The ERICA ALGORITHM for ABR TRAFFIC in ATM NETWORKS

Ibrahim Koçyigit
Department of Electronics, Faculty of Engineering,
Uludag University, Görükle, Bursa TURKEY
E-mail: kocyigit@uludag.edu.tr

Emrah Yürüklü
Department of Electronics, Faculty of Engineering,
Uludag University, Görükle, Bursa TURKEY
E-mail: emrah79@yahoo.com

Keywords*;* *ATM, ABR, CBR, ERICA scheme*

ABSTRACT

In this paper, we explain a congestion avoidance switch algorithm called ERICA (Explicit Rate Indication for Congestion Avoidance) for ABR traffic management in ATM networks, and present some simulation results for a simple ATM network using this algorithm. The networks using this scheme monitor the load on the link and determine a load factor, the available capacity and the number of active VCs (virtual channels). The network will use these information to keep the link utilization high, i.e. advice the sources about the rates they should transmit. So this way sources can use network resources much more efficiently, with smaller queuing delays and transient response.

I. INTRODUCTION

The ATM Forum Traffic Management Specification introduces ABR service category details precisely like source and destination system behaviours. But it gives switch behaviour specification coarsely. So this means several switch algorithms can be developed. ERICA algorithm was presented at the ATM Forum in Feb. 1995.
Firstly, we have to say what the congestion is for ATM networks. Congestion causes cell losses in excess of traffic contract that was made while establishing the connection. So we have to do some controls on ATM network. ERICA is a traffic control algorithm, which sets actions to avoid congestion.
As we mentioned before, the ABR service category uses the leftover bandwidth after the other service categories transmitted. For this reason we can say ABR service is not intended for real time applications. The ABR service provides better service for data traffic by periodically advising sources about the rate, which they should be transmitting. The switches monitor their load, compute the available bandwidth and divide it fairly among active flows. This allows competing sources to get at least the minimum bandwidth that is required for connections survive. The ABR flow control uses some special ATM feedback cells named RM (Resource Management) cells.
The RM cells are sent periodically and turned around by the destinations periodically (see figure 1). When the RM cells came back to the source (a backward RM cell), it contains the information about the situation of the switches on the route. Basically a RM cell has some fields to provide feedback like; Explicit Rate (ER), Congestion Indication (CI) and No Increase (NI). When starting at the source, the ER field is usually set to PCR (Peak Cell Rate), and the CI and the NI flags are clear. On the path, each switch reduces the ER field to the maximum rate that the switch can support and sets CI and NI if necessary. When a source receives a RM cell, it sets its allowed cell rate (ACR) using its current ACR, CI and NI flags, and the ER field of the RM cell.

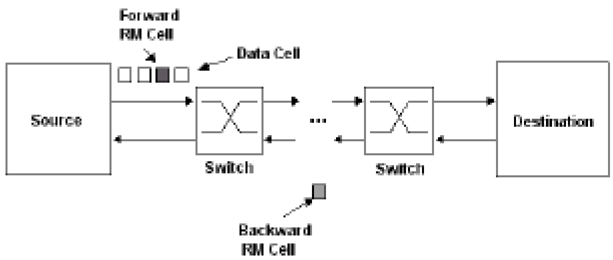

Figure 1. RM Cell Path

We assume that there are two service categories are mostly used, ABR and CBR. We will use these two most popular service categories in our simulations. Of course ABR has the low priority, i.e. its cells will be transmitted after CBR cells are transmitted. In practice, minimum capacity should be reserved for processing aggregate ABR traffic when there is contention.

