EXPERIMENTAL EVALUATION OF SCALABILITY AND RELIABILITY OF A FEEDBACK-BASED UPC-PARAMETERS RENEGOTIATION MECHANISM

Safiullah Faizullah[1], and Arshad M. Shaikh[2]

[1]Rutgers University, Piscataway, NJ, USA
[2]Isra University, Hyderabad, Pakistan

ABSTRACT

As a result of rapidly changing traffic characteristics in QoS-enabled networks oftentimes renegotiating the bandwidth requirements are needed. Once renegotiation is started, the sender keeps this process of invoking renegotiation until new requirements can be fulfilled (or until connection is eventually terminated.) The frequency of polling the network is delicate balance between huge traffic overhead traffic with decreased throughput and under-utilization of the network. While driving optimal follow-up rate is a hard problem, several efficient solutions have been proposed. We have earlier proposed a Scalable Feedback Based UPC-Parameters Renegotiation Protocol for ATM networks which gives efficient solution to this problem. The proposed solution minimizes the overhead by shifting the repeated polling away from the senders/users. Experimental evaluation of scalability and reliability aspects of our solution is presented in this paper.

KEYWORDS

ATM, Reliable, Scalable, QoS, UPC-Parameters, Renegotiation, VBR, Advertisement

1. INTRODUCTION

With increased usage of network and computer-based multimedia applications, stringent requirements are demanded by the connections generated by these applications. This, when combined with frequently changing traffic profile in the course of the connections generated by such applications like VBR [1], real bandwidth-on-demand from a QoS-enabled network, such as ATM, is required. Applications using VBR video in which instantaneous bit-rate varies widely with seen content and encoder's state. Moreover, the users of a software based multimedia application may want to resize a video window and consequently request a larger or smaller image resolution; fast-forward, rewind, or pause can be requested from a video server; or the user may suddenly start to browse an image database, etc. QoS-enabled networks are intended to provide support for such video and multimedia applications via VBR service class. Upon requesting to use such applications, the initial traffic profile representing these connections is defined through a set of traffic descriptors called UPC that includes *peak-rate*(λ_p), *burst-length*(*bl*) and *sustained-rate*(λ_s). The network admits a VBR connection based on its declared

DOI : 10.5121/ijma.2014.6103

UPC. It is expected that the terminal device will comply with the declared UPC once the connection is established. The network may enforce the declared UPC using a leaky-bucket based network policing. A source complies with its UPC if it produces up to *bl* consecutive cells with inter-cell spacing of up to $1/\lambda_p$ and, until the cell-counter is cleared, all the remaining cells arrive at inter-cell spacing of up to $1/\lambda_s$. Each time a cell is admitted the leaky bucket counter is incremented. The leaky-bucket counter is constantly decremented at the rate of λ_s. Violating cells are usually marked by the policer and discarded at the switch node where it encounters congestion. To avoid cells being discarded, the source must regulate its traffic to fit to the initially declared UPC. This may be done at the source with a combination of a leaky-bucket traffic shaper, which mirrors the policing mechanism, and source rate control. Dynamic bandwidth renegotiations are becoming popular alternatives to this static scheme. In these techniques applications/encoders initiate renegotiation every time bandwidth requirements change. If the newly requested bandwidth is not fully granted, the sender keeps on invoking the follow-up renegotiations with some frequency. While these techniques have obvious performance improvements [3-7] over their static counterparts, they have inherent disadvantages. Invoking follow-up renegotiations too infrequently results in under-utilization of the network. While polling the network too often induces a huge overhead traffic and results in a decrease in the network throughput if the network is congested. Predicting an optimal rate for follow-up renegotiations is a hard problem.

2. BACKGROUND

In the literature, we find many proposed solutions to the problem of dynamic resource allocation [2-13]. The authors of this work have proposed a renegotiation protocol based on network feedback mechanism [2, 3, 9]. We have reviewed and highlighted the shortcomings of the prominent strategies in our earlier work— thus, motivating our original work presented in [9] and we have studied the scalability of our solution in [2, 3] which showed that our approach is very promising. Now we study our scheme for reliability and show more experimentation for scalability and present the results of simulations conducted in this paper. In our approach the sender invokes a follow-up renegotiation only when the network signals it for the availability of some bandwidth.

In addition, the sender does not initiate a new renegotiation with new requirements if its previous request was not fully granted thus not unnecessarily increasing the overhead traffic. Yet any request to decrease the UPC requirements is renegotiated immediately. Thorough evaluation of the proposed scheme via extensive simulation experiments were conducted with results showing that the proposed method has significant performance benefits over its polling-based counterparts and it is scalable. Moreover, in this paper utilizing further simulation results we show that the proposed protocol is not only scalable but is it is reliable.

