DVA D05 TCP Modelling Exercises

February 10, 2008

Exercise 1

We have seen in the lecture how the simple TCP equation, Eq 1 below, was derived[1].

$$X = \frac{s}{R\sqrt{\frac{2p_{le}}{3}}} \tag{1}$$

This equation gives the sending rate X [bytes/s] as a function of packet size s [bytes], round-trip time R [s], and packet loss event rate p_{le}. Note that p_{le} actually is the packet loss event rate, not the packet loss rate. A packet loss event is the loss of one or more packets during the same RTT. By using loss events instead of packet losses the TCP behavior of not halving the window more than once per RTT can be approximated. The mapping between packet loss rate and packet loss event rate depends on the packet loss distribution. For the rest of this lab you can assume that p_{le} =packet loss rate.

a.) Calculate the achieved throughput for the following set of parameters: s = 1500bytes, R = 10,20,50,100,200 ms and $p_{le} = 1 * 10^{-4}$, $1 * 10^{-3}$, $1 * 10^{-2}$, $5 * 10^{-2}$.

b.) Create one or more plots that show the throughput of TCP as a function of packet loss event rate for the different round trip times.

Exercise 2

We also saw that a slightly more refined variant of the throughput formula exists, that also captures the effect of the timeouts. Equation 2 below provides an expression that in addition to the window variation due to timeouts and duplicate acknowledgments.

$$X = \frac{s}{R\sqrt{\frac{2bp_{le}}{3}} + t_{RTO}3\sqrt{\frac{3bp_{le}}{8}}p_{le}\left(1 + 32p_{le}^2\right)} \tag{2}$$

This equation gives the sending rate X [bytes/s] as a function of packet size s [bytes], round-trip time (i.e RTT) R [s], number of received packets per ack b, retransmission timeout t_{RTO} [s] and packet loss event rate p_{le}. Assume b = 1and $t_{RTO} = 4 * R$ for the following calculations.

a.) Calculate the achieved throughput for the following set of parameters: s = 1500bytes, R = 10,20,50,100,200 ms and $p_{le} = 1 * 10^{-4}$, $1 * 10^{-3}$, $1 * 10^{-2}$, $5 * 10^{-2}$.

b.) Create one or more plots that show the throughput of TCP as a function of packet loss event rate for the different round trip times.

[1] Which happens to be practically identical to the formula you used in Lab 1, exercise B6-

c.) For what range of packet loss event rates are the difference in sending rate between Equations 1 and 2 more than 1%?

d.) Again use the RTT samples from http://www.cs.kau.se/~perhu/labdata.data. What does the loss rate need to be for the two equations to create a statistically significant difference at the 95% confidence level?

e.) As was discussed during the lecture, most TCP implementations employ *delayed acknowledgments*. This is can be modeled by Eq. 2 by setting $b = 2$. How much percentual decrease does the use of delayed acks create for $s = 1500$bytes, $R = 50$ ms and $p_{le} = 1 * 10^{-4}$, $1 * 10^{-3}$, $1 * 10^{-2}$, $5 * 10^{-2}$. (To extend this, you can also plot the decrease over loss event rate for some different RTTs.

Exercise 3

We have seen in the slides that a model can be created that approximates the delay of transferring some object by taking into consideration the effect of the underutilization during slow start, and assuming that no losses occur.

a.) Use the model in the slides to calculate the time it takes to transfer a 26312 bytes HTML page from a web server over a link which has an RTT of 55 ms, and an available throughput of 2Mbps, and no loss.

b.) Now make a calculation by hand using a time-sequence diagram for each exchange, and consider the effects of delayed acks and also consider the space taken by the involved headers. What time do you arrive at? how large is the percentual difference?

c.) Now assume that the 5th packet gets lost. How long will it take the connection for this case?

d.) We know that Equation 2 describes only the steady state behavior, and does not take the slow start effects into account. However, we are interested in finding out how large effect this might have. Calculate the throughput according to Eq. 2 and compare that to the effective throughput you got for question c above.