II. THE BASIC ERICA ALGORITHM

The ERICA Algorithm operates at each output port (or link) of a switch. The switch periodically monitors the load on each link and determines a load factor (z), the ABR capacity and the number of currently active VCs (virtual channels). In every averaging (measuring) interval (the interval between two RM cells), these quantities will be updated. Further, the switch gives no or one new feedback per source in any averaging interval. The load factor (also called overload factor) is calculated as the ratio of the measured input rate at the port to the target ABR capacity.

$$z \leftarrow \frac{ABR\ Input\ Rate}{Target\ ABR\ Capacity} \quad (1)$$

$$Target\ ABR\ Capacity \neg Fraction ´ Total\ ABR\ Capacity \quad (2)$$

$$Total\ ABR\ Capacity \neg Link\ Capacity\text{-}CBR\ Capacity \quad (3)$$

Fraction is the value of the function f (Q), called “queue control function”. We will describe this function later. The load factor, *z*, is an indicator of the congestion level of the link. High load factor values are undesirable, because it indicates excessive congestion, so low values indicate underutilization. The goal of the switch is to hold *z* in the neighbourhood of unity.

The fair share of each VC is computed as follows:

$$FairShare \leftarrow \frac{Target\ ABR\ Capacity}{Number\ of\ Active\ VCs} \quad (4)$$

The above steps are executed at the end of the switch averaging interval.

The switch allows each source sending at a rate below the *FairShare* to rise their rate to *FairShare.* If the source does not use all of its *FairShare,* then the switch fairly allocates the remaining capacity to the sources which can use it. For this purpose, the switch calculates the quantity of *VCShare:*

$$VCShare \leftarrow \frac{CCR[VC]}{z} \quad (5)$$

If all VCs changed their rate to their *VCShare* values, then in the next cycle the switch would experience unit overload. Hence *VCShare* aims to bring the system to an efficient point. Although *FairShare* tries to ensure the fairness, this point might not be fair. Because fairness is not always necessary in some kind of situations. A combination of these two quantities is used to rapidly converge to the steady state conditions as follows:

$$ER \neg Max\ (FairShare,\ VCShare) \quad (6)$$

ER cannot be greater than *Target ABR Capacity*. So if the calculated *ER* is greater than *Target ABR Capacity*, our new *ER* value will be computed as follows:

$$ER \neg Min\ (ER,\ Target\ ABR\ Capacity) \quad (7)$$

Sources are allowed to send at a rate of at least *FairShare* within the first round-trip of the RM cell. This ensures the minimum fairness between the contending sources. If the *VCShare* value is greater than the *FairShare* value, the source will be able to transmit its data at *VCShare*, so the link won’t be underutilized. This step also allows an unconstrained source to convergence its max-min rate. Step (6) of ERICA is one of the key innovations. Because it improves the fairness at every cycle, even under overload conditions. Step (7) is implemented to ensure that the bottleneck *ER* reaches to source. As we pointed out before, each switch computes the minimum of the *ER* it has calculated at step (7) and the *ER* value in the RM cell. This value is inserted in the *ER* field of the RM cell:

$$ER\ in\ RM\ cell \neg Min\ (ER\ in\ RM\ cell,\ ER) \quad (8)$$

A flow chart of the basic algorithm is presented in figure 2. The flow chart shows steps to be taken on three possible events: at the end of an averaging interval, on receiving a cell (data or RM), and on receiving a backward RM cell. By the way we have to say, “CCR [VC]” means “this VC’s CCR”.

The description we have given is the core of the ERICA algorithm. However, we haven’t mentioned, but this core algorithm has some drawback cases. So in order to overcome this disadvantages, some enhancements made on this basic algorithm. As we know, today ERICA+ is developed. Although we won’t discuss the entire ERICA+, we will represent one of its special queue control functions.

A simple queue control function is the constant function, i.e., a fixed parameter. This function (called “*Target Utilization (U)*”) is used in earlier ERICA versions. The drawbacks of a constant function are; it restricts the system utilization to a maximum of *U* in steady state, the system cannot achieve a queuing delay target, and it does not provide compensation when measurement and feedback are affected by errors. The alternative is to have f (Q) vary depending upon the queuing delay. A number of such functions can be designed. One such adequate function for LANs (small round-trip), and WANs (low error/variance) was found by Raj Jain. We will use this function for our simulations in future.

