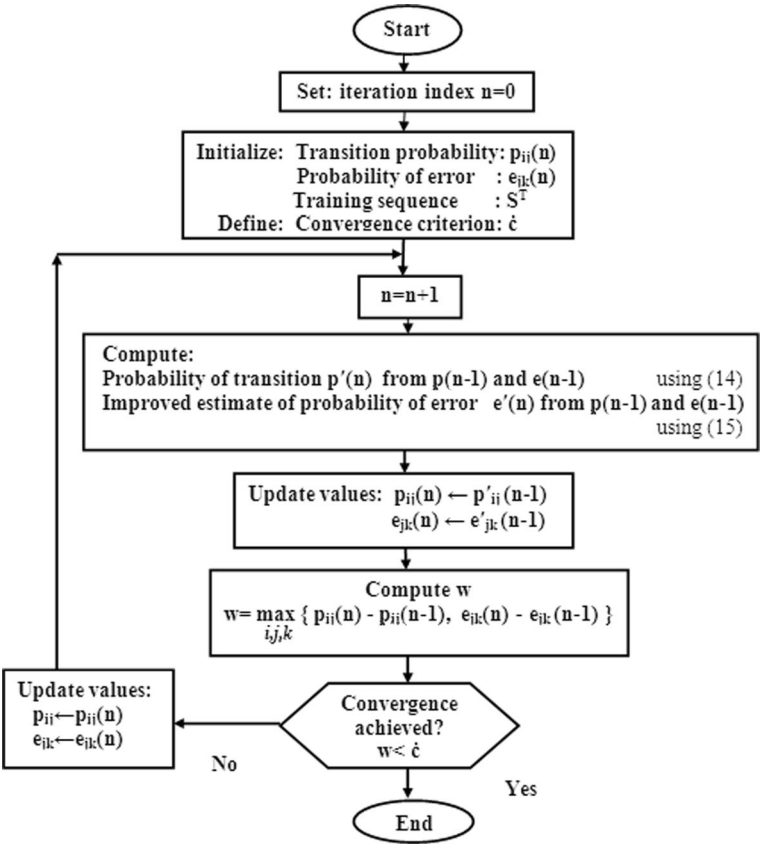

Fig. 9 Flowchart of algorithm 3: Forward-backward machine-learning

Computation has been started with an estimate of p_{ij} and e_{jk} and to calculate improved values of them until convergence criterion, ċ is achieved. In this estimation, $x_i(t)$ is the probability that the scheduler is in state $\ddot{w}_i(t)$ and has generated the error sequence up to step t. Similarly, $y_i(t)$ to be the probability that the model is in state $\ddot{w}_i(t)$ and will generate the rest of the error sequence. An improved value can be calculated by defining $z_{ij}(t)$ - the probability of transition between $\ddot{w}_i(t-1)$ and $\ddot{w}_j(t)$, given the model generated the entire training visible sequence S^T by any path. $z_{ij}(t)$ is defined as follows:

$$z_{ij}(t) = \frac{p_{ij} e_{jk} x_i(t-1) y_j(t)}{P(S^T|\dot{c})} \tag{14}$$

where $P(S|\dot{c})$ denotes the probability that the model generated sequence S^T. Let, p'_{ij} is the the estimate of the probability of a transition from $\ddot{w}_i(t-1)$ to $\ddot{w}_j(t)$. The value of p'_{ij} can be found by taking the ratio between the expected number of transitions from $\ddot{w}_i$ to $\ddot{w}_j$ and the total expected number of transitions from $\ddot{w}_i$.

$$p'_{ij}(t) = \frac{\sum_{t=1}^{T} z_{ij}(t)}{\sum_{1}^{T} \sum_{k} z_{ik}(t)} \tag{15}$$

Similarly, an improved estimation of e'_{jk} can be calculated,

$$e'_{jk}(t) = \frac{\sum_{t=1}^{T} \sum_{l} z_{jl}(t)}{\sum_{t=1}^{T} \sum_{l} z_{jl}(t)} \quad s(t) = s_k \tag{16}$$

Improved estimates for p_{ij} and e_{jk} are repeated using Eqs. (15) and (16) until the change is significantly less than convergence criterion ċ. In this estimation, ċ has been set at 0.001.

