

## **Broadband Networks**

**Prof. Karandikar**

**Electrical Engineering Department**

**Indian Institute of Technology, Bombay**

**Lecture - 18**

**Analysis of TCP**

So, as we had seen that the TCP has 2 phases; one, we call it to be a slow start phase and in the slow start phase, the congestion window doubles after every round trip times, after every acknowledgement, the congestion window increases by 1. So, when the acknowledgement for all the packets transmitted during a window had come; then the window size actually gets doubled. So, this is what is called as the slow start phase. So, even though the increase is faster but still we call in to be a slow start phase because otherwise historically the TCP used to start the window from window size equal to the receivers advertised window size.

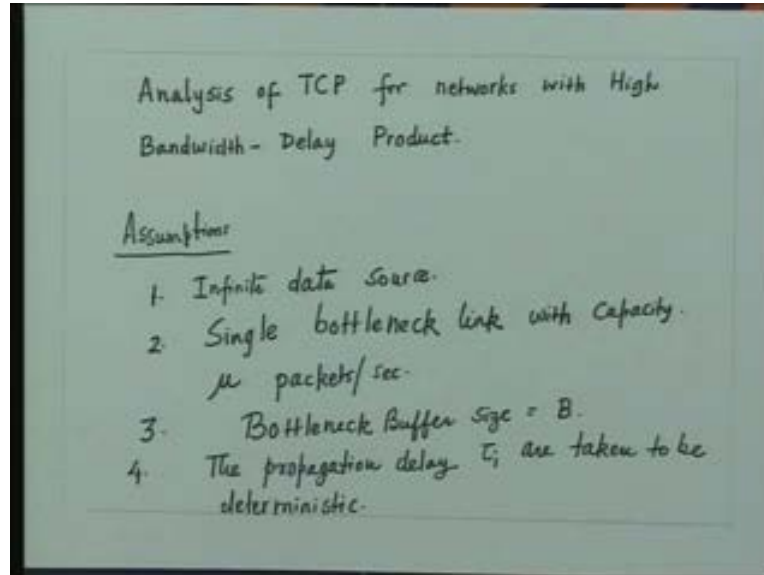
So, in the slow start phase, the congestion window increases exponentially **and it increases exponentially** till it reaches a window size which we call it to be a slow start threshold or a slow start threshold variable. So, it reaches the window size upto that and after that the TCP goes into the congestion avoidance space and in the congestion avoidance space, we follow an additive increase and multiplicative decrease procedure.

Now however, in the slow start phase if you are exponentially increasing the window size and if the TCP detects the packet loss, then again the window size drops to 1 and the TCP enters into a slow start phase again. However, if there is a no packet loss and the and the window size increases upto the slow start threshold window size and after that it enters into the congestion avoidance phase; then if the TCP detects the packet loss due to the triple duplicate acknowledgement, then the window size drops to half of the window size at which the packet loss is detected and again the TCP enters into the congestion avoidance phase only.

Other hand, if the packet loss is detected due to the retransmission timer timeout; then the window size drops to 1 and source enters again into the slow start phase followed again by the congestion avoidance phase. But in the second case that in the second timer, it enters into the slow start phase; the value of the slow start threshold variable is set equal to half of the window size at which the packet loss was detected.

Now, today what we would try to do is that we will try to analyze this slow start phase and the congestion avoidance phase and determine the throughput for such a TCP source with slow start phase and congestion avoidance phase in networks which have high bandwidth delay products.

(Refer Slide Time: 4:44)



So, there are certain assumptions. We will state these assumptions, so the assumptions are; we assume that we have an infinite data source that means the source always has a data to transmit, we also assume that there is a single bottleneck link with capacity of  $\mu$  packets per second, we assume that the bottleneck buffer size is  $B$  and the propagation delay  $\tau_i$  are taken to be deterministic.

So, what we are assuming is that there is a single data source, one data source and there is one receiver and in the network there is one bottleneck node, we assume a single bottleneck node and that bottleneck node has a buffer capacity of  $B$  and the node serves the packet with the rate of  $\mu$  packets per second and the propagation delay  $\tau$ , the propagation delay  $\tau$  that is assumed to be constant. So, this we call it to be, so let us say  $\tau$ . So, if we define the window evolutions of...

Let us define the normalized buffer size.

(Refer Slide Time: 6:50)

Normalized Buffer size

$$\beta = \frac{B}{\mu\tau + 1} = \frac{B}{\mu T}$$

$T = \tau + \frac{1}{\mu}$

↑ propagation delay      service time

(B.W. × Delay)

if  $\beta \leq 1$  (large Bandwidth-Delay product).

