PAPER

Fair Scheduling for Delay-Sensitive VoIP Traffic

Shawish AHMED[†a)], **Xiaohong JIANG**[†,††b)], ***Nonmembers***, ***and*** **Susumu HORIGUCHI**[†c)], ***Member***

SUMMARY With the wide expansion of voice services over the IP networks (VoIP), the volume of this delay sensitive traffic is steadily growing. The current packet schedulers for IP networks meet the delay constraint of VoIP traffic by simply assigning its packets the highest priority. This technique is acceptable as long as the amount of VoIP traffic is relatively very small compared to other non-voice traffic. With the notable expansion of VoIP applications, however, the current packet schedulers will significantly sacrifice the fairness deserved by the non-voice traffic. In this paper, we extend the conventional Deficit Round-Robin (DRR) scheduler by including a packet classifier, a Token Bucket and a resource reservation scheme and propose an integrated packet scheduler architecture for the growing VoIP traffic. We demonstrate through both theoretical analysis and extensive simulation that the new architecture makes it possible for us to significantly improve the fairness to non-voice traffic while still meeting the tight delay requirement of VoIP applications.

key words: *packet scheduling, VoIP, Deficit Round-Robin, fairness, Token Bucket, IP networks*

1. Introduction

Nowadays, there are about 1 billion fixed telephone lines and 2 billion cellphones in the world that are served by the traditional PSTN systems. It is expected, however, that in the near future most of these voice services will be provided by IP networks known as the voice over Internet Protocols (VoIP) [1]. This migration is due not only to the significant cost saving but also to the benefits of integrating voice and data into a single infrastructure.

In the VoIP applications, voice packets are transmitted through IP networks, incurring transmission, queueing and propagation delay at each hop along their paths. Since both delay and delay variation (jitter) may significantly affect the quality of voice services [1]–[3], therefore the end-to-end delay and jitter control need to be carefully addressed in the VoIP applications. In general, the end-to-end delay can be regarded as the sum of a constant delay component and a random one. The constant delay component is mainly determined by the signal transmission and propagation time along physical links, so it is in general uncontrollable. The random delay component, on the other hand, is the queueing delay encountered by a voice packet in each IP router along its path. Since the queueing delay of a packet is mainly controlled by the packets scheduling scheme adopted inside the IP routers, the packet scheduling schemes should be deliberately designed to support the VoIP applications.

The current IP networks are developed mainly for supporting non-real time best effort applications, like file transfer and email, which are not delay or delay jitter sensitive. Thus, the available packet scheduling schemes (e.g. Deficit Round Robin [2] and Weighted Fair Queuing [3]) are designed to provide a fair bandwidth sharing among network traffic without a deliberate consideration about their delay performance. It is notable that current VoIP traffic, although they are delay and delay jitter sensitive, constitutes only a very small fraction of overall traffic in the current IP networks. Therefore, the current packet schedulers guarantee the tight delay and delay jitter requirements of VoIP packets by simply assigning them the highest priority [4], [5]. This simple priority policy is acceptable as long as the amount of voice traffic is relatively very small in comparison with other non-voice traffic. With the notable expansion of VoIP applications and thus a rapid growth of VoIP traffic, however, the above simple priority policy for VoIP will significantly sacrifice the fairness deserved by the non-voice traffic while adopting the fair schedulers alone, as an alternative solution, will severely degrade the quality of VoIP applications. Therefore, new schedulers should be developed for future IP networks to efficiently handle the impacts that will arise with the expected growth of VoIP traffic and the delay-sensitive traffic in general.

In this paper, we shall introduce a new packet scheduling architecture that has the capabilities to both meet the tight delay requirement of VoIP applications and avoid the aggressive resource unfairness to other non-voice traffic. We will show that by combining the conventional Deficit Round-Robin (DRR) scheduler with a packet classifier, a Token Bucket and a resource reservation scheme in an integrated architecture, we are able to support the VoIP traffic with nearly the same delay performance achieved by the current pure Strict-Priority scheduler while allowing a high degree of bandwidth sharing to other non-voice traffic, so a graceful trade-off can be initialized between priority and fairness in a VoIP-capable IP network. The performance of the proposed new scheduling architecture is demonstrated by both mathematical analysis and experimental simulation, where the detailed analysis about its fairness, packet delay, delay jitter and the voice flow queue length are provided.

Manuscript received August 7, 2008.
Manuscript revised February 27, 2009.

[†]The authors are with the Graduate School of Information Sciences, Tohoku University, Sendai-shi, 980-8579 Japan.
[††]The author is a visiting researcher of the State Key Laboratory for Novel Software Technology, Nanjing University, China.
a) E-mail: ahmedmg@ecei.tohoku.ac.jp
b) E-mail: jiang@ecei.tohoku.ac.jp
c) E-mail: susumu@ecei.tohoku.ac.jp

DOI: 10.1587/transcom.E92.B.3115

The remainder of the paper is organized as follows: Sect. 2 reviews the related work. Section 3 introduces the new scheduler. Section 4 provides the mathematical analysis of the proposed scheduler and Sect. 5 includes the experimental simulation results. Finally, Sect. 6 concludes this paper.

2. Related Work

Numerous scheduling algorithms (schedulers) have been proposed to guarantee either the minimum queueing delay or a fair resource sharing for the ongoing traffic in the IP network, see for example [2], [3], [6]–[8], [10], [11]. Currently, voice traffic constitutes only a very small portion of total traffic in IP networks, so the simple Strict Priority (SP) scheduling scheme is usually adopted as a low-cost solution to support the delay sensitive voice traffic [4], [5].

In the SP scheme, the high priority queue is served first until it is empty, and then the packets in the low priority queue are served. This simple priority policy can easily fulfill the QoS requirement of the delay-sensitive traffic. Unfortunately, the main disadvantage of SP scheme is that the low priority queues may suffer from bandwidth starvation if the higher priority queues saturate the link bandwidth [4], so it may introduce a significant unfairness problem to other non delay-sensitive traffic, in particular with the notable expansion of VoIP applications as expected in the future IP networks. Other priority-based scheme, like the low latency queueing scheduler (LLQ) [18], currently implemented by Cisco in their routers, is also promising to guarantee the delay of real-time services. The LLQ is actually a combination of SP scheme and Class-Based Weighted-Fair Queueing (CBWFQ) [19], and it is currently the recommended queueing function for VoIP applications, and it will also work well with video conferences. However, the LLQ also share the same main drawbacks of priority-based schemes, i.e., they can guarantee this promising performance only when the volume of delay-sensitive traffic is very small otherwise a significant bandwidth starvation (unfairness) problem will occur to other low priority traffic.