The rest of the paper is organized as follows: Section III briefly describes our feedback-based approach. Implementation issues are discussed in Section IV. Section V presents experimental methodology and simulation results. Finally, Section VI gives some concluding remarks.

3. FEEDBACK BASED UPC RENEGOTIATION

We outline our Feedback Based Renegotiation scheme that relies on feedback from the network instead of polling the network continuously. The sender needs not necessarily initiate renegotiation if the UPC requirements change. It invokes renegotiation for more bandwidth only if the previous demand was not unfulfilled. On the other hand, any requests to decrease the UPC requirements are renegotiated immediately. Also, if the newly requested bandwidth is not granted, the sender does not disturb the network by invoking follow-up renegotiations. The network keeps track of the unfulfilled requests and informs the sender whenever some bandwidth is available.

We use Fig. 1 to describe our strategy. As some bandwidth is released on a link (e.g., on SW1-SW2 link in Fig. 1), the switch immediately before the link (e.g., SW1) informs all the senders who may potentially use this link (e.g., VBR31, VBR32, VBR41, VBR42, VBR51, VBR52, VBR61, VBR62), that some bandwidth is now available in the network. Senders of active connections that needed more bandwidth but were denied earlier can invoke renegotiation at this point (after a random wait to avoid simultaneous initiation of requests). Note that when some bandwidth is released on a VP, each switch on that VP will send feedback to all the potential senders. As the newly available bandwidth might not be on the VP on which the sender needs more bandwidth, as such this feedback results in huge rather useless traffic not only due to the delivery of feedback information to unconcerned senders but also due to the unsatisfiable renegotiation requests from these senders. We improve this by making the switch only advertise the availability of more bandwidth only to those senders which have active connections (VC's) via that link (for example VBR31 and VBR61 in Fig. 1).

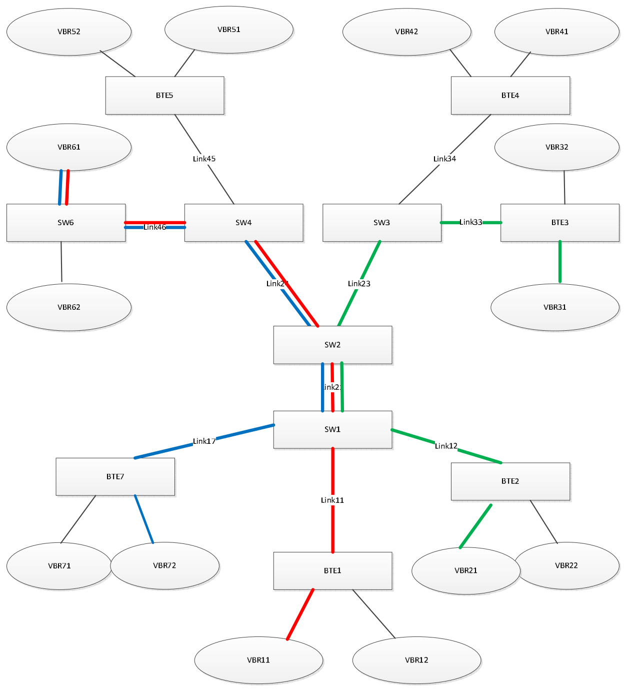

Figure 1. Sample Network

With this enhancement, the switch advertises only to those senders whose connections are bottlenecked at the next link. We maintain the bottleneck information by keeping a flag "NextLinkBottleneck" for every VC passing through a switch. Note that we flag a link as bottleneck for a connection only if the request for more bandwidth is not fully granted. Hence, this mechanism will give feedback only to those who are interested in getting more bandwidth; which is a plus point. During CAC as well as during re-negotiations, the flags are updated appropriately. On receiving feedback from the network, the senders wait for a random period of time before initiating follow-up renegotiations. This provides fairness and decreases the chance of simultaneous initiations.