8.1 Stability and accuracy of run-time TPM estimation

As the process load varies on a demand basis within the system, the PLR changes accordingly. Therefore, the elements of 'E', the error probability matrix too changes with respect to time and iterations. After 900 iterations the system simulates the newly estimated model having modified TPM. In this *learning*, Forward-backward algorithm is guaranteed to converge to a maximum log likelihood ratio as shown in Fig. 10.

This convergence signifies stability of the system. The accuracy of the proposed scheduler is validated by comparing the run-time error patterns for initially considered TPM and for the estimated regenerated one. These patterns are illustrated in Fig. 11.

The two run-time error patterns are almost *identical*, confirming accuracy of the proposed model.

9 System performance analysis of QUEST

The run-time PLRs for individual traffic flow in QUEST are illustrated in Fig. 12.

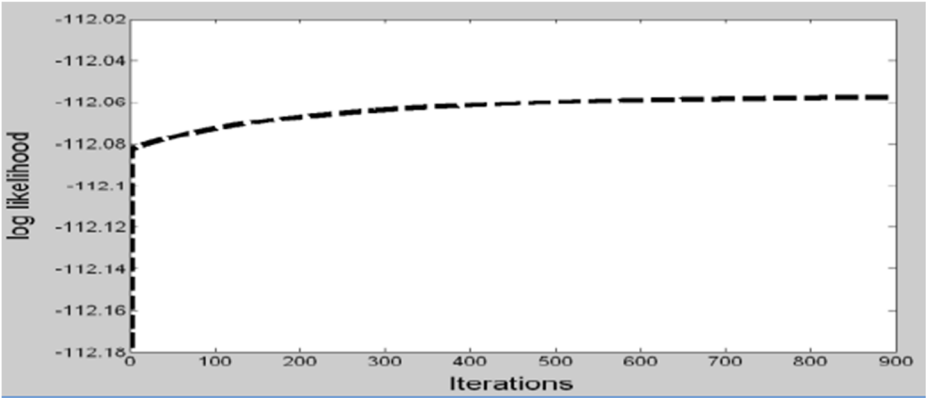

Fig. 10 Plot of log likelihood with respect to no. of Iterations

The figure illustrates that the VoIP traffic in QUEST has a minimum value of PLR with increasing normalized load compared to IPTV and HTTP. The rise rate of run-time PLR for HTTP traffic is significantly highest.

A comparative performance analysis of PLR (here, denoted as P_e) for current state-of-the-art scheduling algorithms - earliest deadline first (EDF), deferred preemption (DP), accuracy-aware EDF (A-EDF) with respect to QUEST for increasing normalized loads are illustrated in Fig. 13.

The L1, L2 cache miss errors and deadline miss errors for aforementioned scheduling algorithms with typical values of L1 = 32 KBytes and L2 = 256 KBytes at a normalized load of 0.9 are depicted in Fig. 14.

It is observed from Figs. 13 and 14 that the QUEST scheduler outperforms other scheduling schemes and offers the lowest value of PLR. The PLR is reduced by 37 % in QUEST compared to A-EDF with lower values of cache and deadline misses. For QUEST, the improvement is due to use of Hidden Markov Model (HMM) filter (Baum-welch based) which is a probabilistic model applicable for finite and discrete