$$\Rightarrow \mu T \geq B$$

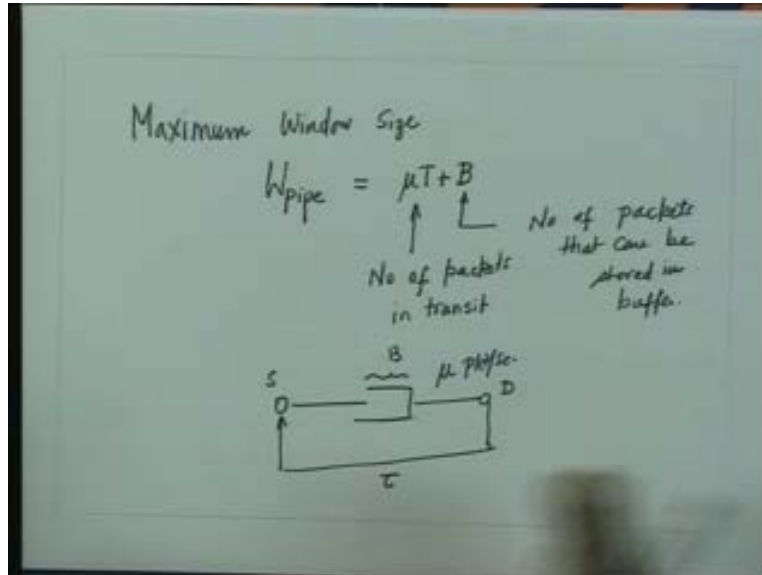
So, the normalized buffer size beta is defined to be B upon mu tau plus 1. So, this is the normalized buffer size. So, let us define T to be equal to tau plus 1 upon mu. So now, this is the propagation, this is the propagation delay and this is the service time. So, this is the total round trip time. So, tau therefore donates the round trip time.

So, this I can write to be B upon mu T. Now, mu is the service rate, T is a round trip time; so therefore, this is the delay and this is the bandwidth. So, if **B is less than or equal** beta is less than or equal to 1 that means we have the case of large bandwidth delay product. Essentially, it means that the bandwidth delay products, so this is what is called as the bandwidth into delay product, this mu T is greater than or equal to the bottleneck buffer size.

So, we would like to analyze the situation or the throughput characteristics of a TCP source in networks which are having large bandwidth delay products. By large bandwidth delay products we say that the product of the bandwidth into the delay and where the delay happens to be the sum of the propagation delays plus the service time that is greater than or equal to the buffer size, that is greater than the bottleneck buffer capacity.

Now, let us try to therefore analyze such networks which are having large bandwidth delay products.

(Refer Slide Time: 9:44)



So, the analysis of such networks which are having large bandwidth delay products: now, the maximum window size that can be there, so we call it to be the  $W_{\text{pipe}}$ , the maximum window size or  $W_{\text{max}}$  will be  $\mu T$  plus  $B$ . Now, these many number of packets can be there in the transit and these many number of packets that can be stored in the buffer; so, this is sort of a **maximum buffer size** maximum window size. This indicates what can be the maximum window size in the network. That is the maximum window size actually denotes what is the maximum number of unacknowledged packets that can be there in the network.

Now, the maximum number of unacknowledged packet that can be there in the network is equal to the number of packets that can be there in the transit plus the number of packets that are actually queued up in the buffer. So,  $\mu T$  plus  $B$  actually indicates the maximum number of unacknowledged packets that can be there in the networks and therefore by definition, it will be the maximum window size.

Now, let us try to see how the window evolution takes place during the slow start phase; the things will become then clear. So, the situation is something like this that there is a source, we got a bottleneck buffer of size  $B$  and this is serving at a rate of  $\mu$  packets per source and we have the propagation delay of  $\tau$  and this source packets at the rate of  $\mu$  packets per second and this is the destination and the propagation delay is  $\tau$ . So, this model actually is a very simplified model of the TCP connection between a source and a destination in a large bandwidth networks.

So, **now let us try so** our assumption is that a source has always data to send, therefore it is an infinite data source. Our assumption is also that the propagation delay is assumed to be constant; usually in practice, the propagation delay may not be constant but it is assumed to be constant here and there is only one single bottleneck node and then that bottleneck node has a capacity of holding  $B$  packets and it has the service rate of  $\mu$  packets per second.

So now, let us try to see how the window evolution takes place during the slow start phase. Now, the window evolution that takes place in the slow start phase is something like this that - let us say; time, the packet which has been acked, window size, packet number and then queue length.

(Refer Slide Time: 12:56)

Slow Start Phase

Time	Packet Ached	Window	Packet No	Queue Length
0		1	1	1
T	1	2	2, 3	2
2T	2	3	4, 5	2
$2T + \frac{1}{\mu}$	3	4	6, 7	3
3T	4	5	8, 9	2
$3T + \frac{1}{\mu}$	5	6	10, 11	3
$3T + \frac{2}{\mu}$	6	7	12, 13	4
$3T + \frac{3}{\mu}$	7	8	14, 15	5
4T	8	9	16, 17	2

Now, if you see at time 0, window size is 1 in the slow start phase. So, we release the packet number 1 and **the queue length** that packet will go into this buffer, this packet will go directly into the buffer when the packet number 1 is released and it will sit here and then the queue length will be 1. Now, after T time unit interval, at time T; now note that this T here consists of tau plus 1 upon mu that is a propagation delay plus service time. So, when this packet 1 has been released here; after the time T, the packet that has been acknowledged will be the packet number 1 and when this packet is acknowledged, the window size becomes 2 and then we will release the packet 2 and 3, that will be released and these 2 packets will go into the buffer and sit there.

So now, what is happening? During the slow start phase; initially we start from the window size equal to 1, we release that packet, when that packet's acknowledgement comes back after a round trip time and that is denoted by capital T here, when the packets acknowledgement comes back after the round trip time, the window size now doubles. So, the window size becomes 2. The window size, for every acknowledgement; the window size increases by 1 in the slow start phase.

So, when this particular packets acknowledgement comes back, the window size becomes 2. So, we release now the 2 packet, these 2 packets go and sit into the buffer. Now, if you see, the packet number 2 which was released earlier in this case, the packet number 2. When that gets serviced after a time interval of T that means at time 2T, the packet which will get at 2; now when this packet comes here, 3 actually starts getting serviced.

