Delay-based Congestion Control Mechanism for Video Services
Mechanism including Backward Loading and Real-time Modes

Juha Vihervaara and Pekka Loula
Pori Campus, Tampere University of Technology, Pohjoisranta 11, 28100 Pori, Finland

Keywords: Delay-based Congestion Control, Video-on-demand Service.

Abstract: Currently the congestion control of the Internet is implemented through overprovisioning and TCP protocol. Unfortunately, TCP is not completely suitable for the use of video services. However, recent studies have shown that these video services represent over half of Internet traffic, with a growing trend. There are also arguments against massive overprovisioning. Due to these factors, there is a need to equip video services with proper congestion control. Unfortunately, most of the congestion control mechanisms developed for the use of video services can only offer low priority services or TCP-friendly real-time services. This paper provides a study in which a new delay-based congestion control mechanism is presented. This mechanism can offer congestion control services for both of these service types.

1 INTRODUCTION

Network operators have relied on overprovisioning and TCP congestion control in order to avoid congestion in their networks and to provide an acceptable service level for their customers. Up to the present time this has been reasonable because most of the traffic has been TCP-based. However, video-based services have become a common source of network traffic (Cisco, 2012). As is well known, video-based services have often used the UDP protocol to implement their transport services. Therefore, these video services operate without any kind of congestion control support.

Broadly speaking, based on these changes in Internet traffic types, there are two possibilities for implementing congestion control in the future. First, significant overprovisioning can be used to ensure the proper function of networks. However, nowadays there are some arguments against massive overprovisioning. For example, massive overprovisioning with unnecessary high power consumption is against green Internet ideology (Gupta, 2003). (Huang, 2005) says that the values of the overprovisioning factor of about 5 or even higher are not uncommon in large IP backbones. Self-similar traffic (Wornell, 1996) also sets challenges for overprovisioning. The studies have shown that variable-bit-rate video, which is transmitted over the Internet, is often self-similar (Rikli, 2011).

The second possibility is to use slight overprovisioning and also equip video-based traffic with congestion control. The current solution, where we use congestion control only with TCP traffic, will not be reasonable in the future because TCP traffic will then no longer play the major role (Cisco, 2012). It could be said that TCP and UDP traffic will take on the opposite roles to former years.

There are two different kinds of transfer modes needed among video services. First, there is a need for a backward loading mode where the delay and bandwidth demands are moderate. For example, the whole video can be loaded to the user equipment before it is watched. Also, proxy servers can be used so that videos can reside near customers. In this case, the service provider can load for example films to proxy servers by using the backward loading mode. Secondly, there is also a need for a real-time mode where the delay and bandwidth demands are important. The user may want to watch a live broadcast. The backward loading mode has to work like a low-priority service where the bandwidth is given away to other connections when the load level of the network is high enough. In contrast, the real-time mode always wants its fair share of the bandwidth.

There are many congestion control mechanisms which are suitable for either low priority service or real-time service. But there has been little research work into developing an integrated mechanism

Vihervaara J. and Loula P..
Delay-based Congestion Control Mechanism for Video Services - Mechanism including Backward Loading and Real-time Modes.
DOI: 10.5220/0004794501270134
In *Proceedings of the 10th International Conference on Web Information Systems and Technologies* (WEBIST-2014), pages 127-134
ISBN: 978-989-758-023-9

suitable for both cases. The aim of this paper is to study one such kind of integrated congestion control mechanism. In this case, the term integrated means that these two mechanisms are as convergent as possible.

The sending rate can be adjusted based on the packet losses or delays. Both indicators are utilized with the mechanism of this study, but it can be said that the algorithm is more delay-based than loss-based. The reason for emphasizing the more delay-based approach in this study is that it generates a suitable platform for implementing the background loading mode. In this paper, we try to find out if the delay-based approach is suitable for this kind of congestion control mechanism with the help of simulations.

2 RELATED WORK

There are many different kinds of congestion control algorithms which can be used by video services. Some delay-based algorithms and some algorithms for multimedia streaming are presented below. All of them are suitable for a low priority or real-time service. However, none of them are suitable for both service types.

