A High-Speed Self-Similar ATM VBR Traffic Generator

Patrick Droz (dro@zurich.ibm.com) Jean-Yves Le Boudec (leboudec@di.epfl.ch)

IBM Research Division
Zurich Research Laboratory
8803 Rüschlikon, Switzerland

Laboratoire Réseaux de Communication
Swiss Federal Institute of Technology (EPFL)
1015 Lausanne, Switzerland

ABSTRACT Recent analysis of real network traffic has shown that a large number of traffic sources produces a traffic stream that is self-similar over several time scales. Motivated by this observation a high-speed Asynchronous Transfer Mode (ATM) traffic generator based on fractional Brownian motion is presented. The traffic simulator generates call arrivals as well as the desired traffic parameters for each call. It is therefore an ideal tool to analyze a wide spectrum of networking functions such as switch performance, routing, and call admission control algorithms. The presented algorithm can be highly parallelized, thus generating traffic logs in a period several orders of magnitude below the actual simulated time period.

1 Introduction

In many different areas where long time series [Taq86] are available, certain correlation structures have been observed that manifest a self-similar (fractal) character. To give a few examples:

- Hydrology [Man65, Hur51], the study of the amount of daily precipitation or the influx/outflux of lakes, reservoirs, or rivers.
- Economics [Gul95, Mul95, Pet91] where the time series consist of foreign exchange rates or trading values of stocks and bonds.
- Communications [Lel94] where the time series consist of traffic measurements or occurrences of errors [Man65].

For all the above time series, a memory effect has been discovered. In other words, the past influences the future to a certain extent. The size of this long-term memory varies among different systems. The memory effect can also be expressed in terms of a positive or negative correlation. A positive correlation means that when the series increases during a certain interval, the probability is higher that it will increase again during the next interval. In the same way it is more likely that a decreasing interval will be succeeded by another decreasing interval. A system that has neither a positive nor a negative correlation does not have a memory, so successive values are independent of each other. In the case of networking traffic measurements, positive correlation has always been observed.

Many traditional traffic simulators assume uncorrelated memoryless behavior. For instance on/off sources with independent identically (exponential) distributed (IID) on/off periods produce uncorrelated traffic. The aggregation of on/off sources thus produces a traffic stream of white noise around the aggregated mean rate very rapidly with an increasing number of sources. In the case of correlated traffic, bursts appear over a wide range of time scales and the smoothing effect appears far more slowly.

Another important point is that traffic generators for high-speed cell-based networks such as ATM [Pat93, Atm93] must be capable of producing huge traffic logs in a short period of time. With increasing link speeds up to Gbit/s or even Tbit/s, event-driven simulation becomes increasingly crucial, especially for traces spanning periods of several hours. Here methods are needed to produce traffic as a whole (fluid-flow approach) compared to the aggregation of single sources.

One of the key advantages of ATM is the concept of Quality of Service (QoS) assurance for reserved connections. A user can specify a sustainable cell rate, a peak cell rate, and a maximum burst size. A traffic generator for reserved traffic must also produce values for these traffic parameters. So far only self-similar traffic generators for connectionless traffic [Lel94, Rob95, Lau95] such as Ethernet LANs have been proposed. Recent studies [Bla95] show that the connection setup/release rates, currently in the range of 10^2 per second, will have to increase drastically to cope with future demands. In the near future a traffic parameter renegotiation process will be introduced that allows connections to change their contract with the network without tearing down the connection first and then reestablishing it with a different set of traffic parameters.

In summary, we have established the following list of design goals to be fulfilled by a traffic generator.

1. Produce self-similar traffic and call arrivals/departures over a wide time span and with different Hurst parameter values (see below).
2. Generate call arrivals/departures and the actual traffic.
3. Produce traffic parameters with every call arrival/departure.
4. Produce huge traffic logs (at least up to days).