So, the window size **now becomes**, when the packet number 2 has been acknowledged; the window size becomes 3. Already 3 is there in the transit and therefore we can now release the

packet number 4 and 5. And since, 3 is there in the transit, these 2 packets will go and sit into the buffer and therefore the queue length is again 2. Now, **the packet number 3 will** the acknowledgement for packet number 3 will come back to the source after another interval of  $1 \mu$  because the difference between 2 and 3 will be  $1 \mu$ , where  $1 \mu$  is the service time. The packet that gets acknowledged is 3 and when the packet that gets acknowledged is 3, the window size becomes 4.

Now, when the window size becomes 4; now at this point when 3 was in the transit, the packet number 4 was getting served and since the window size is 4, packet number 5 is already there in the buffer, we can release the packet number 6 and 7. So therefore, 5, 6, 7 will be there in the queue, so the queue length is 3; 4 will be there on the transit and therefore the total window size is 4 and that indicates the maximum number of unacknowledged data that can be there in the transit.

Now similarly, **at  $3T$  and the  $3T$**  at time  $3T$ , we now get the acknowledgement from the packet number 4 which was released at time  $2T$  and then the window size becomes 5 and now the packet that will be released will be 8 and 9. And, these 8 and 9 packets will go and sit into the buffer. Note that the packet 5, 6, 7, they are there in the transit **5, 6**. So therefore, at  $3T + 1 \mu$ , you get the packet which is acknowledged is 5; the window size becomes 6, the packet that are realized are 10 and 11 and then the queue length becomes 3.

Note that in the queue, the packet number 9, 10, 11 will be there and packet 6, 7, 8 are there in the transit and then they will get acknowledged here. So, we will have  $3T + 2 \mu$ , where we get the packet number 6 acknowledged. The window size becomes 7 and we release the packet number 12 and 13 and the queue length becomes 4.

Similarly, we get after  $3T + 3 \mu$  and  $3T + 3 \mu$ , we will get the acknowledgement for 7 and then the window size will become 8. And 8, we will release the packet number 14 and 15 because note that there are already 4 packets in the queue and 3 packets in the transit. So therefore, the window size in that case becomes 5 and so on. Now, at  $4 \mu$ :  $4 \mu$  is essentially nothing but the  $4T$  and at this point, we will receive the acknowledgement of 8 here.

So, this way the evolution of the window takes place in the slow start phase and as you can see from this example that the window is essentially evolving in an exponential manner during the slow start phase. You can see here, at the time  $4T$ , the acknowledgement for 8 comes, the window size becomes 9 and you will acknowledge packet number 16 and 17. So, the window size is 9, the acknowledgement for packet number 8 has come and note that all the packets that is 9, 10, 11, 12, 13, 14 - they are all in the transit and therefore the queue length again will be 2.

So therefore, this illustrates the exponential nature of the window evolution. Now, let us define each of these that is  $T$  and this interval **and this interval** to be the mini cycles.

(Refer Slide Time: 20:48)

Handwritten notes on a whiteboard:

- $i^{\text{th}}$  Mini cycle denoted by  $[iT, (i+1)T]$
- Let  $W(t) =$  window size at time  $t$
- In  $(n+1)^{\text{th}}$  mini cycle
 
$$W\left(nT + T + \frac{m}{\mu}\right) = 2^{n+m+1} \quad \text{for } 0 \leq m \leq 2^n - 1$$
- In  $n^{\text{th}}$  mini cycle
 
$$W\left(nT + \frac{m}{\mu}\right) = 2^{n-1+m+1} \quad 0 \leq m \leq 2^{n-1} - 1$$

So, we define the  $i^{\text{th}}$  mini cycle to be denoted by  $i T$  into, this indicates our  $i^{\text{th}}$  mini cycle. Now, let us say that  $W(t)$  denotes the window evolution, window size at time  $T$ . Now, if you see here, if you observe the behavior here; this is the first mini cycle, the second mini cycle, the third mini cycle, the fourth mini cycle and so on.

So, if you see that in  $n$  plus  $1^{\text{th}}$  mini cycle **in  $n$  plus  $1^{\text{th}}$  mini cycle**; the window size  $W(nT + T + m/\mu)$ , so the window size at  $(nT + T + m/\mu)$  is given by  $2^{n+m+1}$  for  $m$  lying between  $2^{n-1}$  and  $0$ . Or in other words, if you see in  $n^{\text{th}}$  mini cycle **in  $n^{\text{th}}$  mini cycle**, it will be  $W(nT + m/\mu)$  which will be equal to  $2^{n-1+m+1}$  for  $m$  lying between  $2^{n-1}$  and  $0$ .

So, as you can see here, let us say that **if you have** if you have  $n$  is equal to  $3$ , if you keep  $n$  is equal to  $3$  here and then if  $m$  is  $0$ ; the window size is  $3 - 1$ ,  $2$  raised to power  $2$  that is  $4 + 1$  that is  $5$  here. If you take now  $3T + 1/\mu$  that is  $m$  is  $1$ , then it becomes equal to  $5 + 1$  that is  $6$  here and so on. So, this way you can determine what will be the mini cycles, the window size at the  $n^{\text{th}}$  mini cycle.

Similarly, the queue size, **the queue** the queue length during the  $n^{\text{th}}$  mini cycle is given by  **$n^{\text{th}}$  mini cycle is given by**  $Q$  of  $(nT + m/\mu)$  will be equal to  $m + 2$ , again for  $m$  lying between  $0$  to  $2^{n-1}$ . So, this is the queue length. Now, if you see from here in the  $n^{\text{th}}$  mini cycle, the maximum window size will be when  $n$  achieves the value of  $2^{n-1}$  and that will be  $2^{n-1} + 2$ .