A. Algorithms for the Proposed Feedback Based UPC Renegotiation Mechanism

```
/*Algorithm – VBR*/

VBRApp:- InitReneg
if ShortageInBW>0
    cell.PIT=111;
    cell.Direction=1;
    cell.RequestedBW=ShortageInBW;
    cell.AllocatedBW= cell.RequestedBW;
    send(cell);
endif
return;
VBRApp:- ReceiveBackwardRenegCell
AverageBitRate= AverageBitRate+cell.AllocatedBW;
ShortageInBW= ShortageInBW- cell.AllocatedBW;
if ShortageInBW<0 then
    ShortageInBW=0;
endif
return;
VBRApp:- ReceiveFeedbackRenegCell
wait(random);
InitReneg();
return;

/*Algorithm – Switch*/

Switch:-BackwardRenegCell_Inc_BW
if  cell.AllocatedBW= LockedBW[cell.VciVpi] then
    NextLinkBottleNeck[cell.VciVpi]=TRUE;
else
    NextLinkBottleNeck[cell.VciVpi]=FALSE;
endif
NextLink.LockedBW=NextLink.LockedBW -
                        LockedBW[cell.VciVpi];
LockedBW[cell.VciVpi]=0;
NextLink.AvailBW=NextLink.AvailBW -
                      cell.AllocatedBW;
process Reneg-cells on hold();
Forward_cell_to_next_hop(cell);
return;
```

```
Switch:-ForwardRenegCell_Inc_BW
If  cell.AllocatedBW>NextLink.AvailBW -
                         NextLink.LockedBW  then
    if  NextLink.LockedBW>0 then
        put cell on hold queue;
        return;
    else   /* if NextLink.LockedBW=0 */
        cell.AllocatedBW= NextLink.AvailBW;
    endif
endif
LockedBW[cell.VciVpi]=cell.AllocatedBW;
NextLink.LockedBW=NextLink.LockedBW+cell.AllocatedBW;
Forward_cell_to_next_hop(cell);
return;

Switch:-ForwardRenegCell_Dec_BW
NextLink.AvailBW= NextLink.AvailBW+
                        (-cell.RequestedBW);
NextLinkBottleNeck[cell.VciVpi]=FALSE;
process_Reneg_cells_on_hold();
send_feedback_to_bottlenecked_connections();
Forward_cell_to_next_hop(cell);
return;

Switch:-BackwardFeedbackRenegCell
Forward_cell_to_next_hop(cell);
return;
```

4. IMPLEMENTATION ISSUES

Several implementations related issues are detailed here. These issues were faced during the simulation stage of this work.

Signaling: Whenever an application needs to renegotiate the bandwidth, it creates a Reneg-cell specifying its requested bandwidth. The cell travels through all the links on the VP and gathers information about the minimum available bandwidth in the forward direction. On the backward trip it reserves this amount on all links. Finally, it passes this information to the application. Also, the links which become the new bottleneck are flagged during this round-trip. The application then adjusts its bit-rate accordingly.

VBR UPC Parameters: The simulator we have used for simulations uses a different set of UPC parameters which comprises of Peak bit-rate (P), Mean burst length(*bl*), and Mean interval between bursts (MIB). The difference is insignificant as one set can be derived from the other.

We calculate the Average bit-rate as follows:
Average bit-rate = P * *bl* / MIB.

We use this Average bit-rate for renegotiations and bandwidth allocation.

Reneg-cell Format: For our Reneg-cell, we adopt a format similar to that of the ABR's RM-cell with the following modifications. Here PTI is taken to be “111” which was unused previously. The data portion of the Reneg-cell contains three fields: *Direction*, *RequestedBW* and *AllocatedBW.* If Direction has the value “1”, the cell is a forward going Reneg-cell. Otherwise it is a backward going Reneg-cell. The system specifies the amount of required bandwidth in RequestedBW field which remains unchanged throughout the round trip. The AllocatedBW field keeps track of minimum bandwidth (≥0) on the VP.

Feedback Reneg-cell: A Feedback Reneg-cell is distinguished from a regular Reneg-cell by the contents of its AllocatedBW field which in case of Feedback contains a negative value.

Bandwidth Locking: On the forward direction of Reneg-cell, the switching elements lock the min{requested, available}amount of bandwidth. On its way back, when the bandwidth to be allocated is determined, each switch allocates the AllocatedBW amount of bandwidth from the locked bandwidth to this VC and releases the lock.

Wait and Go: While the Bandwidth locking avoids the chance of allocating the same piece of bandwidth to different VC’s. The adverse effect is that a request (forward Reneg-cell) normally locks more amount of bandwidth than what is actually allocated (because of the bottlenecks somewhere ahead), so the next request will see lesser amount of bandwidth which is not necessarily true. Therefore we introduce “wait and go” mechanism for the forward Reneg-cells as follows: if the amount of available bandwidth is less than the AllocatedBW and there is some bandwidth under lock (which could be released by a previous Reneg-cell on its backward journey) then the switch puts this cell on hold until either AllocatedBW amount of bandwidth becomes available or there is no more locked bandwidth. In case there is no more locked bandwidth, the Reneg-cell’s AllocatedBW is reduced to whatever amount is available, and is locked under this cell and the cell proceeds.

Priority to Renegotiation Cells: To better utilize the network resources, feedback cells are sent on priority basis. To implement that we divide the output queue into two sub queues at the scheduler of each switching element - one for priority cells and the other for regular data cells. The size of the output queue is considered to be the sum of the sizes of these two queues. The switch serves the prioritized queue first, before the data queue. If the output queue is reached to its limit, the scheduler will drop cells from the data queue before the prioritized queue.

