Fair Bandwidth Sharing Scheme Based on Upload Traffic Control

A. Skrastins[1], J. Jelinskis[1], G. Lauks[1]
[1]*Institute of Telecommunications, Riga Technical University,*
Azenes St. 12–311, LV-1046 Riga, Latvia
andris.skrastins@rtu.lv

***Abstract*—The current paper presents the evaluation of new fair downstream bandwidth sharing strategy and effective shared link utilization for TCP/IP networks. The proposed method provides download bandwidth fair sharing among multiple data flows based on their upload per-flow rate limitation. We evaluate the current approach by controlling upload rate to obtain fair download bandwidth sharing for each of the flows in the link. The performance of the proposed method is evaluated on physical test network. The experimental results demonstrate that our approach can successfully provide fair bandwidth distribution among multiple data flows and can be promising method for bandwidth management in the real data networks.**

***Index Terms*—TCP/IP, fair bandwidth distribution, artificial neural networks.**

I. INTRODUCTION

Bandwidth management that focuses on fair bandwidth sharing solutions is one of the most important issues today in the field of computer engineering and network management [1]. There is a well-known task to find a balance between fairness and throughput that is topical in many networking scenarios. The fairness becomes an important issue when accessing a commonly shared bandwidth in the link. A wide variety of strategies are available to maintain fairness but most of them focused on fair scheduling strategy where a different flows that wishing to share a common link can be allocated equally or in proportion to their class of service "weights" [2].

A fair queuing algorithm determines which flow to serve next so as to satisfy a certain fairness criterion [3]. The queue management organizes the buffer space and determines which packet is to be discarded next. Typically, fair queuing (FQ) algorithms attempt to approximate the Generalized Processor Sharing (GPS) which is based on a fluid model, and cannot be practiced in the real world, at the packet level [2], [3].

Most FQ solutions maintain one separate queue for an each data source and an each incoming packet is being placed in an appropriate queue. The system serves these queues in a round-robin-like fashion by taking one packet from each nonempty queue in turn. But more often various packets sizes, as well as different round-trip-times, become important reasons for unfair bandwidth distribution among equal priority queues or flows. Deficit Round Robin (DRR) is one of the scheduling algorithms that try to deal with an unfair bandwidth distribution caused by the variable packet size [3].

Ineffective shared link utilization means that the scheduling algorithm uses only *fair packet-discard strategy* to provide fair bandwidth distribution among multiple flows and does not make any analysis of the causes of congestion as well as does not provide any adaptive management of the traffic sources.

TCP is the major transport protocol in use in most IP networks, which itself includes methods for flow control. TCP is a rate-adaptive protocol that includes flow control algorithm that attempts to control a rate at which the source sends all the packets. This adaptation function attempts to achieve the highest possible data transfer rate without triggering consistent data loss [4]. The TCP receiver can reduces the transmission window of sender using duplicate acknowledgement (ACKs) message. An alternative mechanism is the Explicit Congestion Notification (ECN), where the router can send notification about network congestion while marking packets with a congestion bit flag [5]. However, the main idea of the flow control in TCP is to provide maximum available transmission rate for each individual flow rather than a fair bandwidth distribution. The congestion makes the negative impact on the bandwidth effective utilization, as all the packets, in the case of TCP, that were discarded on the congestion points, will be re-transmitted [6].

Our purpose is to adapt to this kind of adaptation approach by controlling the upload rate of the sender to obtain fair download bandwidth sharing for each of user or even for the each of the flow, as well as reduce the probability of re-transmission by providing more effective network resource utilization. We also evaluate to use an Artificial Neural Network (ANN) as a part of the upload and download rate adjustment mechanism.

II. THE CONCEPT OF PROPOSED METHOD

In this paper we show how to control the download rate of the flow by using the upload rate control.

Manuscript received April 2, 2013; accepted August 23, 2013.
This work has been supported by the European Regional Development Fund in Latvia within the project Nr. 2010/0270/2DP/2.1.1.1.0/10/APIA/VIAA/002.

A. ANN Model

ANN is a computational model that shares some of the similar properties of the brain and is widely applied in various fields to overcome the nonlinear relationships.

