World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:2, No:1, 2008

# Comparison of Different Types of Sources of Traffic Using SFQ Scheduling Discipline

Alejandro Gomez Suarez, and H. Srikanth Kamath



***Abstract*—**In this paper, SFQ (Start Time Fair Queuing) algorithm is analyzed when this is applied in computer networks to know what kind of behavior the traffic in the net has when different data sources are managed by the scheduler. Using the NS2 software the computer networks were simulated to be able to get the graphs showing the performance of the scheduler. Different traffic sources were introduced in the scripts, trying to establish the real scenario. Finally the results were that depending on the data source, the traffic can be affected in different levels, when Constant Bite Rate is applied, the scheduler ensures a constant level of data sent and received, but the truth is that in the real life it is impossible to ensure a level that resists the changes in work load.

***Keywords*—**Cbq, Cbr, Nam, Ns2.

## I. Introduction

SCHEDULING disciplines [1] are important because they are the key to fairly sharing network resources and to providing performance – critical applications with performance guarantees.

Scheduling disciplines has 2 orthogonal components:

1- It decides the order in which requests are serviced.
2- It manages the services queue of request awaiting service.

In a data communication network packets belonging to different traffic flows often share links in the way to their destination. When a node cannot send all the packets it receives in a particular moment, packet queues are originated.

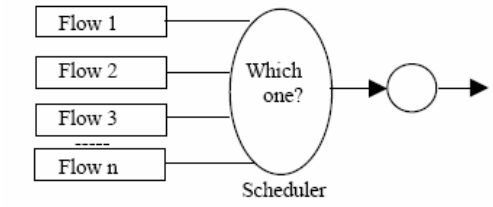


Fig. 1 Scheduler

Manuscript was received on December 07, 2007. This work was done in Department of Electronics and Communication Manipal Institute of Technology Manipal-576104.

Alejandro Gomez Suarez is IAESTE Student from Mexico, Department of Electronics and communications Engineering, Universidad Autónoma de Nuevo Leon, Pedro de Alba S/N Monterrey, Nuevo Leon, Mexico (e-mail:alejandro.gomezsuarez@gmail.com).

H. Srikanth Kamath is Senior Lecturer in Department of Electronics and Communications Engineering, Manipal Institute of Technology, Manipal, Udupi – 576 104, Karnataka, India (e-mail: kamath_srikanth@hotmail.com).

Flow control procedures [2] are required to manage these queues by regulating the order and amount of data each source can transmit.

SFQ [4] algorithm computes both the finish number and the start number of each arriving packet. The start number of a packet arriving at an inactive connection is set to the current round number. A packet's finish number is the sum of its start number and its packets size divided by its weight.

The algorithm is:

1- On arrival, a packet $p_f^i$ is stamped with start tags $S(p_f^i)$ computed as:

$$S(p_f^j) = \max\{v[A(p_f^j)], F(p_f^{j-1})\} \quad j \geq 1 \qquad (1)$$

$F(p_j^f)$= the finish tag is defined by:

$$F(p_j^f) = S(p_i^f) + \frac{l_f^i}{\phi_f} \quad j \geq 1 \qquad (2)$$

Where $F(p_f^0) = 0$ $\phi_f$ is the weight of flow f.

2- Initially the server virtual time is 0. During a busy period, the server virtual time at time t, $v(t)$ is defined to be equal to the start tag of the packet in service at time t. At the end of a busy period, $v(t)$ is set to the maximum of finish tag assigned to any packet that has been served.

3- Packets are serviced in the increasing order of the start tags.

The computation of $v(t)$ in SFQ is inexpensive since it only involves examining the start tag of packet in service.

Fluctuation in service rate may also occur due to variability in CPU capacity available for processing packets.
SFQ for servers with bounded fluctuation in service rate:

### *A. Fluctuation Constrained (FC)*

It has 2 parameters, average rate C (bits/s) and burstiness $\delta(c)$. In a FC server, the time taken to serve packets of aggregate length $w$ in a busy period can exceed the time taken in an equivalent constant rate server by at most $\delta(c)$.

$$T(w) \leq \frac{w}{c} + \delta(c) \tag{3}$$

*B. Exponentially Bounded Fluctuation (EBF)*

It is a stochastic relaxation of the FC server. The probability of the time taken to serve packets of aggregate length $w$ in a busy period deviating by more than $\gamma$ from that in an equivalent period constant rate server, decreases exponentially with $\gamma$.

$$P\left[T(w) > \frac{w}{c} + \delta(c) + \gamma\right] \leq Be^{-\alpha\gamma} \tag{4}$$

To derive fairness guarantee of SFQ, we need to prove a bound on

$$\left|\frac{W_f(t_1,t_2)}{\phi_f} - \frac{W_m(t_1,t_2)}{\phi_m}\right| \tag{5}$$

For any intervals in which both flows $f$ and $m$ are backlogged. We achieve this objective by establishing a lower and upper bound on $w_f(t_1,t_2)$.

