TCP traffic in high speed telecommunication networks. What probability distributions?

Messages that flow from a source to a destination through a network are also known as traffic. This traffic and the network conditions are extremely random in nature.

There are three types of telecommunication networks –telephony (telephone network for voice calls, fax, and also dial up connections), cable-TV networks (cable, web-TV, etc.) and high speed networks such as the Internet. We are concerned with high-speed networks. Some hope that in the very near future, Internet telephony, video-on-demand, networked homes and multimedia applications will replace telephone and Cable-TV networks. But it is believed that the performance of high-speed networks has to improve a lot before that (Gautam, 2003).

Traffic flowing through the networks can be classified into several types. Two of the most common traffic types are ethernet packets/frames and ATM cells. The length or size of an Ethernet packet ranges anywhere from 60 bytes to 1500 bytes and generally follows a bimodal distribution. The length of ATM cells is fixed at 53 bytes. Therefore the network traffic comprises of millions and billions of these little packets or cells (Gautam, 2003). Here we are concerned with packets traffic only, not cell traffic.

The packets arise because when a message needs to be sent from a source to a destination, it is broken down into small packets and transported that way from the source to the destination.

The protocols responsible for this transport of packets over networks are user datagram protocol (UDP) and transmission control protocol (TCP). With UDP, the source does not acknowledge the receipt of packets to the source. TCP is an acknowledgement (ACK) based protocol. In this activity, we are concerned with packets transported by the TCP protocol.

There are standardized ways to obtain data on traffic. See for example the way that the data set we use in this activity was reduced by Jeff Mogul. The site is:

`http://ita.ee.lbl.gov/html/contrib/sanitize-readme.txt`

Usually, these standard ways extract the following variables: the arrival time of the packet, its

source, its destination, its length, its type, etc. The time of arrival and packet size are very important variables, and we will analyze them in this activity.

There are several network performance measures that contribute to the Quality of Service of a network. Among others, we have: (a) loss probability, or the probability of delivering a message with some data loss; (b) delay or latency, or the time between the source sending a message and the destination receiving it; (c) delay-jitter or measure of the variation of the delay; (d) bandwidth or rate at which messages are processed. These measures can be used for optimal design and admission control of the networks. Design deals with buffer sizes, link capacities, network parameters, traffic shaping parameters, and other. Admission control involves rejecting or accepting an incoming request for connection.

Data set

In this activity, we use a dataset obtained from the Internet Traffic Archive

`http://ita.ee.lbl.gov/html/contrib/DEC-PKT.html`

The specific dataset for this activity is dec-pkt-1.tcp which summarizes traces of one hour's worth of TCP traffic between Digital Equipment Corporation and the rest of the world on March 8th, 1995 . It describes 2,153,462 million packets and contains the following 6 variables. The names in italic are the names that appear in the file to be used with R. The dataset is available in the datasets directory of the cs-stats web page. Notice that this data set is the processed version of the raw logs that the servers keep (which is mostly text.. We will work with raw logs later in the quarter).

- *timestamp* of packet arrivals. For the first packet in the trace, this is the raw time. But I removed the integer part.
- *source*: Source host, with a code for confidentiality reasons.
- *destination*: Destination host with code for confidentiality reasons.
- *sourceport*: source TCP port
- *destport*: Destination TCP port
- *databytes*: number of data bytes in the packet, or 0 if none (this can happen for packets that only ack data sent by the other side. We will remove the 0 packages in the activity below.

Activity This activity will be done with R. We will download the data from the CS-STATS web site. Read the data and remove the packets with databytes=0. Find the sample size. We will use only 2.3% of the data, that is 50000 packets, or 102 seconds worth of traffic.

```
These first two lines of code go together in one line... I had to cut them so they would fit..

=================   Reading the data --type all together...

dec1=read.table("http://www.stat.ucla.edu/~jsanchez/oid03/datasets/dec-pkt-1.tcp",

nrow=50000,header=T)   #read only 50000 lines of the data because data set too long

============

dec1[1:5,]     #get a feeling for the structure of this very processed  data

timestamp = dec1$timestamp[dec1$databytes  >  0]  #use only the lines for which databytes not (

databytes = dec1$databytes[dec1$databytes  >  0]

source=dec1$source[dec1$databytes  >  0]

destination=dec1$destination[dec1$databytes  >  0]

sourceport=dec1$sourceport[dec1$databytes  > 0]

destport=dec1$destport[dec1$databytes  >  0]

samplesize=length(timestamp)

samplesize
```