Many different ANN models have been proposed by several authors. In this paper, we concentrate on the most common neural network architecture, which is called the multilayer perceptron (MLP). The MLP networks are used in a variety of solutions, especially if the forecasting is needed, because of the inherent capability of arbitrary input–output mapping [7], [8]. The most common learning rule for MLP is the back-propagation algorithm (BPA). One of the most important features, offered by BPA, is the rapid convergence capability. Each MLP model is composed of a minimum of three layers: an input layer, one or more hidden layers and an output layer. Operation of ANN includes two main steps, where the first step is training of the neural networks and the second is able to forecast relevant output for the set of input data. During the training period, the procedure of the BPA repeatedly adjusts the weights of the connections in the neural network. The magnitude of weight changes is called the learning rate, and is usually chosen experimentally. The amplitude of the output of an each neuron is controlled through the activation function. The actual parameters of the MLP, as well as the implementation details, are defined in Section 3.

B. The Upload Limitation Algorithm

Before we explain the limit calculation algorithm, we should discuss the principle of fairness. The authors of the current paper define the fairness as the proportional bandwidth allocation, according to predefined policy conditions in the congested network.

To design the appropriate upload control system we have to define essential goals to attain. We define the main objective as the ability to provide dynamic upload limit value for each flow. The defined goal includes several tasks for recalculation:

– Tracking of the existing flows by accounting the list of active flows;

– Determining the number of flows – N;

– Keeping the flow array, sorted in descending order according to download rate. The limit should begin with the most active flows (limitation factor is higher for most active flows depending on their download rates);

– Calculating cumulative average download rate D_c [bps] from all the flows

$$D_c = \sum_{j=1}^{N} d_j, \tag{1}$$

where j represents the sequence number of flow in a sorted array and d_j indicates the average download rate [bps] of each flow taking into account the past un predicted samples

$$d_j = (r_{p_j} / m + r_{n_j} / n) / 2, \tag{2}$$

where m and n – number of samples; r_p/m – average download rate of last m samples; r_n/n – average predicted download rate for the next n samples. As network environment is rapidly changing, we assume that the mean values have to be calculated for the short time period (up to 10 seconds or up to 10 samples). Nevertheless, we have to take into account that a too small number of samples could raise undesirable oscillations of d_j;

– The total available bandwidth T_a on the shared link

$$T_a = k \cdot T_{total}, \tag{3}$$

where T_{total} is a total capacity of the shared link and k – ratio (0.1–0.01). The reserved download bandwidth ($T_{total} - T_a$) allows applying a new flow as well as it makes a buffer zone for the possible traffic bursts. It is dependent on the capacity of the total shared bandwidth. The value T_a can varied depending on total shared bandwidth. More often T_a can be chosen from the 95-99 percent of the total bandwidth;

– Calculating C – it indicates how much the amount of the allowed pre-defined downstream bandwidth (T_a) is exceeded

$$C = D_c - T_a. \tag{4}$$

– Dynamic threshold ratio *Thr* is included in the algorithm and is calculated as follows

$$Thr = \begin{cases} \left(\sum_{n=1}^{j} d_j - C \right) / j, \quad \text{while}\left(d_j \le Thr_j\right), & C > 0, \\ (T_a / N) \cdot q, & C \le 0, \end{cases} \tag{5}$$

where $q > 1$ determines how rapid to decrease *Thr* for all flows. This *Thr* ratio allows us to restrict most active flows in balance or helps reducing upload limits to the flows whose download rates are below their allowed rate;

– The *shape* of the upload control function is defined

$$shape_j = U_j \cdot \left(Thr / d_j\right)^2, \tag{6}$$

where U_j is an average upload rate [bps] for each of the flow j. Thr/d_j ratio determines the ability to dynamically adjust a limit value, where $Thr/d_j < 1$ means that the limitation is required, whereas $Thr/d_j > 1$ means the reduction of the limitation.

