On the Delay Bound of Youngest Serve First (YSF) Aggregated Packet Scheduling

TR-ANL-2002-03-04

Li Zhu, Gang Cheng and Nirwan Ansari

March 2002

Electrical and Computer Engineering
New Jersey Institute of Technology
University Heights
Newark, NJ 07029
U.S.A
On the Delay Bound of Youngest Serve First (YSF) Aggregated Packet Scheduling

Li Zhu, Gang Cheng, Nirwan Ansari
Advanced Networking Lab
Elec. And Comp. Enging. Dept.
New Jersey Institute of Technology
University Heights
Newark, NJ 07102, USA

Please address all correspondence to:

Professor Nirwan Ansari
Department of Electrical and Computer Engineering
New Jersey Institute of Technology
University Heights
Newark, NJ 07102

tel: 973-596-3670
fax: 973-596-5680
email: nirwan.ansari@njit.edu

1 This work has been supported in part by the New Jersey Commission on Higher Education via the NJI-TOWER project, and the New Jersey Commission on Science and Technology via the NT Center for Wireless Telecommunications.
Abstract: In this paper, we propose a simple scalable aggregated traffic scheduling scheme, called the Youngest Serve First (YSF) algorithm. We show analytically that YSF can provide bounded end-to-end delay time with high link utilization that may not be possible for the FIFO scheme.

1 Introduction

For the past decade, how to provide end-to-end QoS guarantees has drawn much attention. Conventional solutions to this problem are rooted on per flow based scheduling schemes [1][2]. The most significant drawback of these approaches is the lack of scalability, which could hamper the provision of end-to-end QoS guarantees in large ISPs [3]. Recently, aggregated packet scheduling has been the focus of research as a possible alternative for providing a scalable approach to end-to-end QoS guarantees. In particular, Differentiated Services Working group has proposed RFC 2598 [4], which defines Expedited Forwarding Per Hop Behavior (EF PHB). In this approach, EF traffics, which are regulated at the network edge, share a single FIFO buffer and are scheduled in the aggregated manner in the core network. FIFO packet scheduling is one of the most attractive approaches because of the implementation simplicity.

EF PHB aims to guarantee bandwidth in both large and small time scale, but whether it can guarantee end-to-end QoS still remains unclear. It was believed that the end-to-end QoS in the network could be guaranteed if the link utilization is kept small enough (i.e., less than 50%). Recent studies [3] [5] show that the worst-case end-to-end delay bound through the network is \( H \beta / \{1 - (H - 1)\alpha \} \), where \( H \) is the number of hops along the longest path of all the flows in the network, \( \alpha \) is the link utilization, and \( \beta \) is the constraint on the burstiness of all flows through one link. It is clear that the worst-case delay is bounded only when \( \alpha < 1/(H - 1) \). Thus, the
provisioning power of traffic aggregation is significantly weakened. The reason behind the difficulty of obtaining bounded end-to-end delay for an arbitrary network topology is that in aggregated scheduling, packet delay not only depends on the traffic behavior of the flows sharing the same queue, but also on the traffic patterns in the whole network, even those occurred long time ago [3].

In this paper, we propose a new aggregated packet scheduling algorithm: Youngest Serve First (YSF). In YSF, traffics are shaped at the network edge. We use a label value to indicate the packet state information and encode it in a certain field in each packet header. As packets travel through the network, the encoded information is updated. All the packets are scheduled based on the information carried in the header. Our approach not only has low computational complexity, but it also needs very limited number of bits ($\log_2 H$) to carry the label in the packet headers. Most importantly, it can provide bounded end-to-end delay for any $\alpha < 1$ and any $H$.

The remainder of this paper is organized as follows. In Section 2, we introduce our network model and terminology, and review the related background. In Section 3, using Network Calculus Theory [6][7], we present our YSF scheme and derive its end-to-end delay bound. Finally, discussion and conclusions are given in Section 4.