5. Produce traces that take less time than the simulated time period.

6. Be scalable for different link utilizations, overbooking states, and link speeds.

The paper is structured as follows. In section 2 reasons are given why the simulation of the traffic as a whole is preferred over simulating individual sources and then aggregating them. In section 3 the mathematical definition of self-similarity is given. In section 4 two methods are presented to measure the grade of self-similarity by means of the Hurst parameter H. In section 5 fractional Brownian motion will be introduced. Section 6 presents the new traffic generator. In chapter 7 generated traffic samples will be presented and analyzed with different methods. The final section 8 summarizes the paper.

2 Why Aggregated Traffic?

In future LANs an enormous number of different source types will share the same network resources, e.g. voice, video, *File Transfer Protocol* (FTP), telnet, *World Wide Web* (WWW), and many more. Therefore, if one wishes to simulate a real-world environment such as the LAN of a bank or an airline, the question that arises very soon is, how can a real-world traffic mix be generated? What are the probability distributions of the call arrival times, call durations, QoS parameters, and the actual behavior of all those different sources? In addition, a supplier of networking equipment usually sells products in very different environments. So mixes for different environments are needed even to check, for instance, the performance of an ATM switch with high confidence for all customers.

It is certainly possible to gather statistics for the various sources and then deduce the probability distributions for the large number of parameters. The question is, how feasible is this? The statistics deduced are only valid for the very specific environment from which they were obtained. The statistics would be valid in general for a certain environment if there were unlimited resources available; unlimited resources not only inside the network but also on all clients and servers on the network! In such cases all connections become independent of each other. Such a pathological environment is too much of an assumption. Connections do influence each other and often drastically, e.g. consider the many notoriously overloaded FTP sites all around the globe. It is also unrealistic to believe that one day we will have unlimited resources. This certainly has not happened in the computer industry so far. Human beings have always managed to reach the limits of the available capacity by using new applications. For example, the amount of memory has increased by several orders of magnitude, but we still put more and more memory into our computers to cope with new applications.

Another problem is the fast-growing number of different applications and the extension of existing ones. Recently, WWW traffic has drastically increased. In April 1995, 26.2 % of the traffic on the NSFNet was WWW traffic [Bla95]. In addition, the corresponding protocols have been extended, thus changing the traffic pattern constantly. Furthermore, WWW traffic also depends very strongly on the browser used, as different browsers have different loading strategies. Applications such as WWW can be considered a kind of networking killer. New compression techniques also belong in this category. A higher compression technique on the one hand reduces the mean transmission rate but on the other hand increases burstiness.

3 Self-Similarity

The mathematical definition of self-similarity [Man83, Lel94] is as follows. Let $X = (x_t : t = 1, 2, 3, \ldots)$ be a covariance stationary stochastic process with mean $\mu = E[x_t]$, variance $\sigma^2 = E[(x_t - \mu)^2]$ and an autocorrelation function

$$r(k) = \frac{E[(x_t - \mu)(x_{t+k} - \mu)]}{E[(x_t - \mu)^2]}$$

only depending on $k = 0, 1, 2, 3, \ldots$. Assume an autocorrelation function of the form $r(k) \sim ak^{-\beta}$ as $k \to \infty$ with $0 < \beta < 2$ and a a constant. Now let $X^m = (x_k^m : k = 1, 2, 3, \ldots)$ be the averaged series over nonoverlapping blocks of size m which is

$$x_k^m = \frac{1}{m} \sum_{i=km-m+1}^{km} x_i$$

with $k = 1, 2, 3, \ldots$ and $m = 1, 2, 3, \ldots$. For each m, X^m is a covariance stationary process. X is called exactly second-order self-similar with self-similarity parameter $H = 1 - \beta/2$ if for all $m = 1, 2, 3, \ldots$ $\mathrm{var}(X^m) = \sigma^2 m^{-\beta}$ (σ sample variance) and $r^m(k) = r(k)$, with $k \geq 0$ X is called asymptotically second-order self-similar with self-similarity parameter $H = 1 - \beta/2$ if for all large k: $r^m(k) \to r(k)$, as $m \to \infty$. Here X is called exactly or asymptotically self-similar when the aggregated processes X^m become indistinguishable from their root X in the sense of their autocorrelation function.