So, as you can see here that in  $n^{\text{th}}$  mini cycle, the maximum window size  $W_{\text{max}}$  will be equal to  $2^{n-1} + 2$  and similarly the maximum queue length **maximum queue length** if you want to know, then you have  $Q_{\text{max}}$  which again if you put  $n$  is equal to  $2^{n-1}$ ;

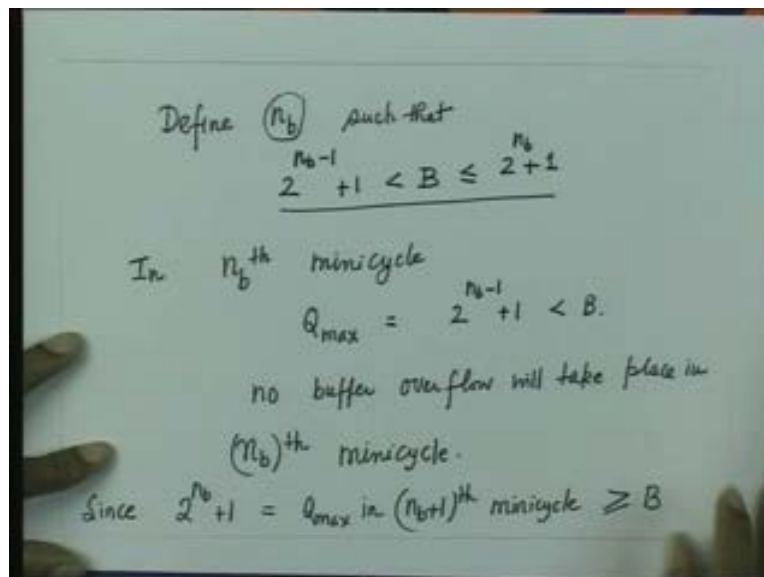
we will have  $2$  raised to power  $n$  minus  $1$  plus  $1$  which is equal to  $2$  raised to power  $n$  minus  $1$  plus  $1$  which we can approximate to be equal to  $2$  raised to power  $n$  minus  $1$ .

So, if you see from here that the  $Q_{\max}$  is approximately equal to  $W_{\max}$  by  $2$ . The  $Q_{\max}$  is approximately half of the maximum window size which will become clearer from this example here. As you can see here that the maximum window size in this mini cycle is  $8$  and half of it is  $4$  which is approximately equal to the maximum queue length which actually here  $5$ .

So, as we have seen that **the slow start** during the slow start phase, the window increases in an exponential fashion and we can write an expression for the window size evolution as well as the queue length evolution. And, we have also seen that the maximum  $Q$  size that is the maximum  $Q$  size will be approximately equal to half of the maximum window size in the  $n$ 'th mini cycle.

Now, let us do one thing that let us define some variable called  $n_b$ .

(Refer Slide Time: 26:35)



So, let us say that **define  $n_b$  such that** define  $n_b$  such that  $2$  raised to power  $n_b$  minus  $1$  plus  $1$  is less than  $B$ . That is  $B$  is the buffer size but the  $B$  itself is less than or equal to  $2$  raised to power  $n_b$  plus  $1$ . So, we choose an  $n_b$  such that  $2$  raised to power  $n_b$  plus  $1$  is greater than or equal to the buffer size but  $2$  raised to power  $n_b$  minus  $1$  plus  $1$  is less than the buffer size.

So, we have chosen some number  $n_b$  such that this relationship holds good. Now, in let us say  $n_b$ 'th mini cycle; what will be the  $Q_{\max}$ ? If you say from our expression for the  $Q_{\max}$  that is the maximum  $Q$  length, the maximum  $Q$  length will be equal to  $2$  raised to power  $n_b$  minus  $1$  plus  $1$ . So, it will be  $2$  raised to power  $n_b$  minus  $1$  plus  $1$ . But since this is less than  $B$ , no buffer overflow will take place in  $n_b$ 'th mini cycle.

So, what we are saying is that choose some number  $n_b$  such that  $2$  raised to power  $n_b$  minus  $1$  that is plus  $1$ , the buffer size  $B$  lies between these  $2$  quantities. So, if you now take  $n_b$ 'th mini



cycle, **if I now take  $n_b$ 'th mini cycle** we know that in  $n_b$ 'th mini cycle, the maximum Q length will reach 2 raised to power  $n_b$  minus 1 plus 1.

Now since by our assumption this quantity is less than the buffer size, no buffer overflow will take place in the  $n_b$ 'th mini cycle. However, since 2 raised to power  $n_b$  plus 1 which is going to be the maximum Q length in the  $n_b$  plus 1'th mini cycle; the buffer overflow can occur in the  $n_b$ 'th plus 1 mini cycle.

(Refer Slide Time: 29:45)

Buffer overflow will occur in  $(n_b+1)$ th mini-cycle.

$$Q(n_b T + T + \frac{m}{\mu}) = \text{queue length in } (n_b+1)\text{th mini-cycle}$$

$$= m+2$$

if  $= B+1$  buffer overflow will occur

$$m = B - 1$$

for this situation, window size  $n_b$

$$W_b = W(n_b T + T + \frac{m}{\mu}) = 2 + m + 1 = 2^{n_b} + B.$$

So, since 2 raised to power  $n_b$  plus 1 which is equal to  $Q_{\max}$  in  $n_b$  plus 1'th mini cycle is greater than or equal to B; buffer overflow will occur in  $n_b$  plus 1'th mini cycle. Now, the question really is that note that the maximum Q length which is going to be there in  $n_b$  plus 1'th mini cycle is greater than or equal to B. That is the maximum Q length but **the Q length can become** the buffer can fall short of the Q length somewhere in between also during the  $n_b$  plus 1'th mini cycle.