This function is the following (also see figure 3):

$$f(Q) = \frac{a \times Q0}{(a-1) \times Q + Q0} \quad \text{if } Q > Q0 \text{ and}$$

$$f(Q) = \frac{b \times Q0}{(b-1) \times Q + Q0} \quad \text{if } 0 \le Q \le Q0$$

f(Q) is a number between 1 and 0 in the range Q0 to infinity, and between *b* and 1 in the range 0 to Q0. Notice that these two functions are rectangular hyperbolic functions. In addition, the function is lower bounded by the queue drain load factor, $QDLF = F_{min}$ (in figure 3).

With this modification we ensure the minimum capacity for ABR service. So our modificated function is:

$$f(Q) = Max\,(QDLF, \frac{a \times Q0}{(a-1) \times Q + Q0}) \quad \text{if } Q>Q$$

This function is implemented to control the queuing delay and to maintain parameter uniformity for different link speeds. ERICA needs four parameters to implement this function: *T0* (target queuing delay which will be converted into *Q0* target queue length), *QDLF*, *a* and *b*.

We chose this function for our simulations, because of the fewer points of discontinuity, smaller number of parameters required, and it's not too complicated to implement.

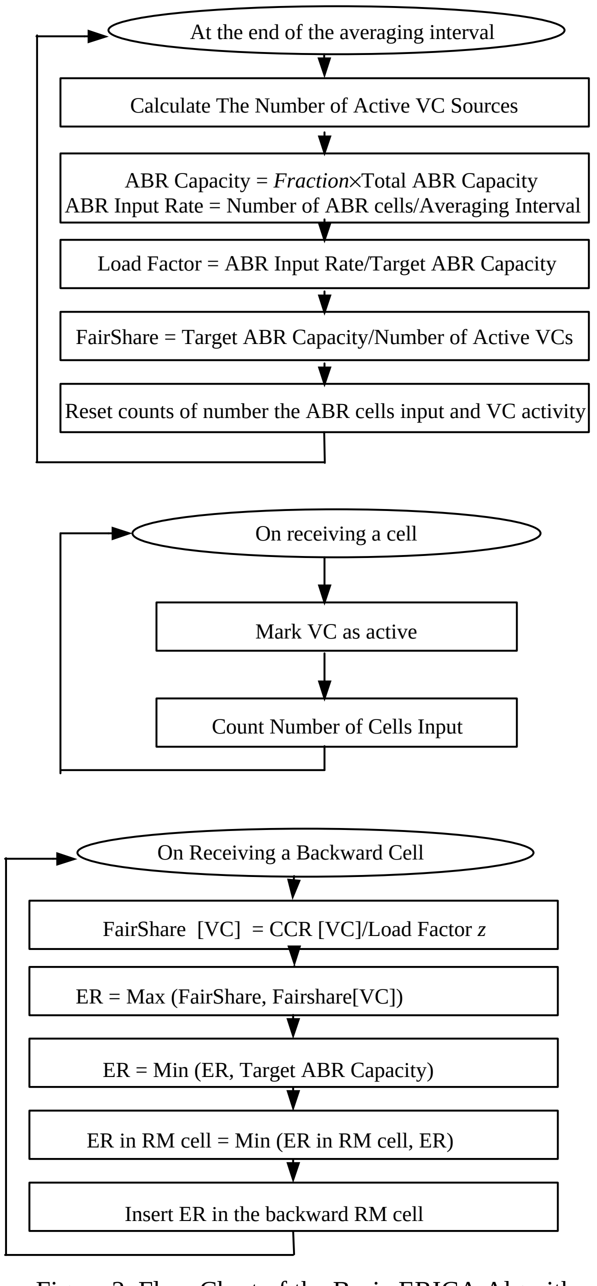

Figure 2. Flow Chart of the Basic ERICA Algorithm

The features of the functions are as follows. It assumes the value of 1 as the desired steady state (which means utility = 100%, queuing delay = *T0*). The parameter *T0* specifies the target delay, but also affects how quickly excess the queues are drained. A larger *T0* results in slower allocation of drain capacity. So *T0* is can be to set to small values as possible, in order to satisfy the primary goal, to quickly drain excess queues. And an important use of this function is to compensate for measurement and feedback errors caused by system (load/capacity/source activity) variation. The parameter *QDLF* defines the tolerance limit of the system to such this variation. If variation is large, other techniques like the use of larger averaging intervals, and long-term averaging metrics must be combined with queue control for ensuring robustness and effective control of the system.

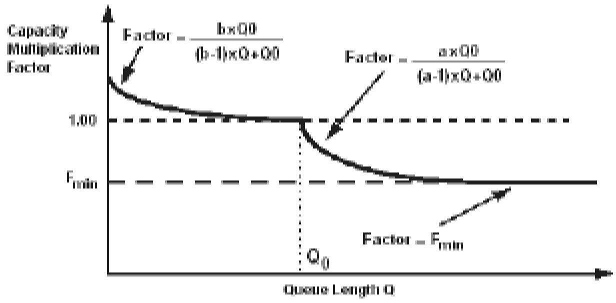

Figure 3. The Queue Control Function

III. SIMULATION SCENARIO