There are three distinct regions for H, $0 < H < 1/2$, $H = 1/2$ and $1/2 < H < 1$. A value below $1/2$ indicates a negative correlation, whereas a value over $1/2$ expresses a positive correlation. In the special case of $1/2$ there is no correlation present.

A more vivid characterization of self-similarity is the scaling law [Pei92] on different time horizons. This means that time series obtained with different sampling frequencies can be scaled such that they look the same. If the sampling frequency is doubled one has to scale the amplitude of the new series with a factor of 2^H to make it look the same. This scaling law is illustrated in section 7 with a traffic sample on different time scales.

Another mathematical property is that the frequency domain spectrum follows a power law and the spectrum disposes of a kind of harmonics.

4 Hurst Parameter Estimation

There are several methods known to measure the Hurst parameter H of a time series, two of which, R/S analysis [Hur51, Man69] and variance time plot [Lel94], will be described. Samples of diagrams can be found in section 7, where artificially generated traffic will be analyzed.

The same mathematical notation as in the previous section is assumed for a time series X.

The definition of the R/S analysis is as follows. Let $W = (w_t = \sum_{u=1}^{t} x_u)$ be a time series consisting of all partial sums of the original time series. Now take the standard deviation over the interval $[t+1, t+s]$:

$$S(t,s) = \sqrt{\frac{1}{s}\sum_{u=t+1}^{t+s} x_u^2 - \left(\frac{1}{s}\sum_{u=t+1}^{t+s} x_u\right)^2}$$

and calculate the range $R(t,s)$:

$$\begin{aligned} R(t,s) &= \max_{0\leq u\leq s}\left[w_{t+u} - w_t - \frac{u}{s}(w_{t+s} - w_t)\right] - \\ &\quad \min_{0\leq u\leq s}\left[w_{t+u} - w_t - \frac{u}{s}(w_{t+s} - w_t)\right] \end{aligned}$$

The ratio $R(s,t)/S(s,t)$ is then called the rescaled range. One has now to select values for s and t and then calculate the rescaled range. The so-called pox diagram is obtained by plotting

$$\log\left(\frac{R(s,t)}{S(s,t)}\right) \text{ against } \log(s).$$

For each s one calculates the mean value for the different k values. With the help of a least-squares fit a straight line is then drawn through these points. The slope of this line is equivalent to the Hurst parameter H. The R/S analysis is known to be very robust against changes in the marginal distribution.

The variance time plot is obtained from the relation $\text{var}(X^m) = \sigma^2 m^{-\beta}$, which was given in section 3 for self-similarity. By plotting

$$\log\left(\frac{\text{var}(X^m)}{\sigma^2}\right) \text{ against } \log(m)$$

one obtains a curve which is a straight line. The slope β of this curve can vary between 0 and -2. The value stands in the following relation to the Hurst parameter H: $H = 1 - \beta/2$. A very efficient way to calculate points is to start with $m = 2$ and then multiply in each iteration by 2.

5 Fractional Brownian Motion

A well-known structure that has self-similar properties is fractional Brownian motion (fBm) [Man68]. Unfortunately, the term is ambiguous. It is sometimes used for the summed deltas (actual position of a particle) and sometimes for the deltas themselves. For clarity the term fBm is used in this paper for the summed increments, whereas the term fractional Brownian noise (fBn) is used for the increments. fBn can be transformed into fBM and vice versa.