Questions 1. What is the sample size after you remove the packages with 0 databytes? Do a histogram of the databytes. Copy and paste it into your editor. What kind of distribution is this? What would explain it? Interpret this histogram. Summarize the data in a way that is appropriate for the kind of histogram you get. You may use any the following commands in R (choose what is appropiate).

```
 hist(databytes,main="histogram of package size")
summary(databytes)
summary(databytes[databytes < 300])
summary(databytes[databytes >=300])
```

Question 2. Do a plot of timestamp (on the horizontal axis) vs databytes (size of the package). Copy it in your editor. Interpret what you see. How does this plot help interpret the histogram in question 1? Are there any particular times at which the frequencies observed in the histogram happen or do the same phenomena seem to be happening all the time.

```
plot(timestamp,databytes,main="trace of package size over time")


plot(timestamp,databytes,type='l',main="trace of package size over time")
```

In the last plot make sure you wrote type="l", the "l" is the letter l, as in line (for line plot)

Question 3 Do the inter-arrival times (time gap between two consecutive arrivals) of packets seem to follow an exponential distribution? Do a graph of the interarrival times, find the summary statistics, and simulate exponential random variables with the same mean and variance. Compare the histogram of the simulated data with that of the actual data. What do you think? What does the qq-plot say? Copy and paste the two graphs into your editor.

```
arrivals = matrix(timestamp)
n=(length(timestamp))-1
interarrivals = matrix(rep(0,n),ncol=1)
for(i in 1:n) {
  interarrivals[i]= arrivals[i+1]-arrivals[i]
}      #we are measuring the time between each two arrivals

par(mfrow=c(2,1))   # open space to put two graphs together
hist(interarrivals,main="histogram of interarrivals in the data")    # don't copy this graph y
summary(interarrivals)
lambda=1/mean(interarrivals)
exponential =rexp(n,lambda)   #generate exponential random variables
hist(exponential,main="histogram of simulated exponential",xlim=c(0,max(interarrival))     #co
dev.off()    #close the graph window
qqplot(interarrivals,exponential,xlim=c(0,max(interarrivals)),ylim=c(0,max(interarrivals)),mai
```

If you wanted to put the graphs on the same scale, you could just add the option xlim=c(lowest number of the two graphs, highest number of the two graphs).

Question 4. Poisson distributions measure counts per unit of time. We can check also if number of packages received per unit of time follow a Poisson distribution or not. A direct way to do that is to plot the histogram and see whether we get a Poisson distribution and compare it with simulated Poisson data. Is the qq-plot consistent with what you see in the histogram?

```
pacpersecond = matrix(c(rep(0,102)))

for(i in 1:102) {
pacpersecond[i]=length(timestamp[(floor(timestamp))==i])
```

```
}   #compute the number of packets per second
pacpersecond=pacpersecond[pacpersecond>0]
summary(pacpersecond)
par(mfrow=c(2,1))  #open room for two graphs
hist(pacpersecond, main="histogram of packets per second")
hist(rpois(length(pacpersecond),mean(pacpersecond)),main="histogram of simulated Poisson",xlim=

Copy-paste the graphs, then type

dev.off()
qqplot(pacpersecond, rpois(length(pacpersecond),mean(pacpersecond)),xlim=c(0,600),ylim=c(0,600)
abline(0,1)
```

Note, to compare the histogram in the same scales you have to use the xlim option with histogram.

Question 5. A more indirect way to determine whether traffic follows power laws is to look at the number of packets per unit of time plotted against time and check this plot for several scales (i.e., first per second, then per minute, etc..). For Poisson data, the plot of the data trace is different for "packets per second" from the data trace for packets per millisecond, that is, the scale at which we measure the rate of traffic matters. For traffic the scale does not matter. See the attached picture 1 where Poisson traces and real traffic data is shown.

Our data set does not cover enough time to be able to do that kind of graph. We will do them when we analyze the whole data set. At the moment, let's just plot the trace of packets per second and describe it.

```
type this next command all in one line. I had to cut it to fit this handout

plot(pacpersecond,type='b',xlab="seconds",ylab="number of packs per second",
main="trace of number of packets per second")
```

References

Gautam, N. (2003) Stochastic Models in Telecommunications for Optimal Design, Control and Performance Evaluation. *Handbook of Statistics, Vol. 21* D.N. Shanbhag and C.R. Rao, eds. Elsevier Science B.V., 2003.

Digital Equipment Corporation. The traces were made by Jeff Mogul (mogul@pa.dec.com) of Digital's Western Research Lab (WRL).

The trace correspond to DEC-WRL-1 in Wide-Area Traffic: The Failure of Poisson Modeling, V. Paxson and S. Floyd, IEEE/ACM Transactions on Networking, 3(3), pp. 226-244, June 1995.