TR-IIS-05-006

Enhanced Bulk Scheduling for Supporting End-to-End Delay Requirements

Yung-Cheng Tu, Meng Chang Chen, Yeali S. Sun and Wei-Kuan Shih

July 2005 || Technical Report No. TR-IIS-05-006

http://www.iis.sinica.edu.tw/LIB/TechReport/tr2005/tr05.html

Enhanced Bulk Scheduling for Supporting End-to-End Delay Requirements

Yung-Cheng Tu[1], Meng Chang Chen[2], Yeali S. Sun[3] and Wei-Kuan Shih[1]

[1]Dept. of Computer Science
National Tsing Hua University
Hsinchu, Taiwan, R.O.C.

[2]Institute of Information Science
Academia Sinica
Taipei, Taiwan, R.O.C.

[3]Dept. of Information Management
National Taiwan University
Taipei, Taiwan, R.O.C.

Abstract

Providing end-to-end delay guarantees for delay sensitive applications is an important packet scheduling issue of routers. In this paper, to support end-to-end delay requirements, we propose a novel network scheduling scheme, called the Bulk Scheduling Scheme (BSS), built on top of existing schedulers of intermediate nodes without modifying transmission protocols on both sender and receiver sides. By inserting TED packets into packet flows at the ingress router periodically, the BSS schedulers of the intermediate nodes can dynamically allocate the necessary bandwidth to each flow to enforce the end-to-end delay, according to the information in TED packets. The introduction of TED packets incurs a lower overhead than the per-packet marking approaches. Three flow bandwidth estimation methods are presented and their performance properties are analyzed. BSS also provides a dropping policy to discard late packets and a feedback mechanism to discover and resolve the bottlenecks. The simulation results show that BSS performs efficiently as expected.

Keywords-*packet scheduling; bulk scheduling; end-to-end delay; QoS*

1. Introduction

Many real-time applications, such as IP telephony, video conferencing, WebTV, and other streaming services over the Internet, rely on networks to support QoS guarantees, typically in the form of bounded end-to-end delays. A common practice in supporting end-to-end delay requirements is to reserve a fixed bandwidth along the path of the packet flow. The bandwidth reservation method is suitable for constant-bit-rate (CBR) traffic, because the nodal delay bound in each node can be accurately estimated and consequently the end-to-end delay can be obtained. With careful arrangement, high network efficiency can also be achieved. However, for most applications, such as video, audio and transaction processing, the traffic is often variable-bit-rate (VBR) or can be described as an on-off model. Methods that combine the bandwidth reservation with buffer management have been proposed for VBR multimedia traffics [1][2][3][4]. These methods require users to give traffic specifications in advance, among which the most notable one is the T-spec in RSVP[5] and ATM[6]; however, it is difficult for users to provide the traffic specifications. Even if the specification is given, the bandwidth allocated to the flow is constant or piecewise constant during the course of the session, which results in either excessive or insufficient bandwidth allocation for the session. Another approach called effective bandwidth or equivalent capacity [7] uses a statistical method to find "just enough" bandwidth to provide a statistical performance guarantee. It also has difficulty in capturing the actual traffic generation process of applications to derive a proper effective bandwidth.

Instead of fixed bandwidth allocation, an alternative is to allocate bandwidth dynamically and/or reschedule packets in real-time during the course of transmission, based on the actual offered load, QoS requirements of the session and status of transmission. Per-packet deadline scheduling, proposed in [8], assumes that a flow has a QoS profile stored in each intermediate node along the flow path. The arriving packet of the flow is assigned a nodal delay bound for each intermediate node and the nodal scheduler schedules the next packet for transmission according to the nodal delay bounds of packets. Compared to rate-based scheduling, this approach can achieve better performance for VBR traffic, because each packet is scheduled according to its own delay bound. However, the QoS profile for each flow must be given to each intermediate node in advance and generally fixed during the course of the session. In addition, this method favors flows with QoS guarantees and may cause starvation of the best-effort flows.

There are other studies related to dynamic bandwidth allocation. In ATM, special cells called Resource Management (RM) cells are used in Available Bit Rate (ABR) service to probe bandwidth in the network [9]. Meanwhile in [10], an ATM block transfer (ABT) scheme is proposed in which the traffic flow is divided into blocks and each block requests bandwidth individually. Another similar approach is proposed in [11], whereby the sender divides the data stream into bursts during the scheduling phase and the first packet of each burst specifies its traffic rate and burst size. At an intermediate node, the nodal scheduler schedules packets based on the requirements of bursts. In this scheme, the sender needs to be able to communicate the application-level semantics to the lower layer protocol entities so they can segment the byte stream into bursts, as the packets of a burst are logically related and intended to be scheduled as a unit, such as a frame in a video stream. Furthermore, this approach requires a major modification to the applications, protocols and transmission systems.

Although the above works tackle the dynamic bandwidth requirement of VBR traffic, they do not address the nodal and end-to-end delay requirements. Some researches on the guarantee of delay bound have focused on the single-node case. Such approaches derive a worst-case delay bounds at a single node and the end-to-end delay bound is obtained by adding up nodal delays [12][13]. These approaches do not consider the advantage of dynamic scheduling at intermediate nodes to expedite packets with longer delays in previous nodes, or to loosen bandwidth allocation to the packets ahead of their schedules.

To overcome the above problems, we propose an end-to-end bulk scheduling scheme to support end-to-end delay requirements for delay sensitive applications. The idea of the proposed scheme is to insert special control packets, called TED (Traffic Specification with End-to-end Deadline) packets, into flow periodically. The TED packets carry information about the residual delay bounds for the schedulers in the intermediate nodes to dynamically allocate the necessary bandwidth to the flow in order to meet the end-to-end deadline. The insertion (deletion) of TED packets is performed at the ingress (egress) routers by the lower layer protocol entities independent of existing systems and applications.

With the bulk scheduling scheme, each nodal scheduler at an intermediate node on the flow path schedules packets per bulk according to the associated TED packet and only the contents of TED packets are modified by the intermediate nodes. Compared to per-packet marking approaches (such as PHBs[14]), our scheme has lower computational overhead for intermediate nodes. The bulk scheduling scheme does not change any information in the data packets, which allows end-to-end secure transmissions (such as IPsec[15]) possible. Furthermore, the proposed scheduling algorithms consider the delay conditions that TED packets have experienced, rather than just the fixed priority or deadline which was assigned when packets were dispatched by the sender. The TED-based scheduler is implemented on top of the existing scheduler of the intermediate nodes so that the original treatment to non real-time flows can be preserved. There are enhancements to BSS, including drop missed policy and feedback mechanism, proposed to further improve the performance.

The remainder of this paper is organized as follows. In Section 2, we discuss the previous works about dynamic scheduling with delay information. In Section 3, we present the proposed end-to-end bulk scheduling scheme in detail, together with BSS flow bandwidth estimation methods for scheduling packets in the intermediate nodes. The analyses of the bulk scheduling scheme are given in this section. In Section 4, we propose two enhancements of bulk scheduling scheme, the dropping policy and feedback mechanism. In Section 5, we present the simulation results, which show the characteristics and strengths of our proposed scheme. Finally, in Section 6, we give our conclusions.

2. Previous Works

2.1. Per-packet scheduling

Our objective is to support end-to-end delay requirements for real-time applications within a dynamically changing network environment. The method for recording information in real-time traffic is also used by Zhu et al. [16]. They propose a per-node deadline-curve scheduling scheme to guarantee the end-to-end delay bound. The per-node deadlines of a packet are specified by the sender and the scheduler in each intermediate node uses the Earliest Deadline First scheme. However, since the schedulers in the intermediate nodes have no means of knowing the source arrival time of a packet, each packet needs to carry some time stamp information in its header.

The method proposed in [16] transforms the deadline information at each node into nodal delay bounds. If the actual delay of a packet in a node is less than its nodal delay bound, the saving delay time is added to the delay budget, which is recorded in the packet header and used to compute the nodal delay bound in the next node. Suppose the initial nodal delay bound in the k-th node for flow i is denoted as $d_{i,k}$ and the actual delay time from the packet leaving its sender node is u, then the nodal delay bound, which is used for scheduling when the packet arrives at the k-th node, is the summation of initial nodal delay bounds from the first to the k-th node minus the actual delay u. If a packet experiences less delay time than the nodal delay bound, it will have looser nodal delay bound for scheduling at subsequent nodes. Conversely, if a packet misses the nodal deadline at previous nodes, it will have a higher priority to be served at later nodes. In [16], Zhu et al. also prove that the proposed per-node deadline-curve scheduling scheme can guarantee the end-to-end delay bound.

However, to obtain the delay budget for packets, the scheme must record the delay condition in each packet header. This will cause two problems: 1) there may not be enough space to record the information in the packet header; and 2) the per-packet computation and marking is a heavy overhead for an intermediate node.

2.2. Nodal delay assignment problem

With regard to the problem of how to assign the per-node delay, Vagish et al. [17] assume that the traffic rate specification and the load of each node are known before connection is established. They propose several strategies for assigning specific delay values for all nodes along the routing path, so that the end-to-end delay requirement of the connection is satisfied. Their objective is to maximize network resource utilization, i.e. the number of connections accepted. The approach of the algorithm given in [17] is to achieve a balance among the amounts of resources used at each node. Suppose there are N nodes along the routing path of flow i and the end-to-end delay bound of flow i is d_{max}. Then, the assignment of the delay bound at the k-th node, denoted by $d_{i,k}$, is given by

$$d_{i,k} = d_{\max}/N \qquad \text{where } k = 1 \text{ to } N \tag{1}$$

As the optimal value may not be feasible because of insufficient network resources, Vagish et al. first compute the upper bound and lower bound of the feasible delay at each node. They then assign the feasible delay to each node between the upper and lower bounds. Since the computation of the feasible delay at each node is based on the information received before the connection is established, the accurate upper and lower bounds are difficult to obtain if the load of each node has a large variance. The traffic rate specification is also hard to formulate if the traffic of a specific connection has a real-time variable bit rate. Also, the assignment of per-node delay is pre-determined and cannot be changed after the connection is established.

3. Bulk Scheduling Scheme (BSS)

Rather than fixed rate allocation or per-packet scheduling, we divide the flow load into segments, called "bulk" in this paper, and dynamically allocate bandwidth for each bulk. In the proposed bulk scheduling scheme, *Traffic Specification with End-to-end Deadline* (TED) packets are introduced and inserted into flow traffic periodically. With the information contained in TED packets, the BSS scheduler can determine how much time the TED packet will need in the current node with respect to the residual end-to-end delay bound and residual hops count. We will prove later that the end-to-end delay bounds of all packets can be guaranteed if the end-to-end delay time of TED packets is bounded. This means the scheduler only needs to dynamically adjust the bandwidths for TED packets, instead of adjustment for each packet. Considering a flow path, the ingress node is responsible for the insertion of TED packets into a flow periodically with the period I_{TED}. Meanwhile, the intermediate nodes schedule and service packets based on the information in the TED packets and modify TED contents, and the egress node removes TED packets when they arrive. Packets in a flow are served in a first-in-first-out (FIFO) manner, as shown in Fig. 1.

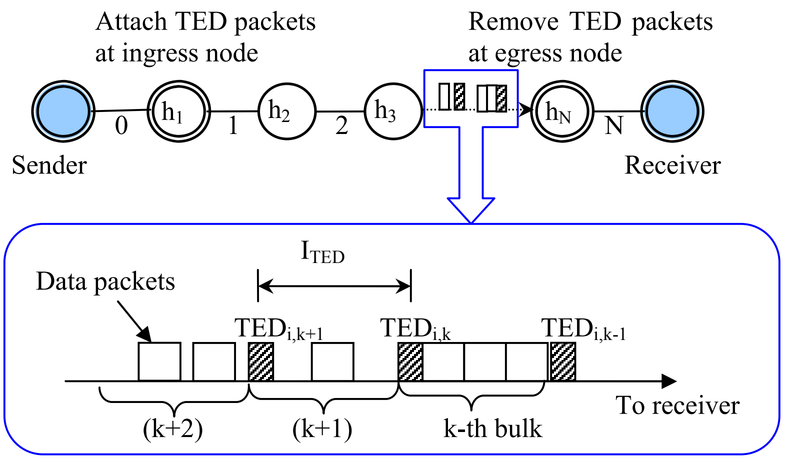

Figure 1. TED-based bulk scheduling scheme overview.

Definitions of bulk – Let $TED_{i,k}$ denote the k-th TED packet of flow i. We define the bulk as the packets between two adjacent TED packets, i.e. the k-th bulk in flow i contains all packets between $TED_{i,k-1}$ and $TED_{i,k}$

3.1. TED packets and End-to-End delay

The end-to-end delay of a packet is computed as the period from the time a packet arrives at the ingress node until the time it reaches the egress node. Here, we prove that if the delays of TED packets are guaranteed, then the delay bounds of the packets in the bulks are guaranteed as well.