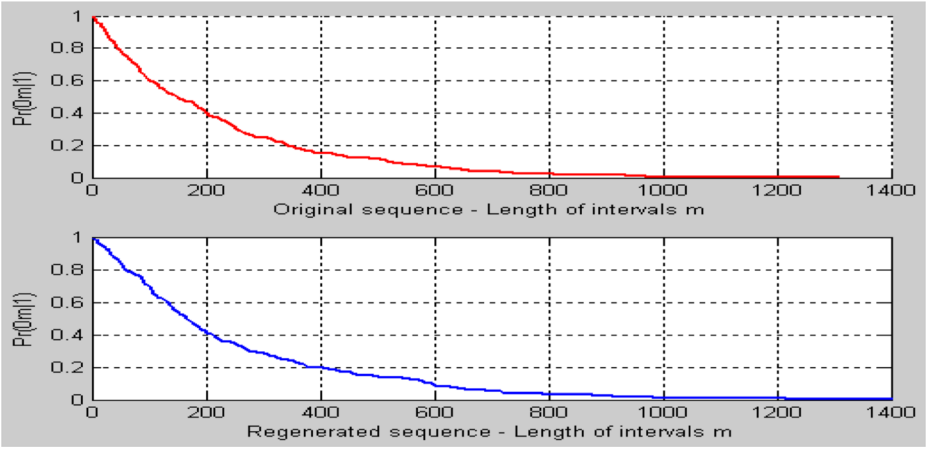

Fig. 11 Pr (0^m|1) for initial model and for newly estimated (regenerated) model

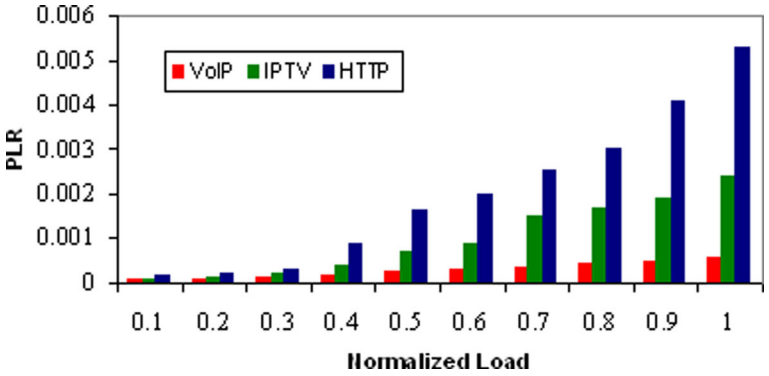

Fig. 12 PLR for each multimedia IP traffic

process states. In contrast, A-EDF uses Kalman filter for process state estimation. Kalman filter is a special case of HMM applicable only for continuous and infinite states for a linear state space model which is not valid in digital embedded systems. Further, Kalman filter assumes Gaussian noise, whereas HMM filter makes no such assumptions and is thus more *general and accurate*. Furthermore, EDF and A-EDF have no explicit control on utilization, leading to *unacceptably high* deadline miss rates at *heavy loads*. In stark contrast, QUEST enforces utilization close to 100%, making lower deadline misses even at heavy loads. This *conclusively* establishes QUEST's *superiority* over EDF and A-EDF.

10 Test-bed implementation for QUEST

The performance of the proposed QUEST scheduler was validated in NetFPGA® [25] – a renowned open platform for high-performance networking router using field-