5. Experimental Methodology And Results

In this section, we detail the experimentation methodology that we adopted for this work. Both the traditional renegotiation algorithms and our feedback based renegotiation algorithm were incorporated into an off-the-shelve simulator that we will simply refer to as NIST in this paper. We needed to add some missing functionalities to the simulator. Specially, we converted the static CAC module to handle dynamic call admission and termination. Further, we modified the VBR application to change its UPC requirements over time.

So that we compare repeated renegotiation approach versus proposed network feedback mechanism, we conducted large number of experiments with varying follow-up rates, and at least two experiments with the proposed technique and studied the utilization/throughput of the network. Figure 2 shows one of the model networks used in the simulations. For scalability, we used a variation of network topologies. Table 1 shows a sample variation in UPC parameter requirements; these were changed for each run of the simulation. Note that the sample networks are kept congested since each source's average bit-rate is quite larger than the capacity of the links which is fixed as 50Mbps for all links in these experiments. The network configuration varied from simple network (Fig. 1 and Fig. 2) to more complex/campus like network (as shown in Fig. 4 where the internal network shown as cloud varied from simulation to simulation.) We can see in the simulation results (Fig. 3, from [3, 9]) that the network remained under- utilized for low follow-up rate. For higher follow-up rates the renegotiation overhead is so much that it reduces the throughput drastically. On the other hand, the network throughput using the proposed feedback based technique is clearly better than the repeated follow-up scheme. Figure 5 and Fig. 6 show that the proposed scheme is highly scalable as the renegotiations overhead is relatively low and remain so under varying congestions for various network topologies. The networks studied varied from simple, to campus like, and finally complex wide area network configuration. Each of these networks we studied under varying network congestions (low demand, medium demand and high demand scenarios.) As expected, the renegotiation overhead varied but remained bounded with low unutilized bandwidth.

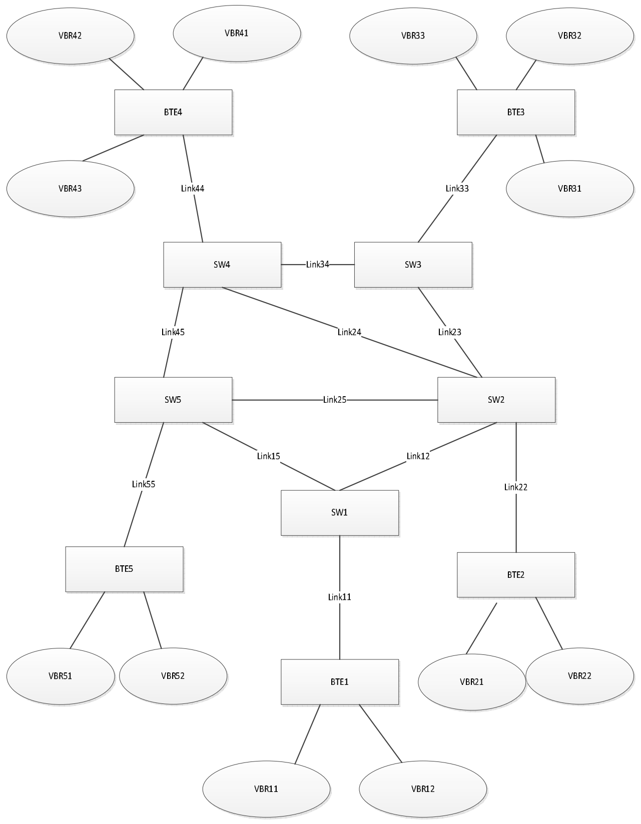

Figure 2. Results- the proposed scheme vs. others.