If flow $f$ is backlogged throughout the interval $(t_1,t_2)$, then in an SFQ server.

$$\phi_f(v_1 - v_2) - l_f^{max} \leq w_f(t_1,t_2) \tag{6}$$

Where $v_1 = v(t_1) and\ v_2 = v(t_2)$

## II. Software used During Simulations

For this project, the simulations were done in NS2 [3] software which is an event driven network simulator developed at UC Berkeley that simulates variety of IP networks. It implements network protocols such as TCP and UPD, traffic source behavior such as FTP, Telnet, Web, CBR and VBR, router queue management mechanism such as Drop Tail, RED and CBQ, routing algorithms such as Dijkstra, and more. NS also implements multicasting and some of the MAC layer protocols for LAN simulations. The NS project is now a part of the VINT project that develops tools for simulation results display, analysis and converters that convert network topologies generated by well-known generators to NS formats. Currently, NS (version 2) written in C++ and OTcl (Tcl script language with Object-oriented extensions developed at MIT) is available.

Ns2 can be built and run both under Unix and Windows. Instructions on how to install ns2 on Windows can be found at: http://www.isi.edu/nsnam/ns/ns-win32-build.html. However, the installation may be smoother under Unix.

**Starting ns**

NS is started with the command 'ns <tclscript>' (assuming that you are in the directory with the ns executable, or that your path points to that directory), where '<tclscript>' is the name of a Tcl script file which defines the simulation scenario (i.e. the topology and the events). You could also just start ns without any arguments and enter the Tcl commands in the Tcl shell, but that is definitely less comfortable.

Everything else depends on the Tcl script. The script might create some output on stdout, it might write a trace file or it might start nam to visualize the simulation.

**Starting NAM**

Nam can either be started with the command 'nam <nam-file>' where '<nam-file>' is the name of a nam trace file that was generated by ns, or it can be executed directly out of the Tcl simulation script for the simulation which the user wants to visualize. [3]

**Starting ns**

NS is started with the command 'ns <tclscript>' (assuming that you are in the directory with the ns executable, or that your path points to that directory), where '<tclscript>' is the name of a Tcl script file which defines the simulation scenario (i.e. the topology and the events). You could also just start ns without any arguments and enter the Tcl commands in the Tcl shell, but that is definitely less comfortable.

Everything else depends on the Tcl script. The script might create some output on stdout, it might write a trace file or it might start nam to visualize the simulation.

**Starting NAM**

Nam can either be started with the command 'nam <nam-file>' where '<nam-file>' is the name of a nam trace file that was generated by ns, or it can be executed directly out of the Tcl simulation script for the simulation which the user wants to visualize. [3]

