Adaptive Load Balancing using Application Metric to Avoid the Bandwidth Starvation Problem for WWW Application due to BitTorrent Application

Sanon Chimmanee

Faculty of Ph.D Information Technology, Rangsit University, Thailand

***Abstract*—This paper presents a new methodology for alleviating a bandwidth starvation problem of WWW (HTTP) application due to using BitTorrent application at the same time. The proposed approach is to separate these applications on the different internet connections. The Application Metric Classifier is applied to be a dynamic decision of load balancer for adaptively distributing Internet applications between the internet connections. This results in optimizing both a service quality for HTTP and network resource utilizations simultaneously.**

Index Terms— *Load Balancing, BitTorrent, QoS*

I. Introduction

NOWADAYS, many users are using Peer-to-Peer (P2P) application to share very large files e.g. multi-media files over the Internet [1],[2].

In BitTorrent (the popular P2P application), a large file is divided into a large number of smaller data blocks, each usually between 32K and 256K, and these blocks are then shared independently [1]. Client can download multiple chunks of the file in parallel. In addition, the client can share individual data blocks for upload to other clients even if it has not finished downloading the entire [1]. Therefore, using BitTorrent transferring (downloading) the large file is a cause of very high traffic volume compared to traditional download using FTP and HTTP [2]. This means that high bandwidth is used by P2P and results in less bandwidth for other applications like web browsing (HTTP application). BitTorrent often put huge pressures on Internet operators and Internet Service Providers (ISP) because the bandwidth costs to the upstream ISPs and inter exchange carriers is more expensive [3],[4].

Broadband ISP would like to limit the BitTorrent traffic in order to solve a problem of bandwidth starvation for other applications [2]-[4]. For example, HTTP (WWW application) customers who are in the same apartment as a few BitTorrent customers often have a big problem of bandwidth starvation although ISP provides a big Internet connection for this place. This means that the HTTP customers are provided lower level of service than they are actually paying for. Traditional BitTorrent uses ports in the range of 6881-6889. If these ports are busy for some reason, BitTorrent will allocate connection port dynamically [4], [5].

Corresponding author e-mail: schimmanee@ yahoo.com

Therefore, traditional rate-limiting device which uses port number to identify BitTorrent packets may not limit bandwidth of BitTorrent traffic [2], [4]. The rate-limiting or rate-shaping device has to identify the BitTorrent traffic by inspecting the TCP/IP packets, identifying distinct packet types based on the data they contain [3]-[5]. This means that some processing at application layer of TCP/IP stack is required to identify BitTorrent pattern. Therefore, all Internet applications will take additional time-delay at the rate-limiting device known as a processing delay problem. Additionally, P2P traffic identification using application signature may be not reliable. This is because that the application signature analysis for P2P traffic has to be modified when a new version of P2P application is released [4].

Nowadays, many ISP often has two Internet connections for redundancy or load sharing solution. Traditionally, there are three basic load balancing techniques for router or gateway [6]-[8]: Per-packet, Per-destination and Per-flow. However, a main target of these load balancing techniques is only effective network resource (bandwidth connection) utilization without a target of supporting quality of service (QoS) [9]-[12]. Per-port load balancing method [13] is not popular. This is because that it may be not effective network resource utilization since layer-4 (TCP or UDP) sessions are fixed for each internet connection.

This paper proposes to solve the problem of bandwidth starvation of HTTP application by using the adaptive Per-port load balancing approach [9]-[12]. The proposed methodology separates HTTP traffic from P2P traffic to one internet connection by using the port number 80, which processes at Transport layer of TCP/IP stack. By this approach, it is not necessary to know which port numbers are used for P2P traffic. Thus, using this methodology allows the packet to be faster process compared to the existing method. From experiments in section 4, this results in a good service quality for WWW application while P2P traffic still takes place at the same time since HTTP and most P2P traffic are served with different internet connection. The proposed solution applies Application Metric Classifier to be a load balancer for dynamically distributing Internet applications between Internet connections. By this adaptive methodology, the proposed load balancing can optimize both a good QoS for HTTP users and network resource utilization simultaneously.

