# Implementation of Packet Scheduling With Buffer Management to Improve Network Performance

Dr. R. Shanmugasundaram [#1], V. Azhaharasan [#2], Dr. S. Pannirselvam [*3]

[#1, *3] *Associate Professor, Department of Computer Science,*

[#2] *Assistant Professor, Department of Information Technology,*

[1, 3] *Erode Arts & Science College (Autonomous),*

*Rangampalayam - 638009, Erode District, Tamil Nadu, India.*

[2] *Gobi Arts & Science College (Autonomous),*

*Gobichettipalayam – 638 453, Erode District, Tamil Nadu, India.*

[1] rsaravinth@sancharnet.in, [2] azhahau.ind@gmail.com, [3] pannirselvam_s@yahoo.com

*Abstract-* **In the wireless networks, achieving efficient communication without packet loss is an important problem. In the network data frames are classified into packets before sending the network. In the streaming applications the packets are dependent to each other. So, if there is a drop in single packet there is no use in the whole sequence. Due to network traffic, all packets of a frame reach to the destination is a vital problem. If we retransmit the packets produce the delay constraints and increases the cost. The main objective of this paper is to reduce the packet delay and increases the packet delivery ratio in the streaming applications. So, we use buffer management algorithms to carry out the packet discard. By using the concept of packet scheduling we consider both packet delay and also the packet size. So, this can enhance the flexibility of network resource management and improvement in quality of service.**

*Keywords-* **Packets, Scheduling, Buffer Management, Network.**

## I. INTRODUCTION

Wireless network refers to any form of computer network that utilizes some form of wireless network connection. It is a process by which homes, telecommunications networks and enterprise installations avoid the costly process of introducing cables into a building or as a connection between various equipment locations.

Wireless telecommunications networks are generally implemented and administered using radio communication. This implementation takes place at the physical level of the OSI model network structure.

### *A. Significance of Wireless network*

The Wireless networks provide an reasonably priced and simple way to share a single Internet connection among a number of computers. Sharing data is far easier with a wireless network once a network drive is installed. The network drive can be an external hard drive hooked up to the network hub. Once set up all computers have access to storage space on the drive. This allows you to save important documents onto the drive form one computer, then access the information on a second computer.

The wireless networks also lets you right to use files and printers from anywhere in your home. It allows you to coordinate files you have on your laptop with your home computer, and you can easily throw files between computers as well. Using a wireless network to transfer files is faster than transfer them via e-mail or burning them to a CD.

Because printers connected to one of the computers on a network are shared by all the computers on that network, you can write documents anywhere in your home, press the ‘print' button, and collect the printed files from a printer that is connected to another computer.

### *B. Structural design of Wireless network*

The wireless communication revolution is bringing fundamental changes to data networking, telecommunication, and is making integrated networks a reality.

A wireless local-area network (LAN) uses radio waves to connect devices such as laptops to the Internet and to your business network and its applications. When you connect a laptop to a WiFi hotspot at a cafe, hotel, airport lounge, or other public place, you're connecting to that business's wireless network.

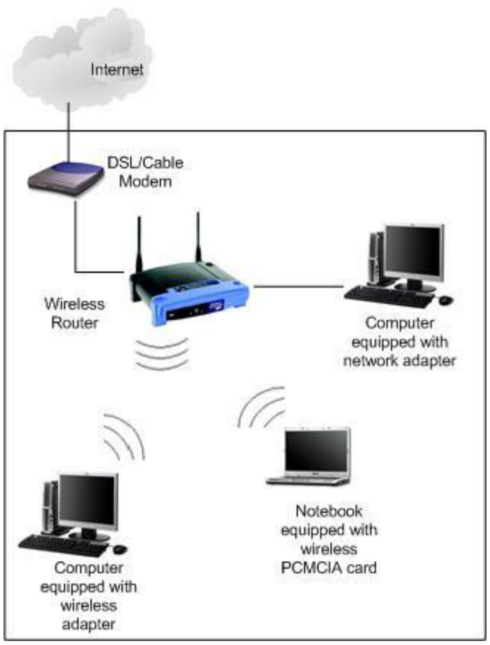


Fig 1: Wireless Network

Wireless technologies employ radio waves and/or microwaves to maintain communication channels between computers. Now that this technology has been developed and continues to be developed since certainly not everything has the Internet on it, the question becomes “How fast can we get this data to a portable device.