Theorem 1. At the receiver, for the k-th bulk of flow i, if end-to-end delay of the $TED_{i,k}$ is bounded, the end-to-end delays of the packets in the k-th bulk are also bounded. Let $\tau_{i,k}^{SEND}(j)$ and $\tau_{i,k}^{RCV}(j)$ be the sending time at the sender and the receiving time at the receiver of the j-th packet in k-th bulk of flow i, respectively. The sending and receiving times of the corresponding TED packet are denoted as $\tau_{i,k}^{SEND}(TED)$ and $\tau_{i,k}^{RCV}(TED)$, respectively. Then we get the following relation,

$$\tau_{i,k}^{RCV}(TED)-\tau_{i,k}^{SEND}(TED)\le d_{\max} \rightarrow \tau_{i,k}^{RCV}(j)-\tau_{i,k}^{SEND}(j)\le d_{\max}+I_{TED} \qquad (2)$$

where d_{max} is the end-to-end delay bound of the TED packets and I_{TED} is the period during which we insert the TED packets into flow traffic.

Proof of Theorem 1.

Since the TED packets are generated periodically with an interval of I_{TED} and packets with the same flow at each node are served in a FIFO manner, the dispatch time of any packet j in the k-th bulk $\tau_{i,k}^{SEND}(j)$ is after the time $\tau_{i,k}^{SEND}(TED) - I_{TED}$ and the arrival time $\tau_{i,k}^{RCV}(j)$ is before the time $\tau_{i,k}^{RCV}(TED)$. Therefore, we get the following relation

$$\tau_{i,k}^{RCV}(j) - \tau_{i,k}^{SEND}(j) \le \tau_{i,k}^{RCV}(TED) - \left(\tau_{i,k}^{SEND}(TED) - I_{TED}\right) \tag{3}$$

and with the end-to-end delay bound of the TED packets d_{max}, we have

$$\tau_{i,k}^{RCV}(TED) - \tau_{i,k}^{SEND}(TED) \le d_{\max} \tag{4}$$

So, from (3) and (4), we can obtain

$$\tau_{i,k}^{RCV}(j) - \tau_{i,k}^{SEND}(j) \le d_{\max} + I_{TED} \tag{5}$$

□

In the following, we suppose the flow path for the flow i from sender to receiver consists of $N+1$ links, labeled as $L_o, L_1, L_2, \ldots L_N$; and N intermediate nodes, labeled as $h_1, h_2, \ldots h_N$. For each link, there are multiple flows sharing the link capacity. For each flow sharing the link capacity of link L_0, it employs per-flow queueing in the node h_n $(1 \le n \le N)$ on a FIFO basis. An intermediate node calculates and may adjust the bandwidth of the flow when receiving a TED packet, rather than serves each flow with a constant weight or priority. By Theorem 1, as long as all the TED packets meet their deadlines, so do all the packets in the flow.

The BSS scheduler in the intermediate nodes performs the following tasks: 1) when a TED packet arrives at an intermediate node, its arrival time is recorded to calculate the remaining time that it can use; 2) when a TED packet becomes the first TED packet in its flow queue, we modify the scheduling priority or bandwidth of the corresponding flow, according to the information in the TED packet. The original intermediate node scheduler then selects the packet to serve according to its scheduling policy; and 3) when the TED packet leaves the intermediate node, we update the residual end-to-end delay bound stored in the TED packet.

We denote the time instance when a TED packet becomes the first TED packet of its flow queue as its scheduling time. In the remainder of this paper, we use $A^{n}_{i,k}$, $S^{n}_{i,k}$ and, $D^{n}_{i,k}$

to represent the arrival time, schedule time and departure time of the *k*-th TED packet of flow *i* in the node h_n. Note that we ignore all negligible delays, such as propagation delay, as they do not affect the results. With the definitions, we get the following equations

$$\tau_{i,k}^{SEND}(TED) = A_{i,k}^{1}, \quad \tau_{i,k}^{RCV}(TED) = D_{i,k}^{N} \tag{6}$$

$$A_{i,k}^{n} = D_{i,k}^{n-1}, \ for\ n = 2\ to\ N \tag{7}$$

$$S_{i,k}^{n} = \max(A_{i,k}^{n}, D_{i,k-1}^{n}), \text{ for } n = 1\ to\ N \tag{8}$$

With these terms, the nodal delay time of $TED_{i,k}$ in the node h_n is $D^n_{i,k}–A^n_{i,k}$. The end-to-end delay of $TED_{i,k}$ consists of the nodal delay times in all nodes that $TED_{i,k}$ passed. We divide the nodal delay time of $TED_{i,k}$ in the node h_n into two parts–the waiting time $w^n_{i,k}$ and processing time $\delta^n_{i,k}$. The waiting time $w^n_{i,k}$, which equals $S^n_{i,k}–A^n_{i,k}$, means the period from the arrival of $TED_{i,k}$ in h_n to the time that it becomes the leading TED packet in its flow queue. The processing time $\delta^n_{i,k}$, which is equal to $D^n_{i,k}–S^n_{i,k}$, is the period from the time that $TED_{i,k}$ becomes the leading TED packet in its flow queue to the departure time of $TED_{i,k}$ in h_n. If $TED_{i,k}$ is the first TED packet in its queue to arrive at h_n, the waiting time is equal to zero. Therefore, the composition of the end-to-end delay of $TED_{i,k}$, i.e., $D^N_{i,k}–A^1_{i,k}$, consists of the waiting and processing times in all nodes that $TED_{i,k}$ passed.

The objective of our scheduler in the intermediate nodes is to minimize the impact of flow *i* to other flows that pass through this node, i.e., h_n, under the constraint that all TED packets can meet their end-to-end delay bounds, i.e.,

$$\sum_{m=1}^{N-1}\left(w_{i,k}^{m} + \delta_{i,k}^{m}\right) \leq d_{\max}, \qquad \text{for all } k \tag{9}$$

However, there is a tradeoff between minimizing the impact on other flows and enforcing end-to-end delay bounds for flow *i*. A short processing time $\delta^n_{i,k}$ of $TED_{i,k}$ in h_n shortens its end-to-end delay, but reduces the bandwidths assigned to other flows and incurs longer delays. Therefore, the interval of TED packets and processing times to TED packets in the intermediate nodes are the major issues of the bulk scheduling scheme.

3.2. TED interval and end-to-end delay bound assignment

According to Theorem 1, the end-to-end delay bound guarantee for all packets is $d_{max}+I_{TED}$ if the end-to-end delay bound guarantee of TED packets is d_{max}. We now present two strategies to assign end-to-end delay bound to TED packets as follows.

1) Explicit end-to-end delay bound assignment

The explicit assignment sets the end-to-end delay bound for TED packets with the value $d_{max}–I_{TED}$. Hence, we can guarantee the end-to-end delay bound for all packets in flow *i* with d_{max}. With this explicit assignment strategy, a large TED interval I_{TED} forces the intermediate nodes schedule TED packets with small end-to-end delay bounds that requires extra bandwidth. On the contrary, a small TED interval results in excessive TED packets

2) Implicit end-to-end delay bound assignments

Although the explicit assignment provides a tight delay bound for all packets, it may result in a high bandwidth requirement and overhead for intermediate nodes. Compared to explicit assignment, implicit assignment provides a relaxed end-to-end delay bound for all packets by setting the end-to-end delay bound for TED packets exactly equal to d_{max}. We will show that with our proposed bulk scheduling algorithms, we can enforce end-to-end delay bounds of d_{max} for all packets by restricting I_{TED} equal to or less than d_{max}/N.

Theorem 2. Suppose the sending rate of the *k*-th bulk of flow *i* at the source and the bandwidth of flow *i* during processing $TED_{i,k}$ in the node h_n is constant. If the processing time of $TED_{i,k}$ in the node h_n is greater than or equal to the TED interval of flow *i*, i.e., $\delta^{n}_{i,k} \geq I_{TED}$, then the end-to-end delay of each packet in the *k*-th bulk is shorter than or equal to that of $TED_{i,k}$. i.e.,

$$\tau^{n}_{i,k}(j) - \tau^{SEND}_{i,k}(j) \leq \tau^{n}_{i,k}(TED) - \tau^{SEND}_{i,k}(TED) \tag{10}$$

where $\tau^{n}_{i,k}(TED)$ and $\tau^{n}_{i,k}(j)$ are the departure times of $TED_{i,k}$ and the *j*-th packet in the *k*-th bulk of flow *i*, respectively.

Proof of Theorem 2.

Let $B^{0}_{i,k}$ be the total amount of data in the *k*-th bulk of flow *i*, and $B^{j}_{i,k}$ be the amount of data that was sent after the *j*-th packet in the *k*-th bulk but before $TED_{i,k}$ of flow *i* plus the size of $TED_{i,k}$. Since the sending rate is constant, the relation between the sending time of the j-th packet and $TED_{i,k}$ is as follows,

$$\tau_{i,k}^{SEND}(j) = \tau_{i,k}^{SEND}(TED) - I_{TED} \cdot B_{i,k}^{0} / B_{i,k}^{j} \tag{11}$$

If the j-th packet leaves h_n before $TED_{i,k}$ arrives h_n, we get

$$\tau_{i,k}^{n}(j) \le \tau_{i,k}^{n}(TED) - \delta_{i,k}^{n} \le \tau_{i,k}^{n}(TED) - I_{TED} \tag{12}$$

Since $B^j_{i,k}$ must be less than $B^0_{i,k}$, combining (11) and (12), we get

$$\begin{aligned} \tau_{i,k}^{n}(j) - \tau_{i,k}^{SEND}(j) &\le \tau_{i,k}^{n}(TED) - \tau_{i,k}^{SEND}(TED) - I_{TED} \cdot \left(B_{i,k}^{0} / B_{i,k}^{j}\right) \\ &< \tau_{i,k}^{n}(TED) - \tau_{i,k}^{SEND}(TED) \end{aligned} \tag{13}$$

Otherwise, let $\overline{B}_{i,k}^{n}$ be the amount of data that queueing before $TED_{i,k}$ plus the size of $TED_{i,k}$ when $TED_{i,k}$ becomes the first TED packet in node h_n. Since the bandwidth is constant while $TED_{i,k}$ is being served, we can get the following relation,

$$\begin{aligned} \tau_{i,k}^{n}(j) &= \tau_{i,k}^{n}(TED) - \delta_{i,k}^{n} \cdot B_{i,k}^{0} / \overline{B}_{i,k}^{n} \\ &\le \tau_{i,k}^{n}(TED) - I_{TED} \cdot B_{i,k}^{0} / \overline{B}_{i,k}^{n} \end{aligned} \tag{14}$$

Combining the (11) and (14) we obtain

$$\begin{aligned} \tau_{i,k}^{n}(j) - \tau_{i,k}^{SEND}(j) &\le \tau_{i,k}^{n}(TED) - \tau_{i,k}^{SEND}(TED) - I_{TED} \cdot \left(B_{i,k}^{0} / \overline{B}_{i,k}^{n} - B_{i,k}^{0} / B_{i,k}^{j}\right) \\ &\le \tau_{i,k}^{n}(TED) - \tau_{i,k}^{SEND}(TED) \end{aligned} \tag{15}$$

Therefore, the end-to-end delay of each packet in the k-th bulk is shorter than or equal to that of $TED_{i,k}$.

□

3.3. BSS nodal Scheduling in the intermediate nodes

Let $\Delta^n_{i,k}$ denote the *processing index* assigned to flow i in node h_n when $TED_{i,k}$ becomes the leading TED packet in h_n. The lower $\Delta^n_{i,k}$ of flow i means it has a higher bandwidth to be served in node h_n. The residual end-to-end delay bound when $TED_{i,k}$ arrives at h_n is represented by $d^n_{i,k}$ and defined as follows:

$$d_{i,k}^{n} = d_{\max}^{e2e} - (A_{i,k}^{n} - A_{i,k}^{1}) = d_{\max}^{e2e} - \sum_{m=1}^{n-1}\left(w_{i,k}^{m} + \delta_{i,k}^{m}\right) \tag{16}$$

We assign the value of $\Delta^n_{i,k}$ based on the expected processing time in the node h_n. We hope that if the actual nodal processing time $\delta^n_{i,k}$ is equal to $\Delta^n_{i,k}$ at each node in the path of flow i, then the end-to-end delay of $TED_{i,k}$ is exactly equal to d_{max}. Note that we use the residual end-to-end delay bounds rather than absolute global deadlines of TED packets for scheduling, so there is no time synchronization problem among all nodes. In the following, we propose three methods for estimating the nodal processing indexes.