The proposed mechanism contains two states. One is an initial state and other is an adaptive state. In the initial state, HTTP traffic that is identified using port 80 is forward to

one of the two internet connections. And forward other applications that are identified using other ports to the other Internet connection. This allows HTTP application to obtain the good service quality. In adaptive state, some application that its traffic volume does not degrade the service quality of HTTP traffic can be adaptively forwarded to the same bandwidth connection as HTTP application by making decision of Application metric Classifier. Therefore, all HTTP users will get the good QoS and two Internet connects can be used effectively simultaneously.

The rest of paper is organized as follows. Section II states problem addressing and previous work. The proposed adaptive port-load balancing mechanism is presented in Section III. Section IV describes a network configuration, experiment setup. Section V gives experiment result and discussion. Section VI offers conclusions and further work.

II. Background

A. Problem Statement

Traditionally, a problem of bandwidth starvation for HTTP due to P2P is solved by using a bandwidth limiting device that requires an advance process of identifying P2P traffic at the application layer of TCP/IP stack [2]-[4]. This method adds time-delay for all internet applications.

The application signature analysis for identifying P2P traffic has to be upgraded when a new version of P2P application is released [4]. Therefore, currently used P2P traffic detection method may be not accuracy completely.

Traditional load balancing techniques in the gateway (e.g. router) are designed without a target of service quality for desired application [9]-[12]. Additionally, it excludes a target for the problem of bandwidth starvation for HTTP due to P2P.

Typically, the per-flow load balancing is popular for implementation. Because the per-packet load balancing has a problem of out-of-order packets and the per-destination requires a high memory for keeping destination IP addresses.

B. Previous Work

Since the traditional load balancing is designed excluding a target of the QoS requirement, Reference [9]-[12] proposed a novel adaptive per-port load balancing that is a load balancing method at the transport layer (TCP or UDP). Application Metric (AM) Classifier adaptively classifies Internet applications into one of two classes. One is First Class and the other is Economic Class. There are two targets of the proposed load balancing. One is to maintain QoS requirement for desired application. Other is a basic target of normal load balancing technique that is an efficient utilization of network resources (bandwidth connections).

Application Metric (AM) Classifier uses an Application Metric to evaluate a cost for each Internet application based on a type of delay-sensitive application, current traffic volume and packet size. Internet traffic can dynamically balance between Internet connections with respect to a QoS requirement for the desired application because the Application Metric is the weight adjustment equation. The mechanism of AM is the Neuro-fuzzy methodology that uses Fuzzy control to adjust weights of neural network. Therefore, the change in environment (e.g. traffic volume, network usage) effects directly to the application cost. It should be note that AM equation that is implemented in the initial state is a feedforward neural network so it does not take time for learning. Additionally, this Neuro-fuzzy method is not used for traffic classification like [4], [14].

Reference [10] presented Application Metric (AM) equation that is only specifically designed for VoIP application. Then, AM equation is developed for general delay-sensitive Internet application e.g. Telnet [9].

III. Adaptive Load Balancing Using Application Metric

Based on assumption that there are two Internet connections. This paper applied the AM Classifier (adaptive load balancing using Application Metric) to solve the problem. There are two objectives of this methodology. One is how to maintain QoS for HTTP traffic and other is how to use network resource (bandwidth connection) effectively. Internet applications will be classified into two classes. One Internet connection is dedicated for the first class and other Internet connection is assigned for the economic class. This mechanism contains two states as below.

A. Initial State

In this state, there is only HTTP as member of the first class and other applications are member of the economic class as mentioned in subsection II-B.