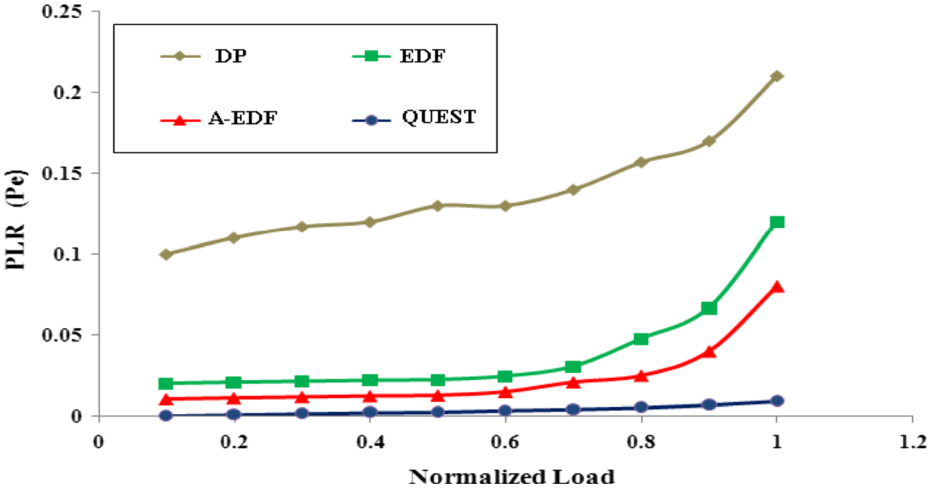

Fig. 13 PLR for DP, EDF, A-EDF, QUEST

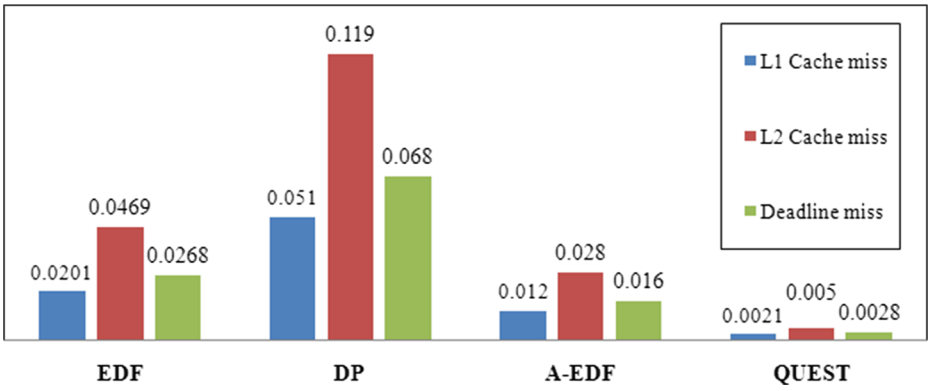

Fig. 14 Cache and deadline miss errors for DP, EDF, A-EDF, QUEST

programmable gate array (FPGA) hardware. The platform was customized for the implementation of the reconfigurable scheduler QUEST in the following experimental setup shown in Fig.15.

The QUEST was implemented in a router which was placed between an ISP gateway and a multiport switch. The NetFPGA®- router was connected