1) Global Even Distribution (GED)

The first method is based on the scheme proposed in [16] and the nodal delay assignment in [17]. The end-to-end delay bound is evenly allocated to all nodes in the path. Each node gets its initial nodal delay bound when the connection is established. The estimation of $\Delta^{n}_{i,k}$ is as follows

$$\Delta^{n}_{i,k} = d^{n}_{i,k} - (N-n)\cdot d_{\max} / N - w^{n}_{i,k} \tag{17}$$

2) Residual Even Distribution on Arrival (RED-A)

In the second method, we calculate the processing index $\Delta^{n}_{i,k}$ based on the residual end-to-end delay bound when $TED_{i,k}$ arrives at h_n. We follow the guideline that the residual end-to-end delay bound is evenly allocated among residual nodes. Therefore, the expected nodal delay bound of $TED_{i,k}$ in h_n is $d^{n}_{i,k}/(N\text{-}n+1)$, and the $\Delta^{n}_{i,k}$ is

$$\Delta^{n}_{i,k} = d^{n}_{i,k} /(N-n+1) - w^{n}_{i,k} \tag{18}$$

3) Residual Even Distribution on Schedule (RED-S)

In this method, we also allocate the residual end-to-end delay bound evenly among the residual nodes. Unlike RED-A, we take the waiting time of $TED_{i,k}$ in h_n into account when we determine the residual nodal delay bound in this method. Note that the actual residual end-to-end delay bound when $TED_{i,k}$ becomes the leading TED packet, i.e., the schedule time of $TED_{i,k}$ in h_n, is $d^{n}_{i,k} - w^{n}_{i,k}$. Since we want to allocate the residual end-to-end delay bound evenly among all residual nodes, the $\Delta^{n}_{i,k}$ is calculated as follows

$$\Delta^{n}_{i,k} = (d^{n}_{i,k} - w^{n}_{i,k}) /(N-n+1) \tag{19}$$

In the three methods above, the residual end-to-end delay time $d^{n}_{i,k}$ and residual hops count N-$n+1$ are stored in TED packets, while the waiting time $w^{n}_{i,k}$ is measured by the intermediate node h_n. If the nodal processing time is guaranteed to be less than the processing index, and $w^{n}_{i,k}+\Delta^{n}_{i,k}$ is less than $d^{n}_{i,k}$ at each node h_n, then the TED packet $\mathrm{TED}_{i,k}$ will definitely meet its end-to-end deadline.

In the literature, the General Processor Sharing (GPS) [18] scheduler is recognized as a scheduling policy that can guarantee the delay bound and achieves the fairest service for each flow. Without loss of generality, in this paper, we assume a General Processor Sharing (GPS)-based scheduling algorithm, such as Weighted Fair Queueing (WFQ) [18], Worst-case Fair Weighted Fair Queueing (WF^{2}Q) [19], and Deficit Round Robin (DRR) [20], is used as the scheduling algorithm at each router. We classify flows into two groups, T-flows and N-flows. T-flows represent the set of flows that submit the delay bound requirements via TED packets and N-flows otherwise. Each flow has a demand rate r_i. For N-flows, the demand rate is a constant value, while for T-flows, their demand rates are changed from time to time and calculated as follows

$$r_i(t) = \overline{B}_{i,k}^{n} / \Delta_{i,k}^{n}, \quad \text{for } S_{i,k+1}^{n} > t \geq S_{i,k}^{n}, \tag{20}$$

where $\overline{B}_{i,k}^{n}$ is the amount of data that queueing before $TED_{i,k}$ plus the size of $TED_{i,k}$ when $TED_{i,k}$ becomes the leading TED packet in node h_n. The BSS scheduler satisfies T-flows' demands first and then allocate the residual bandwidth to N-flows. Fig. 2 shows the BSS scheduler architecture, where TB and NB are the set of all backlogged T-flows and N-flows respectively. Note that the scheduler is work-conserving, so the actual bandwidth for T-flows is more than its demand rates if link capacity C is more than the summation of the demand rates of all T-flows and there are no backlogged N-flows. Table 1 lists the actual bandwidths of the scheduler under different conditions. We can see that only in case (A) the demand rate of T-flows cannot be satisfied, but compared to weighted scheduling sharing bandwidth with other N-flows, the BSS scheduler still provides better service for T-flows. And with some admission control and bandwidth reservation using a bandwidth reservation protocol, such as RSVP[5] or ST2[21], the occurrence of case (A) can be controlled. Besides, in the case (C), the scheduler only provides the bandwidth that each T-flow demands, so the N-flows can actually get higher service quality than weighted scheduling.

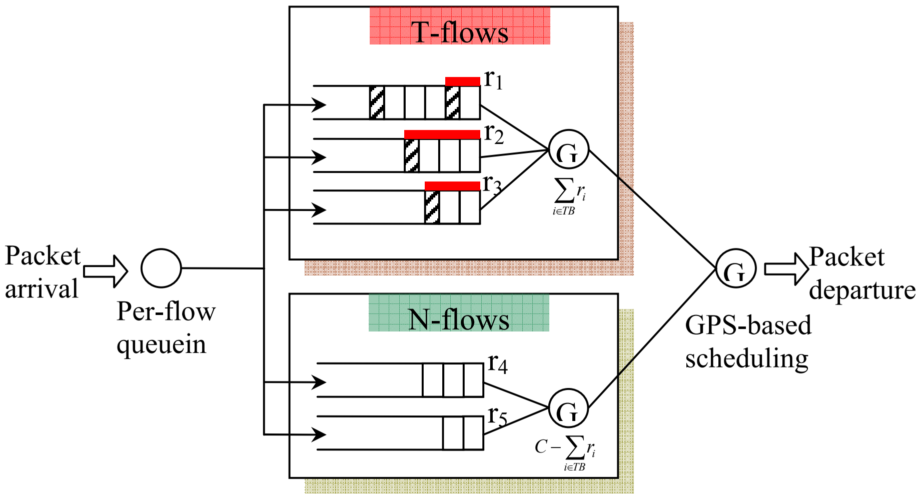

Figure 2. The scheduler architecture in the intermediate nodes.

Table 1. The Actual Bandwidths under Different Conditions in the intermediate nodes

	Case (A)	Case (B)	Case (C)
	$\sum_{i \in TB} r_i \geq C$	$\sum_{i \in TB} r_i < C \wedge NB = \phi$	$\sum_{i \in TB} r_i < C \wedge NB \neq \phi$
T-flows	$r_i \cdot C \Big/ \sum_{i \in TB} r_i$	$r_i \cdot C \Big/ \sum_{i \in TB} r_i$	$r_i \cdot$
N-flows	0	0	$r_i \cdot \left(C - \sum_{i \in TB} r_i \right) \Big/ \sum_{i \in NB} r_i$

3.4. Analyses of Bulk Scheduling Algorithms

In the following analyses, we focus on the value of the processing index $\Delta^n_{i,k}$ assigned by different nodal processing index estimation methods. A lower $\Delta^n_{i,k}$ means a higher bandwidth, which will cause a longer nodal delay for the other flows in the same node. Therefore, a good nodal processing index method should assign the value of the processing index as high as possible, without missing the end-to-end deadline. First, we consider the case where the nodal processing time, $\delta^n_{i,k}$, is exactly equal to its processing index, $\Delta^n_{i,k}$, in each node h_n. If $TED_{i,k}$ has a nodal processing time $\delta^n_{i,k}$ less than $\Delta^n_{i,k}$ in the node h_n, the residual end-to-end delay bounds at later nodes will become larger. Consequently, in the three proposed methods, later nodes will have higher processing indices than in the case where $\delta^n_{i,k}$ is equal to $\Delta^n_{i,k}$. We will discuss the impact of $\delta^n_{i,k}$ exceeding $\Delta^n_{i,k}$ in node h_n for $TED_{i,k}$ later.

Case 1) Each TED packet $TED_{i,k}$ has the nodal processing time $\delta^n_{i,k}$ exactly equal to its processing index $\Delta^n_{i,k}$ in each node h_n.

Property 1. For the three algorithms(GED, RED-A and RED-A), if the waiting time of $TED_{i,k}$ in node h_1 is equal to zero, then the transmission index of $TED_{i,k}$ in node h_1 is equal to $d_{\max}/N$. That is

$$w^1_{i,k} = 0 \;\rightarrow\; \Delta^1_{i,k} = d_{\max} / N \tag{21}$$

Proof of Property 1

Since $w^1_{i,k}=0$ and $d^1_{i,k}=d_{max}$, $\Delta^1_{i,k} = d^1_{i,k} - (N-1)\cdot d_{\max} / N = d_{\max} / N$ for algorithm GED, and for algorithms RED-A and RED-S, $\Delta^1_{i,k} = (d^1_{i,k})/(N-1+1) = d_{\max} / N$.

□

We divide the case 1 into two parts: $d_{\max} / N \le I_{TED}$ *and* $d_{\max} / N > I_{TED}$.

Case 1-Part 1) If $d_{max}/N \le I_{TED}$, then $\Delta^n_{i,k}=d_{max}/N$ for the three algorithms.

In this case, $\Delta^1_{i,k}$ for all k is equal to d_{max}/N, which can be proved by induction. Because $TED_{i,1}$ is the first TED packet, its waiting time in all nodes must be equal to zero, i.e., $w^n_{i,1}=0$ for all n, and $\Delta^1_{i,1}= d_{max}/N$, according to property 1. Next, let $\Delta^1_{i,k} = d_{\max}/N$, then $w^1_{i,k+1}=0$, because $d_{\max} / N \le I_{TED}$. Similarly, we can get $\Delta^1_{i,k+1} = d_{\max}/N$ by property 1. Therefore, $\Delta^1_{i,k} = d_{\max}/N$ for all k.

We now prove that $\Delta^n_{i,k} = d_{\max}/N$ for all n in all nodes by induction. If the transmission indices of $TED_{i,k}$ from h_2 to h_n are all equal to d_{max}/N, i.e., $\Delta^m_{i,k} = d_{\max}/N$ for $2 \le m \le n$, we get the residual end-to-end delay bound $d^{n+1}_{i,k} = d_{\max} - n\cdot d_{\max}/N$. Because the waiting times for $TED_{i,1}$ in all nodes, i.e., $w^n_{i,1}$, are always equal to zero, we get $\Delta^{n+1}_{i,1} = d_{\max} / N$ for the three algorithms. Following the steps in the above proof in node h_1, we get $\Delta^{n+1}_{i,k} = d_{\max} / N$ for all k in node h_{n+1}. Finally, $\Delta^n_{i,k} = d_{\max}/N$ for all TED packets at all nodes.

Case 1-Part 2) If $d_{\max} / N > I_{TED}$, *then* $\Delta^n_{i,k} = d_{\max} / N \geq I_{TED}$ *for the three algorithms*

- For algorithm GED and RED-A

As in part 1, $\Delta^n_{i,1}$ is equal to d_{max}/N at all nodes, because their waiting times are equal to zero. Since $d_{max}/N > I_{TED}$, the waiting times for $TED_{i,2}$ at all nodes are $d_{max}/N - I_{TED}$. For GED and RED-A, the expected nodal delay time at each node is d_{max}/N. The waiting time is subtracted from the calculation of the transmission index, so that the transmission index, $\Delta^n_{i,k}$, for $k \geq 2$ is equal to I_{TED} because $d_{\max} / N - (d_{\max} / N - I_{TED}) = I_{TED}$.

- For algorithm RED-S

As mentioned earlier, $\Delta^n_{i,1}$ is equal to d_{max}/N at all nodes, because its waiting time is always equal to zero. For $k \geq 2$, we show that the value of $\Delta^n_{i,k}$ is always more than I_{TED}.

Property 2. For RED-S, if the waiting time $w^n_{i,k}$ is equal to zero, then the transmission index of $TED_{i,k}$ in node h_n is equal to that in h_{n-1}, i.e.

$$w^n_{i,k} = 0 \ \rightarrow \ \Delta^n_{i,k} = \Delta^{n-1}_{i,k} \tag{22}$$

Proof of Property 2

From the definition of the residual end-to-end delay bound, we get the following equation

$$\begin{aligned} d^n_{i,k} &= d_{\max} - (A^n_{i,k} - A^1_{i,k}) \\ &= d_{\max} - (A^{n-1}_{i,k} - A^1_{i,k}) - (D^{n-1}_{i,k} - A^{n-1}_{i,k}) \\ &= d^{n-1}_{i,k} - w^{n-1}_{i,k} - \delta^{n-1}_{i,k} \end{aligned} \tag{23}$$

Since the waiting time $w^n_{i,k}$ is equal to zero and actual nodal delay time $\delta^n_{i,k}$ is equal to $\Delta^n_{i,k}$, the transmission index $\Delta^n_{i,k}$ can be written as

$$\begin{aligned} \Delta^n_{i,k} &= \frac{d^n_{i,k}}{N-n+1} \\ &= \frac{d^{n-1}_{i,k} - w^{n-1}_{i,k}}{N-n+1} - \frac{\Delta^{n-1}_{i,k}}{N-n+1} \\ &= \frac{N-n+2}{N-n+1} \cdot \Delta^{n-1}_{i,k} - \frac{\Delta^{n-1}_{i,k}}{N-n+1} \\ &= \Delta^{n-1}_{i,k} \end{aligned} \tag{24}$$

Therefore, $\Delta^n_{i,k}$ is equal to $\Delta^{n-1}_{i,k}$.