B. Adaptive State

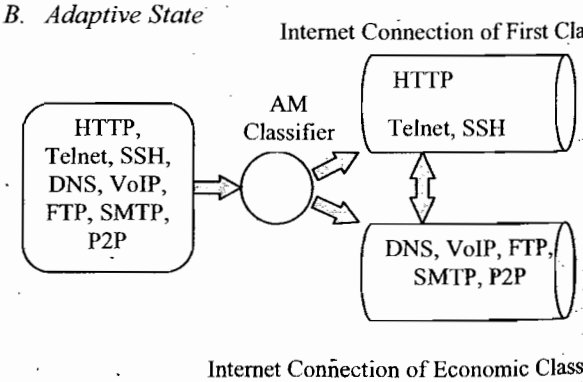

Fig. 1: In the adaptive state, the first-class may contain both the WWW application and other applications. This introduces fairness in the use of bandwidth connections with respect to keep the QoS for WWW application. The target of this state is not only to meet the QoS requirement but to balance the load as well.

In this state, the weights [9] are adjusted by the proposed fuzzy tuning system [9] so that other applications can be in the first class. Thus, the first class may contain the HTTP application and other applications as shown in Fig 1. This results in the optimization of both the utilization of network resources and QoS for the desired application

C. Mechanism of Application Metric Classifier

As mentioned above, the mechanism of Application Metric Classifier is the neuro-fuzzy system. The neural network is shown in the Fig. 2. Equation of this neural network is an application metric equation. By this methodology, the equation can be the weight adjustment equation. The weights in the application metric equation are adjusted by the fuzzy control shown as in Fig. 3. More detail of this neural network equation, fuzzy rule and so on can be found in [9].

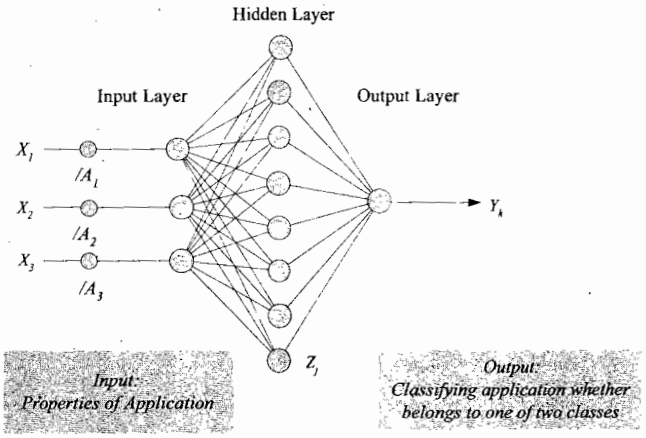

Fig. 2: The multilayer perceptron (MLP) is used to create the application metric formula which is the criterion for evaluating the application cost for each application

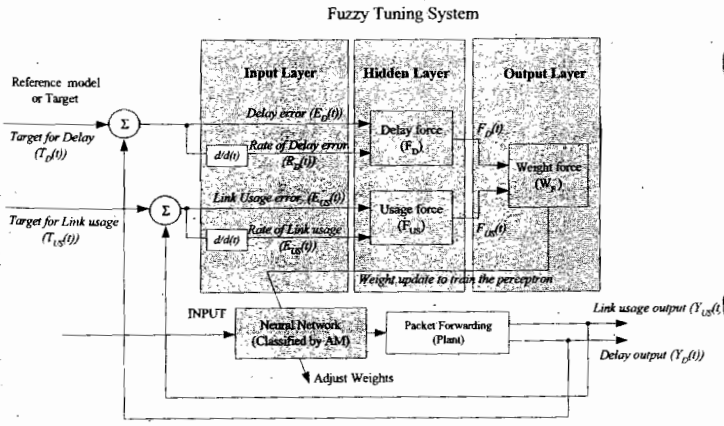

Fig. 3: The proposed mechanism contains the neural network and the fuzzy tuning system known as Hybrid system. Y_{US} is the link usage output, Y_D is the delay output, T_{US} is the link usage target, T_D is the delay target, E_{US} is the link usage error, E_D is the delay error, R_{US} is the rate of E_{US}, R_D is the rate of E_D, F_D is the delay force for maintaining the QoS for WWW (first target), F_{US} is the link usage force for utilizing the usage of link1 and link2 (second target), and W_F is the weight update to tune weights of the neural network

IV. Experiment Configuration

A. Network Configuration

The network configuration of sonetwork co., ltd in Changmai, Thailand was established to provide Internet service for customers at Baanthai apartment and Hillside condominium as shown in a Fig 4. There are 2 internet connections for both sites (Baanthai, Hillside). One is TT&T Internet connection and the other is TOT Internet connection. Each connection is 2 Mbps ADSL with SME class (Business level).