The fairness has been the main target in case of scheduling non delay-sensitive traffic (known as the best effort traffic) and many schedulers have been proposed for the fairness purpose with different degree of complexity. The Fair Queueing (FQ) scheme [3] can achieve almost the perfect fairness with a time complexity $O(logN)$, where N is the number of packet streams that are concurrently active at the gateway or router. The Deficit Round-Robin (DRR) scheduler [2] can provide almost the same fairness like FQ but with only a constant complexity $O(1)$. In DRR scheme, we assign a given quantum to each queue depending on its rate and also use a deficit counter to recode the deficit of this queue from previous rounds, and we serve each nonempty queue in a round robin way based both on the quantum assigned to the queue and also the deficit available to this queue. Many variations of DRR have also been proposed to further improve its fairness without sacrificing its attractive constant complexity, such as the nested deficit round-robin scheduler (NDRR) [20] and the dynamic deficit round-robin scheduler (DDRR) [21]. As a matter of fact, the DRR scheduler and its variations all work based on the round robin mechanism, which mainly focuses on the fair resource sharing among all flows regardless whether they are delay sensitive or best effort traffic flows. Therefore, they were unable to efficiently fulfill the tight delay constraint of the delay sensitive VoIP applications, especially with the expected increases of its traffic volume.

It is worth noticing that another class of weight-based schedulers has also been developed to provide fairness according to the weight of the traffic class, like the weighted fair queueing scheduler (WFQ) [3], the worst-case fair weighted fair queueing scheduler (WF2Q) [23] and the delay optimized worst case fair WFQ scheduler (DO-WF2Q) [24]. The main idea of this type of schedulers is to assign a weight to each flow depending on its traffic volume and then achieve the fairness by emulating the General Processor Sharing (GPS) discipline. Although the weight-based schedulers can achieve a relatively better fairness than the DRR, it suffers not only from a high complexity of $O(N)$, which is significantly higher than the constant time complexity provided by DRR, but also can not meet the tight delay requirements of the VoIP applications.

With the expected expansion of the voice traffic, neither the available SP-Based schemes nor the current fair schedulers will be able to efficiently support both voice and non-voice traffic simultaneously in a way that can easily meet the tight delay requirments of the VoIP applications and also provide fair resource sharing to other low priority traffic. Based on the above observations, we propose in next section a new scheduler for the growing VoIP traffic to achieve a graceful trade-off between delay and fairness in future VoIP-capable networks.

3. A New Scheduler For VoIP

The main idea of our new scheduler is to extend a fair scheduler (the DRR scheduler) by combining it with a resource reservation scheme, a traffic shaper (Token Bucket) and a packet classifier into an integrated architecture. The main objective of the proposed scheduler is to guarantee fairness to the non-voice traffic without significantly sacrificing the delay performance of delay sensitive voice traffic.

3.1 Scheduler Architecture

The basic architecture of our new scheduler is illustrated in Fig. 1, where the three main modules (i.e., packet classifier, Token Bucket and DRR) are shown.

Packet Classifier: The packet classifier module in our scheduler is used to classify the incoming packets into voice and non-voice classes. The packet classification function can be easily implemented based on the idea of packet header inspection [9].

Token Bucket: It works as a traffic shaper, where input traf-

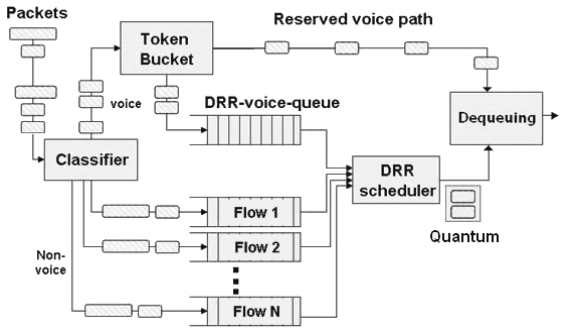

Fig. 1 The proposed packet scheduler architecture.

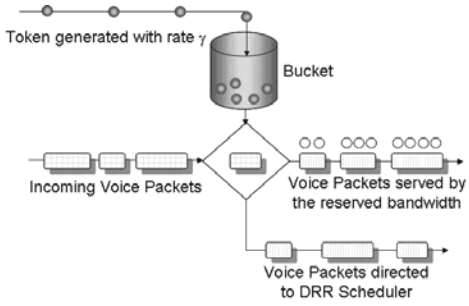

Fig. 2 The operation of Token Bucket.

fic is divided into two output traffic. The first one is a traffic with an upper bound rate that never exceeds the token bucket rate R_{token} regardless of it incoming input rate. The second one is the overflow traffic which will be originated due to this limitation. In our scheduler, the main task of this module is to split the incoming voice flows into two parts, as illustrated in Fig. 2. The first one is a smooth voice traffic composed of a group of voice flows with aggregated rate less than or equal to the R_{token}, and which will be served through a reserved bandwidth equal to this rate. We will refer to this traffic as the reserved voice (RV) traffic. On the other hand, the second voice traffic is composed of the voice flows overflowed from the Token Bucket due to the upper bound rate limitation and they will be re-directed to the DRR module. We will refer to this traffic as the DRR voice (DV) traffic. Within the DRR module, all the voice flows that belong to the DV traffic will compete fairly with other non-voice traffic on the remaining unreserved bandwidth (i.e., the remaining amount of output bandwidth which is not reserved by RV group.)

DRR Scheduler. The DRR module in our scheduler is used to guarantee a max-min fair resource utilization of the remaining unreserved bandwidth among the non-voice traffic and the DV traffic. Actually, the fair resource sharing in our scheduler can also be achieved by adopting other fair schedulers. We choose the DRR scheduler due to its simplicity and its attractive constant time complexity, such that a good scalability can be guaranteed in our proposed scheduler.

3.2 Scheduler Control

Upon the arrival of packets, they will be firstly classified into non-voice class and voice class. Those packets that are classified as non-voice will be directly served through the DRR module. On the other hand, the packets that belong to voice class will be forwarded to the Token Bucket module for further classification, partial of them will be served through the reserved bandwidth with a rate limit R_{token} while the remaining packets will be re-directed to the DRR module.

The main task of Token Bucket module is to allow only to a specific amount of voice traffic with an upper bound rate denoted by R_{token} to be served through the reserved bandwidth, and to redirect the rest overflow of voice packets to the DRR module. The Token Bucket check the voice packet size and compares it with the tokens accumulated by now (i.e., the arrival time of the voice packet) in the bucket, where token is a unit of byte generated with constant rate (i.e., R_{token}) and accumulated in a buffer called Bucket. If this packet size is less than the tokens accumulation, then an amount of tokens that equals to this packet size will be deleted from the bucket and this packet will be served directly through the reserved bandwidth; otherwise, this packet will be re-directed to the DRR module.

The voice flows that will be directed to the DRR module will be buffered in a dedicated queue called the DRR-voice-queue, in which it will be treated a single competitor flow. In DRR module, each queue i is assigned with a quantum Q_i and is associated with a deficit counter DC_i. The quantum Q_i represents the worth of bits that queue i can send in each round, while the deficit counter DC_i is used to recode the deficit of this queue from its previous rounds. The DRR module can handle variable packet sizes in a fair manner by serving all non-empty queues in turn (round-robin mechanism). Once the queue i got its turn, it begins to send out packets subject to the constraint that the summation of their size is less than or equal to Q_i. If there are no more packets in the queue i after the queue has been serviced, then the queue state variable DC_i is reset to zero. Otherwise, the deficit amount of the Q_i is stored in the state variable DC_i. In subsequent round, the amount of usable bandwidth will be the sum of DC_i (from previous rounds) and Q_i. It is notable that our new scheduler architecture is flexible, since it covers pure DRR and Strict-Priority scheduler as two special cases.

