CHAPTER 5
CONCLUSIONS

In Sec. 3.2 we found analytically that the buffer requirement for CLR<1% for a H of 0.8 was 70. The increase from the Poisson Buffer requirement (11) was by a factor of 6.3. Table 4.1 gives us the increase in buffer requirement obtained by simulations. For a Hurst parameter of 0.85 the increase in buffer size was by a factor 8.5 and for an H of 0.75 the increase in buffer size was by a factor of 6. We see a similar increase in buffer size in both the theoretical and the analytical cases.

From the results obtained we see that a drastic increase in the buffering is required if we want to maintain the CLR. But any increase in buffering is seen to reflect an increase the delay of the cells. Also, for the transport of CBR traffic since the cell delay variation is very high, some additional measures may have to be taken to maintain the guaranteed QoS. This could take the form of preferential treatment at the switches. Source level or switch level moulding of the traffic may be done in order to decrease the burstiness of the traffic.

If the relationship between α and H in (3.17) is found, the 2 Parameter Pareto Process provides us with a general distribution function which can be used to model any self-similar process. This distribution can be used to make a random generator that can produce self-similar traces that can be used in simulations. Though this self-similar trace generator maybe easier to understand than the exiting methods like the Fast FFT method [VP97] or Hosking's Procedure [H84], the time-complexity of this method is high.

REFERENCES

[LT67] : Lajos Takacs, Combinational Methods in the the theory of Stochastic process.

[H84] : J.R.M.Hosking, "*Modeling persistence in Hydrological Time series, using fractional differencing*", Water resources Research, pp. 1898 - 1908, Vol. 2 (12), 1984.

[LW91] : W. E. Leland and D. V. Wilson. " *High time-resolution measurement and analysis of LAN traffic: Implications for LAN interconnection* ". In Proceeedings of IEEE Infocomm '91, pp 1360--1366, Bal Harbour, FL, 1991.

[LTWW94] : W.E. Leland , M.S. Taqqu , D.V. Wilson and Walter Willinger , *" On the Self-Similar Nature of Ethernet Traffic* (extended version) , " IEEE / ACM Trans. Networking , vol. 2 , no. 1, pp. 1-15 ,1994

[MRW94] : Duffy. D. McIntosh, A.Rosenstein .M and W.Willinger " *Statistical Analysis of CCSN / SS7 Traffic Data from Working CCS Subnetworks* ", IEEE Journal on Selected Areas in Telecommunication, April 1994.

[B94] : Jan Beran. " *Statistics for Long-Memory Processes* ". Monographs on Statistics and Applied Probability. Chapman and Hall, New York, NY, 1994.

[PF95] : Vern Paxon and Sally Floyd , " *Wide Area Traffic : The failure of Poisson Modeling* , " IEEE / ACM Trans. Networking , Vol. 3 , pp. 226-244 , 1995 .

[CB95] : Mark E. Crovella and Azer Bestavros "*Explaining World Wide Web Traffic Self-Similarity*". Technical Report TR-95-015.Revised.October 12, 1995

[LTWW95] : Walter Willinger, Murad S. Taqqu, Will E. Leland, and Daniel V. Wilson. " *Self-similarity in high-speed packet traffic: Analysis and modeling of Ethernet traffic* measurements ". Statistical Science, 10(1): 67--85, 1995.

[LTG95] : N.Likhanov, B.Tysbakov, and N.Georganas , " *Analysis of an ATM Buffer with Self-Similar ("Fractal") input Traffic* " , Proceedings of IEEE INFOCOM '95 , 1995

[N95] : Ilkka Norros , " *On the use of Fractional Brownian Motion in the Theory of Connectionless Networks* " , IEEE Journal on Selected Areas in Communications , Vol. 13 , No.6 , 1995

[CROV96] : Mark .E.Crovella and A.Bestavros , "*Self similarity in World Wide Web traffic : Evidence and possible causes* " , Proceedings, ACM Sigmetrics Conference on Measurement and Modeling of Computer.

[ENW96] : A. Erramilli , O.Narayan, and W.Willinger , " *Experimental Queuing Analysis with long Range Dependent Packet Traffic,* " IEEE / ACM Trans. On Networking , Vol. 4 , No.2 , pp. 209-223 , 1996

[LS97] : George.C.Lin and Tatsuya Suda , "*On the impact of Long-Range-Dependent Traffic in Dimensioning ATM Network Buffer* " ,www.ics.uci.edu

[PKC97]: K.Park,G.Kim,M.Crovella,*"On the Effect of Traffic Self-Similarity on Network Performance",* Proceedings of SPIE international conference on Performance and Control of Network Systems, Nov 1997

[PM97] : M.Parulekar and A.Makowski, " *M / G / ¥ Input Processes : A Versatile class of Models for Network Traffic* " , Proceedings of IEEE INFOCOM '97 , 1997.

