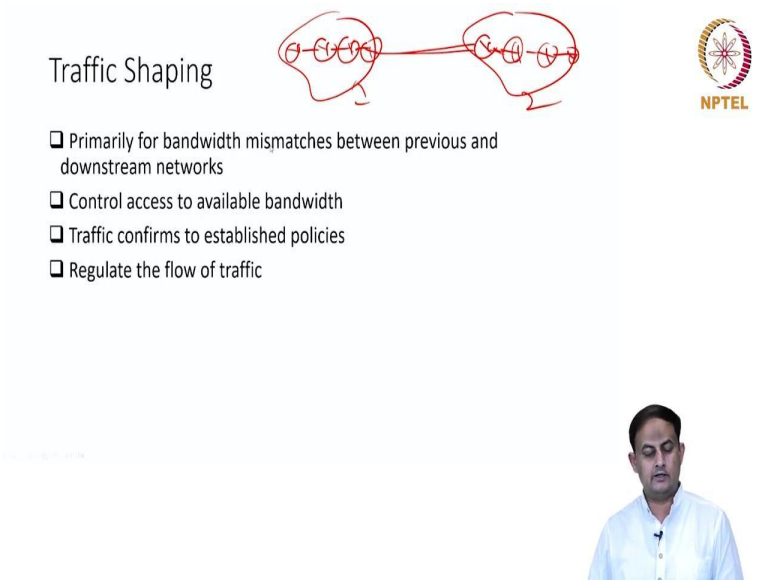
Advanced Computer Networks
Professor Neminath Hubballi
Department of Computer Science Engineering
Indian Institute of Technology Indore

Lecture 19
Traffic Management - Part 6

(Refer Slide Time: 00:23)



So, now, let us look into a little more detail how the shaping operation is done. So, why the shaping operation is required in the network? So, policing is primarily to curtail or limit the rate of transmission from a particular user or the network coming into your network, shaping is basically required for curtailing or reducing the rate of transmission which are exiting your network; you do not want your traffic to exceed the agreed upon limit or maximum ceiling, then how do we exactly do that.

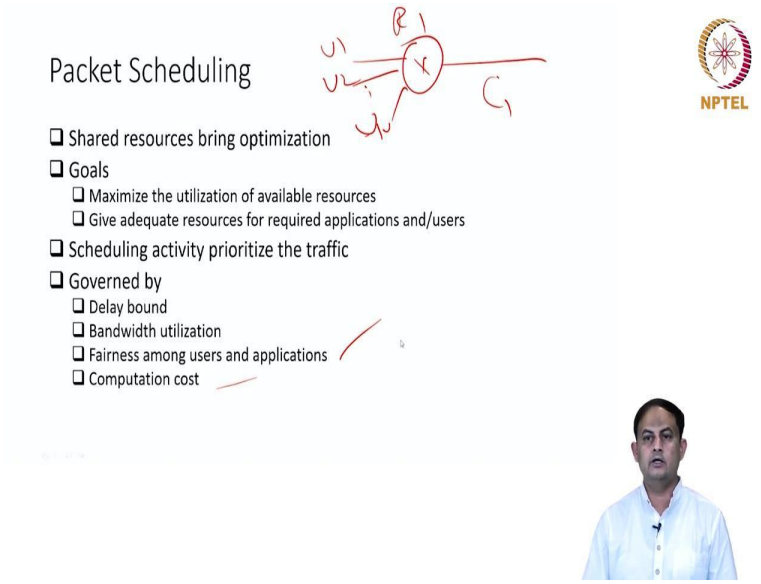
So, Network 1 has got a higher capacity and it is connected to a large number of users, but Network 2 has got a limited capacity and is not able to transmit or handle as many numbers of the packets as network 1. So, the transmission rate mismatch between the 1 and the 2, and particularly the downstream routers are not able to handle the capacity that time you want to reduce, you do not want to burden the downstream router that time this shaping is actually done.