## II. BACKGROUND OF THE STUDY

To enhance the reliability of the transmission in the streaming applications we propose buffer management algorithms. Because in the multimedia applications the data frames are divided into packets. Their packets are interdependent structure. Due to the dependency, if there is a loss in single packet the whole sequence of video application is useless.

### *A. Existing System*

In the existing system in order to overcome the issues of packet losses in the multimedia applications we provide two guidelines for designing buffer management algorithms, and demonstrate their effectiveness. Actually in the wireless medium, before transmission across the network the data frames are split into several smaller sized packets. In the streaming applications the packets are dependent to each other.

The receiving side can make use of the data only if it receives all packets of a frame. The problem of ensuring that all packets of a frame arrive at the destination is crucial one. when considering real-time traffic, such as streaming multimedia traffic, where retransmission of missing packets is not feasible due to delay constraints posed by the application and also to increase the cost.

## III. PROBLEM STATEMENT

In the wireless networks, achieving efficient communication without packet loss is an important problem. In the network data frames are classified into packets before sending the network. In the streaming applications the packets are dependent to each other.

So, if there is a drop in single packet there is no use in the whole sequence. Due to network traffic, all packets of a frame reach to the destination is a vital problem. If we retransmit the packets produce the delay constraints and increases the cost.

The main reason for packet loss in networks are buffer overflows due to congestion. For data streams with packet dependencies, we must differentiate between the packet-level throughput, i.e., the amount of data delivered in terms of packets, and the effective good put, i.e., the amount of data that can be decoded effectively at the receiving end. These two measures can be drastically different, e.g., the throughput may be high, while its good put is very low.

## IV. PACKET SCHEDULING

Packet scheduling is the means by which data (packet) transmission is achieved. The packet scheduler is the traffic control module that regulates how much data an application (or flow) is allowed, essentially enforcing QOS parameters that are set for a particular flow.

The packet scheduler incorporates three mechanisms in its scheduling of packets:

- A conformer
- The packet shaper
- A sequencer

The conformer and sequencer are discussed in more detail in the traffic control documentation. Since the packet scheduler's role is essential to overall traffic control understanding.

The packet scheduler considers the classification provided by the Generic Packet Classifier (GPC), and provides preferential treatment to higher-priority traffic. Consequently, the packet scheduler is the first step (in a sequential view) to ensuring that the prioritized network transmission of packets begins with data that has been deemed most important.

*A. Packet Scheduling Methods*

VPS-TWP: Throughput normalized waiting time priority scheduling

Generally, packets from a class have various sizes and different classes may have different packet size distributions. We describe VPS-TWP as the throughput normalized waiting time priority scheduling for FBS-DD provisioning by further taking the packet size into considerations. At the beginning of scheduling, $TWP_i = \infty$ for $1 \leq i \leq N$. Suppose that class *i* is backlogged at time *t*, $S_i$ (t) is the size of the packet at the head of the class *i* at *t*, and $W_i$ (t) is the waiting time of the packet at the head of the class *i* at *t*. We define the throughput normalized head waiting time of class *i* at *t* as

$$TWP_i(t) = \frac{w_i(t)}{\delta_i s_i(t)}$$

Every time a packet is to be transmitted, the VPS-TWP scheduler selects the backlogged class *j* with the maximum throughput normalized head waiting time,

$$j = \arg\max_{i \epsilon G(t)} TWP_i(t),$$

where *G*(*t*) is the set of backlogged classes at time *t*. Tie is broken by the use of the differentiation priority. The throughput of class *j* is increased by the size of the transmitted packet. Its throughput normalized head waiting time will be minimized as its packet delay will not increase any more. VPS-TWP attempts to minimize the differences between the bandwidth normalized waiting times of successively departing packets. It essentially aims to achieve instantaneous FBS-DD.

VPS-TAD: Throughput normalized average delay scheduling

While VPS-TWP focuses at the instantaneous behavior, we propose the VPS-TAD scheme which focuses on the long-term behavior.

$$\frac{D_i(T, T+t)}{\delta_i B_i(T, T+t)} = \frac{D_j(T, T+t)}{\delta_j B_j(T, T+t)}.$$

That is, the throughput normalized average delay (TAD) factor must be equal in all classes, i.e., $TAD_i = TAD_j$.