2.1 Delay-based Congestion Control

DUAL (Wang, 1992) is one of the first algorithms in this area. In this algorithm, the optimal queue length is defined and then a corresponding delay in that queue length is used as a congestion signal.

TCP Vegas (Brakmo, 1994) is perhaps the most known delay-based congestion control mechanism. It calculates the amount of data queued on the routing path. This calculation is made with the help of RTT samples and sophisticated calculations. If the queue size is inside the wanted thresholds, the sending rate is not changed. Otherwise, the sending rate is decreased or increased. TCP Vegas does not try to be TCP-friendly because it was designed to be used in place of TCP Reno.

TCP-LP (Kuzmanovic, 2006) offers low priority services as compared to the normal best effort service offered by the Internet. The low priority service is implemented with the help of congestion control. To achieve a low priority service in the presence of TCP traffic, it is necessary to detect an incipient congestion earlier than TCP. TCP-LP measures one-way packet delays and employs a simple delay-threshold based method for the early indication of congestion. TCP-LP decreases its sending rate after the smoothed one-way delay exceeds the threshold.

LEDBAT (Shalunov, 2012) also provides low priority services by using one-way delay measurements to estimate the amount of queued data on the routing path. It increases or decreases its sending rate based on the predetermined target. Sending rates are increased and decreased more aggressively if the queuing delays are far from the target.

CDG (Hayes, 2011) is a delay-gradient based congestion control algorithm which no longer requires knowledge of the path-specific minimum delay thresholds for the congestion notification. In fact, CDG does not use delay gradients as a main source of congestion discovery. Instead, delay gradients are used to assist the normal TCP Reno-like congestion control to make the right decisions. Gradients are used to differentiate between congestion and non-congestion losses.

2.2 Congestion Control for Multimedia Streaming

Research work has been very active in this area during the last few decades. There are many proposals for multimedia streaming. One approach is to think that without changes TCP is not suitable for streaming media because it uses retransmissions for reliable communication with head of the line blocking. Therefore, the easiest way to implement congestion control oriented TCP-friendly transport layer protocol for multimedia streaming is to emulate the behaviour of TCP but without TCP's reliability. RAP (Rejaie, 1998) and TCP-RMP (Liang, 2002) can be classified as members of this group. TCP-RMP also does smoothing operations for its sending rate because big rate fluctuations with multimedia streaming are undesirable.

The most known proposal for multimedia streaming is DCCP (Kohler, 2006) and its TCP Friendly Rate Control version, abbreviated as TFRC (Floyd, 2008). The main point with DCCP's congestion control is that congestion control is not a part of DCCP itself. Instead, DCCP allows applications to choose from a set of congestion control mechanisms. At this moment, the main mechanism for multimedia streaming is TFRC. TFRC is designed for applications that need a smooth rate, and many real-time multimedia applications are this type. TFRC is an equation-based congestion control mechanism, which uses TCP's throughput equation (Padhye, 1998) to calculate the allowed sending rate.

Google Congestion Control for Real-Time Communication on the World Wide Web (Lundin, 2012) is a new proposal for this area. It uses delay gradients in a sophisticated way to detect congestion so that different frame sizes are taken into account.

3 CONGESTION CONTROL MECHANISM FOR VIDEO SERVICES

In this section, a new congestion control mechanism is presented. This new congestion control mechanism is called Congestion control for VIdeo to Home Internet Service, or CVIHIS.

3.1 Basic Functions

There are some basic choices for CVIHIS. Typically, this kind of service is implemented in a client-server manner because one video server is serving a great number of clients. Therefore, the natural choice is that CVIHIS is a receiver-based mechanism so that all possible calculations are placed on the receiver side.

The real-time mode wants to get its fair share from the network bandwidth. This means that this mode should be TCP-friendly. However, as usual with this kind of video application, the bandwidth should be used in a much smoother way than TCP's congestion control does. Of course, this new mechanism should be stable and scalable which are typical requirements for all congestion control mechanisms. CVIHIS is an end-to-end mechanism because the key element for scalability is the end-to-end argument described in (Saltzer, 1984). The idea behind this argument is that if you want the mechanism to have scalability, the complex issues should be placed at the network endpoints.