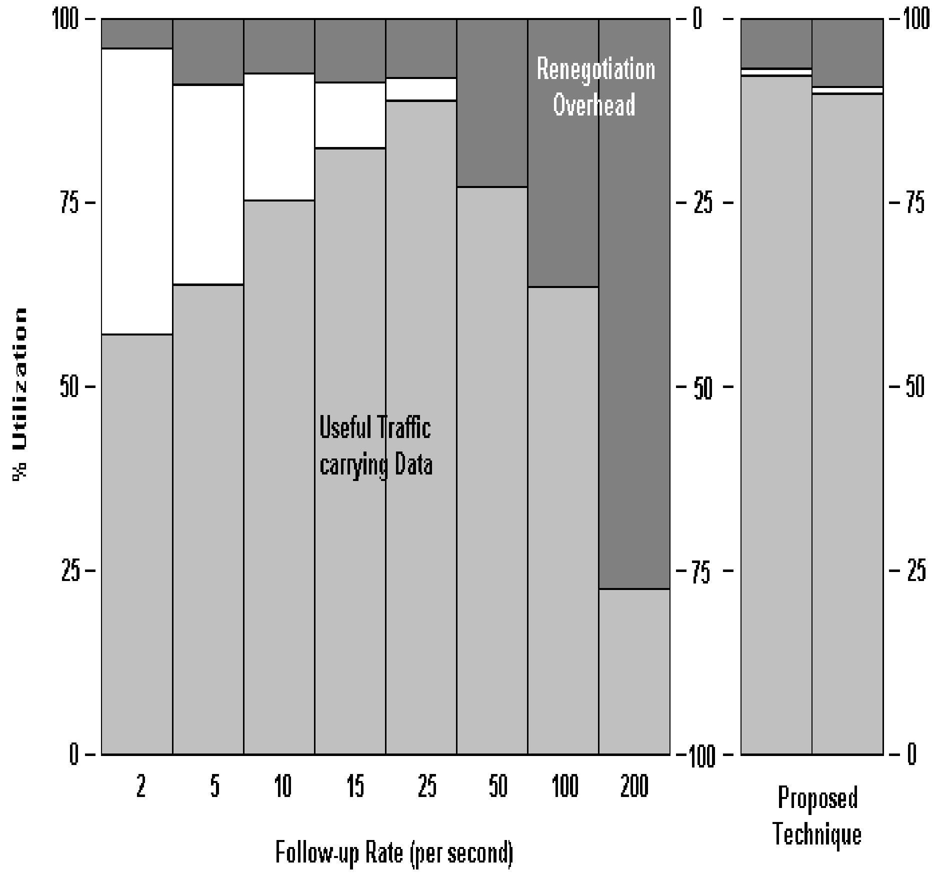

Figure 3. Results- the proposed scheme vs. others

Table 1. Sample UPC Variation in Simulated Networks

We further studied the proposed scheme under several nodes failure scenarios (1-5, 6-10, and 11-15 nodes failing) for various network complexity setups (simple, campus like, and complex) with varying network loads ranging from low demand, medium demand and high demand. The results shows that the proposed scheme is stable with renegotiation overhead remaining bounded with low unutilized bandwidth. We notice that the overhead increases with node failures but the overall renegotiation overhead results are 8% total bandwidth in the worst case scenario. The unutilized bandwidth also remaining low for even the high demand loads on complex networks.

Table 1. Sample UPC Variation in Simulated Networks

	Peak	Burst	Inter-burst	
	Rate	Length	Interval	
Sender	Receiver	(Mbps)	(uSecs)	(mSecs)
vbr11	vbr32	80±30	400±10%	0.8±10%
vbr12	vbr22	65±15	800±10%	1.4±10%
vbr21	vbr12	140±40	500±10%	0.7±10%
vbr22	none	0	0	0
vbr31	vbr21	80±30	400±10%	0.8±10%
vbr32	none	85±20	500±10%	0.8±10%
vbr33	vbr43	90±25	500±10%	1.0±10%
vbr41	vbr33	100±30	600±10%	0.9±10%
vbr42	vbr22	70±20	900±10%	2.0±10%
vbr43	none	0	0	0
vbr51	vbr11	55±40	700±10%	1.5±10%
vbr52	vbr42	80±15	300±10%	0.8±10%

6. Concluding Remarks And Future Work

We have previously introduced a feedback driven scheme for UPC parameter renegotiation. Several simulation experiments were conducted to evaluate the strategy. Results show that the proposed method has significant performance benefits over its polling-based counterparts. Additional simulations conducted under various network topologies and various load support and highlight that the protocol is scalable and reliable. Our future research goal is to study the merit of the protocol further investigated when feedback is delivered by a multicast instead of multiple unicasts. It is also of interest to extend this work to wireless networks. Finally, we would like to implement our scheme using campus-like large real network scenario.