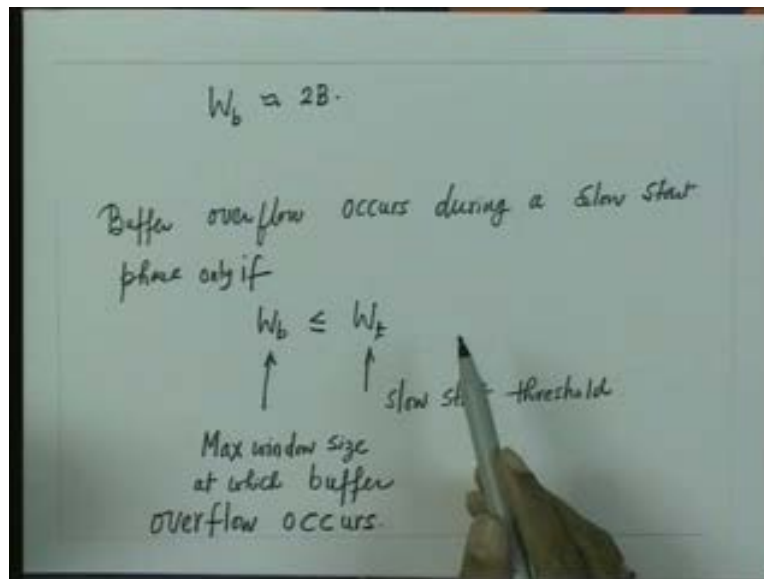
So, we need to determine at what point the Q length exceeds the maximum buffer size. So therefore during this cycle, the Q length will exceed the buffer size for that value of M for which... so this is the Q length in  $n_b$  plus 1'th mini cycle and this in general is given by an expression for m plus 2 and whenever that becomes equal to B plus 1, if it becomes equal to; buffer overflow will occur. So therefore, m should be equal to B minus 1.

So, at this point - m equal to B minus 1, the buffer overflow occurs. So, for this situation, the window size is we call it to be  $W_b$  which is given by W of  $(n_b T$  plus T plus m by mu) and if I keep m equal to B minus 1; so now actually this is equal to 2 raised to power  $n_b$  plus m plus 1 which becomes equal to 2 raised to power  $n_b$  plus B.

So, you can see here from this situation because we have an expression for the Q length variations. For the Q length variations in the **nT mini size**  $n_b$  plus 1'th mini cycle is given by 2

raised to power  $n_b + 1$  and that is what we have written here -  $2^{n_b + 1}$  which is given by  $2^{n_b + B}$ . Now since by definition our buffer  $B$  is less than or equal to  $2^{n_b + 1}$  and  $B$  is greater than  $2^{n_b - 1} + 1$ ; we can approximate the  $W_b$  to be equal to  $W_b$  to be equal to  $2B$ . It is approximated to be equal to  $2B$ .

(Refer Slide Time: 33:14)



Now, **note that now** what is happening? Let us see that the window evolution during the slow start phase is exponential, that is what we have seen. And, we have argued that the buffer will overflow in the  $n_b + 1$ 'th mini cycle if the  $Q$  length exceeds the buffer size and the window size reaches approximately equal to  $2B$ . So, when the window size reaches approximately equal to  $2B$ ; the buffer overflow will occur.

Now, the thing is this to happen that the buffer overflow should occur in the slow start phase; this maximum window size that is  $W$  should be less than or equal to the slow start threshold because **if it is** if it is not less than or equal to the slow start threshold, then actually the slow start threshold itself would have passed.

So, the buffer overflow occurs during a slow start phase **if the** only if  $W_b$  is less than or equal to  $W_t$  which is slow start threshold and  $W_b$  is the max window size at which buffer overflows. So, if this window size  $W_b$  is less than or equal to the slow start threshold, then only the buffer overflow will occur during the slow start phase. **At least** I mean, what we are trying to say is that that a slow start phase should last.

Now, the question is how we have chosen the slow start threshold that is  $W_t$ . So now, let us see how we have chosen  $W_t$ .

(Refer Slide Time: 35:50)

$$W_{\max} = \mu T + B = W_{\text{pipe}}$$
$$W_t = \frac{W_{\text{pipe}}}{2} = \frac{\mu T + B}{2} = \frac{(1 + \beta)\mu T}{2}$$
$$W_b = 2B = 2\beta\mu T$$

Buffer overflow occurs during slow start phase if  $2\beta\mu T \leq \frac{(1 + \beta)\mu T}{2}$

$$\Rightarrow \beta \leq \frac{1}{3}$$

Now, note that the maximum window size that can be there -  $W_{\max}$  can be  $\mu T$  plus  $B$ ; this was our  $w_{\text{pipe}}$  because this is the bottleneck buffer size, so this denotes the maximum number of packets that can be there in the buffer and  $\mu T$  denotes the maximum number of packets that there can be there in the transit. So therefore,  $\mu T$  plus  $B$  is the maximum window size that can be there for this source destination pair **which has the bottlenecks** which has a single bottleneck node with the link capacity of  $\mu$  and the buffer size of  $B$ . So,  $\mu T$  plus  $B$  can be the maximum window size for this source destination pair.