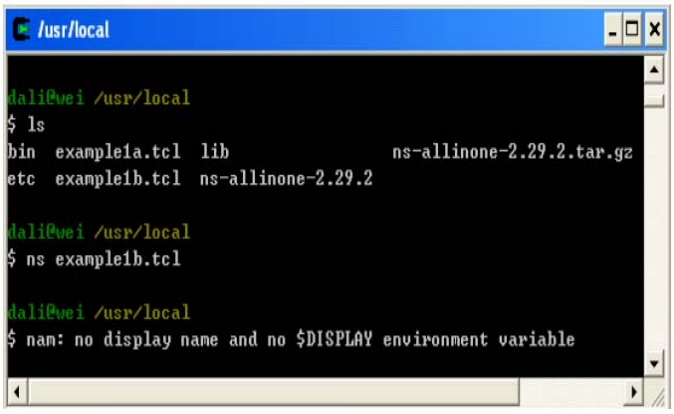


Fig. 2 Environment to call ns scripts

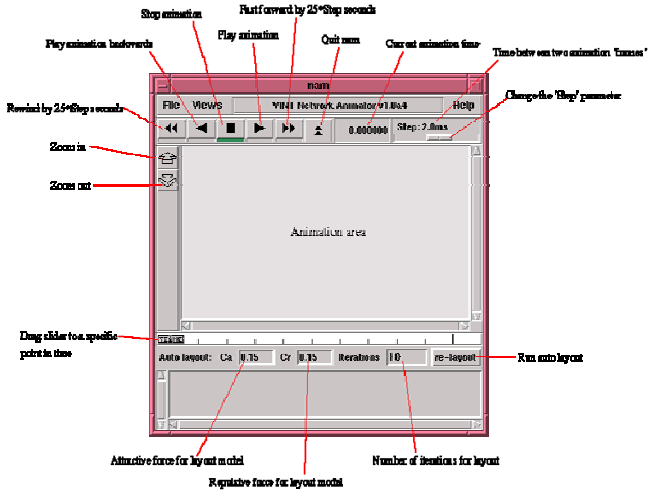


Fig. 3 Nam Animator

**Plotting with Xgraph**

XGRAPH [5] is a plotting utility that is provided by ns. (Sometimes it needs separate compiling using ./configure and then make when at the directory xgraph. Also, sometimes this does not work with the xgraph that arrives with the whole ns single package, and it can then be downloaded and installed separately). Not tat it allows to create postscript, Tgif files, and others, by clicking on the button "Hdcopy". It can be invoked within the tcl command which thus results in an immediate display after the end of the simulation.

As input, the graph command expects one or more ascii files containing each -x -y data point pair per line. For example, xgraph f1 f2 will print on the same figure the f1 and f2.

Some options in xgraph are:

- Title: use –t "title"
- Size: -geometry xsize x ysize
- Title for axis: -x "xtitle" (for the title of the x axis) and –y "ytitle" (for the title of the y axis). [5]
- Color of the text and grid: with the flag –v

An example of the command would be

Xgraph f1 f2 geometry 800 x 400 –t "Loss Rates" –x "Time" –y "Lost Packets"

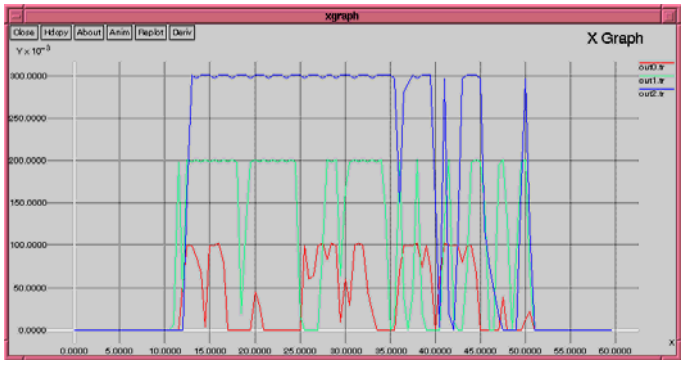


Fig. 4 Plots obtained using XGRAPH

## III. Problem Formulation

Using NS2 software simulator, it is pretended to simulate computer networks where scheduling disciplines are implemented, in this case would be SFQ [4]. When the desired algorithm is applied on the script, the next step is to change the type of traffic that is being used in the network.

During the simulations 4 different types of sources are used:

- Traffic/Expoo
- Application/Traffic/Pareto
- Application/Traffic/Exponential
- Application/Traffic/CBR

Simulations with only one active connection are implemented and after them some others active connections are included to be able to see how the performance of the scheduler is affected, due to this the traffic is also affected.

## IV. Results

The Fig. 4 shows the topology implemented with NS2 Software using Nam animator, where is possible to see our network graphically.

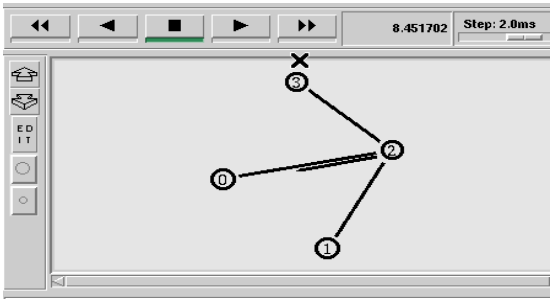


Fig. 5 Topology of the Network

During the simulation it was decided to monitor the number of packets received in a particular link where SFQ is applied.