2 Network Model, Terminology and Background

We assume that all network nodes perform the same packet-scheduling algorithm based on the limited information carried in the packet headers. Before entering the network, each traffic flow $k$ is shaped at the network edge to conform to a token bucket with parameters $(r^k, \beta^k)$, which is the traffic arrival curve satisfying $A^k(t_0, t_0 + t) \leq r^k t + \beta^k$, where $A^k(t_0, t_0 + t)$ is the total traffic from flow $k$ released to the network during time interval $[t_0, t_0 + t]$. Denote $F$ as the
set of flows traversing node \( S \). We assume that for every \( F \) and \( S \), the following conditions hold [3][5]:

\[
\sum_{k \in F} r^k \leq \alpha C, \tag{1}
\]

and

\[
\sum_{k \in F} \beta^k \leq \beta C, \tag{2}
\]

where \( \alpha(<1) \) is the link utilization factor, \( \beta \) is the constraint on the burstiness of all flows through \( S \), and \( C \) is the outgoing link capacity of \( S \). According to [3], \( \beta \) is linearly dependent on \( \alpha \), so we set \( \beta = \tau_0 \alpha \), where \( \tau_0 \) is a constant. In this paper, we adopt the fluid traffic model, but our work can be easily extended to the packet traffic model. The effect of propagation delay is also assumed negligible. The following notations are adopted: \( d_j \) represents the maximum delay (worst-case) experienced by packet \( p \) at the \( j \)th node along its path from the source to the destination, and \( D_j \) represents the maximum total delay (worst-case) experienced by packet \( p \) from the first node to the \( j \)th node (inclusive) along the path. Thus, we have \( \sum_{j=1}^{i} d_j = D_j \).

Next, we briefly review some basic conclusions from Network Calculus Theory [6][7]. For simplicity, we use \( A(t) \) to replace \( A(t_0, t_0+t) \) as the total traffic arrival curve. Denote \( S(t) \) as the traffic service curve, which, in our case, is \( S(t) = Ct \). The amount of packets stored at each node is at most \( B \), which is given by

\[
B = A \otimes S(0) \tag{3}
\]

where \( \otimes \), the deconvolution, is defined by

\[
A \otimes S(t) := \sup_{\tau \in \mathbb{R}} \{ A(t+\tau) - S(\tau) \}. \tag{4}
\]

If the total traffic arrive curve is

\[
A(t) = \alpha Ct + \beta C, \tag{5}
\]
it can be verified that

\[ B = \beta C. \]  

(6)

In Figure 1, \( B \) is the maximum amount of packets stored, which is \( \beta C \) as mentioned above. \( t_n \), the maximum queuing delay, is \( \beta \) in our case if the FIFO scheduling is adopted. \( t_d \) is the maximum burst length or the longest time for the system to clear the queue, as long as work conserving scheduling algorithms are adopted. \( t_d \) can be obtained by solving the following equation:

\[ S(t_d) = A(t_d). \]  

(7)

3 The Youngest Serve First (YSF) Algorithm

We assume the edge node is the first hop for all traffics. Our proposed YSF algorithm works as follows: before entering the network, each packet is labeled with the number one, 1; at each node, the packet with the smallest label value is served first, packets with the same label value are processed in the FIFO manner; the label on each packet is increased by one right before they are transmitted. The label value of a packet indicates the time it has spent in the network, or more precisely, the number of hops it has traversed through the network. That is why our scheme is called Youngest Serve First (YSF).

Naturally, the worst-case delay bound is experienced by packets with \( H \) hops from their source to the destination; \( H \), as defined earlier, is the number of hops along the longest path of all the flows in the network. Consider a packet \( p \). We can see that at different nodes along the path, \( p \) experiences different delay bound. At the first hop, \( p \) has the highest priority due to the smallest label value carried in its header. After \( p \) is transmitted from the first hop node, the label value is increased to two at the second hop node. Thus, \( p \) cannot receive service as long as there are packets with label value one in that node. Intuitively, the longer the packet stays in the network,
the larger the maximum delay it will experience at each node. In other words, \( d_i \leq d_j \), for \( 1 \leq i \leq j \leq H \). For the rest of this paper, traffics are grouped based on the label carried in packet headers.