Exercise 4

During the lectures we have discussed max-min fairness and proportional fairness. Consider the network topology below. Assume all links to have capacity 5 Mbps and delay 10 ms. Now consider the average steady state throughput for the set of long lived TCP flows.

a.) What would a max-min fair allocation of rates be? What would a proportionally fair allocation of rates be?

b.) For the illustrated network, will TCP allocate the rates according to max-min fairness, proportional fairness or something else? Why/why not?

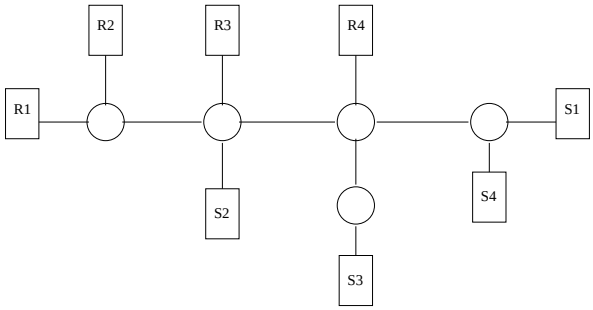

Figure 1: Example Topology

Answers

Exercise 1

a)

```
octave-2.9.15:117> function X=simpletcp(s,R,p)
> X=(s./(R.*sqrt(2.*p./3)))
> endfunction
octave-2.9.15:191> simpletcp (1500,0.01,[0.0001 0.001 0.01 0.05]);
X =
1.8371e+07 5.8095e+06 1.8371e+06 8.2158e+05
octave-2.9.15:192> simpletcp (1500,0.02,[0.0001 0.001 0.01 0.05]);
X =
9.1856e+06 2.9047e+06 9.1856e+05 4.1079e+05
octave-2.9.15:193> simpletcp (1500,0.05,[0.0001 0.001 0.01 0.05]);
X =
3.6742e+06 1.1619e+06 3.6742e+05 1.6432e+05
octave-2.9.15:194> simpletcp (1500,0.1,[0.0001 0.001 0.01 0.05]);
X =
1.8371e+06 5.8095e+05 1.8371e+05 8.2158e+04
octave-2.9.15:195> simpletcp (1500,0.2,[0.0001 0.001 0.01 0.05]);
X =
9.1856e+05 2.9047e+05 9.1856e+04 4.1079e+04
octave-2.9.15:196>
```

Note that if you forget to use the correct element-wise matrix operations you will get erroneous results:

```
octave-2.9.15:104> function X=simpletcp(s,R,p)
> X=(s/(R*sqrt(2*p/3)))
> endfunction
octave-2.9.15:115> simpletcp (1500,[0.01 0.02 0.05 0.1 0.2]',0.00001);
X =
1.0961e+05 2.1923e+05 5.4806e+05 1.0961e+06 2.1923e+06
```

b)

see plot command

Exercise 2

a)

```
octave-2.9.15:131> function X=tcp (s,R,p,b)
> X=s./(R.*sqrt(2.*b.*p./3)+4.*R.*3.*sqrt(3.*b.*p./8).*p.*(1+32.*(p.^2)))
> endfunction
octave-2.9.15:196> tcp (1500,0.01,[0.0001 0.001 0.01 0.05],1);
X =
1.8355e+07 5.7577e+06 1.6850e+06 5.5288e+05
octave-2.9.15:197> tcp (1500,0.02,[0.0001 0.001 0.01 0.05],1);
X =
9.1773e+06 2.8788e+06 8.4249e+05 2.7644e+05
```

```
octave-2.9.15:198> tcp (1500,0.05,[0.0001 0.001 0.01 0.05],1);
X =
3.6709e+06 1.1515e+06 3.3700e+05 1.1058e+05
octave-2.9.15:199> tcp (1500,0.1,[0.0001 0.001 0.01 0.05],1);
X =
1.8355e+06 5.7577e+05 1.6850e+05 5.5288e+04
octave-2.9.15:200> tcp (1500,0.2,[0.0001 0.001 0.01 0.05],1);
X =
9.1773e+05 2.8788e+05 8.4249e+04 2.7644e+04
octave-2.9.15:201>
```

b)

see plot command

c)