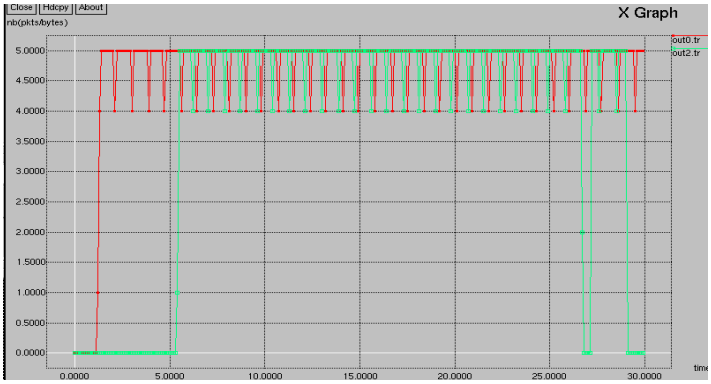


Fig. 6 Type of traffic used is Traffic/Expoo 1 active connection

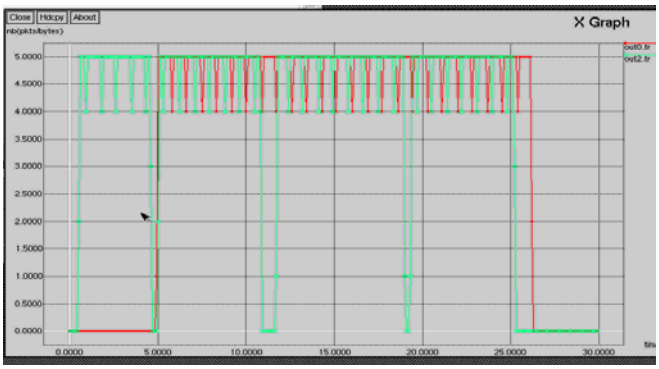


Fig. 7 Traffic/Expoo using more than 1 active connection

The Fig. 6 and Fig. 7 show a similar behavior, in these two graphics the type of traffic used was Expoo, it is possible to see that the number of packets received is between 4 and 5 showing also a little delay and the packets sent back (color green). It is also possible to realize that when the link is shared there are more packets lost.

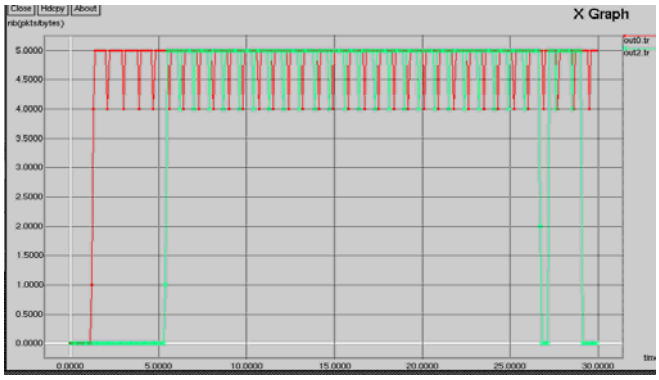


Fig. 8 Type of Traffic is Application/Traffic/Exponential

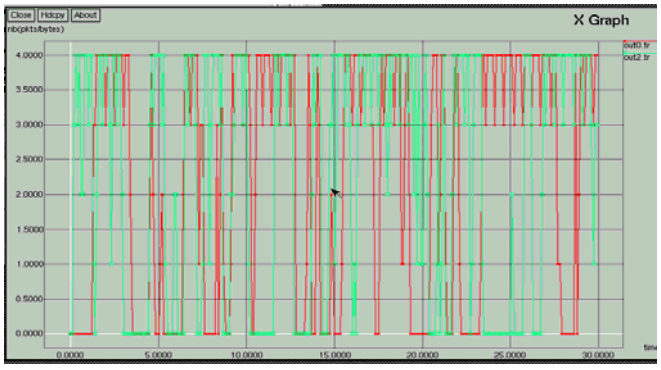


Fig. 9 Traffic Exponential with shared link

On the Fig. 8 and Fig. 9, it can be observed that when there is one active connection, the amount of lost packets is not as much as in the Fig. 8 where there more than one active connection sharing the link.

The Fig. 8 also shows that the lost packets are in both ways, but the first lost is where the information is coming from the destination to the source.