**Definition 1** Let \( F_j \) denote the set of flows, which assume their \( j^{th} \) hop at node \( S \).

Define \( \rho' \) and \( \sigma' \) as follows

\[
\rho^j = \sum_{i \in F_j} \rho^i, \quad (8)
\]

\[
\sigma^j = \sum_{i \in F_j} \sigma^i. \quad (9)
\]

Thus, (1) and (2) can be rewritten as:

\[
\sum_k \rho^k \leq \alpha C \quad (10)
\]

\[
\sum_k \beta^k \leq \beta C. \quad (11)
\]

Next, we will justify our intuition and derive the end-to-end delay bound with respect to the link utilization \( \alpha \), the value of \( H \), and the burstiness constraint \( \beta \).

**Lemma 1** The maximum delay experienced by packet \( p \) at its first hop in the network is \( d_1 = \beta \) (is also \( D_1 \)).

**Proof:** Suppose the first hop node is \( S \). Since packet \( p \) has the smallest label value one, it has the highest priority in the system. Any other packets with larger label value will not affect the service time of \( p \). As a result, \( p \) experiences the worst-case delay when \( S \) is the first hop for all the flows traversing it. In this case, all packets have the same label value or priority. Thus, the scheduling algorithm is just FIFO. According to (10) and (11), the maximum overall traffic arrival curve at \( S \) is \( A(t) = \alpha Ct + \beta C \). According to Figure 1 and the basic results of Network Calculus Theory introduced in Section II, the delay bound is \( \beta \).
Lemma 2 The maximum delay experienced by packet $p$ from its first hop node to the second hop node (inclusive) along the path is bounded by $D_2 = \beta + \frac{\beta}{1-\alpha}$.

Proof: Suppose the second hop node for $p$ is $S$. According to the analysis in Lemma 1, only packets with label value of either one or two can affect the delay time of $p$ at $S$. In order to obtain the worst-case delay bound for flow $j$ at $S$, we assume that only packets with label value one and two traverse node $S$. In other words, only packets experiencing their either first or second hop at $S$ are considered. Since packets labeled with two may have already experienced delay $D_1$, the traffic arrival curve for those flows is $\rho^2(t + D_1) + \sigma^2$ instead of $\rho^2 t + \sigma^2$ [2]. For flows with first hop at $S$, the arrival traffic curve is still $\rho^1 t + \sigma^1$. Therefore, the total arrival traffic curve at $S$ is:

$$A(t) = \rho^1 t + \sigma^1 + \rho^2(t + D_1) + \sigma^2$$

(12)

In order to get the maximum delay, we use the equalities in (10) and (11), which are:

$$\sum_{j=1}^{2} \rho^j = \alpha C$$

(13)

$$\sum_{j=1}^{2} \sigma^j = \beta C$$

(14)

Using (13) and (14), we can rewrite (12) as:

$$A(t) = \alpha(1-x)Ct + x\alpha C(t + D_1) + \beta C$$

$$= \alpha(1-x)Ct + x\alpha Ct + x\alpha CD_1 + \beta C$$

(15)

where $x = \frac{\rho^2}{\alpha C}$. As we can see, the burst size of the overall traffic is changed from $\beta C$ to $x\alpha CD_1 + \beta C$. According to the introduction in Section II, we know that when packet $p$ arrives at $S$, the maximum buffer occupation size $B$ at $S$ is $x\alpha CD_1 + \beta C$. All the packets in the buffer before $p$ arrives will be served before $p$, since their priorities are not lower than $p$; packets that are labeled with two and arrive after $p$ will not affect its departure time, because they carry the same
labels as \( p \) and will be served in FIFO order. However, those packets labeled with one, of which the arrival rate is \((1-x)\alpha C\), will affect the departure time of \( p \) even they enter the queue after \( p \).