Again, the shaping actually uses the same leaky bucket or the token bucket algorithm. So, thereby, by doing the shaping, you want to control access to the bandwidth, how much of the

bandwidth between this and this is used, and how much of the bandwidth from this and this is used.

So, by doing that shaping operation, you were actually confirming to that, and at the same time, you are ensuring that your own traffic going to the second network is not violating the agreed-upon constraint otherwise, he is anyway doing the same policing operation, but when you transmit to the other network, you do not want to violate that agreed upon rate. So, that is where the shaping is actually used. So, you want to regulate your own traffic flow to the other network entering the other network.

(Refer Slide Time: 2:42)



So, that is about the policing and shaping, both of them, we said that use the leaky bucket or the token bucket algorithms. So these are the 2 components of the traffic management operation, and the third component is something called as scheduling. So, we did discuss earlier a little bit when we discussed about traffic management in the first lecture. Here, we look into a little more detail of what is the kind of scheduling operations that are available at our disposal.

And one of the primary things why you want to do the scheduling is: the capacity available at any network is finite in nature; when I say the capacity here, I primarily mean the buffer capacity and also the bandwidth capacity. So, here is a picture let us assume that there is a router, and this router R1 is receiving traffic from the user U1, user 2, and so on to some user 10, and there is a

capacity C that is available, C corresponds to the network bandwidth, the outgoing link connected to the router R1 has got a finite capacity C, and these users U1 to U10 are transmitting parallelly, and they are sharing the same capacity link available. So if I want to give more priority to user 1 transmission, how do I do that? If I want to give the least priority to user 10 transmission, how do I do that? So that differentiation is actually brought by using something called the scheduling operation.

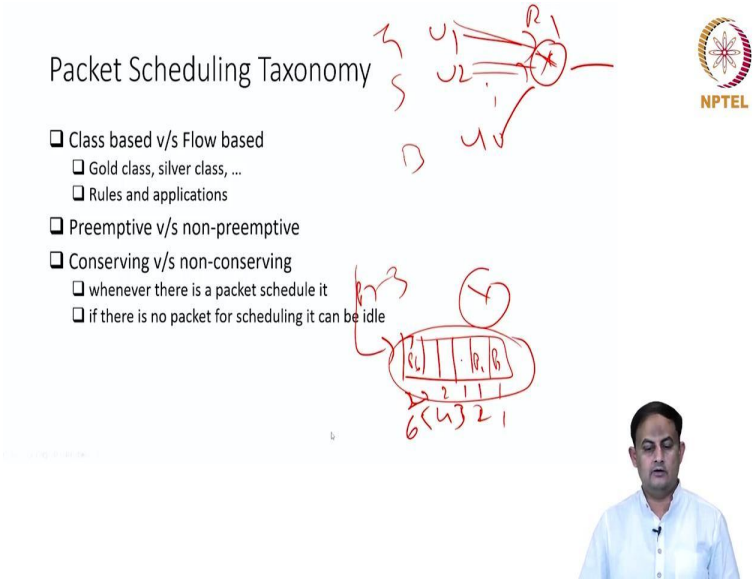
So one simple thing is by sharing the outgoing link you are utilizing, you are bringing the optimization of the available resources. So for the same capacity link, when user U1 is not transmitting, user U2 is able to use it. When both users 1 and 2 are not transmitting, the remaining 8 users are actually sharing that capacity assuming at any point of time, only one user among the 10 is transmitting; he is getting the complete share to the link. So, in nutshell, what we want to do is, whatever the resources are available, I do not want to keep that idle; we want to utilize the fullest capacity. At the same time, with all the available resources, you want to serve more number of users; the best way to do and that too even more number of users with a differentiated service or quality of service. So, one user traffic is given the priority over the other one. So that is what is implemented using the packet scheduling.