□

Property 3. For RED-S, if the waiting time $w^{n}_{i,k}$ is more than zero and the transmission index $TED_{i,k-1}$ in node h_n is more than I_{TED}, then the transmission index of $TED_{i,k}$ in node h_n is also more than I_{TED}, i.e.,

$$(\Delta^{n}_{i,k-1} > I_{TED}) \wedge (w^{n}_{i,k} > 0) \quad \rightarrow \quad \Delta^{n}_{i,k} > I_{TED} \tag{25}$$

Proof of Property 3

From the definition of the residual end-to-end delay bound and the waiting time of $TED_{i,k}$ in the node h_n, we get the following equation

$$\begin{aligned}(N-n+1)\cdot\Delta^{n}_{i,k} &= d^{n}_{i,k} - w^{n}_{i,k}\\ &= d_{\max} - \left(A^{n}_{i,k} - A^{1}_{i,k}\right) - \left(D^{n}_{i,k} - A^{n}_{i,k}\right) + \delta^{n}_{i,k}\\ &= d_{\max} - \left(D^{n}_{i,k} - A^{1}_{i,k}\right) + \delta^{n}_{i,k}\end{aligned} \tag{26}$$

If $w^{n}_{i,k}$ is more than zero, the (26) can be rewritten as

$$\begin{aligned}(N-n+1)\cdot\Delta^{n}_{i,k} &= d_{\max} - \left(D^{n}_{i,k} - A^{1}_{i,k}\right) + \delta^{n}_{i,k}\\ &= d_{\max} - \left(D^{n}_{i,k-1} - A^{1}_{i,k-1}\right) + I_{TED}\\ &= d_{\max} - \left(A^{n}_{i,k-1} - A^{1}_{i,k-1}\right) - \left(w^{n}_{i,k-1} + \delta^{n}_{i,k-1}\right) + I_{TED}\\ &= \left(d^{n}_{i,k-1} - w^{n}_{i,k-1}\right) - \delta^{n}_{i,k-1} + I_{TED}\\ &= (N-n+1)\cdot\Delta^{n}_{i,k-1} - \Delta^{n}_{i,k-1} + I_{TED}\\ &= (N-n)\cdot\left(\Delta^{n}_{i,k-1} - I_{TED}\right) + (N-n+1)\cdot I_{TED}\end{aligned} \tag{27}$$

Therefore, we get the following equation

$$\Delta^{n}_{i,k} = \frac{N-n}{N-n+1}\cdot\left(\Delta^{n}_{i,k-1} - I_{TED}\right) + I_{TED} \tag{28}$$

which shows that if $\Delta^{n}_{i,k-1}$ is more than I_{TED}, then $\Delta^{n}_{i,k}$ is also more than I_{TED}.

□

Property 4. For RED-S, if $d_{max}/N>I_{TED}$ and $\delta^{n}_{i,k}=\Delta^{n}_{i,k}$ for all TED packets in all nodes, then $\Delta^{n}_{i,k}>I_{TED}$ for all TED packets in all nodes.

Proof of Property 4

We prove property 4 by induction. We show that $\Delta^{n}_{i,1}>I_{TED}$ for $TED_{i,1}$ in all nodes first and $\Delta^{1}_{i,k}>I_{TED}$ for all TED packets in node h_1. We then show that $\Delta_{1i,k}>_{ITED}$ if $\Delta^{n-1}_{i,k}$ and $\Delta^{n}_{i,k-1}$ are both more than $I_{TED..}$

First, we know that $\Delta^{1}_{i,1}=d_{max}/N>I_{TED}$. According to property 2, we get $\Delta^{n}_{i,1}=\Delta^{n-1}_{i,1}$ because the waiting time of $TED_{i,1}$ in all nodes is equal to zero. Therefore, $\Delta^{n}_{i,1}>I_{TED}$ for $TED_{i,1}$ at all nodes.

We now show, also by induction, that $\Delta^{1}_{i,k}>I_{TED}$ for all TED packets in the node h_1. Suppose $\Delta^{1}_{i,k}>I_{TED}$, then $w^{1}_{i,k+1}>0$ because $TED_{i,k+1}$ arrives in the node h_1 at time instance $A_{i,k}+I_{TED}$, which is earlier than the departure of $TED_{i,k}$ in h_1, i.e. $A_{i,k}+w^{1}_{i,k}+\Delta^{1}_{i,k}$. With property 3, we get $\Delta^{1}_{i,k+1}>I_{TED}$. Therefore, $\Delta^{1}_{i,k}>I_{TED}$ for all TED packets in the node h_1.

Now, suppose $\Delta^{n-1}_{i,k}$ and $\Delta^{n}_{i,k-1}$ are both more than I_{TED}. If $w^{n}_{i,k}=0$, then $\Delta^{n}_{i,k}=\Delta^{n-1}_{i,k}>I_{TED}$ by property 2. Otherwise, if $w^{n}_{i,k}>0$, then $\Delta^{n}_{i,k}>I_{TED}$ by property 3 because $\Delta^{n}_{i,k-1}>I_{TED}$. Therefore, $\Delta^{n}_{i,k}>I_{TED}$ for all TED packets in all nodes.

□

Table 2 lists the transmission indices, $\Delta^{n}_{i,k}$, in case 1. When d_{man}/N is less than or equal to I_{TED}, the three algorithms have the same transmission indices. However, if d_{man}/N is more than I_{TED}, Algorithm RED-S performs more efficiently than the others because most of its transmission indices are more than I_{TED}.

Table 2. The Transmission Indices for Different Algorithms when Transmission Times Equals Tranmission indices.

	$d_{max}/N\leq I_{TED}$	$d_{max}/N>I_{TED}$
GED	d_{max}/N	d_{max}/N or I_{TED}
RED-A	d_{max}/N	d_{max}/N or I_{TED}
RED-S	d_{max}/N	Between d_{max}/N and I_{TED}

We have shown that with our three proposed algorithms the transmission indices of the k-th bulk of flow i at all nodes are greater than or equal to I_{TED}, i.e., $I_{TED}\leq\Delta^{n}_{i,k}$, when $I_{TED}\leq d_{\max}/N$ in Table 2. Also, TED packets will not miss their end-to-end deadlines if their transmission times are not more than their transmission indices, i.e., $\delta^{n}_{i,k}\leq\Delta^{n}_{i,k}$. If $I_{TED}\leq\delta^{n}_{i,k}\leq\Delta^{n}_{i,k}$, the end-to-end delay bounds of all packets in the k-th bulk are guaranteed according to Theorem 2. Otherwise $\delta^{n}_{i,k}<I_{TED}$, in which case the end-to-end delay bounds of all packets in the k-th bulk are also guaranteed since these packets has less end-to-end delay

than in the case of $I_{TED} \le \delta^n_{i,k} \le \Delta^n_{i,k}$. Therefore, with implicit end-to-end delay bound assignment for TED packets, we can enforce end-to-end delay bounds for all packets by setting the I_{TED} to any value less than $d^{e2e}_{\max} / N$.

Case 2) Consider that in node h_n, the $\delta^n_{i,k}$ is more than $\Delta^n_{i,k}$ with $\hat{\Delta}^n_{i,k}$, i.e. $\delta^n_{i,k} = \Delta^n_{i,k} - \hat{\Delta}^n_{i,k}$.

In case 1, we suppose that each TED packet, $TED_{i,k}$, has the nodal processing time, $\delta^n_{i,k}$, which is exactly equal to its processing index $\Delta^n_{i,k}$ in each node h_n. In the case 2, we consider the condition where $\delta^n_{i,k}$ exceeds $\Delta^n_{i,k}$ in node h_n for $TED_{i,k}$. Note that in the following, we compare all the results with $\delta^n_{i,k}=\Delta^n_{i,k}$, described in case 1.

Case 2-Part 1) The change in the processing index of the next TED packet, i.e., $\Delta^n_{i,k+1}$, in node h_n for the three methods.

Since the increase in $\delta^n_{i,k}$ may add to the waiting time of $TED_{i,k+1}$, the $\Delta^n_{i,k+1}$ with $\delta^n_{i,k} = \Delta^n_{i,k} + \hat{\Delta}^n_{i,k}$ would be less than that with $\delta^n_{i,k}=\Delta^n_{i,k}$,depending on the change of $w^n_{i,k+1}$. Let $\hat{w}^n_{i,k+1}$ denote the change of $w^n_{i,k+1}$. Then, we get the following equation

$$\hat{w}^n_{i,k+1} = \max(D^n_{i,k} + \hat{\Delta}^n_{i,k}, A^n_{i,k+1}) - \max(D^n_{i,k}, A^n_{i,k+1}) \qquad (29)$$

- For GED and RED-A

From the definitions of GED and RED-A, we know that the increase in $w^n_{i,k+1}$ will cause the processing index of $TED_{i,k+1}$, i.e., $\Delta^n_{i,k+1}$, to be decreased by $\hat{w}^n_{i,k+1}$. Therefore, the processing index $\Delta^n_{i,k+1}$ with $\delta^n_{i,k} = \Delta^n_{i,k} + \hat{\Delta}^n_{i,k}$ will be less than that with $\delta^l_{i,k}=\Delta^n_{i,k}$ by $\hat{w}^n_{i,k+1}$.

- For RED-S

From the definition of RED-S, we get the following equation

$$\Delta^n_{i,k+1} = (d^n_{i,k+1} - w^n_{i,k+1})/(N-n+1) - \hat{w}^n_{i,k+1} /(N-n+1) \qquad (30)$$

The reduction in the processing index of $TED_{i,k+1}$ in the node h_n using RED-S is only $\hat{w}^n_{i,k+1} /(N-n+1)$, which is less than that of GED and RED-A.

Case 2-Part 2) The change of processing index in the next node h_{n+1}, i.e. $\Delta_{i,k}^{n+1}$, for the three methods.

Since the nodal delay time of $TED_{i,k}$ in node h_n is increased by $\hat{\Delta}^{n}_{i,k}$, the waiting time of $TED_{i,k}$ in the next node h_{n+1} is reduced by $\overline{w}^{n+1}_{i,k}$, where the value of $\overline{w}^{n+1}_{i,k}$ is

$$\overline{w}^{n+1}_{i,k} = \hat{\Delta}^{n}_{i,k} - \left(\max(D^{n+1}_{i,k-1}, A^{n+1}_{i,k} + \hat{\Delta}^{n}_{i,k}) - \max(D^{n+1}_{i,k-1}, A^{n+1}_{i,k})\right) \tag{31}$$

The residual end-to-end delay bound of $TED_{i,k}$ in node h_{n+1} is also reduced by $\hat{\Delta}^{n}_{i,k}$. With the definitions of these three methods, we get the new processing index for the three methods as follows.

- For GED

$$\begin{aligned}&(d^{n+1}_{i,k} - \hat{\Delta}^{n}_{i,k}) - (N-n-1)\cdot d^{e2e}_{\max} / N - (w^{n+1}_{i,k} - \overline{w}^{n+1}_{i,k}) \\ &= \Delta^{n+1}_{i,k} - \left(\hat{\Delta}^{n}_{i,k} - \overline{w}^{n+1}_{i,k}\right)\end{aligned} \tag{32}$$

- For RED-A

$$\begin{aligned}&(d^{n+1}_{i,k} - \hat{\Delta}^{n}_{i,k}) / (N-n) - (w^{n+1}_{i,k} - \overline{w}^{n+1}_{i,k}) \\ &= \Delta^{n+1}_{i,k} - \left(\hat{\Delta}^{n}_{i,k} / (N-n) - \overline{w}^{n+1}_{i,k}\right)\end{aligned} \tag{33}$$

- For RED-S

$$\begin{aligned}&(d^{n+1}_{i,k} - \hat{\Delta}^{n}_{i,k} - (w^{n+1}_{i,k} - \overline{w}^{n+1}_{i,k})) / (N-n) \\ &= \Delta^{n+1}_{i,k} - \left(\frac{\hat{\Delta}^{n}_{i,k} - \overline{w}^{n+1}_{i,k}}{N-n}\right)\end{aligned} \tag{34}$$

Table 3 lists the reduction of the processing indices $\Delta^{n}_{i,k+1}$ and $\Delta^{n+1}_{i,k}$ when $\delta^{n}_{i,k} = \Delta^{n}_{i,k} + \hat{\Delta}^{n}_{i,k}$. This analysis shows that of the three methods: 1) the RED-S has the smallest decrease in $\Delta^{n}_{i,k+1}$, and 2) the RED-A has the smallest decrease in the processing index $\Delta^{n+1}_{i,k}$.