Our simulation scenario is a simple bottleneck configuration with two nodes. We have 3 TCP transmitters connected to node 1, and 3 receivers of these transmitters respectively connected to node 2. And also we added a CBR source to node 1, and a CBR destination to node 2. We used three different distances for the link length between two nodes: 10km, 100km, 1000km. We neglect the distances between the nodes and the users (see figure 4).

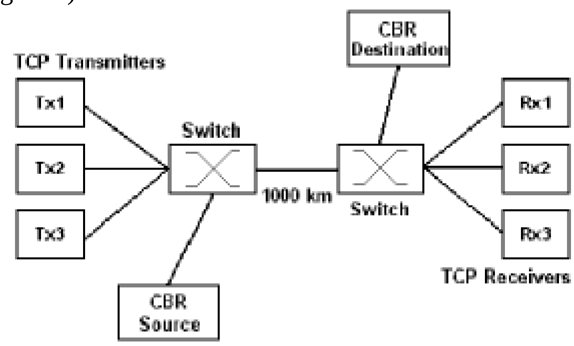

Figure 4. The Bottleneck Topology

The three TCP connections are used for large file transfers. But they just get the remaining bandwidth left from CBR. The link bandwidth for all links is 150 Mbps. All of the TCP transmitters and receivers have the same characteristics. Their maximum window size is 30 segments. And it is supposed that TCP protocol always transmits segments of the maximum possible size (we set this parameter to 9180 bytes). The CBR user is an on-off CBR traffic producer. Its mean on time and mean off time is 100 ms.

In this configuration, TCP users have ABR service categories. So their ABR parameters are PCR=150 Mbps, MCR=0.1 ms, RIF=1, RDF=1, NRM=32, TBE=512, ICR=5 Mbps. As we mentioned before, we use ERICA for congestion control and Generic Cell Rate Algorithm (GCRA) for traffic control. The ERICA parameters are as

follows: δ=0.1 (It's a small fraction that was added to *z*, in order to converge to max-min allocations), a=1.15, b=1, T0=10 ms, QDLF=0.5. The averaging interval is set to 100 cells (it means there is one RM cell in a 100 cells) You can find detailed information about these parameters in Reference [1].
All the links run at 150 Mbps. And all their buffers (switch buffers) are to set to either 1000 cells or 5000 cells. We adjusted the user buffers to 20000 cells, in order to prevent user losses.
We will examine ERICA by changing the demand rate of CBR source, and observing the receiver goodput (the useful throughput, at the TCP receivers, obtained considering the received data but discarding all the faulty and the retransmitted segments) and the good efficiency (the ratio between the goodput and the total offered load of TCP connections) of the TCP traffic. And we will do this examination for different link lengths and buffers.
We use CLASS (ConnectionLess ATM Services Simulator) program for our simulations. It is an ATM simulator, which was developed by Italian Telecommunication and Network System Research Center. Its version is 6.20h and it is sufficient for most of all ATM network topologies.