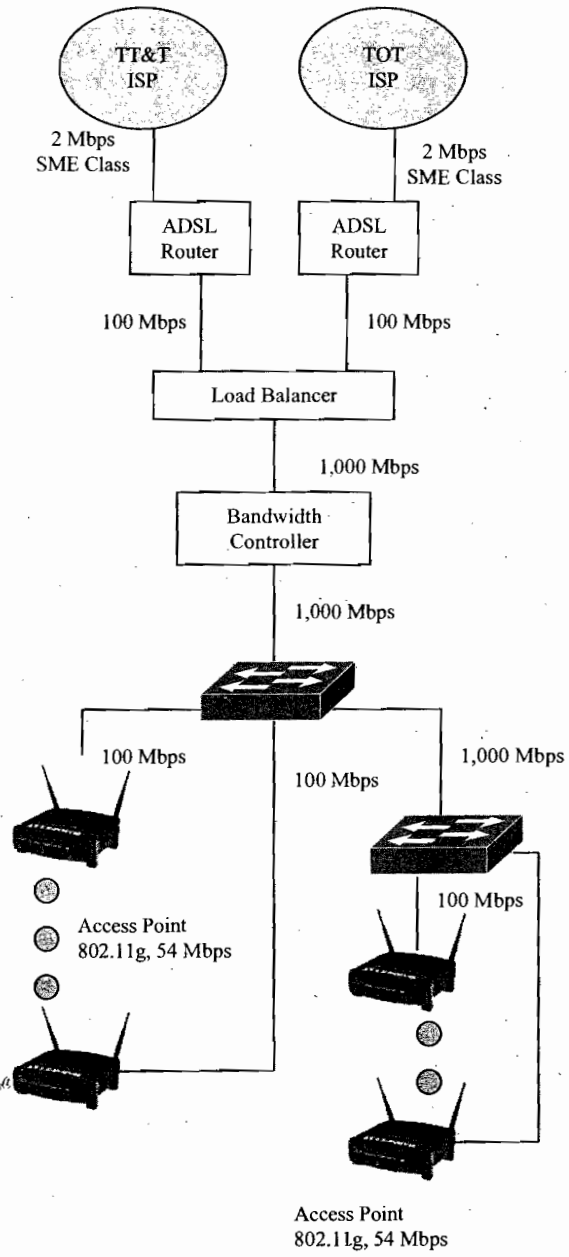

Fig. 4 shows the network configuration for experiments

There are four methodologies for performance evaluation. Thus, there are four experiments. Network configuration of experiment 1 is shown in Fig. 4. A load balancer with per-flow load balancing mode is implemented on Linux Server (IBM xSeries 100). The Netfilters [15] (i.e. iptables) is used to enable the IBM Linux Server to be the load balancer. A bandwidth controller is also implemented on the onLinux Server (IBM xSeries 100). The iptables with a modification of rope script [16] and CBQ [17], [18] (i.e. tc command) are used to enable the IBM Linux Server to be a rate-limiting BitTorrent application. The 10 kbps is allowed for each BitTorrent flow.

Network configurations of experiment 2 to 4 are the same as the experiment 1. The difference is that there is no the bandwidth controller. The difference in experiment of 2 to 4 is a load balancing methodology. In the experiment 2, the per-flow load balancing mode is implemented. In the experiment 3, the static per-port load balancing mode is implemented. WWW packet (HTTP application) is forwarded to TT&T Internet connection and other IP packet (Other application) is forwarded to TOT Internet connection. In the experiment 4, the proposed adaptive load balancing with application metric is implemented. TT&T Internet connection is dedicated to the first class and TOT Internet connection is dedicated to the economic class.