There are several ways known in the literature to produce fBm and fBn. Some of these possibilities are

- *Fourier transform:* generate a frequency spectrum and then apply the inverse of a fast Fourier transform to it [Che95, Man77].
- *Displacement Process:* exploit the fact that fBm is the limit of a fractional Poisson field [Nor94, Man83].
- *Pareto distribution:* aggregate several on/off sources where the on/off periods are Pareto (heavy tailed) distributed [Geo94].
- *Markov process:* use a Markov processes to produce the high and low-frequency spectra [Man71, Rob95].
- *Random Midpoint Displacement:* sum up properly scaled Gaussian numbers in a recursive manner [Pei92, Fou82].

This paper focuses on random midpoint displacement. Compared to the recursive implementation in the original paper [Fou82] the algorithm is transformed into an iterative one. An iterative implementation is more appropriate to generate huge data sets. In addition, a highly parallelized implementation can be achieved easily.

fBm with a prescribed Hurst parameter in the interval $]0,1[$ is generated as follows:

$$\begin{array}{lrcl}
\text{Initialization:} & x[0] & = & 0 \\
\text{Initialization:} & x[1] & = & G_1 \\
\text{Iteration 1:} & x\left[\frac{1}{2}\right] & = & \frac{x[0]+x[1]}{2} + rG_2 \\
\text{Iteration 2:} & x\left[\frac{1}{4}\right] & = & \frac{x[0]+x\left[\frac{1}{2}\right]}{2} + r^2G_3 \\
\text{Iteration 2:} & x\left[\frac{3}{4}\right] & = & \frac{x\left[\frac{1}{2}\right]+x[1]}{2} + r^2G_4 \\
\text{Iteration 3:} & x\left[\frac{1}{8}\right] & = & \frac{x[0]+x\left[\frac{1}{4}\right]}{2} + r^3G_5 \\
\text{Iteration 3:} & x\left[\frac{3}{8}\right] & = & \frac{x\left[\frac{1}{4}\right]+x\left[\frac{1}{2}\right]}{2} + r^3G_6 \\
\text{Iteration 3:} & x\left[\frac{5}{8}\right] & = & \frac{x\left[\frac{1}{2}\right]+x\left[\frac{3}{4}\right]}{2} + r^3G_7 \\
\text{Iteration 3:} & x\left[\frac{7}{8}\right] & = & \frac{x\left[\frac{3}{4}\right]+x[1]}{2} + r^3G_8 \\
\vdots & & &
\end{array}$$

where G_i denotes normal distributed random numbers. The factor r is responsible for the scaling law and thus related to the Hurst parameter H: $r = 1/2^H$ with $0 < H < 1$. In iteration n, 2^{n-1} new samples are generated. In the case of a parallel implementation the number of processes that calculate samples can be doubled after each iteration. One has only to make sure that random Gaussian numbers are generated with different seeds because on most systems random numbers are reproducible and therefore run through the same sequence. In a parallel implementation one has to create first a couple of border points that can be delegated to newly forked processes that calculate different subintervals.

The actual fBn is then generated by taking the difference between two successive samples. By applying an H estimator to such a time series one can check the accuracy of H.

6 Traffic Generator

The traffic generator [Dro96] presented, is based on the random midpoint displacement algorithm [Pei92, Fou82], but extended to two different levels. First, the call arrival/departure process for the aggregated mean rate is generated with fBn. Second, the actual traffic is generated as fBn around the mean rate.

The following is part of the C-code for the traffic generator. It covers only the first part to generate the call arrivals/departures, i.e. the peak/mean rate pairs.

```
/* generate the mean and peak rate  */
reduction  = 1.0/pow(2.0,H);
scale_mean = 1.0; step = nr_of_reservation_steps-1;
while (step > 1) {
 left = 0; right = step;
 while (right < nr_of_reservation_steps) {
   mean[(left+right)/2] = .5*(mean[right]+mean[left])
        + scale_mean * normal_distribution(0.0,1.0);
  left = right; right += step;
 }
 scale_mean *= reduction; step /= 2;
}
/* generate the noise */
for (i=0;i<nr_of_resolution_steps-1;i++) {
  mean[i] = mean[i+1] - mean[i];
}
max = max_function(mean); min = min_function(mean);
factor = link_speed*over_booking*rho / (max - min);
translation = factor * min * -1.0;
for(i=0;i<nr_of_reservation_steps;i++) {
  mean[i] = mean[i] * factor + translation;
  peak[i] = mean[i]/rho;
}
```