The sending rate can be controlled in a rate-based or window-based manner. A natural choice for CVIHIS, and traditionally a more commonly used approach with multimedia streaming, is the rate-based approach. (Akan, 2004) says that the window-based approach is not suitable for continuous multimedia streaming since its ACK-controlled packet injection method does not maintain smooth rate variation.

If the network does not support explicit congestion feedbacks, the sending rate can only be adjusted based on the packet losses or delays. Both indicators are utilized by CVIHIS, but it can be said that the algorithm is more delay-based than loss-based. The reason to emphasize a more delay-based approach is that it generates a suitable platform for implementing the background loading mode. For example, LEBDAT and TCP-LP offer mechanisms so that low–priority applications can withdraw from using bandwidth after the queuing delay exceeds the predefined target.

CVIHIS uses one-way delays for delay measurements instead of round-trip times. The main motivation behind the choice of one-way delay is that this means the necessary calculations can be made on the receiver side. However, there is a drawback with this one-way delay based mode. The clocks of the end-points have to be synchronized with some level of accuracy because we are measuring one-way delays.

3.2 Backward Loading Mode

The rate adaptation schema of CVIHIS is presented in Figure 1. Based on delay measurements, the minimum and maximum delays are picked up. The minDelay value presents a situation in which the queues of the connection path are empty. The minDelay value only includes propagation delay components. It could be said that the minDelay value is the shortest one-way delay value experienced during the lifetime of the connection.

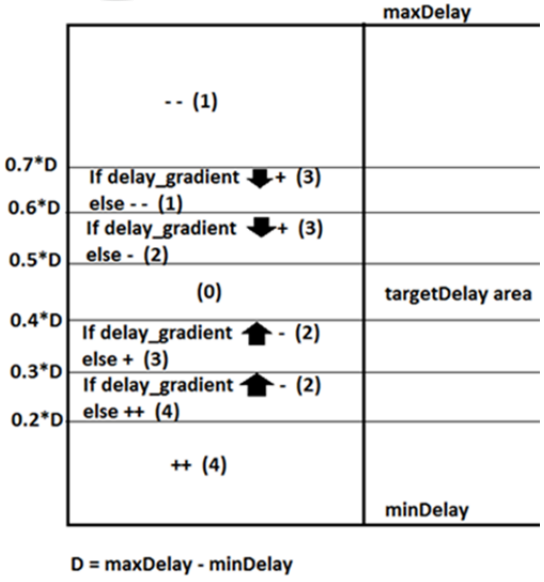

Figure 1: Rate adaptation schema of CVIHIS.

The maxDelay point is found when a packet drop occurs. The delay value of the last received packet before the dropped packet is used for the maxDelay value. In this study, the maxDelay value is totally

updated after each packet drop. However, there can be different kinds of congestion situations in real networks, and these situations generate different kinds of delay values for packet drop situations. Thus, for example, the Exponentially Weighted Moving Average (EWMA) could be used in the calculation of the maximum delay value. The EWMA process would average different kinds of congestion situations.

With the help of the minimum and maximum delay values, the delay space is divided into seven rate adaptation areas. We could also say that the queue of the router is divided into several parts. The centremost delay area is called the targetDelay area. The basic idea is that CVIHIS tries to keep the queue level of the router at the level of this target. Note that this target delay may be over the delay demand of the application. If applications have delay and jitter demands, these demands have to be satisfied by Quality-of-Service mechanisms. CVIHIS is a pure congestion control mechanism, not a Quality-of- Service mechanism.

As shown in Figure 1, the target delay area is not put in the middle of the delay space. Instead, this area is shifted downwards a little so that queuing delays can be kept short. The shiftng used in this case is perhaps too modest for the normal situation. But if the routers are equipped with active queue management mechanisms, this shiftng may be appropriate. The positioning factors for each delay area are presented on the right side of this figure.