As the classification process can be simply accomplished through a packet's header check, and the splitting process depends only of the packets rate regardless to their flows, so it worth to note here that the proposed scheduler does not require any precondition or any auxiliary modules (e.g., flow admission control, or call admission control) for usage.

3.3 Comparison with Relevant Schedulers

The current packet schedulers like Class-Based Weighted-

Fair Queueing (CBWFQ), low latency Queueing scheduler (LLQ) and Strict Priority (SP)scheduler guarantee the tight delay and delay jitter requirements of VoIP packets by simply assigning them the highest priority where the voice packets are served immediately upon their arrival without imposing any limitation on their maximum rate. This policy is acceptable as long as the voice traffic represents a very small portion of the ongoing traffic across the IP networks. With the notable expansion of VoIP applications as expected in the future IP networks, however, this priority policy will introduce a significant unfairness problem to other low priority traffic since it will saturate the link bandwidth while adopting the fair schedulers alone like Deficit Round-Robin (DRR) or Weighted Fair Queueing (WFQ), as an alternative solution, will severely degrade the quality of VoIP applications. Such degradation will occur due to the round robin serving mechanism adopted in such fair schedulers, which will severely increase the queueing latency of the voice packets waiting for their serving round.

In our scheduler, we adopt a new splitting mechanism to the VoIP traffic and bandwidth, in which only a specified portion of the voice traffic will be served by an equivalent amount of reserved bandwidth. In this way, we can control not only the amount of resources occupied by the voice traffic but also initiate a fair competition between the remaining portion of the voice traffic and the other non-voice traffic. Actually, the voice traffic will endure an additional amount of delay in our scheduler due to the fair competition with other traffic. However, we showed in the following sections that by properly controlling such splitting mechanism, a nice trade-off between voice packet delay and fairness deserved by other traffics can be achieved. Moreover, a more practical comparisons between our proposed scheduler performance and the conventional were conducted in Sect. 5. The obtained results notably report the outstanding flexible performance of the proposed scheduler, which can not be achieved by using the probabilistic Priority scheme or even the max-min fair schedulers.

4. Analytical Analysis

In this section, we conduct a detailed analytical analysis of the proposed scheduler. We develop analytical models to analyze the fairness, packet delay/delay jitter and buffer size requirement of the DRR-voice-queue inside the DRR module. The notations employed in our analysis are summarized in Table 1.

4.1 Fairness Analysis

In our analysis, we use the same concept of the throughput Fairness Measure (FM) due to Golestani [12], which measures the worst case difference between the normalized services received by different flows that are backlogged during any time interval. Since we want to measure the throughput fairness among the classes and not the flows, we extend the previous definition to cover the worst case difference between the normalized services received by the voice traffic class and the non-voice traffic class that are backlogged during any time interval.

Definition 1: A class of traffic is backlogged during an interval I if this class is never empty of packets during the interval I.

Clearly, it makes no sense to make this comparison while one of these classes is not backlogged, because the former does not receive any service when it is not backlogged. We assume that each class i has a weight w_i assigned according to its traffic rate. Let $Sent_i(t_1, t_2)$ be the total number of bits sent on the output line by this class interval (t_1, t_2), then the normalized service received by a class i is just $[Sent_i(t_1, t_2)/w_i\ ]$.

In our scheduler, the non-voice packets coming in on different flows are stored in different queues and directly served by the DRR module. For the flow i with rate R_{iDRR}, a weight W_{iDRR} is assigned to it according to the following rule:

$$W_{iDRR} = \frac{R_{iDRR}}{R_{min}} \tag{1}$$

where R_{min} is a small rate designated for weight calculation. The assigned quantum Q_i of flow i is calculated depending on its weight as follow:

$$Q_i = W_{iDRR} \cdot Q_{min} \tag{2}$$

If the minimum quantum Q_{min} is chosen not less than the

Table 1 List of important notations in this paper.

R	Transmission rate of output link (bit/sec).
R_{res}	Reserved transmission rate (bit/sec).
R_{DRR}	Transmission rate handled by the DRR scheduler (bit/sec).
R_v	Voice traffic rate (bit/sec).
R_b	Best effort traffic rate (bit/sec).
R_{vDRR}	offered transmission rate of DRR-voice-queue (bit/sec).
β	Reserved portion of voice traffic (i.e. the splitting ratio) .
R_{token}	Token Bucket rate (bit/sec).
Q_{min}	Minimum quantum value in the DRR scheduler (bit)
Max	Maximum packet size.
$Sent_v(t_1, t_2)$	Amount of voice bits sent in time interval(t_1, t_2).
$Sent_b(t_1, t_2)$	Amount of best-effort bits sent in time interval(t_1, t_2).
N	Number of active queues dedicated to the non-voice traffic in the DRR scheduler .
F	Summation of the active queues' quantum in the DRR scheduler.
Q_b	Summation of all the best effort flows quantum served by the the DRR scheduler (bit)
Q_v	quantum of the DRR-voice-queue.
w_{vDRR}	Weight of DRR-voice-queue.
w_{iDRR}	Weight of flow i inside the DRR scheduler.
w_v	Weight of voice traffic in general.
w_b	Weight of best effort (non-voice)traffic in general.
m	The number of service round received by voice traffic through the DRR module during the interval (t_1, t_2).

maximum packet length Max in the network, then the algorithm time complexity will be $O(1)$. Although the DRR module schedules N of the non-voice flows, each with different rate, weight and quantum, but in our fairness measurement, we refer to them as one aggregated flow (one class) with one rate equal to the summation of their individual rates. We will refer to this class as the Best-effort one with rate R_b, weight w_b and quantum Q_b calculated as follow:

$$R_b = \sum_{i=1}^{N} R_{iDRR} \tag{3}$$

$$w_b = \frac{R_b}{R_{min}} = \sum_{i=1}^{N} W_{iDRR} \tag{4}$$

$$Q_b = \sum_{i=1}^{N} Q_i \tag{5}$$

On the other hand, the voice traffic, with rate R_v, has a weight w_v equal to

$$w_v = \frac{R_v}{R_{min}} \tag{6}$$

This traffic is divided into two parts (please refer to Fig. 2). The first part has an upper bound rate R_{token} that is also equal to $(\beta \cdot R_v)$, where $0 < \beta < 1$. We will refer to β in the rest of this paper as the *splitting ratio*, and it presents the portion of the voice traffic which is served by the reserved bandwidth. The second part of the voice traffic has a rate R_{vDRR}, which can be expressed as $[(1-\beta) \cdot R_v]$ and which is served through the DRR module with weight w_{vDRR} and quantum Q_v assigned to the DRR-voice-queue as follow:

$$w_{vDRR} = \frac{R_{vDRR}}{R_{min}} \tag{7}$$

$$Q_v = w_{vDRR} \cdot Q_{min} \tag{8}$$

We can now express the Fairness Measure of the proposed scheduler as the maximum difference between $[Sent_v(t_1, t_2) / w_v]$ and $[Sent_b(t_1, t_2)/w_b]$ in the backlogged time interval (t_1, t_2). This difference should not depend on the size of the time interval (t_1, t_2). Moreover, If the FM is small, this indicates that the service discipline is closely emulates an ideal fair scheduler.