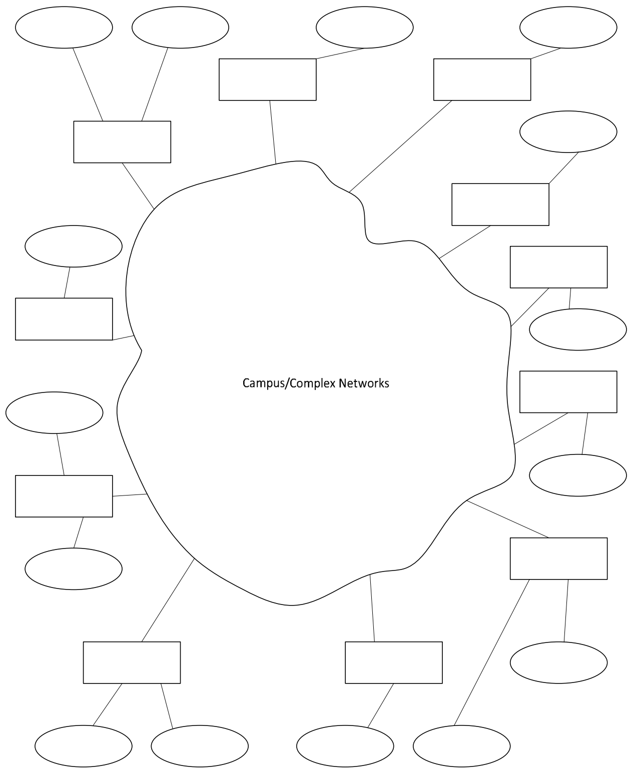

Figure 4. Results- the proposed scheme vs. others.