There are black arrows inside some delay areas. These black arrows represent the direction of the delay. Thus, delay gradients are also used by CVIHIS for rate adaptation. With the help of delay gradients the target delay area can be achieved without significant oscillation. Using only delay gradients for rate adaptation is not recommended because delay gradients are often too sensitive to second-order delay fluctuations (Wang, 1992).

These delay gradients are obtained by comparing the delay values of two consecutive packets. If the arrow is upwards, the delays are increasing and the delay gradient is positive. This also means that the queue is filling up. If the arrow is downwards, the delays are decreasing and the delay gradient is negative. Therefore, the queue is empting.

As can be seen, we have four delay areas with a black delay gradient arrow. Inside these four areas, the rate adaptation command is based on the actual delay value and the value of the delay gradient. The delay value assigns a certain delay area. Inside this delay area, the delay gradient defines the actual rate adaptation feedback. The two extreme delay areas do not use delay gradients for rate adaptation decisions.

This rate adaptation schema tries to achieve two target values. It tries to drive the queue level to the level of the target delay area. This is done with the help of the actual delay values. The adaptation schema also tries to adapt the sending rate to the level of the bottleneck capacity. This is done with the help of the delay gradients. Based on these two targets, one extra rule can be defined. If there is a conflict of interests between these two targets, the sending rate target is favoured.

CVIHIS can adjust its sending rate through five adjustment steps. Four of these steps are presented in Figure 1 by ++, +, -, --. There are two different level steps for both increasing and decreasing the sending rate. These four adjustment steps are quite moderate so that some kind of additive increase and additive decrease behaviour is achieved. The fifth adjustment step is a multiplicative decrease step which is used after a packet drop. During one round-trip time only one multiplicative decrease step is taken. To simplify matters, we can say that CVIHIS imitates AIMD behaviour (Chiu, 1989).

As stated above, the aim is for CVIHIS to be a receiver-based congestion control mechanism. Therefore all conclusion procedures presented in Figure 1 are implemented on the receiver side. Only the rate adaptation command, which is an integer number, is transmitted to the sender. The values of the commands are presented inside parentheses in Figure 1. The sender increases or decreases its sending rate based on these commands if the measured delay is outside the target delay area.

3.3 Real-time Mode

The backward loading mode backs off when it competes with TCP. This means that we must modify the backward loading code so that it will behave in a more aggressive way to be suitable for the real-time mode. We will take an approach in which we push the minimum delay value upwards in a continuous manner. This approach means that the delay areas, which are presented in Figure 1, are also pushed upwards and therefore will CVIHIS behave more aggressively.

As a matter of fact, CVIHIS competes harder and harder during the pushing phase because the delay areas move higher and higher all the time. This upwards pushing is only done when the competing behaviour is really needed. If the last measured delay value is smaller than the pushed minimum delay value, the minimum delay value is set to the

last measured delay value. With this choice, there is no need to change the code of the sender side. In addition, only two code lines have to be added to the code of the receiver side.

This pushing function makes it possible for CVIHIS to compete against the TCP protocol. But this pushing function also gives other benefits. It alleviates the rerouting problem which is a typical problem for delay-based congestion control mechanisms. It also helps the clock synchronization problem.

In fact, the clock offset is not a problem for CVIHIS. CVIHIS divides the area between the values of the minDelay and maxDelay into several delay areas. CVIHIS can do this correctly if the maxDelay is greater than the minDelay. This means that these delay values can also be negative due to the clock offset. Instead of, clock drift can cause problems. If the delay trend is growing, the minDelay will become outdated. So, it is desirable to update the minDelay value from time to time. In this way, the accumulation of clock drift can be removed. Due to this clock drift problem, it is also perhaps clever to equip the backward loading mode with this pushing function in the future. With the backward loading mode, a small pushing factor could be used. The simulation in this study of the backward loading mode did not include this pushing.