Table 3. The reduction of the transmission indices

	$\Delta^{n}_{i,k+1}$	$\Delta^{n+1}_{i,k}$
GED	$\hat{w}^{n}_{i,k+1}$	$\hat{\Delta}^{n}_{i,k} - \overline{w}^{n+1}_{i,k}$
RED-A	$\hat{w}^{n}_{i,k+1}$	$\hat{\Delta}^{n}_{i,k} / (N-n) - \overline{w}^{n+1}_{i,k}$
RED-S	$\hat{w}^{n}_{i,k+1} / (N-n+1)$	$\left(\hat{\Delta}^{n}_{i,k} - \overline{w}^{n+1}_{i,k}\right) / \left(N-n\right)$

4. Enhancements of Bulk Scheduling Scheme

In the bulk scheduling scheme, the scheduler in the intermediate nodes dynamically allocate bandwidth for each bulk to enforce their delay bounds. However, the missed deadline of TED packets still occurs because of the limited bandwidth resources competed by all flows. Here, we propose~~d~~ two enhancements of the bulk scheduling scheme to alleviate these problems – the drop missed policy for late packets and the feedback mechanism to refine the bandwidth allocation.

4.1. Drop Missed Policy

The residual end-to-end delay bound information in TED packets can also be used in the dropping policy which determines whether or not to drop a packet in the intermediate nodes. In the following Theorem 3, we prove that if we detect a TED packet has missed its end-to-end deadline, all packets in the same flow and queued before the TED packet will also miss their end-to-end deadlines. Therefore, we can drop these packets that have missed their end-to-end deadlines and serve other packets in the node to improve the utilization of the node.

Theorem 3. Suppose the packet sequence is not disordered when the packets of each flow pass each node. At any time *t*, if a TED packet $TED_{i,k}$ misses its end-to-end deadline in the node h_n, then all packets in the same flow and queued before $TED_{i,k}$ in h_n also miss their end-to-end deadlines.

Proof of theorem 3

Since $TED_{i,k}$ misses its end-to-end deadline, we can get $t - \tau_{i,k}^{SEND}(TED) > d_{\max}$. Also, the dispatch time of any packet which is queued before $TED_{i,k}$ in the node h_n must be earlier than $\tau_{i,k}^{SEND}(TED)$, so they also miss their end-to-end deadlines at time *t*.

□

The scheduling procedure incorporating with the proposed drop missed policy is shown in Fig. 3, where the *on_schedule($TED_{i,k}$)* procedure is called by the intermediate nodes when a TED packet becomes the first TED packet in its flow queue. Since a TED packet only records the residual end-to-end delay bound when it leaves the previous node rather than the absolute end-to-end deadline, we must calculate the actual residual end-to-end delay bound when it is scheduled, i.e., the recorded residual end-to-end delay bounds in TED packets minus the waiting time of the TED packets. According to Theorem 3, if the TED packet misses its end-to-end deadline and the drop missed policy is applied, we drop the packets queueing before the TED packet. As a result, we can allocate the bandwidth to those packets that could still arrive at their destinations in time.

```
on_schedule(TED_{i,k})
{
  /* waiting time : w */
  w = current – TED_{i,k}.arrival;

  if (TED_{i,k}.residual_d – w < 0) {
    /* TED_{i,k} missed deadline */
    Drop all packets queueing before TED_{i,k}
  } else {
    /* Schedule TED_{i,k} as described in section 3.3)*/
    Calculate transmission index for TED_{i,k};
    Allocate bandwidth for TED_{i,k}
  }
}
```

Figure 3. The scheduling procedure combing with the drop missed policy in the intermediate nodes.

4.2. Feedback mechanism

Once a TED packet misses its end-to-end deadline, besides the drop missed policy, we proposed a feedback mechanism to let the ingress nodes inform the intermediate nodes to speed up for those TED packets that have tendency to miss their end-to-end deadlines. In the proposed bulk scheduling scheme, the processing time $\delta^{n}_{i,k}$ is associated with how we schedule the $\mathrm{TED}_{i,k}$ in h_n. However, the waiting time is determined by the departure time of the previous TED packet in the same flow in h_n. This means the end-to-end delay of $TED_{i,k}$ is not only associated with the scheduling for $TED_{i,k}$ but also the delay conditions of previous TED packets in the same flow. Hence, we get the following property.

Property 5. If the waiting time of $TED_{i,k}$ in h_n is more than zero, then the cumulated delay when $TED_{i,k}$ leaves the node h_n is independent of the delay time that $TED_{i,k}$ has experienced in its past journey, i.e. from h_1 to h_{n-1}. And the cumulated delay, i.e. $D^{n}_{i,k}$–$A^{1}_{i,k}$, can be represented as follows,

$$D^{n}_{i,k} - A^{1}_{i,k} = \left(D^{n}_{i,k-1} - A^{1}_{i,k-1}\right) + \left(\delta^{n}_{i,k} - I_{TED}\right) \quad (35)$$

Proof of Property 5

Because the waiting time is more than zero, $\delta^n_{i,k}=D^n_{i,k}-D^n_{i,k-1}$, and $A^1_{i,k}=A^1_{i,k-1}+I_{TED}$. So we get the following equation

$$\begin{aligned} D^n_{i,k}-A^1_{i,k} &= \left(\delta^n_{i,k}+D^n_{i,k-1}\right)-\left(A^1_{i,k-1}+I_{TED}\right) \\ &= \left(D^n_{i,k-1}-A^1_{i,k-1}\right)+\left(\delta^n_{i,k}-I_{TED}\right) \end{aligned} \tag{36}$$

□

According to property 5, we say TED$_{\text{i,k}}$ has a *deferment* in node h_n if the waiting time of $TED_{i,k}$, i.e., $w^n_{i,k}$, is more than zero. And the *last-deferment hop* of $TED_{i,k}$ represents the last hop that $TED_{i,k}$ has a deferment in the path of flow *i.*

Property 5 shows that the cumulated delay of $TED_{i,k-1}$ will propagate to its next TED packet $TED_{i,k}$ if the waiting time of $TED_{i,k}$ is more than zero. Furthermore, any nodal delay time reductions before the last deferment hop of $TED_{i,k}$ is no use if the departure time of $TED_{i,k}$ at the last deferment is unchanged. Let $TED_{i,k'}$, where $k'<k$, be the TED packet that has no deferment in h_n and the waiting times of $TED_{i,p}$ ($k'<p\leq k$) are all more than zero. With Property 5, the cumulated delay of $TED_{i,k}$ in h_n can be represented as follows

$$\begin{aligned} D^n_{i,k}-A^1_{i,k} &= \left(D^n_{i,k'}-A^n_{i,k'}\right)+\sum_{k'<p\leq k}\delta^n_{i,p}-(k-k')\cdot I_{TED} \\ &= A^n_{i,k'}-A^1_{i,k}+\sum_{k'\leq p\leq k}\delta^n_{i,p} \end{aligned} \tag{37}$$

Theorem 4. Suppose the nodal processing times of $TED_{i,p}$, where $k'\leq p\leq k$, in $h_{n'}$ are all reduced by $\bar{\delta}$ and the nodal delay times in other hops are unchanged. Performing the reduction on the last-deferment hop reduces the end-to-end delay of $TED_{i,k}$ the most..

Proof of Theorem 4.

Let h_L be the last deferment hop of $TED_{i,k}$, we can divide all hops in the path of flow *i* into three groups: 1) the hops before h_L, 2) the last deferment hop h_L and 3) the hops after h_L. If the reduction is performed at a hop in group 1, according to (37), it only affects the value of $A^n_{i,k'}$, which is decreased by $\bar{\delta}$ after reduction. So the end-to-end delay of $TED_{i,k}$ is only decreased by $\bar{\delta}$. If the reduction is performed in h_L, the nodal processing times $\delta^n_{i,p}$, where $k'\leq p\leq k$, are all decreased by $\bar{\delta}$, then the end-to-end delay of $TED_{i,k}$ can be reduced up to

$(k - k'+1) \cdot \overline{\delta}$. Otherwise, if the reduction is performed in a hop which belongs to group 3, the end-to-end delay of *TED*$_{i,k}$ is only decreased by $\overline{\delta}$ because the nodal delays of *TED*$_{i,k}$ are all equal to the processing time in the hops behind h_L. Therefore, performing the reduction on the last-deferment hop reduces the end-to-end delay of *TED*$_{i,k}$ the most.

□

The objective of feedback mechanism is to increase the bandwidths at some nodes for the flows that may miss their end-to-end deadlines so that the missed deadline condition can be reduced or eliminated after refining. Therefore, when a *TED*$_{i,k}$ arrives at its egress node with residual end-to-end delay bound less than a threshold $ACK_{th} \cdot d_{\max}^{e2e}$, the egress node would send a ACK to the ingress node to request a speedup for the flow *i*. According to theorem 4, we will try to remove the deferment condition at the last deferment hop in the speedup and design the feedback mechanism for the bulk scheduling scheme as follows:

1) When a TED packet *TED*$_{i,k}$ has a deferment in a hop h_n, the intermediate node h_n record the node h_n and the expected weight at deferment which is calculated as follows in *TED*$_{i,k}$.

$$\text{expected weight} = \sum_{k' \le p < k} \delta_{i,p}^{n} \Big/ \left(\sum_{k' \le p < k} \delta_{i,p}^{n} - w_{i,p}^{n} \right) \tag{38}$$

Note that we only need to record the last deferment hop, so the TED packet size will not be increased by the number of deferment increases.

2) When the egress node receives a TED packet *TED*$_{i,k}$ with a residual end-to-end delay bound less than the threshold $ACK_{th} \cdot d_{\max}^{e2e}$, it sends an ACK, which contains the last deferment hop and the expected weight, to the ingress node of flow *i*.

3) When the ingress node receives the ACK, it stores the information in later TED packets to inform h_n that the bandwidth of flow *i* must be increased to the expected weight times of original estimation.

With this feedback mechanism, once the same traffic pattern appears, the nodal processing times after speedup in the last deferment hop h_n becomes

$$\hat{\delta}_{i,p}^{n} = \delta_{i,p}^{n} \cdot \left(\sum_{k' \le p < k} \delta_{i,p}^{n} - w_{i,p}^{n} \right) \Big/ \sum_{k' \le p < k} \delta_{i,p}^{n} \tag{39}$$

And the waiting time of $TED_{i,k}$ in h_n becomes

$$\begin{aligned}\hat{w}^n_{i,k} &= A^n_{i,k'} + \sum_{k' \le x < k} \hat{\delta}^n_{i,x} - A^n_{i,k} \\ &= A^n_{i,k'} + \sum_{k' \le x < k} \delta^n_{i,x} - w^n_{i,k} - A^n_{i,k} = 0\end{aligned} \tag{40}$$

In other words, the deferment in h_n no longer exists. It is expected that a new last deferment hop may occur after the speedup. Then, the feedback mechanism and speedup procedure would be applied again, in which case the TED packets must store more than one speedup messages. An n-stages feedback mechanism means it can carry n speedup messages in TED packets and resolve last n deferment hops in its flow path. Note that the feedback is only performed when a TED packet misses its end-to-end deadline, so it will become invalid if the drop missed policy is used.

We show the last deferment marking algorithm in Fig. 4. Since whether a TED packet has a deferment in an intermediate node depends on the waiting time of the TED packet, the marking algorithm is implemented in the procedure when TED packet leaves the node, i.e. *on_departure(TED_{i,k})*. Note that since the expected weight of $TED_{i,k}$ requires the value of the processing times of the TED packets from $TED_{i,k'}$ to $TED_{i,k}$, we store the cumulated processing time from $TED_{i,k'}$ in the variable *cumulate_t*.

```
on_departure(TED_i,k)
{
  /* processing time : t */
  t = current - TED_i,k.schedule;
  /* waiting time : w */
  w = TED_i,k.schedule – TED_i,k.arrival;

  if (w > 0) {
    /* Record information if this hop is a deferment. */
    TED_i,k.last_deferment = TED_i,k.residual_hop;
    TED_i,k.weight = cumulate_t / (cumulate_t-w);
    cumulate_t = cumulate_t + t;
  } else {
    cumulate_t = t;
  }

  /* Update residual delay bounds and hop counts */
  TED_i,k.residual_hop= TED_i,k.residual_hop-1;
  TED_i,k.residual_d = TED_i,k.residual_d–w–t;
}
```

Figure 4. The last deferment marking algorithm in the intermediate nodes.