with the Internet having a speed of 10 Mbps. Three classes of multimedia IP traffic, namely, VoIP (Skype), IPTV (live streaming) and HTTP (web browsing) were being scheduled and executed according to the QUEST. Three laptops were used for receiving each class of traffic and a

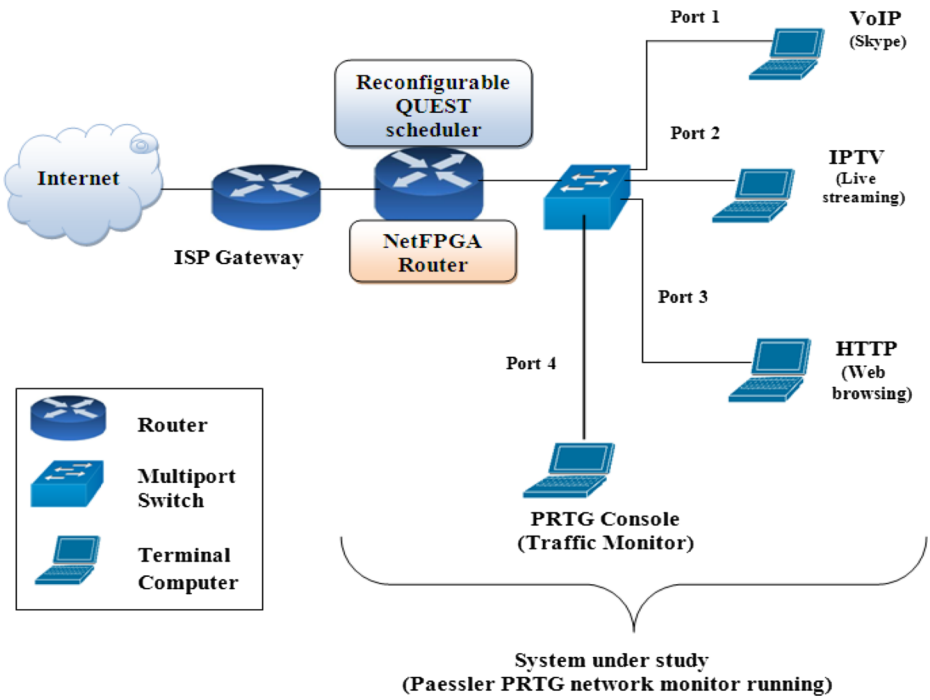

Fig. 15 Test-bed implementation of QUEST

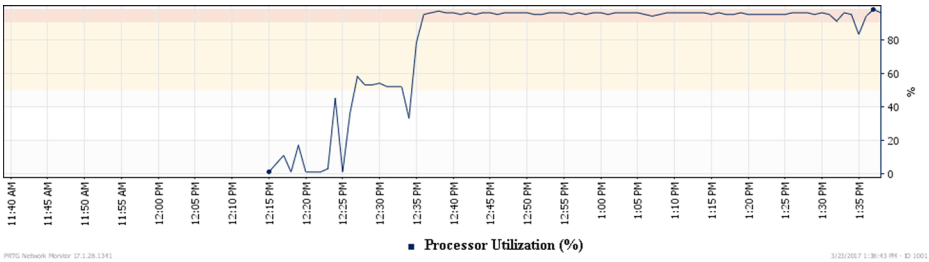

Fig. 16 Trace of processor utilization using Paessler® PRTG network monitor on 23rd March, 2017