The minimum delay value is pushed upwards by the pushing factor so that the current value is multiplied by this factor. Through simulation it was found that a pushing factor of 1.10 is the value of the suitable parameter group. This 1.10 is quite a high value. This high value indicates that CVIHIS must behave in quite an aggressive manner to be TCP-friendly. In fact, (Yang, 2001) shows that TCP is a quick protocol to utilize extra bandwidth.

4 SIMULATION RESULTS

The simulation program used in this work is NS-2 (Fall, 2013). The congestion control mechanism developed in this work is implemented as a new transport layer protocol of NS-2. The new network component objects have been written using C++-language. These new objects have been added to the NS-2 environment.

4.1 Test Network

The structure of test network is presented in Figure 2. There are six end nodes and two core nodes. Nodes 2 and 3 are core nodes. Because we are dealing with the TCP/IP protocol suite, these core nodes present IP routers. In node 2, the queue size of the bottleneck link is 20 packets. Other nodes are end nodes. For example, CVIHIS's sender resides on node 0 and CVIHIS's receiver on node 4. Some end nodes are represented by broken lines because these nodes are present only if three user connections are needed. In the simulation results, CVIHIS will be depicted in red. TCP connections will be depicted in green and blue.

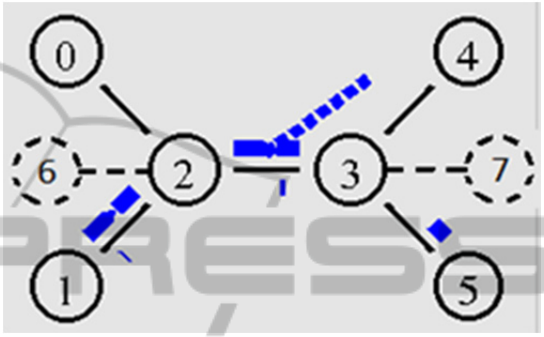

Figure 2: Structure of the test network.

4.2 Backward Loading Mode

Figure 3 presents the results of the simulation in which the back off behaviour of CVIHIS is tested.

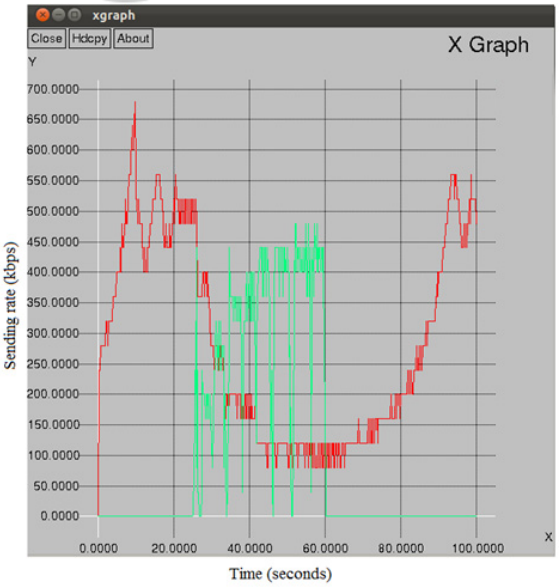

Figure 3: Simulation results of the backward loading.

The CVIHIS connection is started at the beginning of the simulation. The sending rate of CVIHIS is expected to settle to the capacity of the bottleneck link. This initial phase takes about 20 seconds

because no special start phase algorithm is used. After this settling phase, the TCP connection is started. The TCP connection is stopped after the CVIHIS back off action. After that it is checked that CVIHIS increases its sending rate because there is now free capacity in the network.

As can be seen, there is some oscillation behaviour at the beginning of the connection. The first high rate peak is necessary because the CVIHIS queue level probing does not switch on until after the first packet drop. The maxDelay value is not defined before this first drop. The stabilization happens after 20 seconds. As can be seen, the rate also varies after this point. However, this rate variation is generated only by the phase effect of the analysis tool. In order to calculate the sending rates, the packets are grouped. This grouping interval is 0.2 seconds, and it is often not compatible with the packet interval times. As a result, there may be one packet differences between the groups.

This simulation shows that the backward loading mode of CVIHIS works as expected and desired.