Note that given a same interval, bandwidth sharing ratio of two classes is the same as the throughput ratio. The VPSTAD scheme, tailored from PAD , aims to equalize the throughput normalized average delays among all classes so as to achieve the FBS-DD goal. At the beginning of scheduling $TAD_i = \infty$ for $1 \leq i \leq N$. Suppose that a packet is to be transmitted at time *t*. VPS-TAD selects the backlogged class *j* with the maximum bandwidth normalized average delay,

$$j = \arg\max_{i \epsilon G(t)} TAD_i(t).$$

PID Control-theoretic Buffer Management:

A control-theoretic buffer management scheme is to be integrated with the packet scheduling schemes, for the FBS-DD provisioning and proportional loss rate differentiation at the same time. One nice feature of the buffer management based approach is that the packets will be dropped from the tail due to the buffer overflow. This avoids the packet push out issue and facilitates the packet ordering. The buffer management is to dynamically allocate the buffer space into a number of virtual mini-buffers, one mini-buffer for one class. The size of a mini-buffer directly affects a class's loss rate.

(PID) feedback controller to adjust the buffer allocation. Let $l_i$ be the loss rate of class *i*. The goal is to ensure that the observed relative loss rate $l_i$ be proportional to the pre-specified QoS parameter $\delta_i$, that is $l_i / l_j = \delta_i / \delta_j$. Let $L_i$ be the relative loss rate ratio of class *i*, $L_i = \frac{l_i}{l_{1+l_2+\dots l_n}}$. Let $L_i^d$ be the desired relative loss rate ratio of class *i*, that is, $L_i^d = \frac{\delta_i}{\delta_1+\delta_2+\dots\delta_n}$.

During the *kth* sampling period, the relative error is calculated as difference between the desired value and the observed value, that is,

$$e_i(k) = L_i^d(k) - L_i(k).$$

One property of the model is the sum of the relative errors is always zero since $\sum_{i=1}^{n} e_i(k) = 0$.

 

This important property makes it feasible for us to adaptively adjust the buffer allocation for a class independent of the adjustments of other classes while maintaining a constant overall buffer size.

## V. IMPLEMENTATION OF PACKET SCHEDULING

Here we are implementing three modules to reduce the packet delay and increases the packet delivery ratio in the streaming applications.

- Traffic model
- Buffer model
- Buffer Management Algorithm
- Weight Priority Algorithm

### A. *Traffic Model*

Consider a collection of M streams in traffic model of small-sized packets, designated by $S_1, \ldots, S_m$. Each stream $S_m$ is observed as a sequence of frames, $f_i^m$, each consisting of a sequence of exactly k packets, $p_1^{m,i}, \ldots, p_k^{m,i}$. A packet $p_j^{m,i}$ is referred to as the j-packet of frame $f_i^m$, and its arrival time is denoted by $a(p_j^{m,i})$. When referring to packets, we will sometimes omit the frame index i, and use the notation $\{p_j^m\}_{j=0,1,\ldots}$ when referring to the sequence of packets corresponding to stream $S_m$, where $p_j^m$ denotes the jth packet of stream $S_m$, and the $(j\, mod\, k)$ packet of frame $¿\frac{j}{k} \vee ¿^m f_{¿}$ (i.e., the $¿\frac{j}{k} \vee ¿$ th frame of stream $S_m$). The packets of a stream arrive in order, i.e., $a\left(p_j^{mi}\right) \leq a(p_{j+1}^m)$ for all j. The above notation implies the following structure on the arrival of packets in a stream $S_m$ consisting of $r_m$ frames:

$$\sigma = p_0^{1,0}, p_0^{2,0}, p_1^{2,0}, p_0^{2,1}, p_1^{1,0}, p_1^{2,1}, p_0^{1,1}, p_1^{1,1}$$

Fig 2: Example of an interleaved arrival sequence

### B. *Buffer Model*

When the packets go into at a first-in-first-out buffer which can store up to B $\geq$ k packets and which can send out one packet per cycle. At first, the buffer is empty. Every cycle consists of two steps. The first step is called the delivery step, where, if the buffer is nonempty, the packets are transmitted on the link. The second step is called the arrival step. In this step the packets arrives at the buffer. At the discretion of the buffer management algorithm, some packets may be dropped, while other packets are stored in the buffer. The packets are dropped out that depends on the rank of the packets.