The C function *normal_distribution(*μ, σ*)* generates Gaussian random numbers with mean μ and standard deviation σ. On a computer, normal distributed random numbers can be generated (based on the *central limit theorem* (CLT) [Pap91]) with the help of equally distributed random numbers [Pei92], which are usually available on every computer.

The parameter $\rho = m/R$ is used to calculate the peak rate from the mean rate. This value could theoretically be varied but for analysis purposes it is easier to generate different traces with different ρ. In the presented samples ρ was set to 0.3.

After the fBm is generated it is transformed into fBn. Then the minimum, the maximum, the scaling factor, and the translation are determined. The noise is scaled and translated such that it is between 0 and the link speed multiplied with an overbooking factor. As the scaling and translation is a linear transformation, it does not change the correlation structure. It can be shown that the variance time plot as well as the R/S analysis still yield the same results.

After the mean and peak rates are generated, the traffic is generated by further iterations. The traffic is generated as fBn around the mean rate. The scaling for the traffic is controlled by the mean and peak rate in the following way. For a single on/off source with mean m and peak R the standard deviation can be calculated as $\sigma = \sqrt{m(R-m)}$. Viewing the link as a single on/off source is the worst case because there is no smoothing effect at all. The traffic is then scaled such that the standard deviation is proportional to this worst case σ. With the help of an additional parameter the proportion can be varied. For the results presented, $\sigma/4$ was used. If a sample point falls below zero it is clipped to zero. Points that are above the peak rate are also clipped. This means we assume that the user parameter control function is implemented and users are policed. In general the user should be policed right at the access to the network.

The most time-consuming activity of the entire process is the writing of the data to the disk. A sequential iterative implemented version used about 28 s on an RS6000/590 to calculate 4,194,313 measurements including 524,289 call arrivals and departures ((R, m) pairs). Writing these values out to AFS takes about 3.5 minutes (binary format for Matlab). By using several processes that write out the trace, the speed can be improved. For very extensive simulations it is best to avoid writing the trace completely and feed generated traffic directly into the next component, i.e. into a link buffer.

7 Analysis

Figure 1 illustrates self-similarity on different time scales. The Hurst parameter was set to $H = 0.8$. From picture to picture the

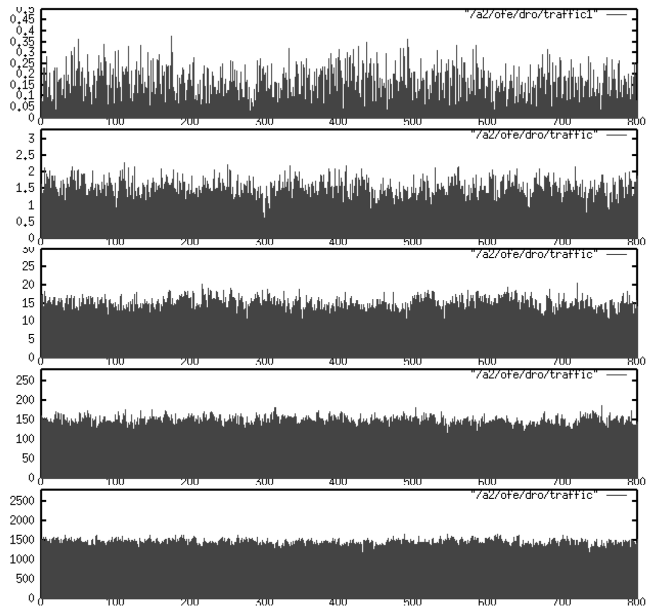

Figure 1: Traffic on different time scales (top: 10^{-3} s, bottom: 10 s).

sampling frequency was reduced by a factor of 10. In the first picture the traffic is normalized to a peak rate of 1.