IV. SIMULATION RESULTS

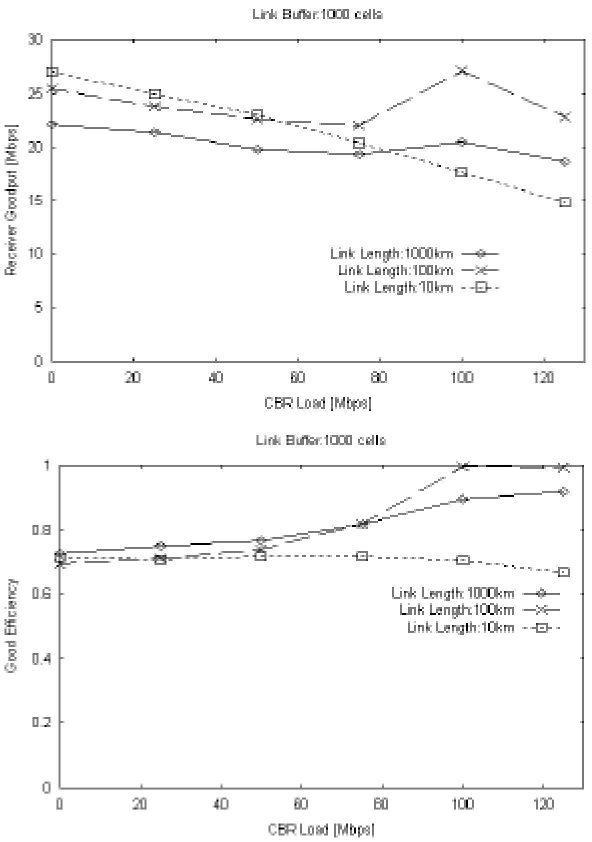

Figure 5. Average receiver goodput and efficiency of TCP connections for link buffer=1000 cells.

As we increase the CBR demand, we saw that TCP receiver goodput is becoming worse and the efficiency is getting better (see figure 5 and 6).
When there is no CBR usage on the links, each TCP connection has 50 Mbps. But in the simulations we see that they all have just about 25 Mbps for the link buffer of 1000 cells. This is because of low link buffer. It is clear that as the buffer sizes get larger; the probability of losing cells becomes lower (The window size issue is also involved). You can see 41 Mbps with the link buffer of 5000 cells. And when TCP gets lower bandwidth allocation its goodput decreases (Because they are able to send less segments). But it's an interesting result that that the rise of the efficiency. The efficiency is supposed to fall down as the CBR load increases. As we know ERICA is a congestion control algorithm. So the ERICA's advantage shows itself from this point of view. At low CBR demands, there are not so large queues, and ERICA does not affect the results so much. But when congestions are existed, ERICA starts to affect the efficiency. This explains the small falling down of the efficiency on the link with the capacity of 5000 cells.