4.3 Real-time Mode

The TCP-friendliness of CVIHIS is tested in this section. Tests are made against the TCP Reno version. In fact, there is more behind this phase because the correct rate adjusting parameters for CVIHIS are searched for at the same time. These parameters can be used for tuning CVIHIS to be TCP-friendly. This suitable parameter group has also been used in the simulation case of the backward loading mode.

This parameter group is presented in Table 1. In fact, we are adjusting the sending gaps of the packets. So if we want to decrease the sending rate, the value of the parameter must be over 1. The four leftmost columns present values for the delay areas presented in Figure 1. MD is the multiplicative decrease factor which is used after packet drops. The value of the pushing factor used by the real-time mode is 1.10. Every data packet generates a rate adjustment command. However, after the multiplicative decrease step it takes one round-trip time before the rate adjustment continues.

Table 1: Adjustment parameters of CVIHIS.

++	+	--	-	MD
0.9976	0.999	1.002	1.001	1.23

In the first simulation case, CVIHIS and TCP start with the initial rate of 300 kbps. The capacity of the bottleneck link is 600 kbps. Both connections start at the same time. The simulation result for this case is depicted in Figure 4. In this simulation case, CVIHIS manages very well. It takes almost exactly its fair share from the capacity of the bottleneck. We have also made other simulations in which the initial rates of CVIHIS and TCP have not been equal. In those cases, CVIHIS managed at an acceptable level but not as perfectly as in this first case.

CVIHIS has also been tested against two TCP connections. The CVIHIS connection and the other TCP connection are sending all the time. Another TCP connection is started and stopped during the simulation. The capacity of the bottleneck link is still 600 kbps.

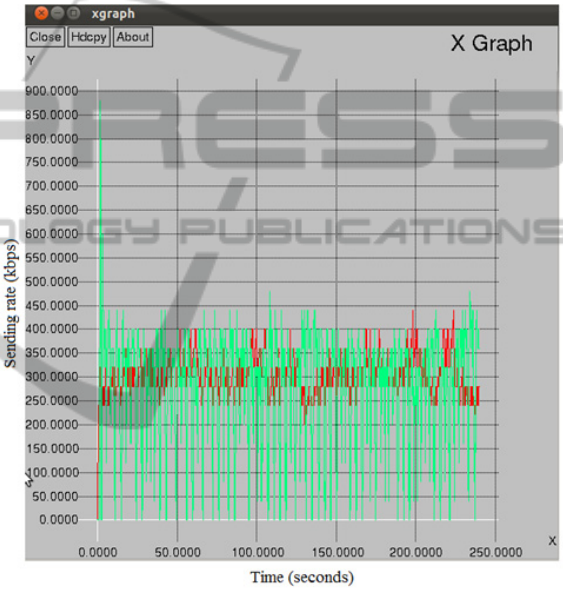

Figure 4: Test against one TCP connection.

In this simulation, it is first investigated if CVIHIS reduces its sending rate after the other TCP connection is started. It is also examined whether CVISIS would increase its sending rate after the other TCP connections is stopped. The simulation result is presented in Figure 5. In this case, CVIHIS manages moderately. It can decrease and increase its rate as required. However, when there are two active TCP connections, CVIHIS exhibits back-off behaviour. The sending rate of 100 kbps is CVIHIS's minimum rate. This moderate minimum rate is used because real-time video services need it.

4.4 Rerouting Problem

One simulation has been done to test the rerouting properties of CVIHIS. The rerouting discovery ability of CVIHIS is based on the pushing of the