So, this is the maximum window size and typically we will say the slow start threshold  $W_t$  to be half of this, that is  $W_{\text{pipe}}$  by 2 and that we will keep it to be equal to  $\mu T$  plus  $B$  by 2 and which denotes to be equal to  $1 + \beta$   $\mu T$  upon 2; if you write this expression in terms of  $\beta$ .

Now,  $W_b$  if we have written to be equal to  $2B$  which we can approximate to be equal to  **$2B$  into which is equal to 2** twice of  $\beta$  into  $\mu T$ , that means a buffer overflow will occur during a slow start phase; buffer overflow occurs during a slow start phase if twice of  $\beta$   $\mu T$  is less than or equal to  $1 + \beta$   $\mu T$  by 2 and which implies that  $\beta$  should be less than or equal to  $1/3$ .

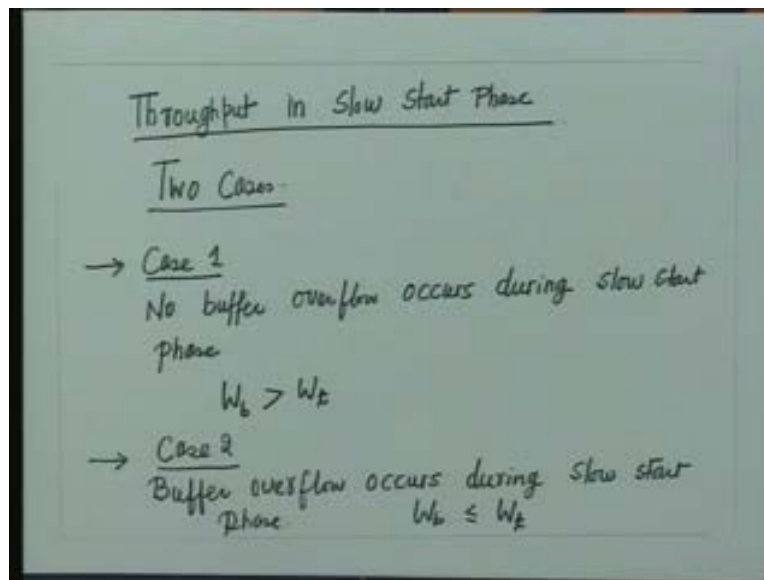
So, what it is defined to say is that if the  $\beta$  that is a normalized buffer size; if it is less than or equal to 0.33, then the buffer overflow can occur during a slow start phase. That means the packet can be lost during the slow start phase. Now, one thing; if the packet is lost during the slow start phase, then 2 things happens. Whenever, the source detects that the packet loss has occurred, it drops the window size to  $W$  equal to 1 and enters into another slow start phase **and it enters into another slow start phase**.

Now, when it enters into another slow start phase; this time this slow start threshold variable will not be the original slow start threshold variable which was used in the previous slow start phase

that will not be. But it will be half of that because whenever you detect a packet loss, you set your slow start threshold variable equal to half of window size at which the packet loss has been detected.

Now, let us try to therefore analyze what will be the throughput of this TCP connection in the slow start phase.

(Refer Slide Time: 39:27)



So, we now try to determine the throughput in the slow start phase. Now in this case, we consider 2 cases. In case one, we assume that no buffer overflow occurs during slow start phase. Now, if you assume that no buffer overflow occurs during slow start phase; that means  $W_b$  at which we are saying that a buffer overflow can occur should be greater than the slow start threshold variable. If it is greater than the slow start threshold variable, then what happens? Our window size increases to the slow start threshold and after that the TCP enters into the congestion avoidance phase.

So therefore, that slow start phase, the clean slow start phase; there is no buffer overflow. **Another** we consider another case where the buffer overflow occurs during the slow start phase. So, we consider two cases: case one - we consider that no buffer overflow occurs during slow start phase that means this case corresponds to the situation of  $W_b$  is greater than  $W_t$ , case 2 - we assume that buffer overflow occurs during slow start phase that means  $W_b$  is less than or equal to  $W_t$ .

Now, in the case 2 situation where the buffer overflow occurs during the slow start phase; there will be 2 slow start phase, we need to consider 2 slow start phase. In one slow start phase, what will happen? Our window size increases to  $W_b$  at which the buffer overflow occurs. The window drops to  $W$  equal to 1 and again enters into the slow start phase. But this time it will not go upto  $W_t$ . It will go upto  $W_t$  but this  $W_t$  will be different than the previous slow start phase  $W_t$ . It will be equal to half of the window size at which the packet loss has been detected.

Now, let us try to therefore determine throughputs of the slow start phase in both the cases. So, we consider the case one first.

(Refer Slide Time: 42:50)

Case 1.

$$W_b > W_t$$

$$W(t) = 2^{t/T}$$

$$T_1^s = \text{Duration of slow start phase.}$$

$$= T \log_2 W_t$$

$$N_1^s = \text{Number of packets, successfully transmitted during slow start phase.}$$