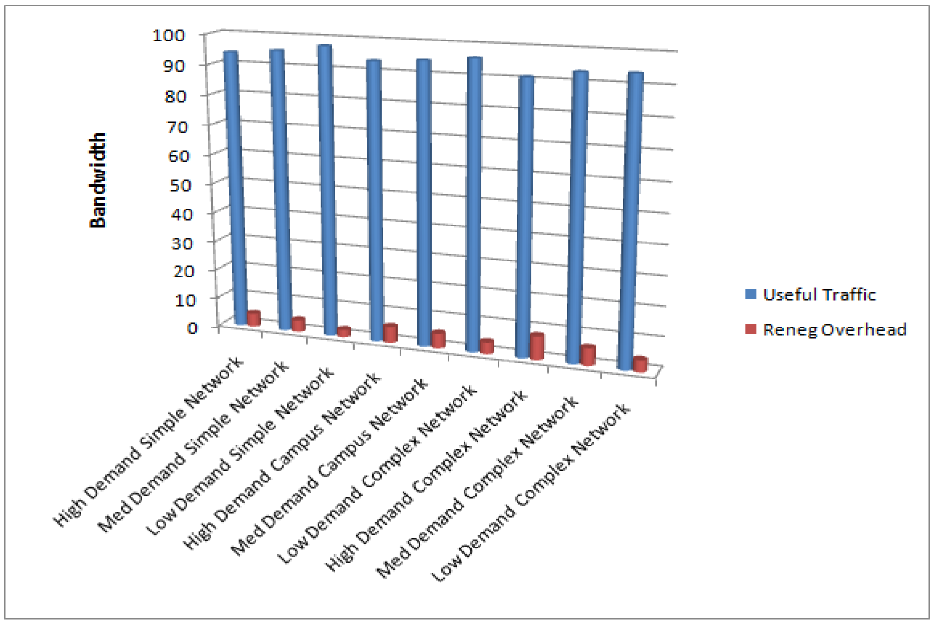

Figure 5. Results- the proposed scheme vs. others

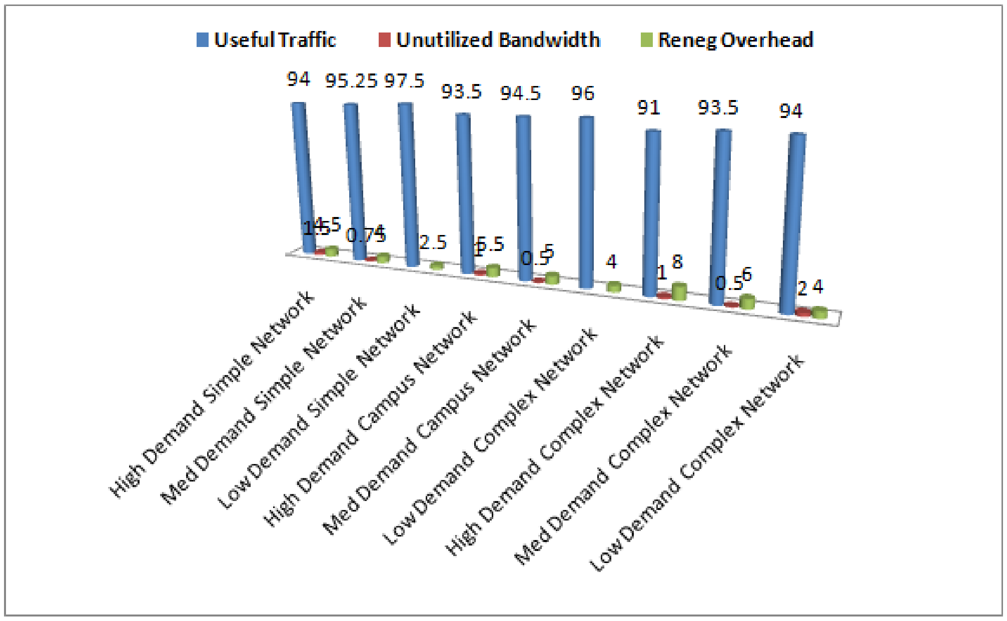

Figure 6. Results- Scalability of the Proposed Scheme

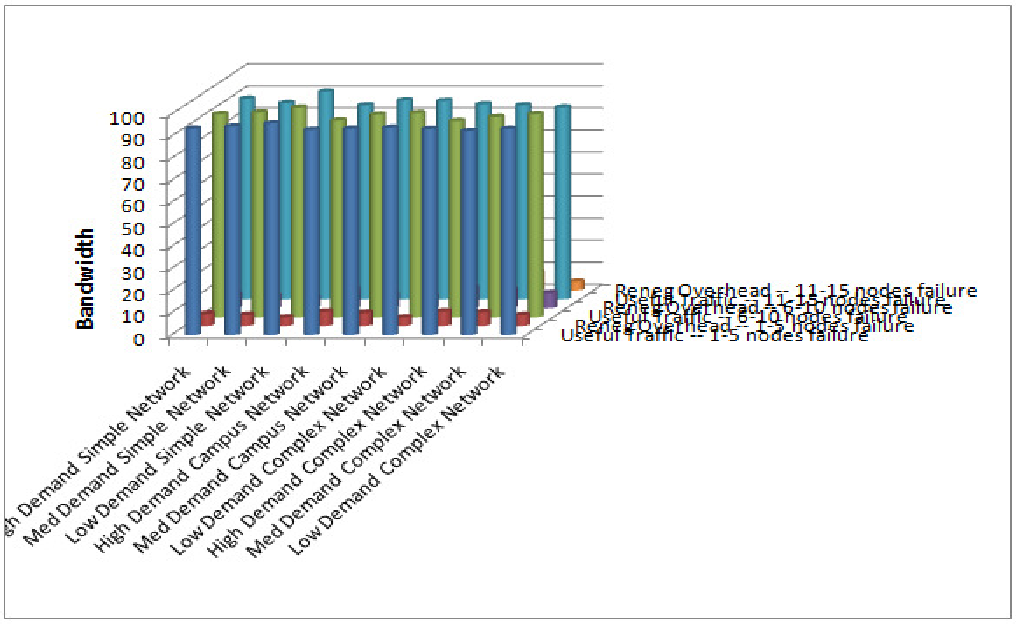

Figure 7. Results- the proposed scheme vs. others.

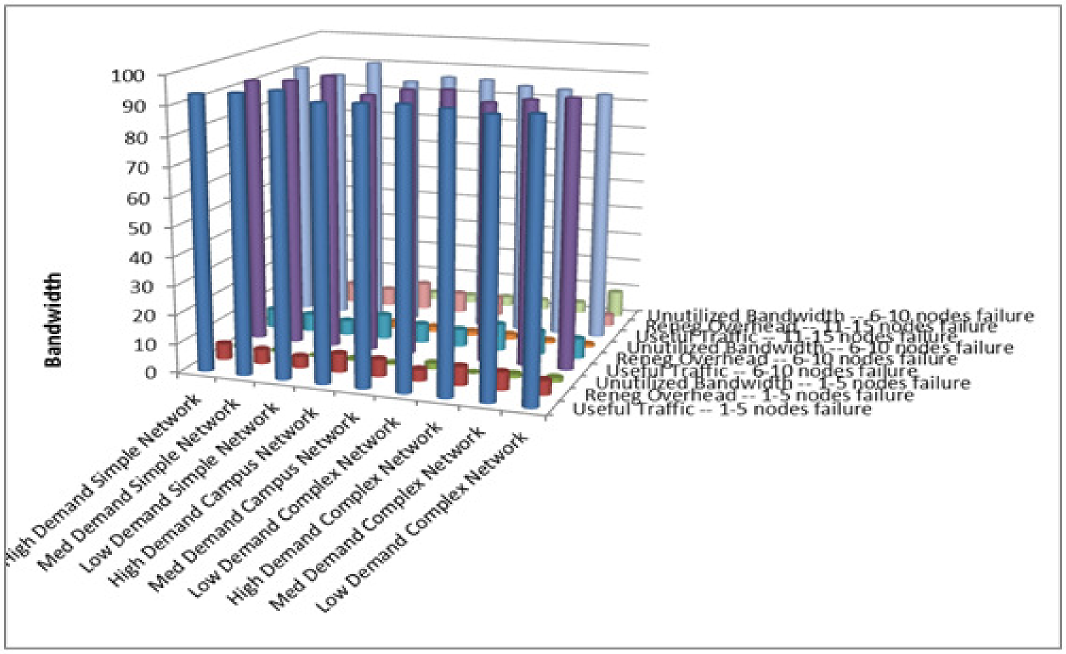

Figure 8. Results- the proposed scheme vs. others.

