Broadband Networks

Prof. Karandikar

Electrical Engineering Department

Indian Institute of Technology, Bombay

Lecture - 13

Rate Proportional Servers

We have been discussing packet scheduling algorithms and till now our objective has been to design packet scheduling algorithms that can provide fairness to various stations in a network node or a router. Now, towards that end we had seen that if you want to provide fairness in the sense of max- min fairness, then the fair scheduling algorithm is a generalized processor shearing algorithm or the fluid flow fair queuing algorithms.

Now, this algorithm assumes that the traffic is fluid in nature and the algorithm works by serving each non empty queue by an infinitesimal amount of fluid. So, if you want to consider this in terms of the traffic being sent in terms of bits, then this algorithm will operate by serving in a round Robin fashion in a bit by bit manner. So, each non empty queue use 1 bit and then the second from the second queue you serve another bit and so on. So, this way you will serve in a round Robin fashion in a bit by bit or the traffic if fluid in nature, then you serve a very small infinitesimal amount of fluid from one queue, then you serve from another queue and so on.

So, the normalized service as a result, the normalized service received by 2 sessions which are backlogged at every instant of time is equal. And, how do you define the normalized service? The amount of fluid which has been served during the backlogged interval that is normalized with respect to the rate that has been allocated to a particular session. So, this we call as the fluid flow fair queuing where the traffic which is fluid in nature is served by the server in such a manner that the normalized service received by 2 sessions which are backlogged that is equal and it increases in an equal fashion.

However, we have also seen that the traffic in practice is not fluid in nature. The minimum entity that we can transmit on the link is a packet and therefore we have to device packet based scheduling algorithms. So, in the previous lectures we were trying to see what kind of packet based scheduling algorithms we can device? So, we said well, let us have a very crude approximation of the generalized processor sharing algorithms or fluid flow faring, fluid flow fair queuing algorithms and which could be a round Robin round Robin packet scheduling algorithms where you serve packet from one non empty queue, then another packet from another non empty queue and so on.

If the different sessions have different rates allocated to them, then you can serve them in proportion to these weights or in proportion their rates. However, disadvantage is that round

Robin packet scheduling algorithms can be unfair over a shorter time scales and more over if the packet lengths are variable in nature; then in order to be fair, we need to know the mean packet length in advance so that we can normalized these weights with respect to the mean packet length and then serve the packets.

So, we have an improved version of the round Robin algorithms which was called as deficit round Robin algorithm and the deficit round Robin algorithms maintains the credit for each session and then serves the packet. The advantage of the deficit round Robin algorithm is that it does not require the knowledge of the mean packet length in advance. However, it is still suffers from the same disadvantage which the weighted round Robin suffers and which is that the algorithm can be unfair over a shorter time scales.

So now, then the question is that what kind of packet based scheduling algorithms which we can device which approximates the fluid flow versions of the fair queuing algorithm, which comes closer in terms of fairness to the fluid flow fair queuing algorithms? So, we then said that we can, what we can do really is that we can emulate the fluid flow fair queuing algorithms and determine the order in which the packets are leaving the system, had they been served with the fluid flow fair queuing algorithms

Now, if in the actual packet scheduling algorithms, if we serve the packets in the increasing order of their departure times in the fluid flow fair queuing algorithms; we are likely to be fair. If the question however is that in order to know the departure exact departure times of the packets in the fluid flow fair queuing algorithms is not possible. So therefore, we define a quantity something some something called as a virtual departure time or virtual finish times.

So, what we do is that we define a quantity which is called as the virtual finish time and this virtual finish time is the value of the virtual time when the packet will leave the system in the fluid flow fair queuing and then we serve the packets in the increasing order of the virtual finish times. Now, it turns out that computing the virtual finish times is conceptually quite simple. It leads to a very recursive way of computing the virtual finish times if we know the virtual finish times of the previous packets in your queue.