Thus, the effective total arrival traffic curve that can determine the departure time of \( p \) is

\[
A_{\text{eff}}(t) = (1-x)\alpha Ct + (x\alpha CD_1 + \beta C)
\]

instead of (15). If we replace \( A(t) \) in Figure 1 with \( A_{\text{eff}}(t) \), \( t_d \) is the maximum delay time \( d_2 \) for \( p \) at its second hop node \( S \). Using (7), \( d_2 \) can be determined from the following equality:

\[
Cd_2 = A_{\text{eff}}(d_2) = (1-x)\alpha Cd_2 + (x\alpha CD_1 + \beta C).
\]

Thus,

\[
d_2 = \frac{x\alpha D_1 + \beta}{1-(1-x)\alpha} = \frac{(x\alpha + 1)\beta}{1-(1-x)\alpha}.
\]

Since

\[
\frac{\partial d_2}{\partial x} = \frac{-\alpha^2}{(1-(1-x)\alpha)^2} \beta < 0,
\]

when \( x = 0 \), \( d_2 \) reaches its maximum

\[
d_2 = \frac{\beta}{1-\alpha}
\]

Hence,

\[
D_2 = D_1 + d_2 = \beta + \frac{\beta}{1-\alpha}
\]

After packet \( p \) takes its second hop and moves to the \( j \)-th (>2) node, there are possibly more packets with smaller label values or higher priorities at that node. Intuitively, \( p \) could experience longer delay at those nodes. Next, in Theorem 1, we derive the delay bound as \( p \) moves closer to its destination.
**Theorem 1** The maximum delay experienced by packet \( p \) from its first hop node to the \( k^{th} \) hop node (inclusive) along the path is bounded by

\[
D_k = D_{k-1} + \frac{\beta + \alpha D_{k-2}}{1-\alpha}
\]  

for any \( k \geq 3 \).

**Proof:** We will use induction to complete the proof. We can reasonably define \( D_0 = 0 \). From Lemma 1 and Lemma 2 we have

\[
D_2 = D_1 + d_2 = \beta + \frac{\beta}{1-\alpha} = D_1 + \frac{\beta + \alpha D_0}{1-\alpha}
\]  

(23)

Assume

\[
D_k = D_{k-1} + \frac{\beta + \alpha D_{k-2}}{1-\alpha}
\]  

(24)

We shall next show the above expression holds for \( k+1 \). Let node \( S \) be the \((k+1)^{th}\) hop of packet \( p \), only packets with label not larger than \( k+1 \) can affect the delay of \( p \) at \( S \). In other words, only packets assuming their \( j^{th} \) (\( j \leq k+1 \)) hop at node \( S \) will be considered. The overall arrival traffic curve can be written as:

\[
A(t) = \sum_{j=1}^{k+1} \{\rho^j(t+D_{j-1}) + \sigma^j\}
\]  

(25)

Define \( x_j = \frac{\rho_j}{\alpha C} \), and also note that \( \sum_{j=1}^{k+1} x_j = 1 \). Then we can rewrite (25) as:

\[
A(t) = \sum_{j=1}^{k+1} \{\rho^j(t+D_{j-1}) + \sigma^j\} = \sum_{j=1}^{k+1} \rho^j(t+D_{j-1}) + \sum_{j=1}^{k+1} \sigma^j = \sum_{j=1}^{k+1} x_j \alpha C(t+D_j) + \beta C
\]

\[
\leq x_{k+1} \alpha Ct + x_{k+1} \alpha CD_k + \sum_{j=1}^{k} x_j \alpha Ct + \sum_{j=1}^{k} x_j \alpha D_{k-1} + \beta C
\]

\[
= x_{k+1} \alpha Ct + x_{k+1} \alpha D_k + (1-x_{k+1}) \alpha Ct + \sum_{j=1}^{k} x_j \alpha CD_{k-1} + \beta C
\]
\[ x_{k+1}^\alpha C t + (1-x_{k+1})^\alpha C t + [x_{k+1}^\alpha CD_k + (1-x_{k+1})^\alpha D_{k-1} + \beta C] \] (26)