REFERENCES

[1] MPEG-4 Standards ISO/IEC JTC1/SC29/WG11 N2201, '98.

[2] S. Faizullah and A. Shaikh, "A Scalable Feedback-Based UPC-Parameters Renegotiation Scheme", The 2nd International Conference on Network, Communication and Computing (ICNCC 2013), Kuala Lumpur, Malaysia, Dec. 29-30, 2013. http://www.icncc.org/ (published in IJSPS 2013.)

[3] S. Faizullah and A. Shaikh, "A Scalable Feedback-Based UPC-Parameters Renegotiation Scheme", International Journal of Signal Processing Systems Vol.1, No. 2, 2013, pp 302-306.

[4] R. Ulrich and P. Kritzinger, "Managing ABR Capacity in Reservation-based Slotted Networks", Tech. Report, ICSI, 1996.

[5] K. Nahrstedt and J. M. Smith, "Revision of QoS guarantees at the application network interface", Technical Report, University of Pennsylvania, Distributed Systems Lab, 1994.

[6] S. Chong, S. Li, and J. Ghosh. "Dynamic Bandwidth Allocation for efficient transport of real-time VBR video over ATM", Proc. of the IEEE INFOCOM Conf., pp. 81-90, 1994.

[7] Vikas Tyagi, Vivek Tyagi and Smita Sharma, 'Traffic Shaping by Statistical Characterization in ATM Networks", Intl. Journal of Essential Science Vol.5, No.1 2011, pp 78-82.

[8] D. J. Reininger, D. Raychaudhuri and J. Y. Hui, "Bandwidth renegotiation for VBR video over ATM networks", IEEE Journal on Selected Areas in Communications, Volume 14 Issue 6, September 2006, pp 1076-1086.

[9] S. Faizullah and A. Shaikh, "A Feedback-Based UPC-Parameters Renegotiation Strategy", Proc. Of the Internl Conf. On Parallel and Distributed Processing Techniques and Application (PDPTA '99), Las Vegas, NV, Jun 28-Jul 1, '99, pp. 937-943.

[10] Daniel Reininger, Gopalakrishnan Ramamurthy, Dipankar Raychaudhuri, "VBR MPEG Video Coding With Dynamic Bandwidth Renegotiation", IEEE International Conference on Communications,. 95-R-008 (AP)

[11] B. Mark and G. Ramamurthy, "Real-time Estimation and Dynamic Renegotiation of UPC Parameters for Arbitrary Traffic Sources in ATM Networks", IEEE/ACM Transactions on Networking (TON), Volume 6 Issue 6, Dec. 1998, pp 811 – 827

[12] M. T. Andrade and Artur Pimenta Alves, “Experiments with Dynamic Multiplexing and UPC Renegotiation for Video over ATM”, Proc. NETWORKING '00/Proceedings of the IFIP-TC6 / EU Com. Intl Conf. on Broadband Com., High Perf. Networking, and Perf of Com. Networks, pp 895-907

[13] Alins, J., J. Pegueroles, J. Mata, and L. J. de la Cruz Llopis, "Two-level renegotiated constant bit-rate algorithm (R-CBR) for stored video services over QoS networks", IASTED international Conference on Communications Systems and Networks, 2002.

AUTHORS

Safi Faizullah received his Ph.D. in Computer Science from Rutgers University, New Brunswick, New Jersey, USA in 2002. He also received MS and M. Phil. Degrees in Computer Science from Rutgers University, New Brunswick, New Jersey, USA in 2000 and 2001, respectively. Dr. Faizullah also earned his BS and MS degrees in Information and Computer Science from KFUPM, Dhahran, KSA in 1991 and 1994, respectively. His research interests are in computer networks, mobile computing, wireless networks, distributed and enterprise systems. He has authored over twenty refereed journals and conference papers. Dr. Faizullah works for Hewlett-Packard and he is a Visiting Scholar/Adjunct Professor of Computer Science at Rutgers University. He is a member of IEEE, SCIEI, PMI and ACM.

Arshad M. Shaikh received his MS degree in Computer Science from Rutgers University, New Brunswick, New Jersey, USA in 2002. He also received MS and BS from National University, Karachi, Pakistan. His research interests are in programing languages and computer networks. He has authored several conference papers. Mr. Shaikh worked for National University, Karachi, Virtual University, and Isra University, Hyderabad, Pakistan as lecturer.