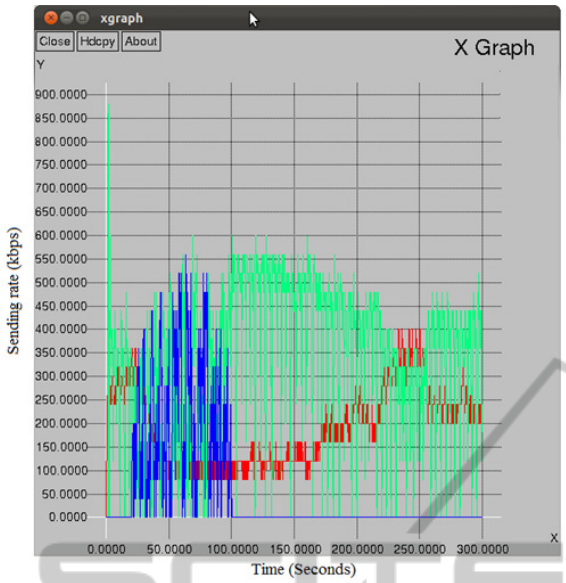

Figure 5: Test against two TCP connections.

minimum delay value and updating of the maximum delay value.

The network layout has been changed for this simulation. There are two possible routes between nodes 2 and 3. There is a direct default route and a backup route. The default route is switched off twiceo times during the simulation. The capacity of the default route is 700 kbps and the capacity of the backup route is 400 kbps. The simulation result is presented in Figure 6. As can be seen, CVIHIS can observe route changes and it can accommodate its sending rate according to the new route.

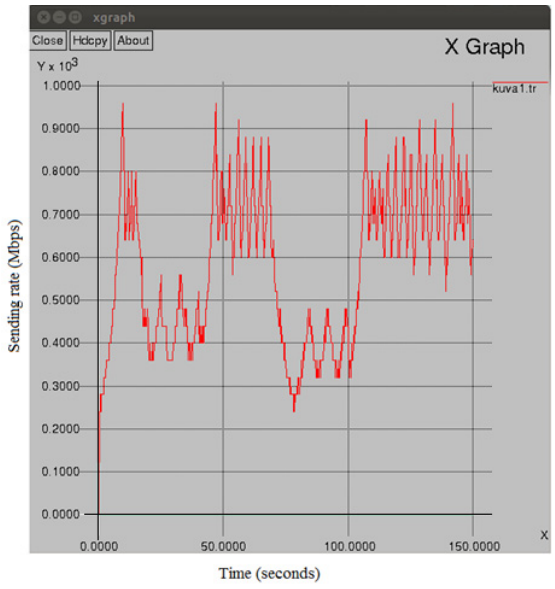

Figure 6: Testing rerouting properties of CVIHIS.

5 CONCLUSIONS

With the help of simulation it was found that this kind of delay-based congestion control approach could be suitable for video services. CVIHIS managed in an acceptable way.

However, it was found that CVIHIS behaves in quite a parameter sensitive way in terms of TCP-friendliness. Small changes to the values of the adjusting parameters caused changes to the TCP-friendliness. This kind of parameter sensitiveness can be problematic. It can also mean that minor changes in the network environment generate problems for TCP-friendliness. Therefore, it is also a good idea to simulate CVIHIS against other TCP versions, like TCP New Reno.

On the other hand, these kinds of findings come as no big surprise. It is a delicate task to make different kinds of mechanisms work together in a harmonious way. TCP and CVIHIS are very different kinds of congestion control mechanisms. TCP is a loss-based mechanism and CVIHIS is merely a delay-based mechanism. TCP is a window-based mechanism and CVIHIS is a rate-based mechanism. Widmer's paper (Widmer, 2001) analyses different kinds of congestion control mechanisms. This paper shows that rate-based congestion control mechanisms have some trouble with TCP-friendliness because TCP uses window-based rate control. Consequently, extra simulations are needed to prove the usefulness of this CVIHIS approach on a large scale.

Some elaboration work is also needed. For example, due to the pushing operation of the minimum delay value, the sending rate of the real-time mode fluctuates. This can be seen in Figure 6. Therefore, some smoothing function could be added in future work. For example, some filtering procedure could be put on the top of the current algorithm to make a low pass filtering procedure.

Instead of specifying a complete protocol, the aim of this study was to specify only a congestion control mechanism that could be used in different parts of the protocol stack. Only a basic mechanism was defined and all kinds of special cases were left outside this presentation. Therefore, all kinds of special cases, such as the start phase of the connection, need extra study in the future.