So, as I said, the goal here is to maximize the utilization of the available, at the same time, you want to give a fair share of utilization to all the transmissions or all the users; I do not want to maximize the transmission of certain user traffic at the expense of the other users i.e., without giving any share to the other user, I do not want to hog the network, one user traffic is only taking the complete bandwidth, that situation I do not want. So, any scheduler that you designed needs to ensure that these two conditions: one is maximizing utilization of the available bandwidth or the resources, and at the same time you want to be fair to all the users or all the applications or all the network from which you are actually receiving the traffic. There are a bunch of algorithms or mechanisms that can use. So, this scheduling operation is, in turn governed by how much bandwidth you want to give. So, policing, shaping, and scheduling operations work in tandem.

And so something which is not confirmed, you want to apply the traffic shaping or the policing action there you first decide, then you put it inside the buffer, and from that buffer, you actually alter the scheduling operation, and then you bring the notion of the priority. So, this operation

which requires you to alter the order in which the packets are exiting your router, requires you to do some kind of computation, and you want to bring fairness among the users and the applications.

(Refer Slide Time: 07:35)



So, there are a bunch of scheduling algorithms available, these scheduling algorithms are broadly classified into different categories one is called as class-based scheduling algorithms and flow-based scheduling algorithms. So, the difference is the following, in class-based scheduling, you assign a class or label to the end user traffic.

So, let us go back to the previous diagram, router R1, and you got user U1, U2, so on U10. And there is an exit link or outgoing link. And I mark user U1 as a Gold Class customer, user U2 Silver class customer, user U10 as something called a Bronze customer, and I am going to assign gold customers always have the highest priority, silver customers have got the less priority, bronze customers have the least priority something like that. So, I map the users into some available classes, and then each class is given a priority, and accordingly, you do the scheduling operation; that is how this scheduler works.

The other way of doing the scheduling is flow-based scheduling. So, user 1 traffic is marked as one flow, user U2 traffic is marked as the second flow, and user U10 traffic is marked with some other flow, and then you write a rule by using a combination of the source, whose users traffic it

is, which application it is. So, from the same user, if multiple applications might have a different priority assigned to them, you can bring a fine-grained notion of the priority when you do the flow-based scheduling. So, you can map it to the earlier discussion that we had with respect to packet classification.

Packet classification is actually done with a set of rules, and using those rules, you are actually doing the differentiation and exactly the same thing is happening here, the scheduling operation, scheduling is one way to implement what the classifier is actually telling you. So this is a flow, you want to take this action, then how do I do that, I tell my scheduler to do work accordingly, I want to transmit this faster, and quicker, I want to delay this, I want to drop this, accordingly your scheduler can work, what to drop, and what to transmit quickly. And the rules dictate what to transmit or what to be given priority.

And the second class of the algorithm or the mechanisms or second classification is something with the primitive and the non-primitive scheduling mechanism. So, if you can recollect our earlier discussion, a router has got a queue and the packets are arriving and are put inside this queue, let us say P1, P2, so on P6 are now inside this queue. And let us assume, for the sake of discussion that this queue capacity can accommodate only 6 packets, and 6 packets are already full. And a new packet P7 arrives at the input queue on the line card, and you do not have the capacity to place that, P7.

Now, it depends on the mechanism of how your scheduler is working. If you have assigned the notion of the priority, let us say P1, P2 P3, P4, P5, and P6 are having priority 1, 1, 1, 2, 2, and 2 something like this, and P7 has got the highest priority, let us say 3. Now, a higher-priority queue has arrived at the input port. And you have already marked in what order the packets already queued up or buffered inside your router are going to exit the router; the order of the scheduling is decided maybe I am going to transmit this, this packet in this order 1, 2, 3, 4, 5 and 6, this order is decided and if you are giving priority to P7, now what you do?