renowned Paessler PRTG® network monitor [26] console was connected with a multiport switch to monitor the performance of the QUEST. The *trace* of the run-time processor utilization over a continuous monitor of 1 h 20 min is depicted in Fig. 16. The router was switched on at 12:15 PM. The scheduler adapts itself to reach a steady state processor utilization which is very close to 100% at 12:37 PM. A utilization very close to 100% (within the range of 91 to 97%) was maintained over a period of continuous 58 min except at 1:35 PM when the utilization falls below 90%.

The individual process utilization monitored for each class of multimedia IP traffic is depicted in Fig. 17.

The experimental results indicate that the steady state process utilization ratio in the order of 80:16:4 for VoIP, IPTV and HTTP traffic was achieved. The mean waiting time (in ms) and the run-time PLR (expressed in percentage) of QUEST for varying load over a continuous observation period of 55 min are depicted in Fig. 18.

It is clear from the figure that the maximum value of packet loss rate for QUEST is 0.49%, which is within the standard PLR threshold of 1%. The maximum value of mean waiting time is 8.2 ms which is less than the minimum deadline of 20 ms.

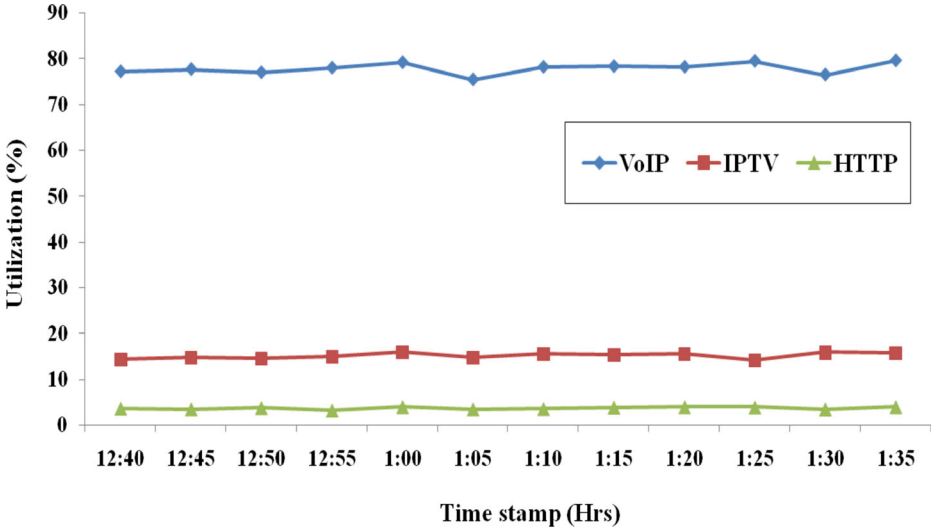

Fig. 17 Time trace of process utilization ratio for VoIP, IPTV and HTTP

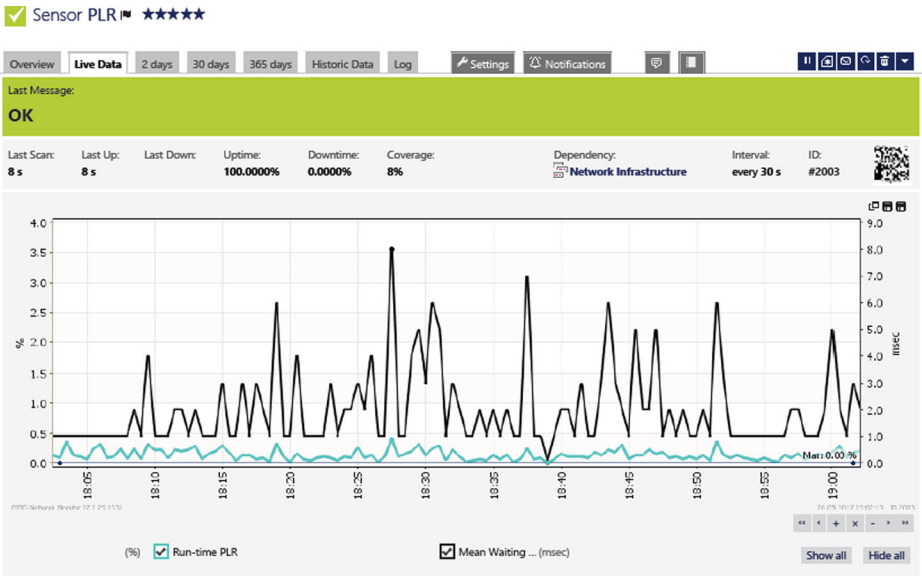

Fig. 18 Trace of PLR and mean waiting time for QUEST with varying load

Detailed experiment characterization is too *long* and is suitable for a separate, forthcoming publication.

11 Conclusion

This paper presents a novel re-configurable QoS-enhanced intelligent real-time packet scheduler - QUEST, for multimedia IP traffic in routers. Machine learning algorithms were used for the first time to our best knowledge to design a QoS-maximized optimal fair stochastic packet scheduler to dynamically optimize the system QoS during run-time. In stark contrast to the schedulers available in the literature, this scheduler was shown to maximize the system-QoS, guaranteeing utilization fixed at 100%. QUEST addresses poor performance of the premier EDF scheduler at heavy loads. Its other *unique advantages*, namely, avoiding priority starvation and arbitrary pre-programming of process utilization ratio, were validated with rigorous simulations. Performance of the scheduler was analyzed using QoS's two most important metrics, namely, packet loss rate and mean waiting time (related to system latency). Simulation results indicate that the performance of the proposed scheduler is substantially superior compared with current state-of-the-art scheduling algorithms. An improvement of 37% in PLR and an improvement of 23% in mean waiting time were obtained over the competing scheduler A-EDF. The accuracy of the QUEST was further established by comparing the run-time error patterns for initial and estimated TPM and they were found to be almost identical. A design for QUEST's implementation in NetFPGA® router has been presented. Extension to fuzzy queueing systems is underway and would be published in forthcoming papers. The dynamic optimization presented in

Section 7 can be further improved by applying stochastic computational intelligence algorithms like simulated annealing (SA), particle swarm optimization (PSO) [40], etc.

Acknowledgements The authors would like to thank Prof. A. K. Jana for his helpful suggestions.