One possible point for future research could be to test the real-time mode of CVIHIS by using another kind of congestion control approach. This other approach could be a loss-based approach so that the behaviour of TCP congestion control is emulated on the receiver side.

REFERENCES

Akan Ö. 2004. *On the throughput analysis of rate-based and window-based congestion control schemes*. Computer Networks 44 (2004). Pages 701-711.

Brakmo L, O'Malley S, Peterson L. 1994. *TCP Vegas: New Techniques for Congestion Detection and Avoidance*. ACM SIGCOMM London, UK. Pages 24-35.

Cisco. 2012. *Cisco Visual Networking Index: Forecast and Methodology*, 2011-2016. Retrieved 2.2.2013 from http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html.

Chiu D, Jain R. 1989. *Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks*. Computer Networks and ISDN Systems 17/1989 Pages 1-14.

Gupta M, Singh S. 2003 *Greening of the Internet*. ACM SIGCOMM, Karlsruhe, Germany.

Hayes D, Armitage G. 2011 *Revisiting TCP Congestion Control using Delay Gradient*. The proceedings of NETWORKING 2011 10th International IFIP TC 6 Networking Conference, Valencia, Spain, May 9-13, 2011.

Huang Y, Guerin R. 2005. *Does Over-Provisioning Become More or Less Efficient as Networks Grow Larger?*. Proceeding of the 13th IEEE International Conference on Network Protocols 2005.

Fall K. 2013. *A collaboratoion between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC. NS-2 Manual.* www.isi.edu/nsnam/ns/doc/index.html.

Floyd S, Handley M, Padhye J, Widmer J. 2008. *TCP Friendly Rate Control (TFRC): Protocol Specification*. RFC5348.

Kohler E., Handley M., Floyd S., 2006. *Datagram Congestion Control Protocol (DCCP)*. RFC4340.

Kuzmanovic A, Knightly E. 2006. *TCP-LP: low-priority service via end-point congestion control*. IEEE/ACM Trans. Netw. 14(4) August 2006. Pages 739–752.

Liang S, Cheriton D. 2002. *TCP-RTM: Using TCP for Real Time Applications.* IEEE International Conference of Network Protocols '02.

Lundin H, Holmer S,. Alvestrand H. 2012. *"A Google Congestion Control Algorithm for Real-Time Communication on the World Wide Web"*. IETF Informational Draft, Version 3.

Padhye J, Firoiu V, Towsley D, Kurose J, 1998. *"Modeling TCP Throughput: A Simple Model and its Empirical Validation",* Proc ACM SIGCOMM 1998.

Rejaie R, Handley M, Estrin D. 1998. *Architectural considerations for playback of quality adaptive video over the internet.* Technical report 98-686, CS-USC, November 1998.

Rikli N. 2011. *Self-similarity and stationarity of increments in VBR video.* King Saud University Journal of King Saud University – Computer and Information Sciences 4/2012 Pages 7-16, Retrieved 1.2.2013 from www.ksu.ede.sa.

Saltzer J, Reed D, Clark D. 1984. *End- to- end arguments in system design*. ACM Transactions on Computer Systems 2(4). Pages 277-288.

Shalunov S., Hazel G, Iyengar J, Kuehlewind M. 2012.. *Low Extra Delay Background Transport (LEDBAT)*. RFC6817.

Wang, Z., Crowcroft, J. 1992. *Eliminating periodic packet losses in the 4.3-Tahoe BSD TCP congestion control algorithm.* SIGCOMM Computer Communication. Rev. 22(2). Pages 9–16.

Widmer H, Denda R, Mauv M. 2001. *A Survey on TCP-Friendly Congestion Control.* IEEE Network May 2001, Volume:15 Issue 3,. Pages 28-37.

Wornell G. 1996. *Signal processing with fractals: A Wavelet Based Approach*. Prentice Hall. 176 Pages.

Yang Y, Kim M, Lam S. 2001. *Transient Behaviors of TCP-friendly Congestion Control Protocols.* INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Pages 1716 - 1725, vol.3.