Lemma 1: Consider any execution of the proposed scheduling scheme and any interval (t_1, t_2) of any execution such that voice class is backlogged during (t_1, t_2)). Let m be the number of service round received by voice traffic through the DRR module during the interval (t_1, t_2), Then

$$R_{token} \cdot (t_2 - t_1) + m \cdot Q_v - Max \leq Sent_v(t_1, t_2) \leq R_{token} \cdot (t_2 - t_1) + m \cdot Q_v + Max$$

Proof 1: Since the voice traffic is divided in two parts, the rate of the first part will never excesses the token bucket rate R_{token}. Therefore, the upper bound of service received by this part in time interval (t_1, t_2) is equal to $[R_{token} \cdot (t_2 - t_1)]$. On the other hand, the second part of the voice traffic will be re-directed to the DRR module in which it will be treated as an ordinary flow. It has been proven in [2] that, in the DRR scheduler, the amount of bits sent by any backlogged flow i in time interval (t_1, t_2) is $[m \cdot Q_i \pm Max]$. Therefore, the lemma follows by summing the two parts of the voice traffic. □

The following theorem proves the fairness of the proposed scheduler, and that this fairness is controllable through the splitting ratio.

Theorem 1: For an backlogged interval (t_1, t_2) in execution of the proposed scheduler service discipline

$$\left| \frac{Sent_v(t_1, t_2)}{w_v} - \frac{Sent_b(t_1, t_2)}{w_b} \right| \leq (Q_{min} \cdot (1 + \beta)) + \frac{Max}{w_v} + \frac{N \cdot Max}{w_b}$$

Proof 2: by decomposing the voice class from lemma 1 we get the following equation:

$$\begin{aligned} Sent_v(t_1, t_2) &\leq R_{token} \cdot (t_2 - t_1) + m \cdot Q_v + Max \\ &= \beta \cdot R_v \cdot (t_2 - t_1) + m \cdot Q_v + Max \end{aligned} \tag{9}$$

Thus, we can calculate the normalized service received by the voice class as follow:

$$\frac{Sent_v(t_1, t_2)}{w_v} \leq \frac{\beta \cdot R_v \cdot (t_2 - t_1)}{w_v} + \frac{m \cdot Q_v}{w_v} + \frac{Max}{w_v} \tag{10}$$

From Eqs. (6), (7) and (8)

$$\frac{Sent_v(t_1, t_2)}{w_v} \leq \beta \cdot R_{min} \cdot (t_2 - t_1) + m \cdot (1 - \beta) \cdot Q_{min} + \frac{Max}{w_v} \tag{11}$$

Using the DRR algorithm invariant which state that the difference in the number of round-robin opportunities given to flow i and flow j in the time interval (t_1, t_2) is $|m - m'| \leq 1$, in addition to the DRR scheduler lemma, we can easily show that the non-voice class can receive the following amount of service

$$Sent_b(t_1, t_2) \geq m' \cdot Q_b - N \cdot Max \tag{12}$$

Where N is the number of active queues dedicated to serve the non-voice flows in the DRR module. Then the normalized service received by the non-voice class can be expressed as.

$$\frac{Sent_b(t_1, t_2)}{w_b} \geq \frac{m' \cdot Q_b}{w_b} - \frac{N \cdot Max}{w_b} \tag{13}$$

From Eqs. (4) and (5)

$$\frac{Sent_b(t_1, t_2)}{w_b} \geq m' \cdot Q_{min} - \frac{N \cdot Max}{w_b} \tag{14}$$

By combining Eq. (11) and (14) we get

$$\left| \frac{Sent_v(t_1, t_2)}{w_v} - \frac{Sent_b(t_1, t_2)}{w_b} \right|$$

$$\leq \beta[R_{min} \cdot (t_2 - t_1) - m \cdot Q_{min}] + Q_{min} + \frac{Max}{w_v} + \frac{N \cdot Max}{w_b} \quad (15)$$

The term of $[R_{min} \cdot (t_2 - t_1) - m \cdot Q_{min}]$ contains two different operands; the first one $[R_{min} \cdot (t_2 - t_1)]$ is continuous, in term of time $(t_2 - t_1)$, and the second operand $(m - Q_{min})$ is a discrete, in term of round m. The relation between these two operands is tight because m represents the number of round-robin rounds that can be accomplished in the time interval (t_1, t_2). In order to simplify this relation, we expressed m in term of time as follow:

$$m = \left\lfloor \frac{(t_2 - t_1)}{(Q_v + Q_b)/R_{DRR}} \right\rfloor \quad (16)$$

Where R_{DRR} is the bandwidth handled by the DRR module. Then $(m \cdot Q_{min})$ can be now expressed as follow:

$$m \cdot Q_{min} = \left\lfloor \frac{(t_2 - t_1) \cdot R_{DRR}}{(Q_v + Q_b)} \right\rfloor \cdot Q_{min} \quad (17)$$

From Equation (5), (6), (7) and (8)

$$m \cdot Q_{min} = \left\lfloor \frac{(t_2 - t_1) \cdot R_{min} \cdot R_{DRR}}{(R_{vDRR} + R_b)} \right\rfloor \quad (18)$$

In the worst case, the R_{vDRR} and the R_b will saturate (equal to) bandwidth R_DRR handled by the DRR module, and then, we can approximate $(m \cdot Q_{min})$ as follow:

$$m \cdot Q_{min} \geq \lfloor (t_2 - t_1) \cdot R_{min} \rfloor \quad (19)$$

this simplification means that with the increase of time $(t_2 - t_1)$ the number of round m equivalently increase too and that $m \cdot Q_{min}$ will always be the floor of $[R_{min} \cdot (t_2 - t_1)]$. Therefore, the subtracting of $[R_{min} \cdot (t_2 - t_1) - m \cdot Q_{min}]$ will maximally result into one complete round which is equal to Q_{min}. Thereby, the theorem follows

$$\left| \frac{Sent_v(t_1, t_2)}{w_v} - \frac{Sent_b(t_1, t_2)}{w_b} \right| \leq (Q_{min} \cdot (1 + \beta)) + \frac{Max}{w_v} + \frac{N \cdot Max}{w_b}$$

□

The result of this analysis shows that the proposed scheduler can keep this fairness index bounded and independent of both voice traffic amount and backlog time length. Moreover, this result shows that the increases or decreases of β can partly degrades or enhances the fairness respectively, as long as the $(\beta < 1)$; but if $(\beta = 1)$, the fairness will diverge to infinity because of the disappearance of $[m \cdot (1 - \beta) \cdot Q_{min}]$ from Eq. (15) which means that the whole voice traffic rate will be served by the reserved bandwidth.