So therefore, conceptually we can compute the virtual finish times in very easy manner. But it turns out that in practice, its real time implementations may be very difficult because it requires computation of the virtual time and what is virtual time? Virtual time is actually keeping track of the progress of the work done in the system and the virtual time is actually a monotonic functions and it changes its slope whenever the set of backlog sessions changes.

And since, in the real practice the number of such sessions which can change their state from backlog to non backlogged or vice versa can be order, can be of the order of the total number of sessions and since the total number of sessions in a typical node or a router can be quite large; so therefore the computational complexity associated with a real time implementation of the virtual time can be of the order of the total number of sessions. And, as a result the real time implementation of the weighted fair queuing becomes very difficult.

Then we say that well since the virtual time is actually indication of the progress of the work done in a system and since the tag of the packet which is currently in service is an indication of the virtual finish times, then why not we approximate the virtual time from the finish tag of the packet which is currently in service.

So, this way we try to… Earlier you know what we have done? We are saying that we will define the finish tag from the virtual time of the systems. Now, we are saying that if suppose a packet is already tagged, then we can estimate the virtual time from the tag of the packet which is currently in service and this way we can also recursively compute the estimate, rather we can recursively estimate the virtual time of the system. This is of course an approximation and since computation of the virtual time is determined from the finish tag of the packet currently in service and the finish tags of the packets will be computed from the virtual time; so the algorithm is therefore called as a self clocked fair queuing algorithm.

Now, when we define a self clocked fair queuing algorithm; obviously, this algorithm does not require emulation of the fluid flow fair queuing algorithms and as a result therefore, it is computationally quite efficient. But then the question really is that can we or how can we compare the fairness of the self clocked fair queuing algorithms? How can we say that this algorithm is fair? Then we argued that we can define something called as the relative fairness and the relative fairness means the difference between the normalized services received by 2 backlogged sessions in the packet based fair queuing algorithm that is in the self clocked fair queuing algorithm.

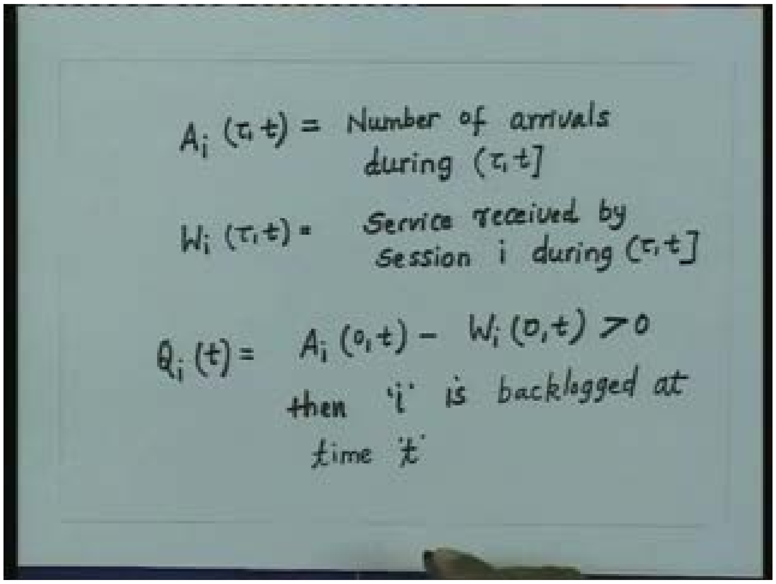
If this difference between the normalized service received by the 2 sessions is bounded, then obviously the algorithm is fair and if this bound if this bound is minimum if this bound is minimum, then obviously we are having a we are having a packet scheduling algorithms which is relatively fair. In the sense that this algorithm is guaranteeing that any 2 sessions which are backlogged; the difference between the normalized services is not very large, it is very low.

Then we can argue that such a packet based fair queuing algorithm is good and it turns out that for any packet based fair queuing algorithm, the relative fairness that is the difference between the normalized sessions received by 2 backlogged sessions can be derived and from that perspective, it turns out that self clocked fair queuing algorithm comes very close to an optimal packet based fair queuing algorithms. So, this is what we have discussed in the previous lectures.