Upload regulation: C in (5) operates as a reference point for the upload limitation; therefore we define two operation states:

1. In (5) $C > 0$ indicates that the limitation should be performed, as T_a is exceeded. As one can see in the (6) - the relation Thr/d_j determines the limitation ratio which effectively and in balance limits the flows that exceeds their permitted download rate more, as well as makes it possible to the flows, that do not exceed their rate, to recover by reducing their upload limit;
2. On the contrary $C \le 0$ indicates that T_a is not exceeded and there is no necessity for any upload restriction, this allows us to decrease upload limitations for all of the flows.

III. EXPERIMENTAL NETWORK AND GENERAL RESULTS

A. Description of Testbed

The main goal of the testbed is to emulate the network between Local Area Network (LAN) and Wide Area Network (WAN), where the commonly shared link is located between the LAN's gateway and the WAN's router (Fig. 1). The WAN provides connectivity to Internet services whereas the LAN side consists of the users and management server, which controls the fair bandwidth sharing among the connections of users to the Internet servers.

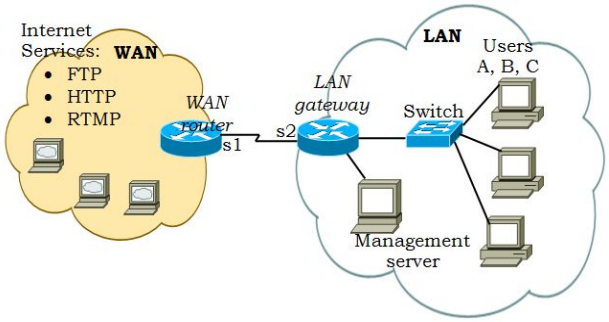

Fig. 1. Network topology of testbed.

The shared link is implemented between two Cisco 2800 series routers with the total capacity of the shared link (T_{total} = *1984Kbps*) and total available bandwidth T_a was 95 % of $T_{total.}$ The ratio q was equal to 1.1. FIFO (First-In First-Out) queuing was set on each router interface.

Management server dynamically controls the configuration of the router via telnet protocol while traffic accounting information is obtained using SNMP protocol. We named interface toward LAN as *private* and toward WAN as *public* on LAN gateway.

The tasks of the LAN router are as follows:

– To monitor traffic load on the shared link;
– To make user marking on the private interface;
– To classify the marked user on the public interface;
– To provide virtual queues for each of the users on the public interface;
– To keep an accounting for the upload/download load of each of the users.

The outline of the management server is as follows:

Step 1. To capture total traffic load and rate of each flow on the shared link via SNMP protocol;

Step 2. To determine the flows that should be restricted as well as flows whose restrictions should be reduced;

Step 3. To maintain and train the ANN instance for each of the user as well as to make traffic prediction for the next few second and to calculate the upload limits;

Step 4. Finally, to set the calculated limits on the router's virtual queues.

The virtual queues on the LAN router were implemented using Cisco Modular Quality of Service Command Line Interface (MQC) [9]. The management server controls the virtual queues and operates in a second time scale. The recalculation period was set to four seconds in our testbed.

The proposed practical realization of the management software was made using C# programming language, where ANN (MLP with BP) was realized using NeuronDotNet library. The actual parameters of (ANN) MLP were set as follows:

– The learning rate was set to 0.3;
– The default weight on arcs (random number 0.1- 0.3);
– The sigmoid activation function was set for all neurons.

The topology of MLP network consists of 40 neurons in the input layer, two hidden layers (3 neurons in each of the layers) and 5 neurons in the output layer. Each of the training set consists of 40 input samples (where 20 samples refer to the latest upload and next 20 to the latest download rate) and 3 output samples include the 3 latest samples of the download. The length of the training set was set to 10, and hereafter was successively updated with the latest training samples by deleting the oldest ones. One training cycle was made of 100 training loops.

B. Performance Results

The aim of this work is to demonstrate fair sharing capabilities and resource utilization efficiency for the proposed bandwidth sharing approach. In addition, we also evaluate the possibility of improving the efficiency of the proposed method by using ANN for short time traffic load forecasting.