4.2 The Voice Packet Delay and Delay Jitter Upper Bound

As the voice packet can be served through the reserved bandwidth with probability β or through the DRR scheduler with probability $(1 - \beta)$, so the average voice packet delay D_{voice} can be simply as follows:

$$\begin{aligned} D_{voice} &= D_{vRES} \times p(D_{vRES}) + D_{vDRR} \times p(D_{vDRR}) \\ &= D_{vRES} \times \beta + D_{vDRR} \times (1 - \beta) \end{aligned} \quad (20)$$

Where, D_{vRES} and D_{vDRR} represent the average delay that a voice may encounter if it is served through the reserved bandwidth or the DRR module, respectively. With respect to those voice packets which are served by the reserved bandwidth, their average packet delay D_{vRES} is in term of μ sec and hence we neglect it. For the rest of the voice packets, they will suffer from delay due to the fair competition with other non-voice packets in the DRR module and the above equation will be simplified in terms of $[D_{vDRR} \times (1 - \beta)]$. In general, the DRR scheduler belongs to the class of LR servers. S. Kanhere and H. Sethu report in [17] a tight upper bound of the packet latency θ_i of any flow i in the DRR scheduler, then later, Anton Kos et al correct this upper bound in [13]. In our analysis we used the former upper bound especially with the lack of accurate delay distribution for this type of scheduler. The following equation shows the delay upper bound θ_i for any flow i.

$$\theta_i \leq Q_i \left(\frac{1}{R_i} - \frac{1}{R} \right) + (Max - 1) \left(\frac{Q_i}{F \cdot R_i} N + \frac{1}{R_i} - \frac{2}{R} \right) \quad (21)$$

We utilize such upper bound in our analysis instead of D_{vDRR}, and hence we adjust the definition of D_{voice} from "the average voice packet latency" to "the maximum average voice packet latency" and also we change the equality sign in Equation 20 into a less than in the following new form of such equation:

$$D_{voice} \leq (1 - \beta) \left[Q_v \left(\frac{1}{R_{vDRR}} - \frac{1}{R_{DRR}} \right) + (Max - 1) \left(\frac{Q_v}{F \cdot R_{vDRR}} N + \frac{1}{R_{vDRR}} - \frac{2}{R_{DRR}} \right) \right] \quad (22)$$

On the other hand, the delay jitter may occur between two packets of the same voice flow if one of them is served directly by the reserved bandwidth (i.e., it will not suffer from delay) and the other is served by the DRR module (i.e., it will suffer from delay due to fair competition with non-voice traffic). The upper bound of such delay jitter D_{jitter} is equal to maximum packet delay that any packet may encounter in the DRR module as derived in The following equation

$$D_{jitter} \leq \left[Q_v \left(\frac{1}{R_{vDRR}} - \frac{1}{R_{DRR}} \right) + (Max - 1) \left(\frac{Q_v}{F R_{vDRR}} N + \frac{1}{R_{vDRR}} - \frac{2}{R_{DRR}} \right) \right] \quad (23)$$

In general the packet delay of any flow i in the DRR module is directly related to its relative weight w_i and its

assigned quantum Q_i as a result (i.e., the weight of the observed flow with respect to the weight of the other competitor flows); So, a flow with a relative small weight usually suffer from a high packet delay. In our case, the increase of splitting ratio results in a decrease of the amount of voice traffic R_{vDRR} served by the DRR module and hence its relative weight w_{vDRR} and quantum Q_v too. Although the splitting ratio does not appear in the above equation, it actually controll both of R_{vDRR} and Q_v. This is why, the delay-jitter will increase with the increase of the splitting ratio. Finally, We have to notice that only $[(1-\beta)100]$ percent of the voice traffic that may suffer from this delay jitter, and which can be recovered through the jitter buffer at the receiver node.

Given a predetermined upper bound of both packet delay and delay jitter, the network designer will be able to calculate the optimum value of β by solving the above two equations (i.e., Eqs. (22) and (23)).

4.3 The DRR-Voice Queue Buffer Size

Generally, the buffer size should be large enough to allow the voice application to endure the normal variance in the scheduler latency without suffering from packet loss. In the proposed scheduler, those voice packets, which are redirected to the DRR module and buffered in the DRR-voice-queue, will wait for their turn of service according to the DRR mechanism; therefore, we estimate the proper buffer size of the DRR-voice-queue. Due to the availability of the upper delay bound only, we have estimate the maximum buffer size B_{vDRR} of the DRR-voice-queue depending on the maximum delay jitter D_{jitter} as of follow:

$$B_{vDRR} = D_{jitter}(1-\beta)R_v \tag{24}$$

This equation was derived in agreement with the well known thumbing rule [14], which states that the maximum buffer size of any queue is equal to the product of both the maximum delay and the service rate of this queue. This equation is verified through simulation experiments in the next section.

5. Simulation Results

In this section, we wish to answer the following questions about the performance of the proposed scheduler:

- Is the fairness independent of the backlogged time interval's length? Is the fairness sensitive to the splitting ratio change? Does the workload variation of both voice and non-voice traffic affect the fairness?
- How does the splitting ratio variation affect the delay and delay-jitter performance of the voice traffic? What is the delay performance of the voice packets served through the DRR module especially with the increase of their workloads?
- does the buffer size model for the DRR-voice-queue reflect the simulation case or not?
- What is end-to-end VoIP QoS performance of the proposed scheduler in practical situation?

5.1 Default Simulation Setting

Unless otherwise specified, the default for all the later experiments is as specified here. We simulate the behavior of our scheduler in a single node with OC1 output link capacity (51.89 Mbps). This node receives voice and non-voice flows generated according to a Poisson distribution. The voice flows were generated according to the G.729A codec specifications with 8 kbps for every single voice stream with 20 ms of voice payload. On the other hand, 60% of the non-voice packets sizes were settled as 44 bytes, 20% as 550 bytes and the rest 20% as 1500 bytes. Every point in the following fairness or delay graphs was derived in simulation interval, typically 2000 sec.

5.2 The Fairness Analysis

We designed a set of experiments to examine the fairness provided by the proposed scheduler to both voice and non-voice traffic in general. The following experiments are conducted under different backlogged time interval's lengths, various splitting ratio and several workloads. Due to the relation between these parameters, we have devised experiments to isolate each of them, and then we tested every parameter individually.

In the first experiment, we examine the difference between the normalize service received by both backlogged voice and non-voice traffic during different backlogged time interval's length. We generate both voice and non-voice traffic with workload equal to 0.4 for each of them, and fix the splitting ratio to 0.5. The normalized throughput of both voice and non-voice traffic has been acquired at different backlogged time lengths typically equal to 10, 100 and 1000 sec. The simulation results, as illustrated through the pie chart in Fig. 3, show that the resulting normalized throughput percentage of both voice and non-voice traffic classes is almost equal even with the increase of backlogged time lengths. This result confirms that the fairness provided by the proposed scheduler is independent of the backlogged time length.