5. Performance Evaluations

In this section, we evaluate the missed deadline ratio and end-to-end delay performances of the proposed bulk scheduling scheme via simulations. All simulations were performed using the ns-2 network simulator [22] with minor modifications. The missed deadline ratio of a flow is defined as the number of missed deadline packets (including dropped packets) divided by the total number of offered packets. We ignore all uncontrollable and negligible delay such as link propagation delay in our simulation. If not specified, the default traffic source of each flow follows the exponential ON/OFF model with mean burst/idle period *5000/5000* ms and sending rate during burst period *512*Kbps. The period of each simulation is 3000 seconds. The default value of I_{TED} for each flow is equal to its initial nodal delay bound, i.e. d_{max}/N, and the implicit end-to-end delay bound assignment for TED packets is used. For the flows with end-to-end delay guaranteed, i.e. T-flows, we use the missed deadline ratio as quality of service index, marked on y-axis of the left hand side. Although the end-to-end delay bound is not guaranteed for N-flows, we show the average end-to-end delay of N-flows, marked on the y-axis of right hand side, as the indication of the scheduler impacts on N-flows.

5.1. Single-hop case

In this set of single-hop simulations, the characteristics of BSS are fully examined for different environments. First, we consider the network topology where five flows share a single link as shown in Fig. 5. Each flow i is with the sender S_i and receiver R_i. Three T-flows, Flow *1*, *2* and *3*, have end-to-end delay bounds of *500*ms, *1000*ms and *1500*ms, respectively; while the other two flows, Flow *4* and *5*, are N-flows with end-to-end delay bounds of *500*ms each. All packets of the five flows pass through the link between node h_1 and node h_2, where the link capacity is *1500*Kbps.

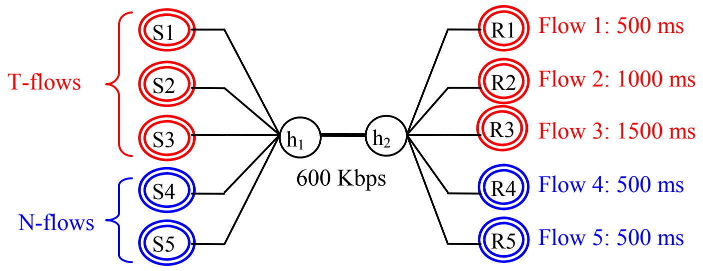

Figure 5. The network topology of single hop simulation.

5.1.1. Comparisons with weighted scheduling

In the first simulation, we compare our bulk scheduling scheme with weighted scheduling because of its popularity. We run a series of simulation with different weight assignments to T-flows for weighted scheduling that the simulation results are in Fig. 6 and Fig. 7, in which the weight settings for Flow *1/2/3* are *(x+0.6)/(x+0.3)/x* and *(x*1.2)/(x*1.1)/x*, respectively, with *x* from *1* to *4* while the weights of N-flows are equal to one. As the weight assignment is non-trivial, we use the above two general weight assignment methods with the coefficients selected from numerous simulations that perform reasonably well. Note that the dotted lines in Fig. 6 and Fig. 7 are the results of our bulk scheduling scheme that the lower three dotted lines are for T-flows and the others is for N-flow. The results show that the bulk scheduling scheme has better performances than weighted scheduling in all kinds of weight assignments. The weight assignments *5.2*, *3.9* and *2.6* for flow *1*, *2* and *3* achieve best performance for weighted scheduling for the simulation setting; however, the N-flows suffer from high miss ratio. As the weight assignments are obtained from numerous simulations, it is not clear how to select the proper weights. In addition, when the network environment is complicated as the Internet, it is very difficult to find proper weights for T-flows for weighted scheduling.

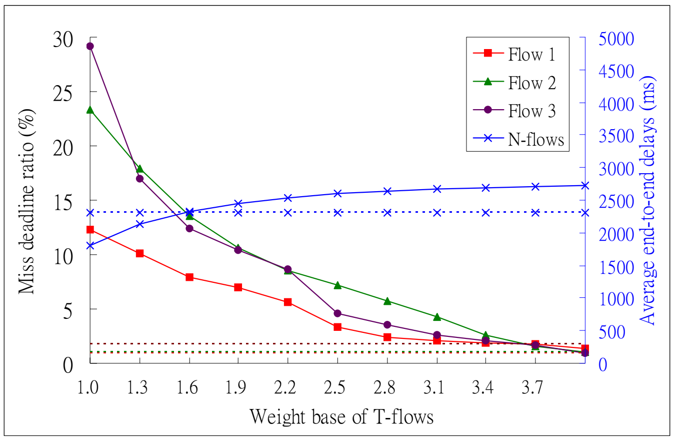

Figure 6. The performances of weighted scheduling with weights of Flow 1, 2 and 3 as (x+0.6), (x+0.3) and x.

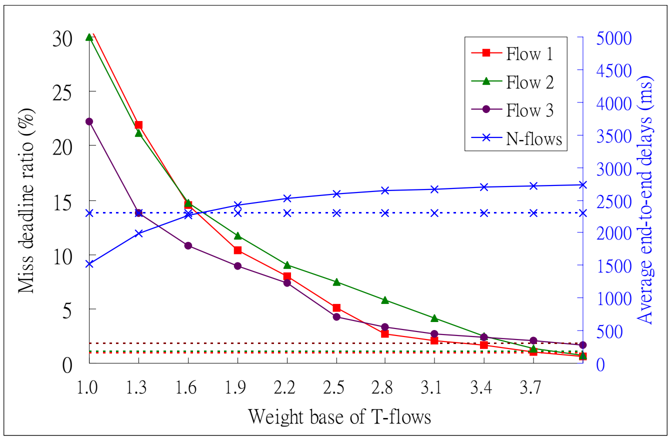

Figure 7. The performances under weighted scheduling with weights of Flow 1, 2 and 3 as (x*1.2), (x*1.1) and x.

In Table 4, we list the results of the simulation with the same network environment using the bulk scheduling scheme. Since there is only one hop in this simulation, GED, RED-A and RED-S produce the same result. Hence, only one result is shown in the table. We can see that with the bulk scheduling scheme, all T-flows have low missed deadline ratios although their average end-to-end delays increase. Furthermore, the bulk scheduling has less impact on N-flows while improving the missed deadline performance of T-flows.

Table 4. The Simulation Results of Bulk Scheduling

	BSS	
	Avg. delay	Miss ratio
Flow 1	174 ms	0.96 %
Flow 2	373 ms	1.04 %
Flow 3	602 ms	1.78 %
N-flows	2302 ms	50.26 %

5.1.2. Performance with different TED interval lengths

In order to see the impact of the length of TED intervals, we repeat the experiments in the previous simulation with different I_{TED}'s from *0.1* to *0.9* of end-to-end delay bound for each flow. Fig. 8 shows the performances of all flows under different I_{TED}'s with explicit end-to-end delay bound assignment. We can see that the end-to-end delays and missed deadline ratios of T-flows are reduced when $\mathrm{I}_{\mathrm{TED}}$ increases since, the bulk scheduling scheme will schedule TED packets with smaller end-to-end delay bounds if the value of I_{TED} becomes larger. Although larger I_{TED} results in better performance for T-flows and has less overhead, the performance of N-flows will be degraded. Fig. 9 shows the performances of three flows under different TED interval with implicit end-to-end delay bound assignment for TED packets. We can see that almost all T-flows can meet their end-to-end deadlines when TED interval is less than end-to-end delay bound, which is consistent with Theorem 3. Also, the missed deadline ratio increases when TED interval becomes larger, which is consistent with Theorem 1.

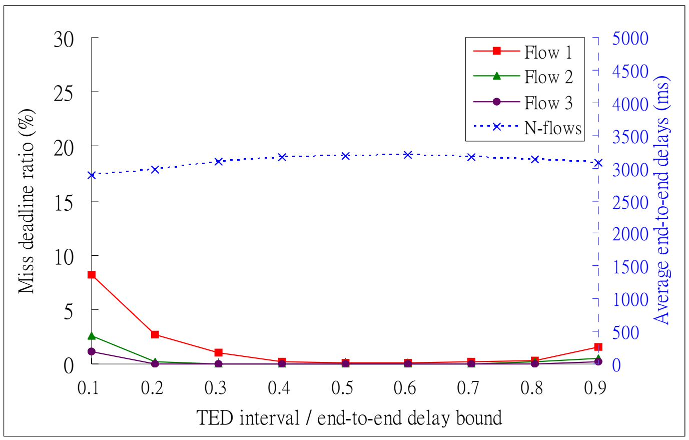

Figure 8. The performances under different I_{TED} with explicit end-to-end delay bound assignment for TED.

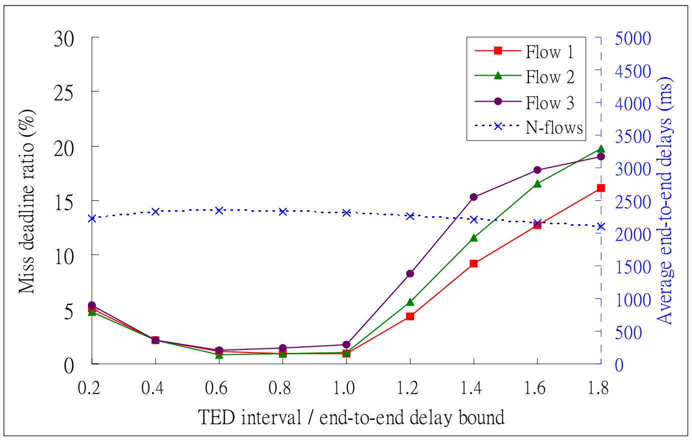

Figure 9. The performances under different I_{TED} with implicit end-to-end delay bound assignment for TED

5.1.3. Performance with different burst periods

We examine the effects of bulk scheduling scheme on flows with different bursty characteristics. Fig. 10 shows the effects with the average lengths of burst period are from 1000ms to 9000ms. The missed deadline ratio increases when the flow traffic has longer burst period.

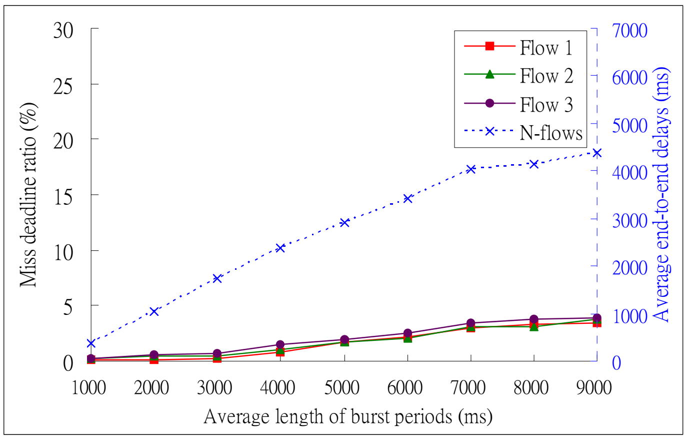

Figure 10. The performances of bulk scheduling scheme under different burst periods.

5.1.4. BSS in a lossy network environment Performance with different burst periods

In a lossy network, the TED packets may be lost during transmission. In this simulation, we set the packet loss rate between 0% and 20% to examine the performances of the bulk scheduling scheme when packets may be lost. The results are shown in Fig.11. We can see that the packet loss has little impact on our bulk scheduling scheme since the bulk scheduler maintains the bandwidth of the previous bulk when a TED packet is lost. The average missed deadline ratio of T-flows increases slightly when packet loss rate increases because fewer TED packets results in rougher end-to-end delay guarantee. To overcome this problem, users may choose the explicit end-to-end delay bound assignment for TED packets, which provide tighter end-to-end delay guarantee for data packets. Fig. 12 shows the results of bulk scheduling scheme with explicit TED delay bound assignment.

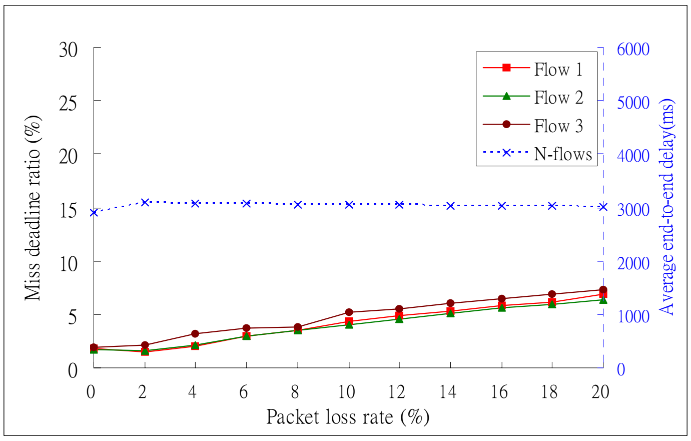

Figure 11. The performances of BSS with implicit TED delay bound assignment under different packet loss rates.