Test for approximate value:

```
octave-2.9.15:205> p=[0.00001 0.0001 0.001 0.01 0.05]
p =
1.0000e-05 1.0000e-04 1.0000e-03 1.0000e-02 5.0000e-02
octave-2.9.15:207> tcp (1500,0.01,p,1)./simpletcp(1500,0.01,p)
X =
5.8090e+07 1.8355e+07 5.7577e+06 1.6850e+06 5.5288e+05
X =
5.8095e+07 1.8371e+07 5.8095e+06 1.8371e+06 8.2158e+05
ans =
0.99991 0.99910 0.99108 0.91719 0.67295
```

Iterate to find answer:

```
octave-2.9.15:211> while (tcp (1500,0.01,p1,1)./simpletcp(1500,0.01,p1))>0.99
> p1=p1+1e-6
> end
```

p1 = 0.0011230

X = 5.4272e+06

X = 5.4821e+06

octave-2.9.15:212>

So the answer would be for loss event rate > 0.00112

d) When using the most appropriate statistic method, (i.e before and after comparison), there will always a statistically significant difference. When calculating the confidence intervals for the two means (tcp and simpletcp) it should be possible to get loss rate value that creates a 95% significant difference.

e)

```
octave-2.9.15:228> ((tcp (1500,0.05,p,1).-tcp(1500,0.05,p,2))./tcp(1500,0.05,p,1))*100
X =
1.1618e+07 3.6709e+06 1.1515e+06 3.3700e+05 1.1058e+05
X =
8.2151e+06 2.5957e+06 8.1426e+05 2.3829e+05 7.8189e+04
X =
1.1618e+07 3.6709e+06 1.1515e+06 3.3700e+05 1.1058e+05
ans =
```

```
29.289 29.289 29.289 29.289 29.289
octave-2.9.15:229>
```

As can be seen, the difference is not dependent on the loss rate.

Exercise 3

a)

Assume MSS= 1460bytes.

```
octave-2.9.15:239> O=26312*8
O = 210496
octave-2.9.15:240> S=1460*8
S = 11680
octave-2.9.15:241> R=2e6
R = 2000000
octave-2.9.15:242> RTT=0.055
RTT = 0.055000
octave-2.9.15:243> K=ceil(log2(O/S+1))
K = 5
octave-2.9.15:244> K=ceil(log2(O/S+1))Q
octave-2.9.15:244> Q=floor(log2(1+RTT/(S/R)))+1
Q = 4
octave-2.9.15:245>del=O/R+2*RTT+4*(RTT+S/R)-(2^4-1)*S/R
del = 0.37101
octave-2.9.15:246>
```

b)

The difference for this case will be caused by the additional time taken for the delayed acknowledgment, and the fact that the window opens up slower. Recall that delayed acknowledgments wait up to 200ms for a second segment to arrive, and if a second segment arrives within that time only generates one acknowledgment for the two segments. If no second segment arrives, then an acknowledgment will be generated 200ms after the reception of the first and only segment. When you draw up the message exchanges, the figure should generate in the following expression:

```
octave-2.9.15:24> del2=O/R+2*RTT+5*(RTT+S/R)-11*S/R+0.2
del2 = 0.65521
octave-2.9.15:25> del=O/R+2*RTT+4*(RTT+S/R)-(2^4-1)*S/R
del = 0.37101
octave-2.9.15:26> del2/del
ans = 1.7345
```

So this more detailed calculation gives a transfer time that is 73 per cent higher that the formula in the slides.

c)

The answer on this question depends on which TCP variant that is drawn, but in general the answer will be at least 0.1 s larger than for b. Doing a calculation using limited transmit gives:

```
octave-2.9.15:28> del2=O/R+2*RTT+8*(RTT+S/R)-12*S/R+0.2
del2 = 0.83189
octave-2.9.15:29> O/del2
ans = 2.5303e+05
```

So the effective throughput for this case was 253 Kbps, or one eighth of the available bandwidth of 2Mbps.

d)

The Since there are 20 packets sent in the direction from the server (not counting the packets for connection tear down..) we can assume a packet loss event rate of 0.05. When entered into the equation we get:

```
octave-2.9.15:434> tcp(1500,0.055,0.05,2)
X = 7.1081e+04
ans = 7.1081e+04
octave-2.9.15:435>
```

So, the approximation done by the steady state formula is much more conservative.