So, all these packets have not exited the router they are queued up, but you have decided in what order you are going to send them. So if your scheduler is preemptive, what it means is I am going to halt the already scheduled set of packets if the arrival of a new packet with the highest priority will take precedence. I am going to halt the transmission of scheduled packets'

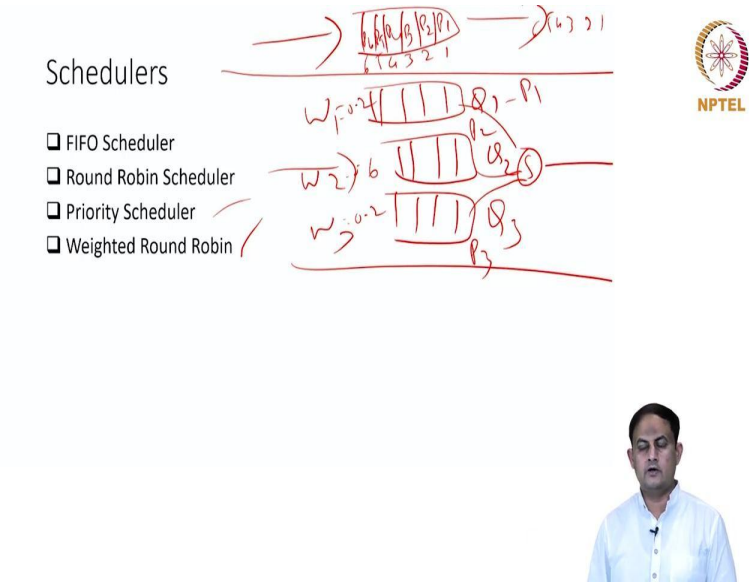
transmission, and you transmit the highest priority packet now. This is called as preemptive scheduling.

And in the non-primitive scheduling: although this might be the highest priority queue, already I have made up my mind in what order I want to transmit it, and I am not going to disturb that. But if I have the available capacity, I will put it inside the queue now, starting from that point onwards I am going to decide in what order the subsequent packets are going to be transmitted that is called as non-preemptive both versions you can make use of.

So, the next classification is something called a conservative or non-conservative. In conservative technology, you decide to transmit it whenever there is a packet available if there are no packets available, then you may not be able to schedule the transmission.

In the non-conservative mechanism, you can decide, even when the packets are available, then also you may not be able to transmit them, that is the difference. So, if there is a capacity transmitted at a certain point of time, you might also decide to keep the link idle. So, that is the three different categorizations of the scheduling algorithms or the mechanism that you want you can use inside your router.

(Refer Slide Time: 14:09)



So now after having understood the class of the schedulers available, now, the question that I want to ask is how do I implement these schedulers, what type of schedulers are actually

available? So, there are many types of schedulers available for packet scheduling. So, the simplest form of the schedule is something called the FIFO scheduler. So, this we discussed earlier, the router has got a queue, and the packets are arriving at the input port, and the packets are exiting at the output port, and in between, you have got this queue.

And what you do is pick up the packets from the input port and put them inside this queue. So let us say P1 is the first packet arriving, P2 is the next one, P3 is the next one, P4 is the next one, P5 and P6, in this order, the packets have arrived at the input queue and the scheduler what it can do is it can schedule the first packet, this is the first one to exit, this is the next one to exit, this is the next one to exit, this is the next one and so forth. The order in which the packets have arrived in the same order the packets are exiting the network that is called as the FIFO scheduler this is the default scheduler that is available inside any router.

So, there is nothing fancy happening here, if there is a capacity inside your buffer or in the queue, you put them inside the queue; otherwise, you discard them, and you make them exit in the order in which they have arrived there is no alteration that is happening here. This is the simplest form of the scheduler that is available.

So, if we can recollect earlier discussions, so, many of the routers may have multiple queues maybe I might have 3 queues let me call those queues as Q1, Q2, Q3, this is Q number 2, this is Q number 3, and there is an output line. The scheduler which is sitting here is picking packets from this queue, it is also picking packets from this queue, and this is picking packets from this queue, and the incoming packets are put inside this Q1, Q2, Q3. And what I can do is I can tell my scheduler yes to pick up packets, one packet from Q1, one packet from Q2, and one packet from Q3, and you go in order, you recycle.