B. *Performance Evaluation*

There are two major performance evaluations as follows: One is a QoS performance for WWW applications. It can be measured from average time-delay of displaying web page (HTTP application) which is measured in unit of second.

The other is a load balancing performance, which can be measured from fairness in use of network resources (two bandwidth connections). Unequal usage of links, measured in percentage of bandwidth link usage, is the difference between two link usages.

C. *Experiment Setup*

Each experiment was constructed as follow. Internet Temporary File was deleted before accessing to web page (e.g. yahoo.com, google.com) and there was at least one BitTorrent user. There were three times for accessing to the same web page for one testing. In each experiment, there were at least 63 tests. The testing was implemented on 9:00-10:00, 12:00-13:00, and 18:00-19:00 for 7 days (Monday-Sunday). The experiments at the Baanthai Apartment were done during August 1-7, 2007. The experiments at the Hillside Condominium were done during August 8-15, 2007

All parameters setting for Application Metric classifier are the same as subsection 5.2 in [9]. The difference is that a target of delay is equal to 4 second because Reference [19] suggests that Web page downloads should complete in a few seconds (e.g., less than 5 second).

V. EXPERIMENT RESULT

A. *Load Balancing Performance*

Methodology	Link usage (%)		Unequal usage of link
	TT&T	TOT	
Per-flow LB	96%	94%	2%
Per-flow LB + Bandwidth Control	90%	89%	1%
Static Per-port LB	68%	98%	30 %
Adaptive Per-port LB	80%	98%	18%

Table 1 shows experiment result of Load balancing performance in percentage.

From our experiment as shown in Table 1, the per-flow load balancing & bandwidth controller methodology is the best performance in load sharing since the per-flow load balancing distributes flow in round robin fashion [8].

The proposed methodology gives a better performance than the static per-port load balancing since it adaptively distributes applications into one of two classes.

B. *Time-delay of web page performance*

Methodology	Average time-delay of displaying web page (second)
Per-flow LB	20.28
Per-flow LB + Bandwidth Control	5.48
Static Per-port LB	4.28
Adaptive Per-port LB	4.48

Table 2 shows experiment result of the average time-delay of displaying web page in unit of second.

From our experiment as shown in Table 2, the proposed methodology gives a better performance in time-delay than both of the per-flow load balancing methodology and the per-flow load balancing & bandwidth controller methodology up to 294 % and 19 %, respectively. This is because the proposed methodology includes a target of service quality for WWW application and operates at the transport layer.

The static per-port load balancing methodology is the best performance in time-delay since it separates WWW application from all other applications. It is better performance in time-delay than the adaptive per-port load balancing only 4.68 %. However, the proposed methodology has a better performance in load balancing than the static per-port load balancing methodology up to 67%.

VI. CONCLUSION

This paper presents the adaptive load balancing technique that includes a target of avoiding bandwidth starvation for WWW application due to BitTorrent. The proposed load balancing gives a better performance in load sharing than the static per-port load balancing method. In QoS performance evaluation, it gives a better performance than both of the per-flow load balancing and the per-flow load balancing & bandwidth controller method. The proposed load balancing can avoid the problem of bandwidth starvation for WWW application and effectively use network resources at the same time. This load balancing methodology can be known as WWW-friendly load balancing.

The adaptive load balancing with Application Metric will be studied to distribute Internet applications among several bandwidth connections in further work.

ACKNOWLEDGMENT

Author thanks Sonetwork co., ltd (Internet Service Provider in Changmai, Thailand) for its support in testing of four experiments.