Exercise 4

a)

For max-min fairness the answer is $T_{R1-S1} = T_{R2-S2} = T_{R3-S3} = T_{R4-S4} = 1.25$Mbps. For proportional fairness (with the utility function set to maximize system throughput) the proportionally fair can be set according to $5 = 3T_F + T_F + T_F + T_F$ where $T_F$equals to one "connection-hop" of system load. Since the R1-S1 covers three hops with competing traffic it counts for $3T_F$. So, for the simplistic example of this exercise the proportionally fair the proportionally fair allocation is $T_{R1-S1} = \frac{1}{6} * 5 = 0.833$Mbps and $T_{R2-S2} = T_{R3-S3} = T_{R4-S4} = \frac{5}{6} * 5 = 4.167$Mbps. For other more complex topologies it can be solved by using the derivative or calculating to fit.

The solution can be verified with the summation formula from the slides using c for the $T_{R2-S2} = T_{R3-S3} = T_{R4-S4}$rates:

```
octave-2.9.15:278> c=4.16667
c = 4.1667
octave-2.9.15:279>
octave-2.9.15:279> x1 = [5-c c c c]
x1 =
0.83333 4.16667 4.16667 4.16667
octave-2.9.15:280> d=0.001
d = 0.0010000
octave-2.9.15:280> x2 = [5-(c+d) c-d c-d c-d]
x2 =
0.83233 4.16567 4.16567 4.16567
octave-2.9.15:281>
octave-2.9.15:281> sum((x2 .- x1) ./ x1)
ans = -0.0019200
octave-2.9.15:282>
octave-2.9.15:282> d=-0.001
d = -0.0010000
octave-2.9.15:283>
octave-2.9.15:283> x2 = [5-(c+d) c-d c-d c-d]
x2 =
0.83433 4.16767 4.16767 4.16767
octave-2.9.15:284> sum((x2 .- x1) ./ x1)
ans = 0.0019200
octave-2.9.15:285>
```

If we try with another value, one sum will be greater than zero:

```
octave-2.9.15:271> c=4.1
c = 4.1000
octave-2.9.15:272> x1 = [5-c c c c]
x1 =
0.90000 4.10000 4.10000 4.10000
octave-2.9.15:282> d=0.001
d = 0.0010000
octave-2.9.15:273>
octave-2.9.15:273> x2 = [5-(c+d) c-d c-d c-d]
x2 =
0.90100 4.10100 4.10100 4.10100
octave-2.9.15:274>
octave-2.9.15:274> sum((x2 .- x1) ./ x1)
ans = 0.0018428
octave-2.9.15:275>
octave-2.9.15:275>
octave-2.9.15:275> d=-0.001
d = -0.0010000
octave-2.9.15:276>
octave-2.9.15:276> x2 = [5-(c+d) c-d c-d c-d]
x2 =
0.89900 4.09900 4.09900 4.09900
octave-2.9.15:277>
octave-2.9.15:277> sum((x2 .- x1) ./ x1)
ans = -0.0018428
octave-2.9.15:278>
```

b)

TCP will behave more like proportional fairness than max-min fairness since the throughput a TCP flow that competes with other TCP flows are dependent on their relative RTTs. Since flow R1-S1 will have a longer RTT than the other flows, it will get less bandwidth than them. However, the RTT dependence of TCP can also create unfairness. If the flows R3-S3 and R4-S4 are subject to the same loss rate, the throughput of R3-S3 will be lower than that of R4-S4 since R3-S4 have a larger RTT. So even though they consume the same amount of constrained resources (one "connection-hop") they will not get the same resources. In a proportional fairness model they would get the same throughput since they consume the same amount of constrained resources.