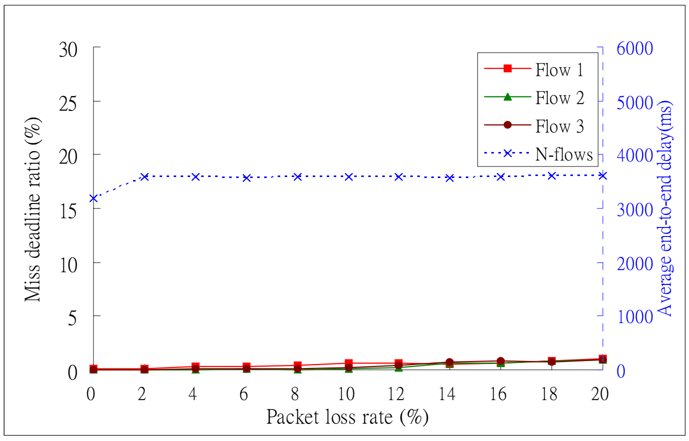

Figure 12 The performances of BSS with explicitt TED delay bound assignment under different packet loss rates.

5.1.5. Performance with different End-to-end delay bounds of T-flows

In this experiment, the end-to-end delay bound of flow 1, 2, and 3 are x, x+500ms, and x+1000ms, where x ranges from 500ms to 3000ms. Fig. 13 shows the performances of the bulk scheduling scheme with different end-to-end delay bounds. The average end-to-end delays of T-flows are increased when they have larger end-to-end delay bounds but the end-to-end delay bounds of T-flows are still guaranteed. We can see that the average end-to-end delay of N-flows are reduced when T-flows allow higher end-to-end delay bounds because bulk scheduling serves T-flows to meet their end-to-end deadlines which may reduce their priorities, and then N-flows can get better service.

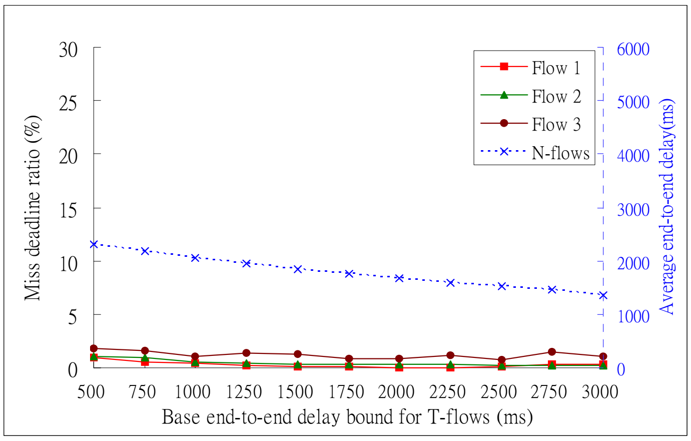

Figure 13. The performances of bulk scheduling scheme with different end-to-end delay bounds.

5.2. Multi-hop case

In the following, we evaluate the performance of bulk scheduling multi-hop flow paths to understand its behaviors in a complicated network environment. Fig. 14 shows the network topology used in this simulation, where Flows 0, 1, 2, 3, and 4 have 6, 3, 3, 1, and 1 hops, respectively, and end-to-end delay bounds of 500ms. The link capacity of each link is 1500 Kbps. Assigning a proper weight to each flow for weighted scheduling in a multi-hop case is much harder than in a single-hop case because each flow has a unique flow path and hop count and may overlap with each other. We also run a series of simulation with different weight assignments for T-flows. The results are shown in Fig 15 and Fig. 16. Table 5 lists the results of the simulation with different nodal processing index estimation methods. As described in earlier analysis, the RED-S has less impact on the lower priority flows, so it achieves better fairness among T-flows.

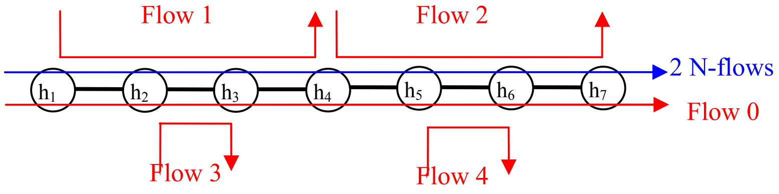

Figure 14. The network topology for multi-hop simulation.

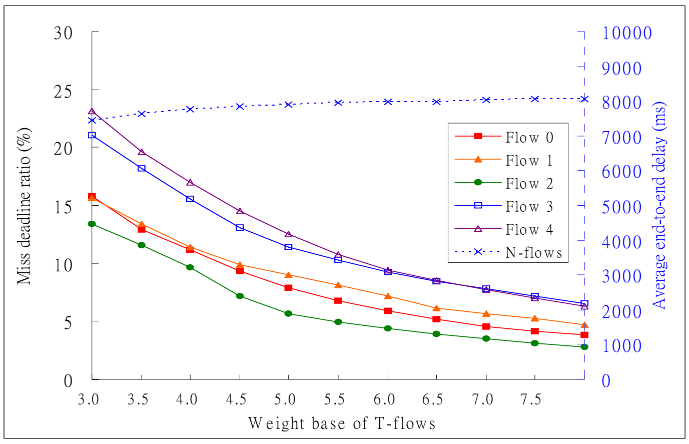

Figure 15. The performances of weighted scheduling with weights of Flow 1, 2 and 3 as (x+0.6), (x+0.3) and x in multi-hop case

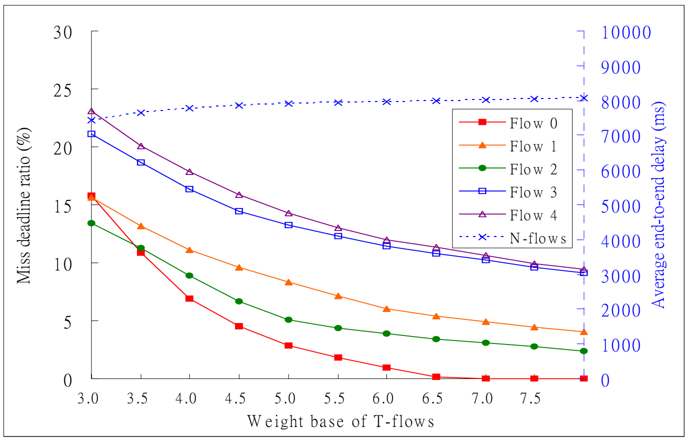

Figure 16. The performances of weighted scheduling with weights of Flow 1, 2 and 3 as (x*1.2), (x*1.1) and x in multi-hop case

Table 5. The Simulation Results of Bulk Scheduling in the Mutli-hop Case.

	GED		RED-A		RED-S	
	Avg. delay (ms)	Miss ratio (%)	Avg. delay (ms)	Miss ratio (%)	Avg. delay (ms)	Miss ratio (%)
Flow 0	287	5.53	198	1.82	238	5.22
Flow 1	168	1.61	147	0.74	153	1.29
Flow 2	156	0.98	134	0.77	137	0.91
Flow 3	239	8.42	247	10.55	241	6.24
Flow 4	243	8.25	246	8.24	240	6.05
N-flows	8322	79.92	8371	80.20	8363	80.15

We repeat the simulations used in the single-hop case under different end-to-end delay bounds and end-to-end burst periods as in single-hop case,-for the multi-hop case, and show the results of flow 0 under different nodal processing index methods and weighted scheduling (which labeled as WS), where the weight settings for Flows 0, 1, 2, 3, and 4 are 8.6, 8.3, 8.3, 8, and 8, respectively, that has the best performance in the previous weighted scheduling simulation, in Fig. 17 and 18. As expected, its performance deteriorates when the traffic is burstier, and it has a lower miss ratio when its end-to-end delay bound increases. We can see that the performances of BSS are similar to the WS with best weights assignment. Note that the flow 0 under RED-A always has better performance, but gives higher impact to other flows. Flow 0 under GED has a higher missed deadline ratio because GED spends the saved delay budget on a congested node.

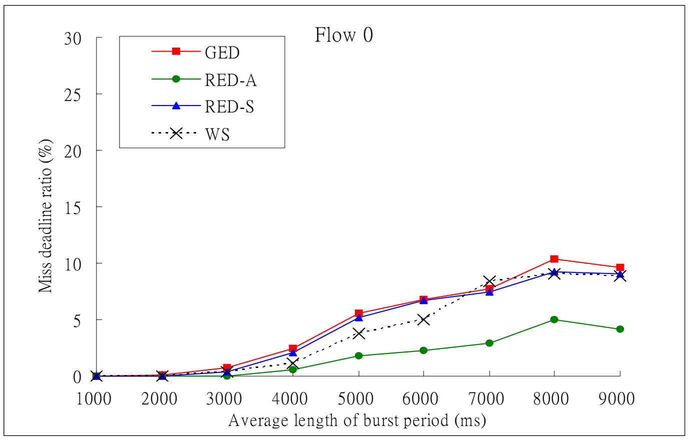

Figure 17. The performances of Flow 0 under different burst periods in multi-hop case.

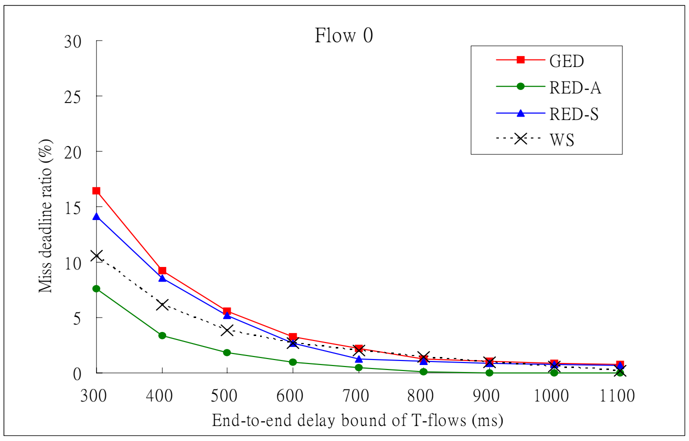

Figure 18. The performances of Flow 0 under different end-to-end delay bounds for T-flows in multi-hop case.

5.3. Performance Improvements from drop missed policy and feedback mechanism

In the previous simulations, we did not apply any enhancements, such as the drop missed policy and feedback mechanism proposed in Section 4, because the enhancements have little effect when most packets of T-flows have already met their end-to-end deadlines. In the following simulations, we show the improvement resulting from these enhancements in the network topology in Fig. 19. There are 12 T-flows (one for 6-hops, two for 3-hops, three for 2-hops and six for 1-hop) and six 1-hop N-flows, one for each link. All flows have the same flow specification as an exponential model traffic flow with a mean burst/idle period is 5000/5000 ms and the sending rate in burst period is 512 Kbps with link capacity of 1500Kbps for each link. Clearly, the network is heavily loaded. Also note that when applying drop missed policy, the dropped packets are treated as missed deadline packets in calculating the missed deadline ratio.

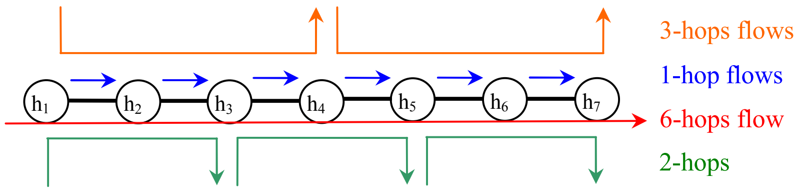

Figure 19. The network topology for the simulation in section 5.3.

Fig. 20 shows the results under different end-to-end delay bounds for all flows without applying the drop missed policy and feedback mechanism. We only list the result of the RED-S method here because all the three methods have the similar results. We can see that the missed deadline ratios of T-flows, especially the multi-hop flows, increase when end-to-end delay requirements decrease Fig. 21 is the result of applying the drop missed policy that the missed deadline ratios of all flows are reduced. We can see that the average end-to-end delay for N-flows lightly increases when end-to-end delay requirements of T-flows increase from 500ms to 2000ms. This is due to fewer TED packets being dropped for relaxed delay requirement. Fig. 22 shows the results with only applying 1-stage feedback mechanism (ACK_{th}=0). In this figure, 1-hop T-flows have large improvement in the missed deadline ratio but others deteriorate slightly because the 1-stage feedback mechanism only resolves one deferment in the flow path. Fig. 23 shows the results with the 3-stage feedback mechanism (ACK_{th}=0). We can see that the performances of 2-hop flows also improves substantially but 3-hop flows have less improvement because the bandwidth resources are exhausted after resolving two deferments. The results of bulking scheduling with the drop missed policy and feedback mechanism (ACK_{th}=0) are the same as Fig. 21, since if all missed deadline packets are dropped, then the egress node will not receive any missed deadline TED packets and the feedback mechanism has no use.