### C. *Buffer Management Algorithm*

There are two design criteria for the buffer management algorithm,

- No-regret policy. Once a frame has a packet admitted to the buffer, make every attempt possible to deliver the complete frame

- Ensure progress. Ensure the delivery of a complete frame as early as possible. To implement the first criterion, we will use a dynamic ranking scheme for the traffic. The second criterion takes form in the usage of preemption rules. The balancing between the two criteria is done by a definition of the delicate interplay between the ranking scheme and the preemption rules

### D. *Weight Priority Algorithm*

We present the main algorithm called WEIGHTPRIORITY, which follows the design criteria. In the beginning of the arrival step of any cycle t, and for every frame $f_i^m$ define its rank at t by

$$r_t\,(f_i^m) = (\, w_t\,(f_i^m, m\,)$$

For completeness, we also define a tie breaking rule for frames of the same stream, where given any two such frames corresponding to the same stream, we consider the frame with the lower frame index as having the higher rank.

We say a frame is alive if none of its packets have been dropped yet, and a frame $f_i^m$ is said to be active at time t if fm i is alive at time t, and $f_i^m > 0$. Note that by the definition of the streams, at any time t, at most one frame can be active in every stream. Since the rank of a frame depends on its weight and its invariant stream index, the rank of a frame does not change during the arrival step.

 

However, a frame can stop being alive during the arrival step due to having some of its packets dropped.

## VI. CONCLUSION

The buffer management scheme is used for avoiding the packet losses. Because in the streaming applications each and every packets depends on other packets. So, if there is a loss in any packet the whole sequence is useless. To avoid these problems we use buffer management schemes to discard some of the packets that depends on the ranking schemes. In addition to that we use packet scheduling methods to improve the quality of service in the streaming applications.

In the packet scheduling schemes both packet size and packet delay is taken into considerations. After that control-theoretic buffer management scheme is proposed. The packet scheduling with buffer management provides delay and bandwidth differentiation in an integrated way. The packet scheduling schemes are able to achieve the FBS-DD provisioning at different workload conditions.

Securing multimedia data has become of utmost importance especially in the applications related to military purposes. Using innovative encryption algorithms for video sequences are necessary for protect the data. This can be done in future work.

## REFERENCES


1) Z.-L. Zhang, S. Nelakuditi, R. Aggarwal, and R.P. Tsang, “Efficient Selective Frame Discard Algorithms for Stored Video Delivery Across Resource Constrained Networks,” Real-Time Imaging, vol. 7, no. 3, pp. 255-273, 2001.

2) E. Gurses, G.B. Akar, and N. Akar, “A Simple and Effective Mechanism for Stored Video Streaming with TCP Transport and Server-Side Adaptive Frame Discard,” Computer Networks, vol. 48, no. 4, pp. 489-501, 2005.

3) A. Kesselman, Z. Lotker, Y. Mansour, B. Patt-Shamir, B. Schieber, and M. Sviridenko, “Buffer Overflow Management in QoS Switches,” SIAM J. Computing, vol. 33, no. 3, pp. 563-583, 2004.

4) A. Kesselman, Z. Lotker, Y. Mansour, B. Patt-Shamir, B. Schieber, and M. Sviridenko, “Buffer Overflow Management in QoS Switches,” SIAM J. Computing, vol. 33, no. 3, pp. 563-583, 2004.

5) C. Dovrolis, D. Stiliadis, and P. Ramanathan. Proportional differentiated services: Delay differentiation and packet scheduling. *IEEE/ACM Trans. on Networking*, 10(1):12–26, 2002.

6) J. Liebeherr and N. Christin. JoBS: Joint buffer management and scheduling for differentiated services. In *Proc. Int’l Workshop on Quality of Service*, pages 404–418, 2001.

7) J. Wei, C.-Z. Xu, and X. Zhou. A robust packet scheduling algorithm for proportional delay differentiation services. In *Proc. of Globecom*, 2004.

8) A. Awad, M.W. McKinnon, and R. Sivakumar, “Goodput Estimation for an Access Node Buffer Carrying Correlated Video Traffic,” Proc. IEEE Seventh Symp. Computers and Comm. (ISCC), pp. 120-125,2002.


 