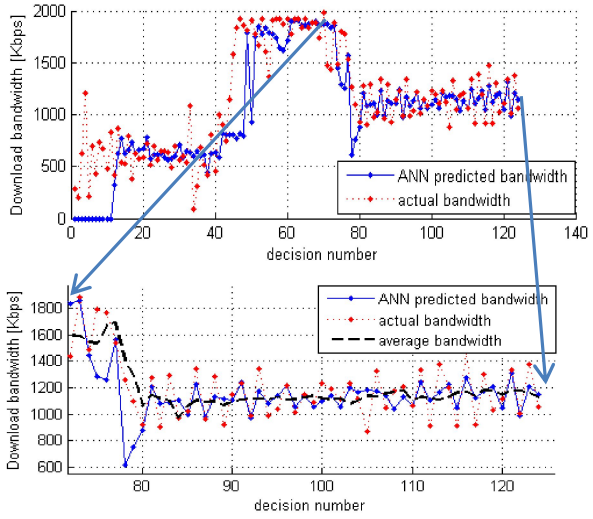

Fig. 2. ANN adaptation and prediction capabilities.

We examine the ability of ANN to forecast short time traffic load as well as an adaptation capability to the rapid traffic changes. In this paper ANN was used to improve the decision making process for upload traffic control, where not only the average 5 s actual download rate was taken into account, but also the ANN average 3 s forecasted download rate (2). However, the experiments have showed that the ANN is not able to adapt to rapid traffic changes and as a result the adaptation is too slow to use it in the online mode. Such an approach does not bring significant benefits (Fig. 2.). A similar result can be obtained if the past traffic sampling only is used in (2).

The bandwidth sharing performance of the proposed method is evaluated using TCP traffic under different application layer protocols. The experiment includes three users where each of them has established one flow. The first user-A makes file download via Hypertext Transfer Protocol (HTTP), user-B streams online video via Real Time

Message Protocol (RTMP) and user-C makes the file download via FTP (Fig. 2).

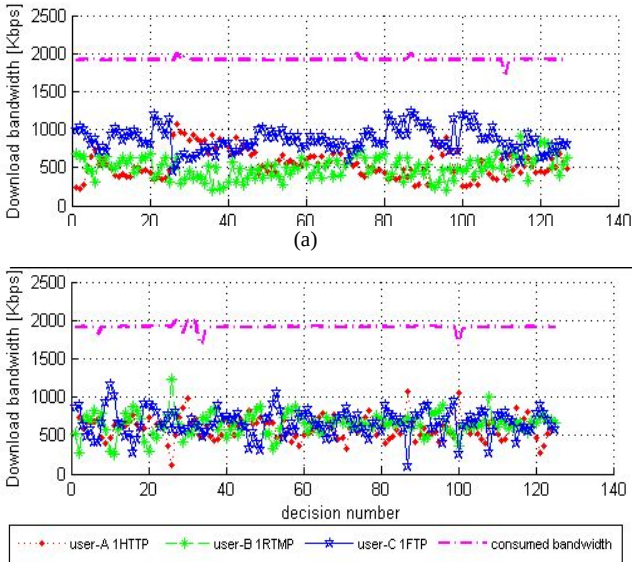

Fig. 3. Example of download bandwidth distribution without (a) and with (b) proposed method.

TABLE I. RESULTS OF FIG. 2: AVERAGE FLOW RATE.

Proposed scheme	User A [Kbps]	User B [Kbps]	User C [Kbps]	Total [Kbps]
Without	554	501	866	1921
With	601	660	665	1926

The fairness evaluation with and without the proposed method is depicted on Fig. 2(a) and Fig. 2(b) respectively. Figure 2(a) demonstrates how the different types of application protocols can contribute unequal bandwidth sharing. Figure 3 shows how the proposed algorithm maintains the fair traffic distribution among the users. For clarity, the average flow rates for each of the user are summarized in Table I. However, as we see in Table I that frequent traffic changes in short time scale reduce the possibility of fairer bandwidth allocation as well as the minimal value of the limitation step that was 1 Kbps, but the upload traffic from each of the users varied from 10 Kbps to 20 Kbps. But this level of fairness may be sufficient in most cases, even if we take into account that the high packet level precision for a fair bandwidth sharing is not always required.