The inequality in (26) is based on the fact that \( D_i < D_{k-1} \) if \( i < k-1 \). The third term in the last equality stands for the maximum traffics queued in the system when packet \( p \) joins the queue. The first term can be viewed as the traffics which assume the \((k+1)\)th hop at node \( S \), and arrive after \( p \), and thus cannot affect \( p \)'s departure time. The second term represents the traffics with smaller label values, which also arrive after \( p \), that they can affect the departure time of \( p \). Using the similar argument in the proof of Lemma 2, the effective arrival traffic curve is

\[ A_{eff}(t) = (1-x_{k+1})^\alpha C t + x_{k+1}^\alpha D_k + (1-x_{k+1})^\alpha D_{k-1} + \beta C \] (27)

By solving for \( d_{k+1} \) in the following equation

\[ Cd_{k+1} = A_{eff}(d_{k+1}) = (1-x_{k+1})^\alpha C d_{k+1} + x_{k+1}^\alpha CD_k + (1-x_{k+1})^\alpha CD_{k-1} + \beta C \] (28)

we get

\[ d_{k+1} = \frac{x_{k+1}^\alpha D_k + (1-x_{k+1})^\alpha D_{k-1} + \beta}{1-(1-x_{k+1})^\alpha} \] (29)

Thus,

\[ \frac{\hat{c}d_{k+1}}{\hat{c}x_{k+1}} = \alpha \frac{[(1-\alpha)D_k - D_{k-1} - \beta]}{(1-(1-x_{k+1})^\alpha)^2} = \frac{\alpha [(1-\alpha)(D_{k-1} + \frac{\alpha D_{k-2} + \beta}{1-\alpha}) - D_{k-1} - \beta]}{(1-(1-x_{k+1})^\alpha)^2} \]

\[ = \frac{\alpha(D_{k-2} - D_{k-1})}{(1-(1-x_{k+1})^\alpha)^2} < 0. \] (30)

(24) has been used to reach the third equality in (30). So, when \( x_{k+1} = 0 \), \( d_{k+1} \) reaches its maximum value. From (29),

\[ d_{k+1} = \frac{\alpha D_{k-1} + \beta}{1-\alpha} \] (31)

and

\[ D_{k+1} = D_k + d_{k+1} = D_k + \frac{\alpha D_{k-1} + \beta}{1-\alpha}. \] (32)
Using the recursive relationship in (32) with the initial conditions stated in Lemma 1 and Lemma 2, we have

\[ D_j = \frac{\beta}{\alpha} \left\{ \frac{r_2 - \alpha - 1}{r_2 - r_1} r_1^j + \frac{r_2 - \alpha - 1}{r_1 - r_2} r_2^j - 1 \right\} j \geq 3 \]  

(33)

where \( r_{1,2} = \frac{1 - \alpha \pm \sqrt{-3\alpha^2 + 2\alpha + 1}}{2(1 - \alpha)} \) are the roots of the following quadratic equation

\[ r^2 - r - \frac{\alpha}{1 - \alpha} = 0. \]  

(34)

It can be shown from (33) with further algebraic manipulation that \( D_H \propto H \beta \) when \( \alpha \) is very small, and \( D_H \propto (1 - \alpha)^{H/2} \) when \( \alpha \approx 1 \). Based on the above analysis, we can conclude that the link utilization, which is independent of \( H \), can approach one, and the end-to-end delay is also bounded at the same time. Figure 2 shows the performance comparison between YSF and FIFO with \( H = 10 \). The vertical axis is the number of time units (in terms of \( \tau_0 \)). With a given link utilization \( \alpha \), YSF performs much better than FIFO, especially when \( \alpha \) is large.