In Fig. 2 the result of the R/S and the variance time plot analyses are depicted. The R/S estimator yielded $H = 0.796$. For

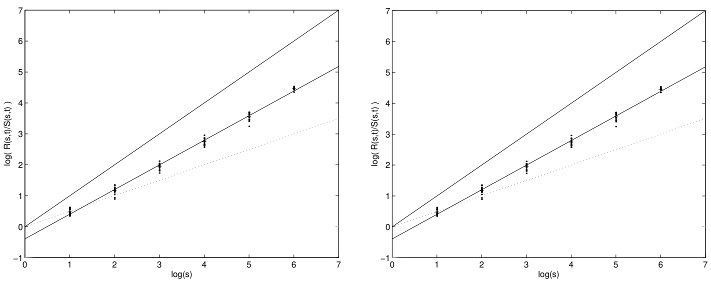

Figure 2: R/S (left): $H = 0.796$, Variance (right) $H = 0.795$.

the trend line a least-squares fit was applied. In addition, two lines with slope 0.5 and 1.0 are shown to make the diagram more readable. The result of the variance-time plot yielded $H = 0.795$. Some measurement points at the beginning and at the end were not considered for the trend line. But there is a large region where the trend is a straight line.

In Fig. 3 the call arrivals and departures with aggregated peak and sustainable rate including the traffic are shown. Every 8 ms there is a change in the reservation state. This is an appropriate value for a switch that handles 125 connection setup/tear-downs per second (or renegotiations). The sampling interval for the traffic was set to 1 ms, which is equivalent to 366 cells on a 155 Mbit/s link.

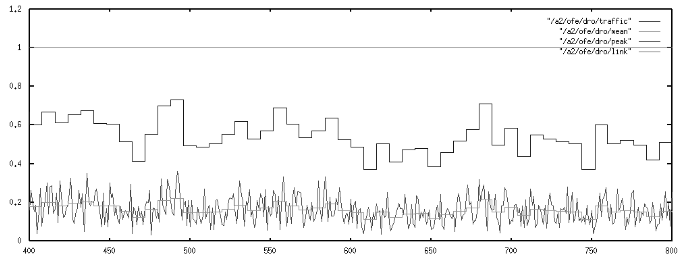

Figure 3: Peak rate, sustainable rate and traffic (time-resolution 10^{-3} s).

The correlation structure can clearly be seen. Increments are more often followed by increments and decrements are more often followed by decrements.

8 Summary and Conclusion

In this paper a high-speed self-similar traffic simulator based on fractional Brownian motion was presented. By means of a parallel implementation of the algorithm, huge data logs can be generated in time periods that are shorter than the simulated time. Compared to other traffic simulators the one introduced here produces the actual traffic as well as the arrival/departure process of ATM connections including QoS parameters. Thus makes it an appropriate tool to study performance issues for reserved traffic.

The generated traffic scaled well to the parameter H. Both analysis techniques R/S analysis and variance time plot estimatee almost the same H value.

Ongoing work focuses on the design of a call admission controller (CAC) based on a fractal traffic pattern.

References

[Atm93] The ATM Forum. *ATM user-network interface specification, version 3.0.* PTR Prentice Hall, 1993. ISBN 0-13-225863-3.

[Bla95] Trevor Blackwell. *Fast SVCs. Preliminary WWW pages*, June 1995. http://www.eecs.harvard.edu/ tlb/fast-svc.html.

[Che95] Ying Chen, Zhiang Deng, and Carey L. Williamson. *A Model for Self-Similar Ethernet LAN Traffic. SCSC 95 Ottawa Ontario Canada*, July 1995.

[Dro96] Patrick W. Droz-Georget. *Traffic Estimation and Resource Allocation in ATM Networks.* PhD Thesis No. 11462, Swiss Federal Institute of Technology, 1996.