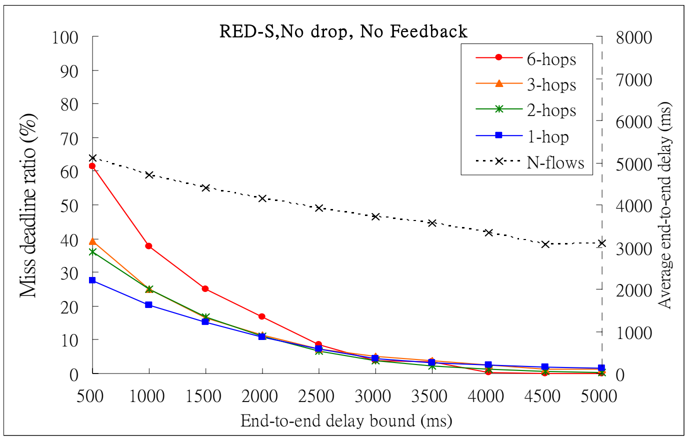

Figure 20. The performances of BSS without any enhancement under different end-to-end delay bounds for T-flows.

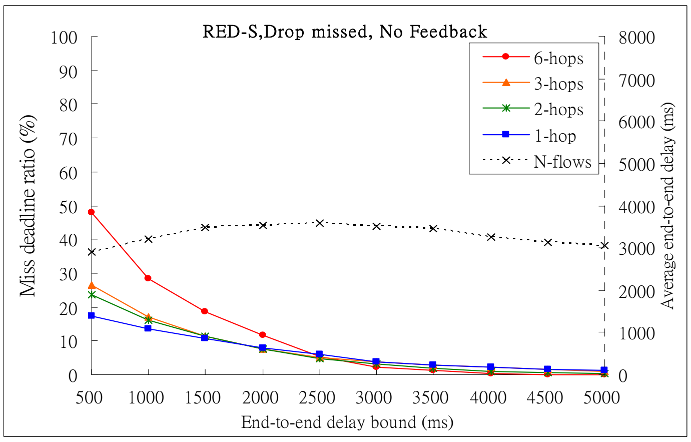

Figure 21. The performances of BSS with drop missed policy under different end-to-end delay bounds for T-flows.

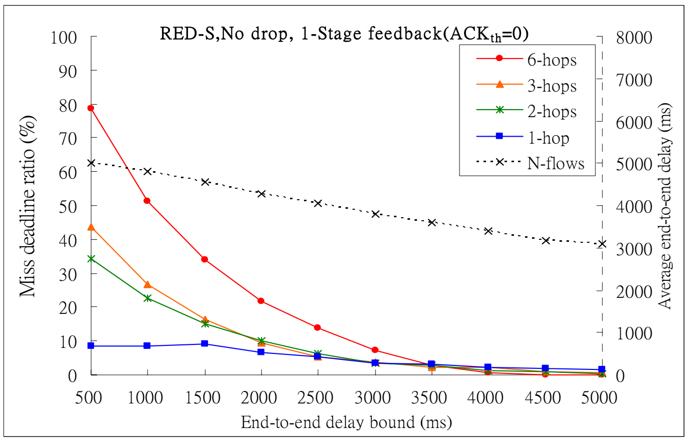

Figure 22. The performances of BSS with 1-stage feedback mechanism under different end-to-end delay bounds for T-flows.

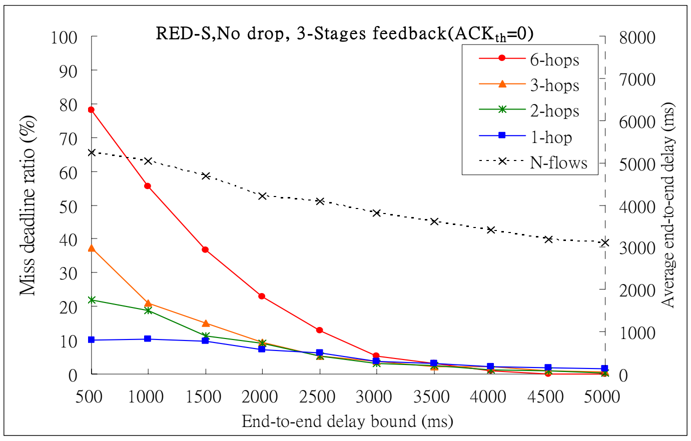

Figure 23 The performances of BSS with 3-stages feedback mechanism under different end-to-end delay bounds for T-flows.

In our feedback mechanism, the feedback threshold, ACK$_{th}$, determines the timing and frequency of feedbacks. In the previous simulation, we set ACK$_{th}$ to zero, which means the feedbacks are sent only when a TED packet has missed its deadline. A higher feedback threshold means the egress router will send feedbacks earlier before a missed deadline occurs. Let the X_α denote the missed deadline ratio when ACK$_{th}$=α, we estimate the improvement of feedback with ACK$_{th}$=α as follows:

$$improvement(\alpha) = (X_0 - X_\alpha) / X_0 \tag{41}$$

In the following simulation, we use the 6-stages feedback mechanism with different ACK$_{th}$ values, which are 0.1, 0.2 and 0.3. Fig. 24 shows the results under three different link capacities 1500, 1700 and 1900kbps. In a heavy loaded network environment, such as the 1500Kbps link capacity, the 1-hop and 2-hops flows have greater improvements because they at most have one or two deferments during transmission and can speedup quickly that results in 3-hops and 6-hops flows have less chance to get enough bandwidth to meet their end-to-end deadlines. When the network load is lighter, such as with 1700Kbps link capacity, we can see that the 3-hops flows also have some improvements; however, 6-hops flows suffer from insufficient bandwidths. And when the link capacity is 1900Kbps, all flows improve with non-zero feedback thresholds. These results also show that a higher feedback threshold usually, but not always, performs better on those flows that have shorter path length because such flows can obtain bandwidth faster. However, this causes other flows that have longer path lengths to have worse performances. Note that all flows have larger levels of improvement when the network has more link capacity because of the smaller base of improvements, i.e., X_0.

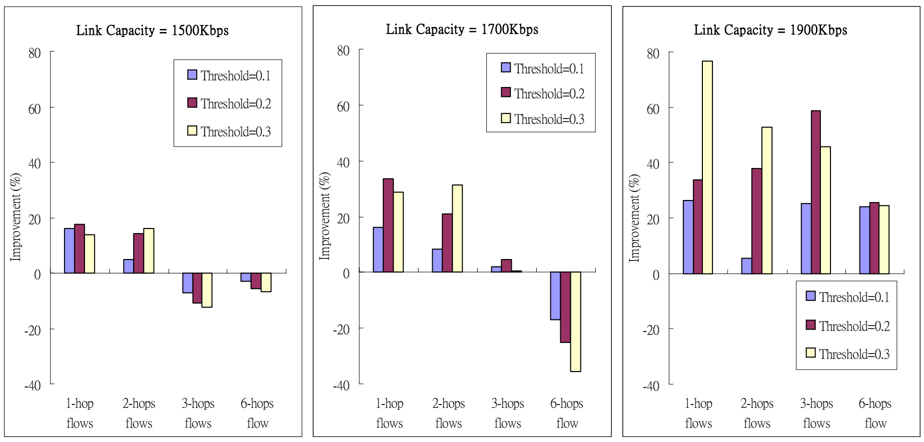

Figure 24. The feedback mechanism with different ACK_{th} when link capacities are 1500, 1700 and 1900Kbps.

6. Conclusion

To summarize, the bulk scheduling scheme with three different nodal processing index estimation methods provides a novel framework to guarantee the end-to-end delay bound for real-time flows with variable-bit-rate traffic. We have proved that if the end-to-end delay of TED packets is guaranteed, the end-to-end delay of all packets is also guaranteed. In the proposed design, the TED packet only contains the residual end-to-end delay bound and residual hops count that the TED packet size can be less than 50 bytes. If we insert TED packets every fifty milliseconds, the overhead is about 1 KBps and very affordable. We performed extensive simulations to fully examine the characteristics of the bulk scheduling scheme under various transmission environments. The results show that the bulk scheduling scheme can generally provide better end-to-end delay guarantees than rate based scheduling algorithms. In addition, it is difficult to assign proper weights to the flows in a rate based system.

Using the dropping policy, which only needs to consider the information stored in the TED packets, the bulk scheduling scheme provides even better performance. We have shown that the last-deferment intermediate node has a major adverse effect in supporting end-to-end delay bound. This means that the last-deferment condition is the first thing to resolve to improve end-to-end delay. The proposed feedback mechanism aims to discover and resolve the last-deferment condition and further improve the performance. The simulation results in Fig. 22 and Fig. 23 show the effectiveness of the proposed feedback mechanism. The timing to start the feedback mechanism is determined by the ACK_{th}, and as we can see from Fig. 24, an early start of feedback mechanism gives the flow advantage in competing for bandwidth and helps in reducing missed deadline ratio. In a congested network, the competition apparently favors flows with fewer hops. However, the giving of an ACK_{th} value for real-time flow and its effects to improve end-to-end delay still needs further studies

References

[1] S. Jamaloddin Golestani, "Congestion-free communication in high-speed packet networks," in *IEEE Transactions on Communications*, pp. 1802-1812, December 1991.

[2] D. Clark, S. Shenker, and L. Zhang, "Supporting real-time applications in an integrated services packet network: Architecture and mechanism," in *Proceedings of ACM SIGCOMM '92*, pp. 14-26, Aug. 1992.

[3] H. Zhang and D. Ferrari, "Rate-controlled static-priority queueing," in *Proceedings of INFOCOM'93*, pp. 227-236, March 1993

[4] S. Floyd and V. Jacobson, "Link-sharing and resource management models for packet networks," in *IEEE/ACM Transactions Networking*, vol. 3 pp. 365-386, Aug 1995.

[5] R. Braden et al., "Resource reservation protocol (RSVP) – version 1 functional specification," *IETF RFC 2205*, Sept. 1997.

[6] ATM User-Network interface (UNI) signaling specification version 4.1, ATM Forum, April 2002.

[7] G. Kesidis, J. Walrand and C.S. Chang, "Effective bandwidths for multiclass Markov fluids and other ATM sources," in *IEEE/ACM Transactions on Networking*, Vol. 1, pp. 424-428, 1993.

[8] D. Ferrari and D. Verma, "A scheme for real-time channel establishment in wide-area network, " in *IEEE Journal on Selected Areas in Communications*, pp. 368-379, April 1990.

[9] T. M. Chen, S. S. Liu, and V. K. Samalam, "The available bit rate service for data in ATM networks," in *IEEE Communication Magazine*, pp. 56-71, May 1996.

[10] Traffic control and congestion control in B-ISDN. *ITU-T Rec. 1.371*. No;. 6-14, 1995.

[11] G. G. Xie and S. S. Lam, "Real-time block transfer under a link-sharing hierarchy," in *IEEE Transactions on Networking*, vol. 6, no. 1, Feb. 1998.

[12] A. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The multiple node case, " in *IEEE/ACM Transactions on Networking*, vol. 2, pp. 137-150, Apr. 1994.

[13] L. Georgiadis, R. Guerin, V. Peris, and K. N. Sivarajan, "Efficient network QoS provisioning based on per node traffic shaping," in *IEEE/ACM Transactions on Networking*, vol. 4, pp. 482-501, Aug 1996.

[14] I Stoica, H. Zhang, S. Shenker, R. Yavatkar, D. Stephens, A. Malis, Y. Bernet, Z. Wang, F. Baker, J. Wroclawski, C. Song and R. Wilder, "Per hop behaviors based on dynamic packet states," *Internet Draft*, http://www.cs.cmu.edu/~istoica/DPS/draft.txt.

[15] S. Kent and R. Atkinson, "Security architecture for the Internet protocol," *IETF, RFC 2401*, Nov. 1998.

[16] K. Zhu, Y. Zhuang and Y. Viniotis, "Achieving end-to-end delay bounds by EDF Scheduling without traffic Shaping," in *Proceedings of IEEE INFOCOM 2001*, vol. 3 , pp. 1493 -1501, April 2001.

[17] A. Vagish, T. Znati and R. Melhem, "Per-node delay assignment strategies for real-time high speed network," in *Proceedings of Globecom'99*, pp. 1323-1327.

[18] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control -- The single-node case," in *IEEE/ACM Transactions Networking*, vol. 1, no. 3, pp. 344-357, Jun. 1993.

[19] J. Bennett and H. Zhang, "WF2Q: Worst-case fair weighted fair queueing," in *Proceedings of IEEE INFOCOM 1996*, pp.120-128, Mar. 1996.

[20] M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," in *Proceedings of ACM SIGCOMM'95*, pp. 231-242, Aug. 1995.

[21] L. Delgrossi and L. Berger. (1995). "Internet Stream protocol version 2 (ST2), protocol specification – version ST2+," *RFC 1819*, Internet Engineering Task Force, August 1995

[22] The network simulator NS (version 2), http://www.isi.edu/nsnam/ns/