It may seem very natural to adopt another scheme, namely Oldest Serve First (OSF), in which the packets with the largest label value are served first instead of being served last in YSF. Next, we will show that YSF performs better than OSF.

**Lemma 3** In OSF, for any packet \( p \) with \( H \) hops to its destination, the worst-case delay experienced till its \((H - 1)^{th}\) hop (inclusive) is \( D_{H-1} = \frac{(H - 1)\beta}{1 - \alpha H} \).

**Proof:** In order to get the worst-case delay, we assume that the node \( S \) is the \( k^{th} \) (\( 1 \leq k \leq H - 1 \)) hop node along \( p \)'s path to its destination, and all packets from other flows assume their \( H^{th} \) hop at \( S \). According to OSF, packet \( p \) has the lowest priority and can only be served when no other
packets are in the corresponding node. Since other flows may all experience the maximum delay $D_{H-1}$, the total arrival traffic curve is

$$A(t) = \alpha C(t + D_{H-1}) + \beta C. \quad (35)$$

Then, $d_k$, the delay bound of $p$ experienced at the $k^{th}$ node can be obtained by solving the following equation:

$$S(d_k) = Cd_k = A(d_k) = \alpha C(d_k + D_{H-1}) + \beta C. \quad (36)$$

Thus, we get

$$d_k = \frac{\alpha D_{H-1} + \beta}{1 - \alpha}. \quad (37)$$

On the other hand, we also have

$$D_{H-1} = \sum_{k=1}^{H-1} d_k = (H-1) \frac{\alpha D_{H-1} + \beta}{1 - \alpha}. \quad (38)$$

Hence,

$$D_{H-1} = \frac{(H-1)\beta}{1 - \alpha H} \quad (39)$$

From Lemma 3, we can see that $D_{H-1}$ is bounded only when $\alpha < \frac{1}{H}$. Since $\alpha$ can approach one in YSF, YSF achieves higher link utilization. From (39), we also see that in OSF, when $\alpha$ approaches one, the end-to-end delay bound goes to infinity. However, in YSF, the end-to-end delay bound always remains finite as long as $\alpha < 1$. Thus, YSF also performs better than OSF in terms of the end-to-end delay bound.

4 Discussion and Conclusions

In this paper, we have proposed a new aggregated traffic scheduling scheme, Youngest Serve First (YSF), and derived its end-to-end delay bound. YSF has been proven to have the following
merits. First, the link utilization $\alpha$ in YSF can approach one, regardless of the network topology and the value of $H$; second, the end-to-end delay bound in YSF is much smaller than that in FIFO, and thus better end-to-end delay bound can be guaranteed. Even with an additional required complexity, which is rather low, the above merits warrant YSF preferable over FIFO. At each node, there are $H$ different label values. Thus, we need at most $H$ queues, which correspond to different label values. Packets are placed into different queues based on their label values and the backlogged queue with the smallest label value is served first. Therefore, YSF is scalable because we only need to manage $H$ queues, no matter how many flows traverse each node. We also note that only $\log_2 H$ bits are required to encode the label. We may use either the TTL field or the TOS field in the IP header to realize YSF, but this issue is beyond the scope of this paper. Ideas from timestamp based scheduling algorithm such as WFQ [3] and WF$^{2}$Q [5] may be incorporated in designing aggregated traffic scheduling scheme. However, YSF possesses the following merit not shared by the timestamp-based approaches: all the nodes in the network need not be synchronized in time, and timestamps need not be computed and updated.

References


Figure Captions:

Fig. 1 Illustration of the basic concept of Network Calculus Theory.

Fig. 2 Performance comparison between YSF and FIFO.
Figure 1
Performance Comparison: YSF vs. FIFO

Figure 2