[VP97] : Vern Paxson ,*"Fast Approximate synthesis of Fractional Gaussian Noise to generate Self-similar Network traffic"*,Computer Communication Review 27(5), pp 5-18, October 97

[TG97] : Boris Tsybakov and Nicolas D. Georganas , "*On Self Similar Traffic in ATM Queues :Definitions , Overflow Probability Bound, and Cell Delay Distribution* , " IEEE / ACM Trans. Networking , Vol. 5 , No. 3 , pp. 397-409 , 1997

[S98] : William Stallings , " *Self Similar Traffic* " . High Speed Networks . Prentice-Hall International Inc. 1998

[VMG] : Attila Vidacs, Sandor Molnar and Geza Gordos ," *The impact of Long Range Dependence on cell loss in an ATM Wide* Area *Network* "

APPENDIX B

The Matlab source codes for various programs used in our project is given here.

- The Fast FFT method for generating Self-Similar Traces

```
clear all;
n=32768;
H=0.985;
j=sqrt(-1);
const1=2*pi/n;
const2=8*H*pi;
const3=2*sin(pi*H)*gamma(2*H+1);
d=-2*H-1;
d1=-2*H;
for i=1:4
   a(i)=2*i*pi;
   b(i)=a(i);
end
for k=1:n/2
   L=const1*k;
   B=(a(1)+L)^d + (b(1)-L)^d + (a(2)+L)^d + (b(2)-L)^d + (a(3)+L)^d +
(b(3)-L)^d...
      +  ( (a(3)+L)^d1 + (b(3)-L)^d1 + (a(4)+L)^d1 + (b(4)-L)^d1
)/const2;
   A=const3*(1-cos(L));
   temp=A*(B+L^d);
   f(k)=temp*exprnd(1,1,1);
   z(k)=sqrt(f(k))*exp(j*rand*2*pi);
end

z1(1)=0;

for m=1:n/2
   z1(m+1)=z(m);
end

for m=n/2+1:n-1
   z1(m+1)=conj(z(n-m));

end

tr=real(ifft(z1));
tr=tr+abs(min(tr));
tr=tr*1000;
```

- The Program used to solve for (3.3)

```
clear all
d=30;
LamD=0.75*d;
alp=150;
const=1/(d*(LamD*(alp-1)-alp));
```

```
s1=0;
for i=0:10000
   s1=s1+(i*const + 1/d)^(-alp-1);
end
de=200;
w(1)=1;
for j=0:de  %e%
   for k=0:de  %f%
      Q(j+1,k+1)=0;
      for n=0:100 %a%
         if ( ((j+1)>=d) & (n<=(d+k-j-1)) ) | ( ((j+1)<d) & (n<=k) )%b%
            pN(j+1,k+1) = 0;
         elseif ((j+1)<d) & ((k<n) &(d+k-j-1>=n))
            pN(j+1,k+1)=0;
            for s=1:n-k %c%
               sum1=(factorial(n)/(factorial(s+k)*factorial(n-s-
k)))*(s/d)^(s+k)*(1-s/d)^(n-s-k)...
                  *(d-n+k)/(d-s);
               pN(j+1,k+1)=sum1+pN(j+1,k+1);
            end %c%
         else
          pN(j+1,k+1)=1;
         end %b%
      %Q(j+1,k+1)=Q(j+1,k+1)+pN(j+1,k+1)*LamD^n*exp(LamD)/factorial(n);
     Q(j+1,k+1)=Q(j+1,k+1)+pN(j+1,k+1)*((n*const + 1/d)^(-alp-1))*1/s1;
   end %a%
      if (k==0)
         q(j+1,k+1)=0;
      else
         q(j+1,k+1)=Q(j+1,k)-Q(j+1,k+1);
         if q(j+1,k+1)<0
            q(j+1,k+1)=0;
          end

      end
   end  %f%
end  %e%
c=q;
q=q-eye(de+1);

for j=1:de+1
   q(de+1,j)=1;
end
a=zeros(de+1,1);
a(de+1)=1;
w=inv(q)*a;
for i=1:de-1
   x(i)=0;
   for j=i+1:de
      x(i)=x(i)+w(j);
   end
end
x(de)=0;


t=0:de;
```

```
for k=1:100
   f(k)=0;
   for i=1:100
      f(k)=f(k)+w(i,1)*c(i,i+k);
   end
end
f=f/sum(f);
f=f*32768;
buflen(f,0.99)
```