REFERENCES

[1] David Arthur, and Rian Panigrahy, "Analyzing BitTorrent and Related Peer-to-Peer Networks" *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm 2006*, Miami, Florida, January 22 - 26, 2006, pp. 961 - 969

[2] Bandwidth Management for Peer-to-Peer Application, White paper [Online]. Available: http://www.lowth.com/rope/BlockingBittorrent

[3] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-Network Identification of P2P Traffic," *Proceedings of the 13th international conference on World Wide Web 2004*, New York, NY, USA, May 17 - 20, 2004, pp. 512 – 521.

[4] Anssi Tauriainen, "A Self-learning System for P2P Traffic Classification" HUT T-110.551 Seminar on Internetworking, 2005-04-26/27.

[5] BitTorrentTutorial [Online]. Available: http://wiki.theppn.org/index.php/BitTorrent_Tutorial, last modified 1 September 2007

[6] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien and Renata Teixeira, "Avoiding traceroute anomalies with Paris traceroute", *Internet Measurement Conference, October 2006 (IMC'06)*, October 25-27, 2006.

[7] Zenghua ZHAO, Yantai SHU., Lianfang ZHANG., *and* Oliver YANG, "Flow-Level Multipath Load Balancing in MPLS Network", *IEICE Trans. Commun.*, vol.E88–B, No.5, May 2005

[8] Yoji KISHI, Takeshi KITAHARA, Yujin NOISHIKI, Akira IDOUE, *and* Shinichi NOMOTO, "An Adaptive Traffic Load Balancing Method for Multi-Hop Mesh Networks in Broadband Fixed Wireless Access Systems", *IEICE Trans. commun.*, vol.E88–B, No.4, April 2005

[9] S. Chimmanee, K. Wipusitwarakun and S. Runggeratigul, "Hybrid Neuro-Fuzzy Based Adaptive Load Balancing for Delay-Sensitive Internet Application", *Journal of Intelligent and Fuzzy System (JIFS)*, vol.16, No.2, 2005, pp. 79-93.

[10] S. Chimmanee, K. Wipusitwarakun and S. Runggeratigul, "Adaptive Per-Application Load Balancing with Neuro-fuzzy to Support QoS for VoIP over the Internet ", *KES'2003, Lecture note in AI*, Oxford, vol.1, pp.533-541, 3-5 September, 2003.

[11] S. Chimmanee, K. Wipusitwarakun and S. Runggeratigul, "Load Balancing for Zone Routing Protocol to Support QoS in Ad Hoc Network (Published Conference Proceedings style)", *Proc. ITC-CSCC 2002*, vol.3, pp.1685-1688. 16-19 July, 2002.

[12] S. Chimmanee, K. Wipusitwarakun, P. Termsinsuwan and Y. Gando "Application Routing Load Balancing (ARLB) to support QoS for VoIP application over VPN Environment (Published Conference Proceedings style)" *Proc. IEEE NCA 2001*, pp.94-99, 11-13 Feb, 2002.

[13] Ivan Pepelnjak, Load Balancing in BGP networks [Online]. Available: http://www.nil.si/ipcorner/LoadBalancingBGP/

[14] A. A. Ali and R. Tervo, "Traffic Identification Using Artificial Neural networkwork (Published Conference Proceedings style)", *Canadian Conference on Electrical and Computer Engineering 2001*, Vol. 1, pp.667-671.

[15] Noris network, What is netfilter.org? [Online]. Available: http://www.netfilter.org/

[16] How to control BiTTorent traffic using a Linux firewall [Online]. Available: http://www.lowth.com/rope/BlockingBittorrent

[17] Tc-cbq-detail [Online]. Available: http://linuxreviews.org/man/tc-cbq-details/

[18] Francisco Burzi, Bandwidth Limiting HOWTO [Online]. Available: http://www.linux-tutorial.info/modules.php?name=Howto&pagename=Bandwidth-Limiting-HOWTO/cbq.html

[19] W. Noureddine and F. Tobagi, "Improving the Performance of Interactive TCP Applications Using Service Differentiation", *Elsevier, Computer Networks*, Volume 40, Issue 1, September 2002, pp. 19-43