In the second experiment, we addressed the impact of the splitting ratio on the difference between the normalize service received by both backlogged voice and non-voice traffic. We fixed the workload of both voice and non-voice

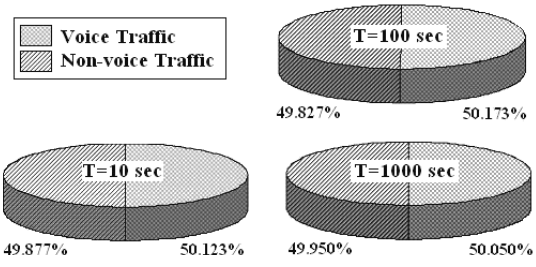

Fig. 3 The normalized throughput percentage of the voice traffic with respect to non-voice traffic in different backlogged time intervals.

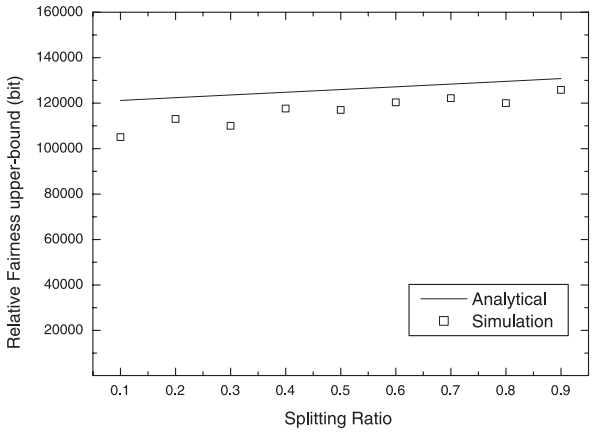

Fig. 4 The normalized throughput of the voice traffic with respect to non-voice traffic at different splitting ratio.

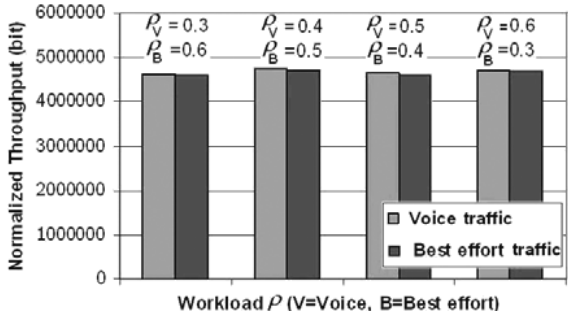

Fig. 5 The normalized throughput of the voice traffic with respect to non-voice traffic at different workloads.

traffic to 0.4 for each of them. The normalized throughput of both traffic has been inspected at different splitting ratio. The result, as shown in Fig. 4, shows an enhancement in the fairness with the decreases of the splitting ratio. This enhancement is due to the decrease of reserved bandwidth offered to the voice traffic, and hence the increase of fair competition between the overflowed voice traffic and the non-voice traffic in the DRR module. We can also notice that the simulation results match well with the upper bound derived in the previous section. Therefore, we considered the result of this experiment a verification of our theorem.

In the last experiment of this set, we examined the effect of the workload variation of both voice and non-voice traffic on the fairness. We fix the splitting ratio at 0.5, and the simulation time period to its default value, then we acquired the normalized throughput of both voice and non-voice traffic under different combination of their workloads. The results, as shown in Fig. 5, record the absence of any effect on the difference between the normalized throughputs of these two classes even with the variation of their traffic's workloads. Through these simulation results, we show that the fairness provided by the proposed scheduler is independent of both backlogged time period and workload variation. On the other hand, we show that the splitting ratio is the only metric which affects on the fairness index.

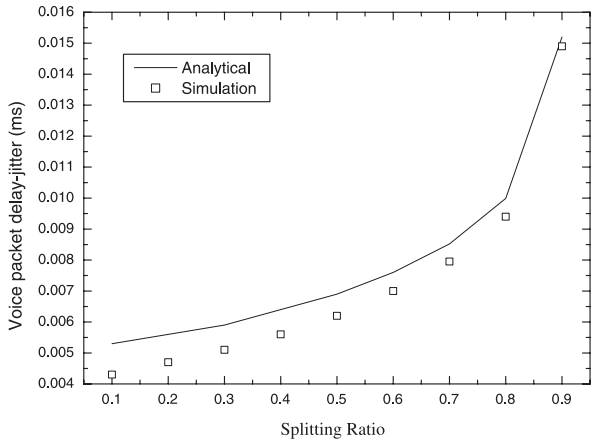

Fig. 6 The delay-jitter upper bound at different splitting ratios.

5.3 The Voice Packet Delay and Delay-Jitter Analysis

In the following experiments, we inspect the delay and delay-jitter performance of the voice traffic in the proposed scheduler under various splitting ratios and compare it with other scheduling schemes, including Deficit Round-Robin (DRR), Weighted Fair Queueing (WFQ), Surplus Round-Robin (SRR), and Strict Priority (SP) schemes.

In our scheduler, two voice packets that belong to the same flow may be separated due to the splitting mechanism adopted in the proposed scheduler. In the first experiment, we inspect the delay-jitter between these two packets. We fix the workload of the both voice and non-voice traffic to 0.4 for each of them. We then record the delay-jitter variation with the the increase of splitting ratio from 0.1 up to 0.9. As shown in Fig. 6, the results indicate that delay-jitter increase with the increase splitting ratio. In general the packet delay of any flow in the DRR module is directly related to its relative weight and hence on its assigned quantum (i.e., the weight of the observed flow with respect to the weight of the other competitor flows); So, a flow with a relative small weight usually suffer from a high packet delay. In our case, the increase of splitting ratio results in a decrease of the amount of voice traffic served by the DRR module and hence its relative weight and quantum too. This is why, the delay-jitter will increase with the with the increase of the splitting ratio. It is also notable that the simulation results of the delay-jitter performance match well with the upper bound derived in the previous section. Therefore, we considered the result of this experiment a verification of our theorem.

In the second experiment, we target to compare the voice packet delay performance of the proposed scheduler in comparison with other scheduling schemes, including Deficit Round-Robin (DRR), Weighted Fair Queueing (WFQ), Surplus Round-Robin (SRR) and Strict Priority (SP) schemes. We fix the workload of the both voice and non-voice traffic to 0.4 for each of them. We then record the delay variation with the the increase of splitting ratio from

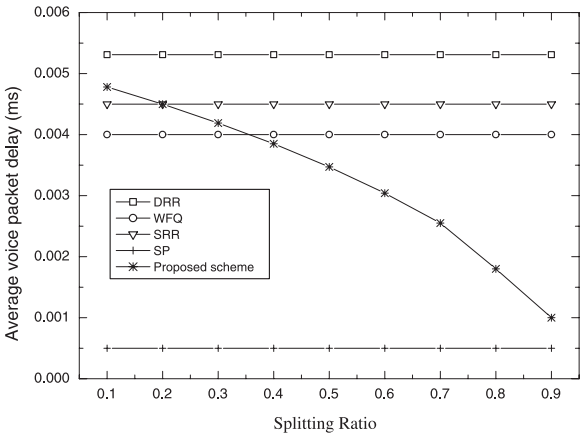

Fig. 7 Voice packet delay of the proposed scheduler in comparison with relevant schedulers at different splitting ratios.