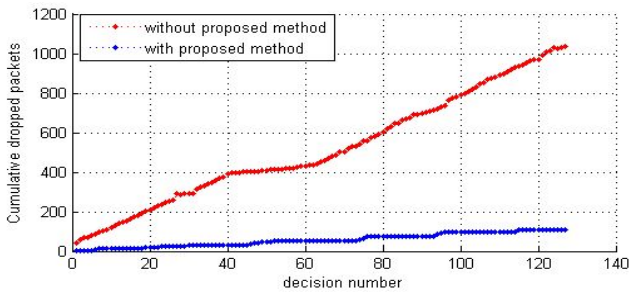

Fig. 4. Cumulative packet loss on WAN router on s1 interface.

One another value for quantitative evaluation of the proposed method is the amount of discarded packets on the s1 interface on the WAN router. As we can see in Fig. 4 proposed method shows promising results for the effective utilization of resources while providing a fair bandwidth sharing.

IV. CONCLUSIONS

In the current paper we provide the detailed description of the evaluation of the new method for fair bandwidth sharing that is based on adaptive upload traffic control strategy for TCP/IP networks. The bursty and continuously changing traffic load practically eliminates the possibility to use ANN to improve the effectiveness of the decision making system of the proposed algorithm. The experimental results demonstrate the resource utilization effectiveness, which is characterized by decreasing number of dropped packets at the same time providing a fair bandwidth allocation strategy. In present days TCP/IP networks does not always require high packet level precision for fair bandwidth allocation, and in many cases it could be sufficient to use the approximate per-session or per-user based fair bandwidth sharing strategy. The proposed method can be used as the potential fair bandwidth control mechanism of LAN or Wireless LAN networks which are operated under the pre-defined shared bandwidth allocation policy. The proposed algorithm can be easily extended using weight-based network bandwidth sharing and its evaluation is one of the objects of the future research.

REFERENCES

[1] M. Kassim, M. Ismail, K. Jumari, M. I Yusof, "A survey: bandwidth management in an IP based network", *World Academy of Science, Engineering and Technology*, vol. 62, pp. 356–363, Feb. 2012.

[2] N. Vaidya, A. Dugar, S. Gupta, P. Bahl, "Distributed Fair Scheduling in a Wireless LAN", *IEEE Transaction on Mobile Computing*, vol. 4, no. 6, pp. 616–629, Nov. 2005. [Online]. Available: http://dx.doi.org/10.1109/TMC.2005.87

[3] K. McLaughlin, S. Sezer, H. Blume, X. Yang, F. Kupzog, T. Noll, "A scalable packet sorting circuit for high-speed WFQ packet scheduling", *IEEE Trans. Very Large Scale Integration Systems*, vol. 16, no. 7, pp. 781–791, Jul. 2008. [Online]. Available: http://dx.doi.org/10.1109/TVLSI.2008.2000323

[4] G. Huston, "TCP Performance", *The Internet Protocol Journal*, vol. 3, no. 2, pp. 2–24, Jun. 2000.

[5] RFC 3168, "The Addition of Explicit Congestion Notification (ECN) to IP". [Online]. Available: http://www.ietf.org/rfc/rfc3168.txt

[6] S. Floyd, K. Fall, "Promoting the use of end-to-end congestion control in the Internet", *IEEE/ACM Trans. Networking TON*, vol. 7, no. 4, pp. 458–472, Aug. 1999. [Online]. Available: http://dx.doi.org/10.1109/90.793002

[7] A. Eswaradass, X. H. Sun, M. Wu, "Network bandwidth predictor NBP: A system for online network performance forecasting", in *Proc. 6th IEEE CCGrid*, IEEE Computer Society, 2006, pp. 265–268.

[8] D. G. Gomes, N. Agoulmine, J. N. de Souza, "IP bandwidth allocation management using agents and neural network approach", in *Proc. 2002 IEEE Workshop on IP Operations and Management*, Dec. 2002, pp. 8–12.

[9] *Modular Quality of Service Command Line Interface Overview*, Cisco. [Online]. Available: http://www.cisco.com