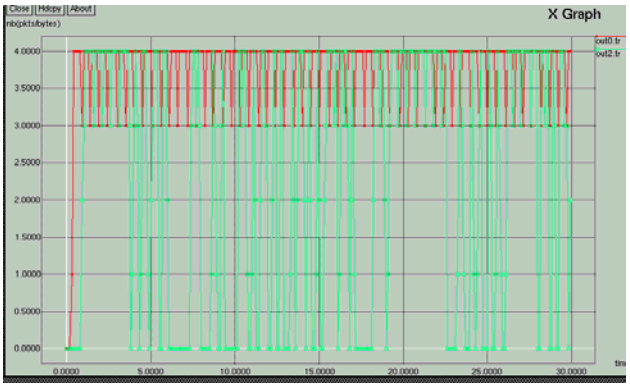


Fig. 10 Monitoring traffic with Traffic/Pareto

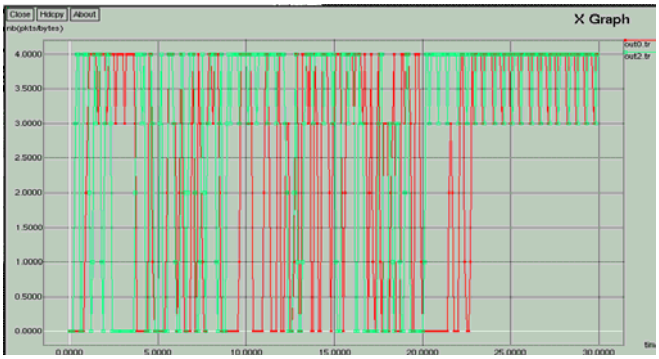


Fig. 11 Traffic Pareto with shared link

The Figs. 10 and 11 show the different behavior of the scheduler, it is clear when the link is shared and the network begins to be saturated the packets that are lost increases to much, even when two different types of traffic are implemented.

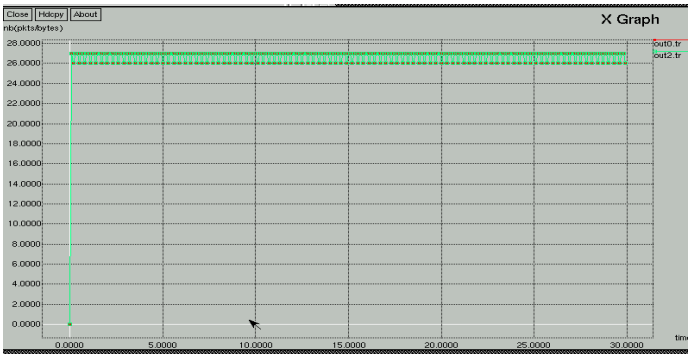


Fig. 12 Traffic CBR with shared link

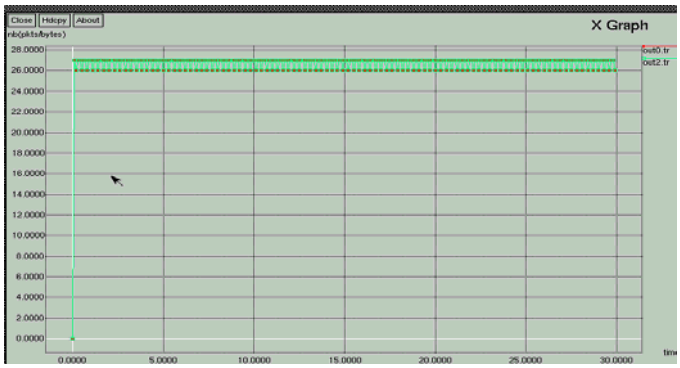


Fig. 13 Implementation of CBR like source of traffic

The Fig. 12 and Fig. 13 show the behavior of traffic when its source is CBR, it is easy to realize that the values of packets that are received are maintained on levels almost constant in both graphs.

We can deduce that the best option to manage the traffic is having CBR (Constant Bite Rate) as a traffic source

## V. Conclusion

After running several simulations it is easy to think that the best option in traffic management is having CBR source because the number of lost packets and the variations of received packets have minimum fluctuations. But the truth is that it is almost impossible to maintain this kind of level of efficacy in network where the packets present different sizes, lengths, even they are not transmitted at the same time, and, the most important thing is the types of traffic are also variable in each connection.

Finally the best option is to implement a fair discipline like SFQ where all the active links are serviced according with the amount of information needed.

It is also truth that the connections which receive more service is that one which is paying for a guaranteed service.

## References

[1] S. Keshav, An Engineering Approach to Computer Networking, Fifth Ed 2002, pages 209 -263.
[2] Rodrigo Sierra, Fair Queuing in Data Networks, Internetworking 2002 pages 1- 6.
[3] Marc Greis Tutorial, http://www.isi.edu/nsnam/ns/tutorial/
[4] NS2 Web Site www.isi.edu/nsnam/ns
[5] Eitan Altman and Tania Jimenez, NS Simulator for Beginners, December 2003, pages 33 – 38.