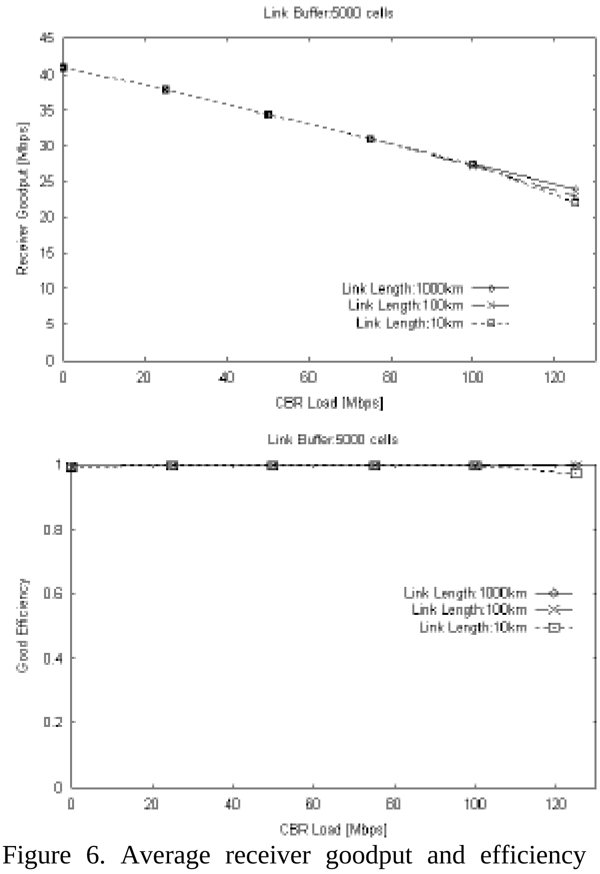

Figure 6. Average receiver goodput and efficiency of TCP connections for link buffer=5000 cells.

As we can see in the figures (especially in figure 5), the long link lengths make receiver goodput worse, and the efficiency better. As the distance gets closer, the round trip time is becomes smaller. So TCP becomes more attack, and it causes more cell losses but sending more segments (so receiver goodput increases).

If we compare the two simulations with the link capacities of 1000 and 5000 cells, we can see easily the recovering effect of the large buffers. Both of efficiency and goodput are getting better as the links have higher capacities. Because, large buffers don't allow high amount of cell losses to occur.

Although there is a break point at about 100 Mbps in figure 5, these results are good for this ATM network configuration. This break point is because of low buffer size and getting closer to limit of the link capacity.

CONCLUSIONS

In this paper, we have described the design and evaluation of ERICA switch algorithm for ABR congestion control. We have also studied on the behaviours of the ERICA scheme. In addition, we present simulation results which were realized on the CLASS simulator. These results illustrating the efficiency and goodput of the ABR traffic. We saw that ERICA is sufficient for the worst case of a ATM network. We can say buffers can be enlarged in order to increase the goodput and efficiency for long distances.

The ERICA scheme has considerably influenced the design of contemporary switch mechanism. Notably, the ATM Forum traffic management specification 4.0 [1] gives ERICA as an example switch mechanism.

REFERENCES

[1] "The ATM Forum Traffic Management Specification Version 4.0", ATM Forum Traffic Management AF-TM-0056.000, April 1996.

[2] Kalyanamaran, S., Jain, R., Fahmy, S., Goyal, R., Vandalore, B., "The ERICA Switch Algorithm for ABR Traffic Management in ATM Networks", Ohio, July 1996.

[3] Kalyanamaran, S., "Traffic Management for tha Avaliable Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) Networks", Ph.D. Dissertation, Dept. Of Computer and Information Sciences, The Ohio State University, August 1997.

[4] Kalyanamaran, S., Jain, R., Fahmy, S., Goyal, R., Lu, F., Srinidhi, S., "Performance of TCP/IP over ABR Service on ATM Networks", Globecom '96, London, 1996.

[5] Charny, A., Clark, D. D., Jain, R., "Congestion Control With Explicit Rate Indication", The Ohio-state University, Ohio.

[6] "Class User Manual", Politecnico di Italy, Dipertimento di Elettronica, Italy, May 1998.

[7] Koçyigit, I., "X.25 Paket Anahtarlamali Telekomunikasyon Sebekesinde Yüksek Debili Veri Iletimi", ELECO 2000, 8-12 December 2000, Sagem-Bursa.

[8] Yürüklü, E., "ATM Sebekelerde Hücre Seviyesi Analizleri ", Bursa, 2001.