References

1. Abhaya VG, Tari Z, Zomaya AY (2014) Performance analysis of EDF scheduling in a multi-priority preemptive M/G/1 queue. IEEE Trans Parallel Distrib Syst 25(8):2149–2158. doi:10.1109/TPDS.2013.171
2. Aboughazaleh N, Mosse´ D, Childers BR, Melhem R (2006) Collaborative operating system and compiler power management for real-time applications. ACM Trans Embed Comput Syst 5(1):82–115. doi:10.1145/1132357.1132361
3. Aljaž Osojnik A, Panov P, Džeroski S (2016) Multi-label classification via multi-target regression on data streams. Mach Learn 1–26. doi:10.1007/s10994-016-5613-5
4. Bril RJ, Lukkein JJ, Verhaegh WFJ (2007) Worst case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption revisited. Proceedings of the 19th Euromicro. Conf. Real-Time System. pp 269–279. doi:10.1109/ECRTS.2007.38
5. Chen S, Nahrstedt K (1998) An overview of quality of service routing for the next generation high-speed networks: problems and solutions. IEEE Netw 12(6):64–79. doi:10.1109/65.752646
6. Chen Y, Farely T, Ye N (2004) QoS requirements of network applications on the internet, information knowledge system managements. IOS Press 4(1):55–76
7. Chib S, Greenberg E (1995) General understanding the Metropolis-hasting algorithm. Am Stat 49(4):327–335. doi:10.1080/00031305.1995.10476177
8. Cristofaro ND, McGill G, Sallahi A, Davis M, Alsibai A, St-Hilaire M (2009) QoS evaluation of a voice over IP network with video: a case study, Proceedings of Canadian Conference on electrical and computer Engineering, St. John's, NL, pp 288–292. ISBN: 978-1-4244-3509-8. doi:10.1109/CCECE.2009.5090139
9. DEVS suite Discrete event system simulator suite, Arizona Center of Integrative Modeling and Simulation of Arizona State University Available: http://acims.asu.edu/software/devs-suite
10. Doran G, Ray S (2014) A theoretical and empirical analysis of support vector machine methods for multiple-instance classification. Mach Learn 97(1):79–102. doi:10.1007/s10994-013-5429-5
11. Elghazel H, Aussem A (2015) Unsupervised feature selection with ensemble learning. Mach Learn 98(1):157–190. doi:10.1007/s10994-013-5337-8
12. Ghaderi M, Boutaba R, Kenward GW (2005) Stochastic admission control for quality of service in wireless packet networks. Lecture notes in computer science series 3462:1309–1320. ISBN: 978-3-540-32017-3. doi:10.1007/11422778_105
13. Ghazel C, Saïdaneb L (2015) Satisfying QoS requirements in NGN networks using a dynamic adaptive queuing delay control method, Proceedings of the 10th international Conference on future networks and communications. Procedia Computer Science 56:225–232. doi:10.1016/j.procs.2015.07.203
14. Greco L, Fontanelli D, Bicchi A (2011) Design and stability analysis for anytime control via stochastic scheduling. IEEE Trans Autom Control 56(3):571–585. doi:10.1109/TAC.2010.2058497
15. Jin X, Min G (2007) Performance analysis of priority scheduling mechanisms under heterogeneous network traffic. J Comput Syst Sci 73:1207–1220. doi:10.1016/j.jcss.2007.02.008
16. John K, Williams JK (2014) Using random forests to diagnose aviation turbulence. Mach Learn 95(1):51–70. doi:10.1007/s10994-013-5346-7
17. Johnston J, Farrington S, Saville R, Szigeti T (2010) Medianet Reference Guide, Cisco, pp12–13
18. Kang K-D, Son SH, Stankovic JA (2004) Managing deadline miss ratio and sensor data freshness in real-time databases. IEEE Trans Knowl Data Eng 16(10):1200–1216. doi:10.1109/TKDE.2004.61