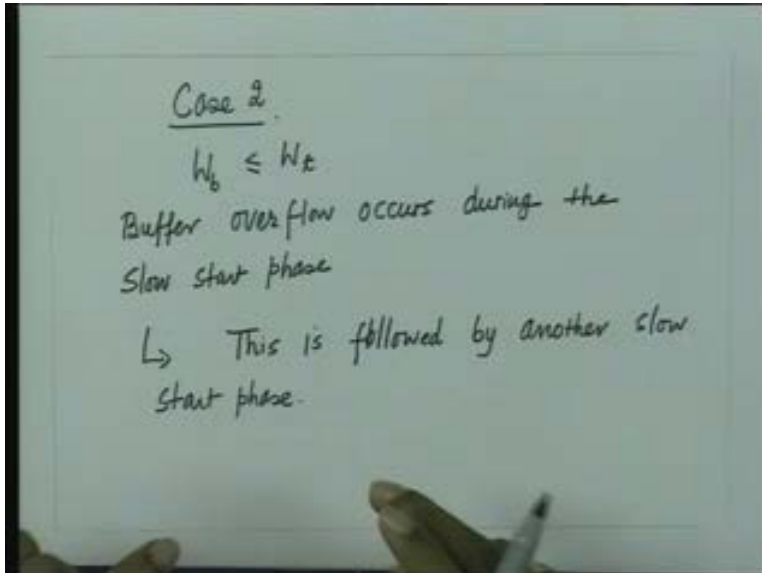
$$= W_t$$

So, case one is that  $W_b$  is greater than the slow start threshold. Now in this case, the slow start phase ends when the window size reaches  $W_t$ . Now, we will approximate the window evolution during the slow start phase to be equal to 2 raised to power  $t$  by  $T$ . Now, note that the window is increasing actually in a discrete fashion as we had seen. And, we had seen that the window evolution is actually given by  $W$  in the  $n$ 'th mini cycle  $W(nT + m \text{ by } \mu)$  by this expression. Now, we can actually have a time approximation of this and we can say that a window evolution is given by  $W_t$  by  $T$ .

Now, let us say that  $T_1^s$  is  $T_1^s$  - slow start threshold denotes the duration of this slow start phase, then this will be equal to  $T$  times  $\log_2$  of  $W_t$ . The slow start phase will end when the window size has reached  $W_t$ . Similarly, let  $N_1^s$  denote the number of packets transmitted during the slow start phase and now the number of packets transmitted during the slow start phase; if you see, the window will approximately equal to the window size.

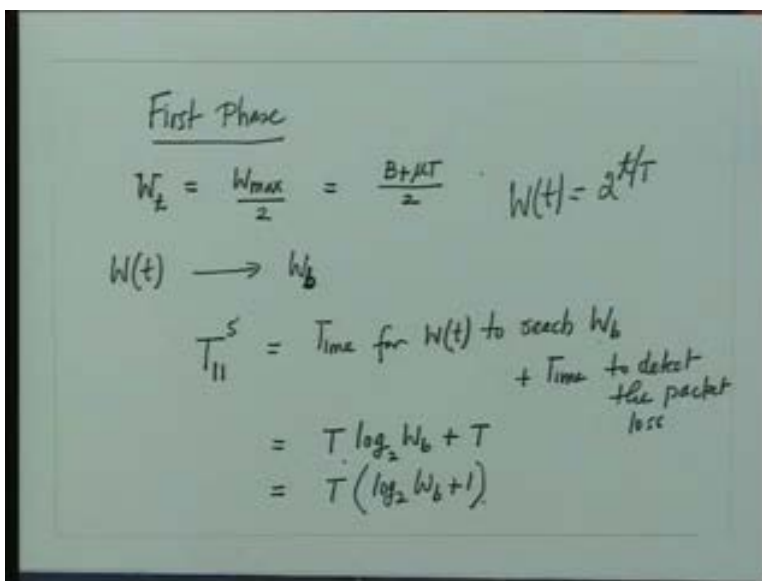
I mean if you can see the evolution of the window size here, let me just show you the evolution of the window size. That will clearly tell you that; as you can see from here that the number of packets transmitted during the slow start phase, number of packets successfully transmitted during the slow start phase is approximately equal to the window size. So, the number of packets successfully transmitted will be approximately equal to the  $W_t$ .

(Refer Slide Time: 45:42)



Now, we consider the case 2. Case 2 is when the  $W_b$  is less than or equal to  $W_t$ ; so therefore, buffer overflow occurs **buffer overflow occurs** during the slow start phase. So, the buffer overflow occurs during this slow start phase, this is followed by another slow start phase. So, the buffer overflow which occurs during the first slow start phase, this will be followed by another slow start phase. So now, let us try to analyze the throughputs in both phases. So, first phase we will consider.

(Refer Slide Time: 47:00)



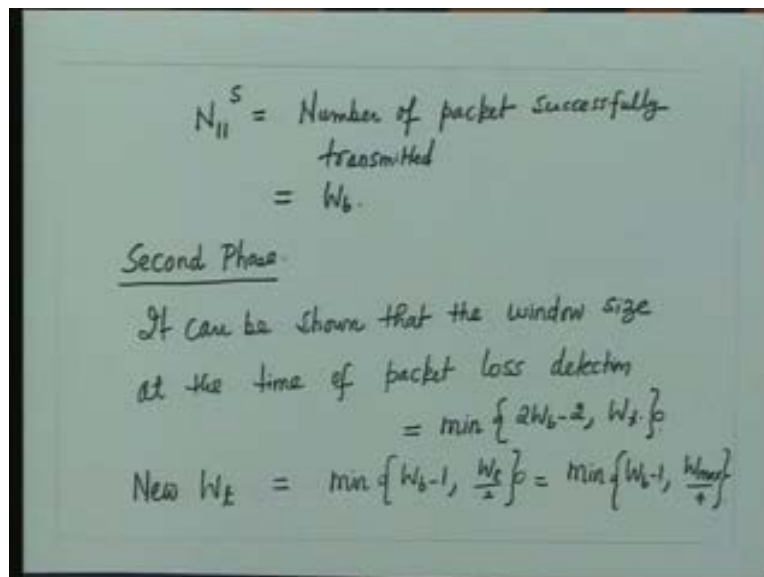
So, the first phase of the slow start phase: in this case, we can assume that the threshold which we have kept to be equal to  $W_{max}$  by 2 and that we have kept equal to  $B$  plus  $\mu T$  by  $t$ . Now,