0.1 up to 0.9. The splitting mechanism (for both voice traffic and bandwidth) adopted in our scheduler offers us the ability to impose a limit on the priority offered to the voice traffic by controlling the amount of traffic served through the reserved bandwidth, while enhancing the fairness deserved by the non-voice traffic by sacrifice a portion of voice packets delay through the DRR module. As shown in Fig. 7, the results of this experiment show that the voice packets delay performance of the proposed scheduler lies in between that of the DRR and the SP schemes. Such behavior indicates the ability of controlling the performance of our scheduler through the splitting ratio to emulate any of the later schemes according to the network policy. We can also notice the performance of the other scheduler, like the WFQ which can report better packet delay performance in compare with the DRR but they still also report a high complexity $O(\log n)$.

5.4 The Buffer Size of the DRR-Voice-Queue

In the proposed scheduler, part of the voice traffic is served through the DRR module, in which the voice packet will be temporarily buffered in a queue called the DRR-voice-queue until it get its chance to be served according to the DRR packet scheduling mechanism. We conducted an experiment to verify our analytical maximum length model in Eq. (23). In our experiment, we fixed the non-voice workload to 0.4 and then inspected the maximum buffer size of the DRR-voice-queue when the voice traffic workload grows from 0.1 up to 0.4. The experiment results and the corresponding modeling results are summarized in Fig. 8, which shows clearly that our analytical buffer size model is very efficient and it can be used to dimension the buffer size of DRR-voice-queue without causing any voice packet dropping problem.

5.5 The Assessment of VoIP QoS

Due to the existence of two paths for the voice traffic in the proposed scheduler, the amount of voice packets served by

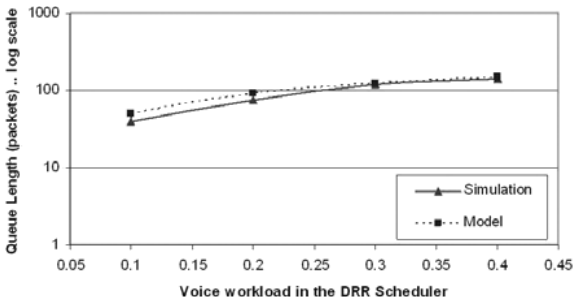

Fig. 8 A comparison between the simulation and analytical max-length of the DRR-voice-queue.

the DRR module exert an additional delay (delay-jitter) because of the competition with the best-effort traffic. Nevertheless, the impact of such delay jitter can be easily eliminated by adopting a small size jitter-buffer at the receiver node. We now report the results of a new simulation experiment. This experiment is designed to investigate the delay-jitter impact on the end-to-end QoS of the VoIP calls delivered by our scheduler in practical situation. The QoS measurements have been conducted using the famous R-score metric [27].

Simulation Settings: Figure 9 shows the network topology used in the experiment. All the links have a bandwidth of 2 Mbps and propagation delay of 1 ms. We generate 100 VoIP flows (each with a rate of 8 Kbps) from S_0 to D_0. In addition to the 100 observed VoIP flows, the background traffic in the system is as follows. There are 10 best-effort flows with rate 80 Kbps from each of the following nodes $(S_1, S_2, \cdots, S_5)$ to $(D_1, D_2, \cdots, D_5)$, respectively. The number of end-to-end hopes is chosen based of similar conditions in [25] and [26]. The VoIP and best-effort flows were generated according the default simulation settings

VoIP QoS Measure: The R-score takes into account one way delay, loss rate, and the type of the encoder. For the G.729A encoder [27], the R-score is as follow

$$R = 94.2 - 0.024d - 0.11(d - 177.3)H(d - 177.3) - 11 - 40\log(1 + 10e) \quad (25)$$

where

- $d = 25 + d_{jitter_buffer} + d_{network}$ is the total one-way delay in ms comprising of 25 ms voice encoder delay, delay in the delay-jitter buffer (30 ms, 40 ms, 50 ms, and 60 ms), packet delay.
- $e = e_{network} + (1 - e_{network})e_{jitter}$ is the total loss rate including network and jitter losses
- $H(x) = 1$ if $x \geq 0$, else 0. R-score should be larger than 70 for acceptable call quality.

The quality of VoIP call is sensitive to delay, delay jitter and loss. In order to maintain a good call quality ($R \geq 70$), the one way delay should be less than 200 ms and the packet loss rate along the path should be less than 5%. With the increase of delay and packet loss, the VoIP quality is deteriorated.

Simulation Results: Figure 10 shows the average R-score for VoIP flows. The R-score of the 40 ms jitter-buffer

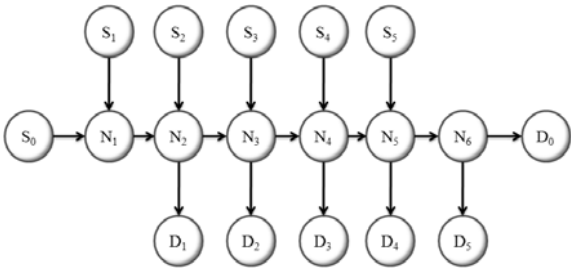

Fig. 9 Simulated network topology.

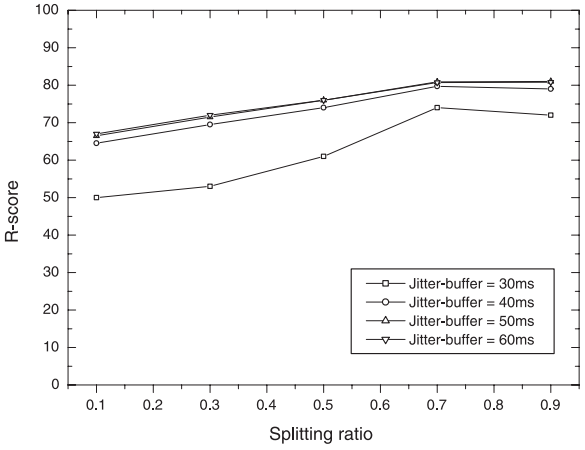

Fig. 10 End-to-end VoIP QoS inspection at different splitting ratios.

reports an acceptable QoS of the delivered VoIP calls. Such performance was significantly better in comparison with the 30 ms jitter-buffer, but almost similar to the 50 ms and 60 ms jitter-buffer cases. Such results indicates that the jitter-buffer can efficiently conceal the impact of delay-jitter even with a small size. It also worth to note that the performance of the proposed scheduler can be easily controlled through the splitting ratio β, and hence the trade-off between delay and fairness can be easily achieved through the proposed scheduler.

6. Conclusions

In this paper, we proposed a scheduling architecture for VoIP application. The new architecture is flexible in the sense that we can easily comprise between the packet delay and fairness through only one control parameter (i.e., the splitting ratio of voice traffic). Based on our new architecture, we proved through both the theoretical analysis and experimental simulation that it is possible for us to offer a certain degree of fairness to non-voice traffic without significantly sacrificing the performance of delay-sensitive voice traffic. We expect that our proposed architecture can efficiently handle the impacts that will will occur more frequently with the expected growth of VoIP traffic in the future voice-intensive IP networks.