So, every time you pick up one packet from Q1, next pick up one packet from Q2, and so forth. So, it is not necessary that you have only three queues, but multiple of them might be available, n queues. So, you sequentially go one after the other, and then you recycle. So, this is called the round-robin scheduler. In a way, the round-robin scheduler is again not doing anything fancy, it is just your packets are segregated and put into different queues, and then it is picking an equal amount of the packets from each of the queues.

So, now only differentiation that you can bring here is, let us say a particular user's packet or particular application's packets are put inside queue number 1, if the rate of arrival of that packet from that application or the user is different, then the number of the packets that are available in the Q1, Q2, Q3 might vary. So different applications are transmitting at a different rate, and assuming for each application, I have a queue, then the number of packets inside these queues might vary. So at any point of time, if certain queues are not full, they do not have the packet to transmit. Let us say, for the sake of discussion, Q2 does not have any packets put inside its queue. Now, the packets from the Q1 and Q3 are picked up, and subsequently, some packets arrive at Q2 now, I go in that order Q1, Q2, Q3, but any number of the queues which are empty, those packets are not picked up from that. Only whenever packets are available in some queues you pick them up from there and schedule them for transmission. So that is how the round-robin scheduler is working.

So, now, the next variation that you can bring is, so here, round-robin scheduler is actually giving equal opportunity to all the queues. So, you are not really able to differentiate, when the transmission rate of all the applications is equally good, or it is just sufficient to put the queues full, then the round-robin scheduler is not exactly doing anything fancy, it is as good as the first in first out scheduler with a slight difference. So, how do I bring the differentiation? so that is where the priority scheduler come.

So, what it does is it assigns a priority to each of these queues Q1, Q2, and Q3. I might tell my scheduler that Q1 has got to highest priority, the next highest priority is Q2, and the next one is Q3. So the scheduler would pick up the packets from Q1, the highest priority queue always; if Q1 is empty, then you go and pick up the packets from Q2, and if Q1 and Q2 both are empty then you pick up the packets from Q3. So that is called the scheduler.

So you might assign different priorities to different queues and then you always look for the higher priority packets from the higher priority queue and then schedule them for transmission. So, what we understood earlier is if use this mechanism, then if there is a constant arrival of the packets of the highest priority queue, then others will not get a chance to transmit they will be independently starving there, so, that is not desirable.

So, remember, the objective of the scheduler is to be fair to others. So, if I do this operation, if I tell my scheduler to always transmit the highest priority packet, then you will not be fair, and the packet might be independently waiting inside the queue. So, how do I exactly avoid that? So, that is where the weighted round robin comes into the picture, where you assign a weight to each of these queues.

Now I do not have the priority, but I have the weight assigned to the queues, W1 is the weight assigned to Q1, W2 is the weight assigned to Q2, and W3 is the weight assigned to Q3. So, for the sake of assumption, let us say the numbers are in fractions, W1 has got 0.2, W2 has got 0.6, and W3 has got 0.2 as a weight. And what it means is if Q1, Q2, and Q3 has got ten packers to transmit, every 2 packets exiting are selected from Q1, there will be six packets selected from the Q2, and another two packets are selected from Q3, together you will be able to transmit ten of them.

So, like that, I can define the notion of the weights, and then whatever the weight is there accordingly, I pick up the packets from different queues and then transmit. That is the weighted round-robin. So I go in a round-robin fashion, but at the same time, I alter the number of the packets picked up from each of the queues, which is dictated by the weight assigned to that queue. And then you transmute it.

Now you can think of a simple utilization of this one is maybe the highest priority one which is a 0.6, is serving the user which has got the highest priority, or the application which has got the highest priority, and the remaining the Q1 and Q3 have got the other priority. The next two applications got the same priority, and they are served from this Q1 and Q3 that is how you utilize the scheduler to bring the differentiated service.