[Fou82] A. Fournier, D. Fussell, and L. Carpenter. *Computer Rendering of Stochastic Models. ACM Comm 25*, pages 371–384, 1982.

[Geo94] Nicolas D. Georganas. *Self-Similar (Fractal) Traffic in ATM Networks. Lecture Notes in Computer Science 868*, September 1994. Second International Workshop, IWACA.

[Gul95] D.M. Guillaume, M.M. Dacorogna, R.R. Davé, U.A. Müller, R.B. Olsen, and O.V. Pictet. *From the Birds Eye to the Microscope. O&A Research Group*, February 1995. http://www.olsen.ch.

[Hur51] H. E. Hurst. *Long-term Storage Capacity of Reservoirs. Transaction of the American Society of Civil Engineers, 116*, 1951.

[Lau95] Wing-Cheong Lau, Ashok Erramilli, Jonathan L. Wang, and Walter Willinger. *Self-Similar Traffic Generation: The Random Midpoint Displacement Algorithm and Its Properties. IEEE ICC*, June 1995.

[Lel94] Will. E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson. *On the Self-Similar Nature of Ethernet Traffic (Extended Version). IEEE/ACM Transactions On Networking*, 2(1), February 1994.

[Man65] Benoit B. Mandelbrot. *Self-Similar Error Clusters in Communications Systems and the Concept of Conditional Stationarity. IEEE Trans. Communications Technology COM-13*, pages 71–90, 1965.

[Man68] Benoit B. Mandelbrot and John W. Van Ness. *Fractional Brownian Motions, Fractional Noises and Applications. SIAM Review 10*, pages 422–437, 1968.

[Man69] Benoit B. Mandelbrot. *Long-Run Linearity Locally Gaussian Process, H-spectra and infinite Variances. International Economic Review*, 10(1), February 1969.

[Man71] Benoit B. Mandelbrot. *A fast Fractional Gaussian Noise Generator. Water Resource Research*, 7(3), June 1971.

[Man77] Benoit B. Mandelbrot. *Fractals: Form, Chance and Dimension.* Freeman San Francisco, 1977. ISBN 0-7167-0473-0. ISBN 0-7167-0474-9.

[Man83] Benoit B. Mandelbrot. *The Fractal Geometry of Nature.* Freeman New York, 1983. ISBN 0-7167-1186-6.

[Mul95] U.A. Müller, R.D. Davé M.M. Dacorogn, O.V. Pictet amd R.B. Olsen, and J.R. Ward. *Fractals and Intrinsic Time - A Challange to Econometricians. O&A Research Group*, February 1995. http://www.olsen.ch.

[Nor94] Ilkka Norros. *A storage model with self-similar input. To appear in Queuing Systems*, 1994.

[Pap91] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGRAW-HILL INTERNATIONAL EDITIONS, 1991. ISBN 0-07-100870-5.

[Pat93] Craig Patridge. *Gigabit Networking.* ADDISON-WESLEY, October 1993. ISBN 0-201-56333-9.

[Pei92] Heinz-Otto Peitgen, Hartmut Jürgens, and Dietmar Saupe. *Chaos and Fractals, New Frontiers of Science.* Springer Verlag, 1992. ISBN 0-387-97903-4, ISBN 3-540-97903-4.

[Pet91] Edgar E. Peters. *Chaos and Order in the Capital Markets.* John Wiley and Sons, Inc., 1991. ISBN 0-471-53372-6.

[Rob95] Stephan Robert and Jean-Yves Le Boudec. *Can self-similar traffic be modeled by Markovian processes? COST 242 Stockholm meeting*, May 1995.

[Taq86] M. S. Taqqu and J. B. Levy. *Using Renewal Processes to gnerate Long-Range Dependence and High Variability. Dependence in Probability and Statistics, Progress in Prob. and Stat.*, pages 73–89, 1986.