References

[1] P.V. Mockapetris, "Telephony's next act," IEEE Spectr., vol.43, no.4, pp.28–32, April 2006.
[2] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round-robin," IEEE/ACM Trans. Netw., vol.4, no.3, pp.375–385, June 1996.
[3] A. Demers, S. Keshav, and S. Shenker, "Design and analysis of a fair queuing algorithm," ACM SIGCOMM, pp.1–12, Sept. 1989.
[4] M.J. Karam and F.A. Tobagi, "Analysis of the delay and jitter of voice traffic over the Interne," INFOCOM, vol.2, pp.824–833, April 2001.
[5] A.P. Markopoulou, F.A. Tobagi, and M.J. Karam, "Assessing the quality of voice communications over internet backbones," IEEE/ACM Trans. Netw., vol.11, no.5, pp.747–760, Oct. 2003.
[6] S. Kanhere, H. Sethu, and A. Parekh, "Fair and efficient packet scheduling using elastic round robin," IEEE Trans. Parallel Distrib. Syst., vol.13, no.3, pp.324–336, March 2002.
[7] A. Striegel and G. Manimaran, "Dynamic class-based queue management for scalable media servers," Real-Time Technology and Applications Symposium, vol.2, pp.228–236, June 2000.
[8] A. Kortebi, S. Oueslati, and J. Roberts, "AImplicit service differentiation using deficit round robi," ITC19, Beijing, Aug. 2005.
[9] S. Oueslati and J. Roberts, "A new direction for quality of service: Flow-aware networking," Proc. NGI, Rome, June 2005.
[10] L. Lenzini, E. Mingozzi, and G. Stea, "Aliquem: A novel DRR implementation to achieve better latency and fairness at O(1) complexity," IWQoS'02, 2002.
[11] S. Ramabhadran and J. Pasquale, "Stratified Round Robin: A low complexity packet scheduler with bandwidth fairness and bounded delay,"ACM SIGCOMMM, pp.239–249, 2003.
[12] S. Golestani, "A self clocked fair queueing scheme for broadband applications," INFOCOM, 1994.
[13] A. Kos and J. Miletic, "Enhanced Latency Analysis of DRR Algorithm," http://alas.matf.bg.ac.yu/mv00023/papers.htm
[14] S. Gorinsky, A. Kantawala, and J. Turner, "Link Buffer Sizing: A new look at the old problem," Proc. ISCC, June 2005.
[15] G. Appenzeller, I. Keslassy, and N. Mckeown,"Sizing router Buffer," SIGCOMM, 2004.
[16] N. Saberi and M.J. Coates, "Bandwidth reservation in optical WDM/TDM star networks," Proc. 22nd Bienn. Symp. on Comm, pp.219–221, 2004.
[17] S.S. Kanhere and H. Sethu, "On the latency bound of deficit round robin," Proc. Int. Conf. on Computer Communications and Networks, 2002.
[18] "The low latency queueing," http://www.cisco.com/warp/public/121/latencyqueueing.html
[19] R. El Abdouni and Khayari, "Class-based weighted fair queueing: Validation and comparison by trace-driven simulation," Int. J. Commun. Syst., vol.18, pp.975–994, Dec. 2005.
[20] S.S. Kanhere and H. Sethu, "Fair, efficient and low-latency packet scheduling using nested deficit round robin," Proc. IEEE Workshop on High-Performance Switching and Routing, May 2001.
[21] K. Yamakoshi, K. Nakai, E. Oki, and N. Yamanaka, "Dynamic deficit round-robin scheduling scheme for variable-length packets," Electron. Lett., vol.38, no.3, pp.148–149, Jan. 2002.
[22] X. Yuan and Z. Duan, "FRR: A proportional and worst-case fair round robin scheduler," INFOCOM, vol.2, pp.831–842, March 2005.
[23] M.J. Karam and F.A. Tobagi, "WF2Q: Worst-case fair weighted fair queueing," INFOCOM, March 1996.
[24] X. Fei and A. Marshall,"Delay optimized worst case fair WFQ (WF2Q) packet scheduling," ICC, 2002.
[25] S. Henning, F. James, and T. Don, "Congestion control for real-time traffic in high-speed networks," INFOCOM, pp.543–550, 1990.
[26] A. Bhargava, J. Kurose, D. Towsley, and L. Van, "Performance com-

parison of error control schemes in high speed computer communication networks," INFOCOM, pp.694–703, 1988.
[27] R. Cle and J. Rosenbluth, "Voice over IP performance monotoring," ACM Computer Communication Review, vol.31, pp.9–24, 2001.

Shawish Ahmed received the B.S. and M.S. degrees from Ain Shams University, Cairo, Egypt in 1997 and 2002, all in Computer Sciences. Since 2005, he is on a fellowship/mission provided by the Egyptian government to get the Ph.D. degree from the Graduate school of Information Science, Tohoku University, Japan. His current research focuses on supporting the Voice over Internet-Protocol (VoIP) applications over wired and wireless networks. He is a student member of the IEEE.

Xiaohong Jiang received his B.S., M.S. and Ph.D. degrees in 1989, 1992, and 1999 respectively, all from Xidian University, Xi'an, China. He is currently an Associate Professor in the Department of Computer Science, Graduate School of Information Science, TOHOKU University, Japan. Before joining TOHOKU University, Dr. Jiang was an assistant professor in the Graduate School of Information Science, Japan Advanced Institute of Science and Technology (JAIST), from Oct. 2001 to Jan. 2005. Dr. Jiang was a JSPS (Japan Society for the Promotion of Science) postdoctoral research fellow at JAIST from Oct. 1999–Oct. 2001. He was a research associate in the Department of Electronics and Electrical Engineering, the University of Edinburgh from Mar. 1999–Oct. 1999. Dr. Jiang's research interests include optical switching networks, routers, network coding, WDM networks, VoIP, interconnection networks, IC yield modeling, timing analysis of digital circuits, clock distribution and fault-tolerant technologies for VLSI/WSI. He has published over 130 referred technical papers in these areas. He is a member of IEEE.

Susumu Horiguchi (M'81–SM'95) received the B.Eng. the M.Eng. and Ph.D. degrees from Tohoku University in 1976, 1978 and 1981 respectively. He is currently a Full Professor in the Graduate School of Information Sciences, Tohoku University. He was a visiting scientist at the IBM Thomas J. Watson Research Center from 1986 to 1987. He was also a professor in the Graduate School of Information Science, JAIST (Japan Advanced Institute of Science and Technology). He has been involved in organizing international workshops, symposia and conferences sponsored by the IEEE, IEICE, IASTED and IPS. He has published over 150 papers technical papers on optical networks, interconnection networks, parallel algorithms, high performance computer architectures and VLSI/WSI architectures. Prof. Horiguchi is members of IPS and IASTED.