when the  $W_t$  when the  $W_t$  when the window, it reaches the  $W_t$  that is a slow start threshold sorry when it reaches  $W_b$  that is the window size at which the buffer overflow occurs; so now let us determine the time to reach the  $T_{11}^S$ . I say that time for  $W_t$  to reach window size  $W$  at which point? The buffer overflow occurs plus the time to detect the packet loss and this is equal and this will be the duration of the slow start phase, this will be the duration of the first phase of the slow start phase. So, this is given by  $T$  of  $\log W_b$  plus  $T$ .

So, by following the same evolution that  $W_t$  is equal to 2 raised to power  $t$  by  $T$ ; the time it takes for  $W_t$  to reach  $W_b$  will be equal to  $T$  of  $T$  into  $\log$  of  $W_b$  plus it takes another round trip time to detect the packet loss. So therefore, we add another  $t$ . So, as a result this time is given by  $\log$  of  $W_b$  plus 1.

Now again, the window size has reached upto  $W_b$ , the window size is  $W_b$ ; so the total number of packets which have been successfully transmitted can again be approximated by  $W_b$ .

(Refer Slide Time: 49:57)

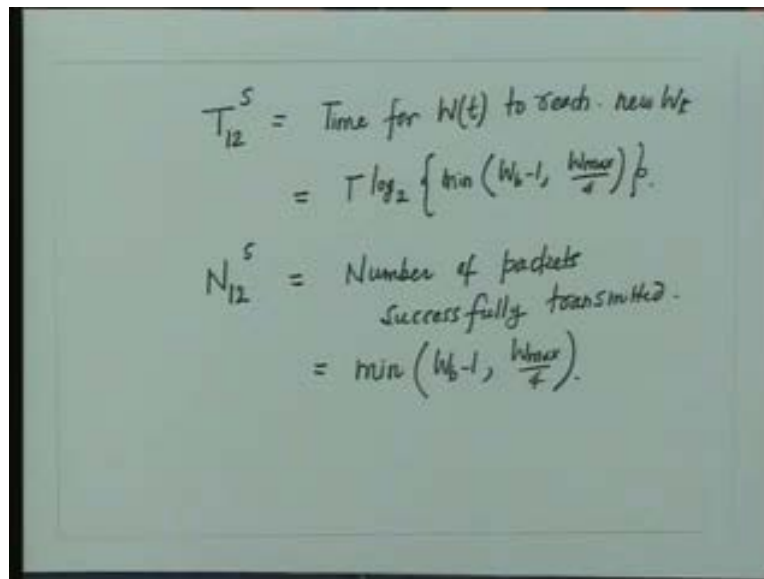


So, we will say that let us denote  $N_{11}^S$  to be equal to number of packets successfully transmitted and this is given by  $W_b$ . So, what we are saying is that we are assuming a condition where the buffer overflow occurs during the slow start phase. So, this is the first phase of the slow start phase. Now, the packet loss is detected, when the packet loss is detected that time, the window will drop to  $W$  by 1 and another slow start phase begins. So now, let us consider the second phase.

Now, in the second phase note that the slow start threshold variable  $W_t$  which was earlier said in the initial condition, which was said equal to  $W_{max}$  by 2; it will not be that, it will be the half of, it will be said equal to half of the window size at which the packet loss was detected. It can be shown that the window size at which the packet loss was detected that the window size at the time of packet loss detection is actually given by minimum of  $2W$  minus 2 into the original  $W_t$  which was there.

So, the new  $W_t$  will be said equal to minimum of  $W_b$  minus 1 into  $W_t$  by 2. And, what was  $W_t$  by 2? It was  $W_{max}$  by 2. So, that will be actually equal to minimum of  $W_b$  minus 1 into  $W_{max}$  by 4. Now, we will now determine that... so, now when the window size reaches this new  $W_t$ ; the slow start phase ends and a congestion avoidance phase begins.

(Refer Slide Time: 53:03)



$$T_{12}^S = \text{Time for } W(t) \text{ to reach new } W_t$$

$$= T \log_2 \left\{ \min \left( W_b - 1, \frac{W_{max}}{4} \right) \right\}$$

$$N_{12}^S = \text{Number of packets successfully transmitted.}$$

$$= \min \left( W_b - 1, \frac{W_{max}}{4} \right)$$

So, we will now determine how much time it takes for the source to remain in this phase and the number of packets that it has successfully transmitted; so, which in our case will be given by  $T_{12}^S$  to be equal to time for  $W_t$  to reach this new slow start threshold which is given by the  $T$  of log of the new slow start threshold which is  $W_b$  by 1  $W_{max}$  and the number of packets successfully transmitted  $N_{12}^S$  is given by the window equal to... is given by this.

So therefore, we have 2 phases: we can determine how much is the time for the slow start phase cycle and we can also then determine how much is the maximum number of packet that has been transmitted during this slow start phase and from this we can determine the throughput for the slow start phase. So, now we will go into determining the throughput for the congestion avoidance phase.



(Refer Slide Time: 54:40)

## REFERENCES

---

T. V Lakshman & U. Madhow.  
' The performance of TCP/IP for networks  
with high bandwidth delay products'  
IEEE / ACM Transaction on networking  
June 1997