19. Khan MA, Ansari AQ (2012) Handbook of research on industrial informatics and manufacturing intelligence: innovations and solutions, IGI Global, EISBN: 9781466602953, pp 395–397. doi:10.4018/978-1-4666-0294-6
20. Kleinrock L (1975) Queueing systems. Theory. Wiley, Hoboke, vol 1, pp 37–51
21. Kooti H, Mishra D, Bozorgzadeh E (2011) Reconfiguration-aware real-time scheduling under QoS constraints. Proceedings of the 16th Asia and South Pacific Design Automation Conference (ASP-DAC), Yokohama, pp 141–146. doi:10.1109/ASPDAC.2011.5722174
22. Liu CL, Layland JW (1973) Scheduling algorithms for multiprogramming in a hard real- time environment. J ACM 20(1):46–61. doi:10.1145/321738.321743
23. Lyngsø RB, Pedersen CNS (2001) Complexity of comparing hidden markov models. Proceedings of the 12th Int. Symp. on Algorithms and Computation, New Zealand, Springer Berlin Heidelberg, ISBN: 978-3-540-45678-0, pp 416–428. doi:10.1007/3-540-45678-3_36
24. Nasri M, Kargahi M, Mohaqeqi M (2012) Scheduling of accuracy-constrained real-time systems in dynamic environments. IEEE Embed Syst Lett 4(3):61–64. doi:10.1109/LES.2012.2195294
25. NetFPGA® hardware platform. Available: http://netfpga.org/site/#/systems/4netfpga-1g/details/
26. Paessler PRTG® network monitor. Available: https://www.paessler.com/homepage
27. Rikli N-E, Almogari S (2013) Efficient priority schemes for the provision of end-to-end quality of service for multimedia traffic over MPLS VPN networks. Journal of King Saud University –Computer and Information Sciences 25(1):89–98. doi:10.1016/j.jksuci.2012.08.001
28. Saeed Ullah S, Thar K, Hong CS (2016) Management of scalable video streaming in information centric networking. Multimedia Tools Applications 1–28. doi:10.1007/s11042-016-4008-8
29. Saleh M, Dong L, (2010) Comparing FCFS & EDF scheduling algorithms for real-time packet switching networks, Proceedings of International Conference on Networking, Sensing and Control (ICNSC), Chicago, pp 698–703. doi:10.1109/ICNSC.2010.5461572
30. Seth K, Anantaraman A, Mueller F, Rotenberg E (2006) FAST: frequency-aware static timing analysis. ACM Trans Embed Comput Syst 5(1):200–224. doi:10.1145/1132357.1132364
31. Szigeti T, Hattingh C (2005) End-to-end QoS network design: quality of service in LANs, WANs, and VPNs. Cisco Press, Indianapolis, pp 110–112
32. Thiébaut D, Wolf JL, Stone HS (1992) Synthetic traces for trace-driven simulation of cache memories. IEEE Trans Comput 41(4):388–410. doi:10.1109/12.135552
33. Toral-Cruz H, Pathan A-SK, Pacheco JCR (2013) Accurate modeling of VoIP traffic QoS parameters in current and future networks with multifractal and Markov models. Math Comput Model 57(11–12):832–2845. doi:10.1016/j.mcm.2011.12.007
34. Wang G (2010) ML estimation of transition probabilities in jump Markov systems via convex optimization. IEEE Trans Aerosp Electron Syst 46(3):1492–1502. doi:10.1109/TAES.2010.5545204
35. Wang X, Bai Y (2016) A modified MinMax -means algorithm based on PSO. Comput Intell Neurosci 2016: 1–13, Article ID 4606384. doi:10.1155/2016/4606384
36. Wang J, Hou YB (2016) Packet loss rate mapped to the quality of experience. Multimedia Tools Applications 1–36. doi:10.1007/s11042-016-4254-9
37. Wang X, Khemaissia I, Khalgui M, Li ZW (2015) Dynamic low-power reconfiguration of real-time systems with periodic and probabilistic tasks. IEEE Trans Autom Sci Eng 12(1):258–271. doi:10.1109/TASE.2014.2309479
38. Zhang Y, Wang S (2015) Detection of Alzheimer's disease by displacement field and machine learning. Peer J 1–29. doi:10.7717/peerj.1251
39. Zhang Y, Dong Z, Phillips P, Wang S, Ji G, Yang J, Yuan TF (2015) Detection of subjects and brain regions related to Alzheimer's disease using 3D MRI scans based on eigenbrain and machine learning. Front Comput Neurosci 9(66):1–15. doi:10.3389/fncom.2015.00066
40. Zhang Y, Wang S, Genlin J (2015) A comprehensive survey on particle swarm optimization algorithm and its applications. Math Probl Eng, Hindawi Publishing Corporation 2015(2015):1–38, Article ID 931256. doi:10.1155/2015/931256
41. Zhoua X, Jianbin W, Xu C-Z (2007) Quality-of-service differentiation on the internet: a taxonomy. J Netw Comput Appl 30:354–383. doi:10.1016/j.jnca.2005.07.001

Suman Paul is currently an Assistant Professor, in the Dept.of Electronics and Communication Engineering, Haldia Institute of Technology, Haldia, West Bengal University of Technology (Maulana Abul Kalam Azad University of Technology West Bengal), India. He received his bachelor's and master's degree in electronics engg and computer science in 2005 and 2008, respectively. He is working in the field of scheduling in embedded systems, network QoS and multimedia IP traffic. He worked as associate researcher in the Indian Institute of Management (IIM), Calcutta in 2007.

Malay Kumar Pandit is currently a Professor, Dept.of Electronics and Communication Engineering, Haldia Institute of Technology, West Bengal University of Technology (MAKAUT, West Bengal), India. He received his B.E and M. E degrees in Electronics Engineering from Electronics and Telecom Engg Dept, Jadavpur University, India in 1989 and 1991, respectively. He received his PhD from UK's renowned Cambridge University in 1996. He did his post-doc from the Optoelectronics Research Centre, City University of Hong Kong till 2002 where he pioneered the use of polymers for optical waveguide applications. He then took a corporate career where he worked in a fibre optic company "FONS (I) Ltd" in the domain of optical networking. His current research interests are in the field of scheduling, network QoS.