Now, we will take this discussion forward by then arguing out that how can we generalize now all these frameworks. Now, till now we had seen particularly in the context of the self clocked fair queuing algorithms that we had defined the notion of something called as the virtual time of the session. So, now we generalized this concept and we say that let us define it to be the potential of a session i. So, we now define the potential of the session i.

So, however, before we define the potential of the session i, let me just summarize some of the definitions that we have been already using in the discussion of the packet scheduling algorithms.

(Refer Slide Time: 13:18)



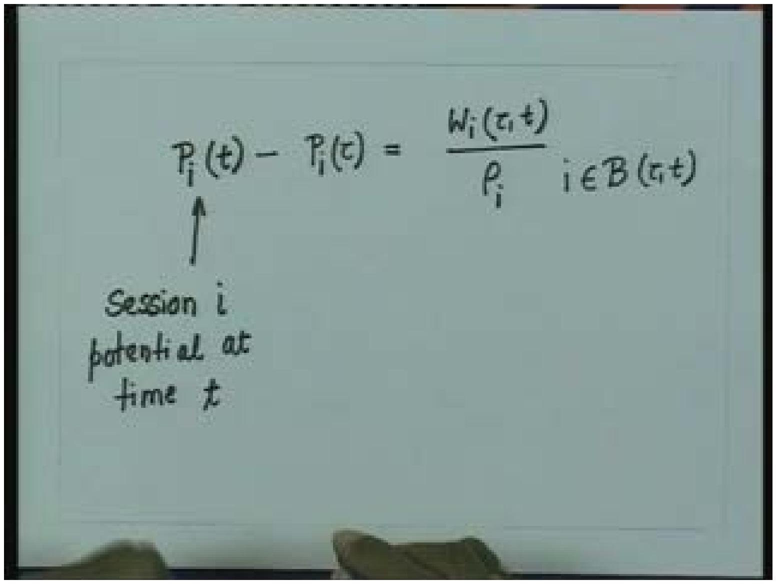
So, we define as A_i tau t to be the number of arrivals, the number of the arrivals during the intervals of tau to t. Now, in the packet based as we had discussed last time also that in the packet based scheduling algorithms, we will assume that when the last bit of the packet has arrived, then only this quantity A_i increases. So, this is like a stair case functions.

Similarly, we define as W_i (tau, t) to be the total number of bits served. So, this is like the service received by the session i service received by session i during tau t. Now, this is also a staircase function. We assume that this W_i that is a service received by the session i increases only when the last bit of particular packet has been served.

So, now we say of course that a session is backlogged Q_i (t) which we are defining it to be the number of arrivals from 0 to t minus the number of departures from 0 to t. So, if this Q_i (t) is greater than 0, then session i is backlogged at time is backlogged at time t. So, this is what we already defined, the definition in the previous discussions also. I am just recapitulating these definitions.

Now, we define something called as the potential of a session i and we say that the potential of this session i which is backlogged increases in proportion to the normalized service it receives during its backlogged sessions. So, during the backlogged sessions, during the backlogged period the session i, so let us say that P_i (t) - this denotes the session i potential at time t.

(Refer Slide Time: 16:16)



So, this denotes the session has the potential time t and P $_i$ (t) minus P $_i$ tau will be equal to W $_i$ (tau, t) upon rho $_i$. So, what we are saying? The potential of the session i during the interval tau to t increases as the normalized service received by this session during this interval. So, the session i is backlogged during the interval tau to t.

Now, all the algorithm let us say in the GPS; what we are doing in GPS? In GPS, the potentials of all the sessions which are backlogged increases exactly by the same amount. So, the potentials of all the sessions which are backlogged in GPS is same. The question really is that when the session which was ideal for a long, when this ideal session becomes backlogged for the first time, what should be its potential?

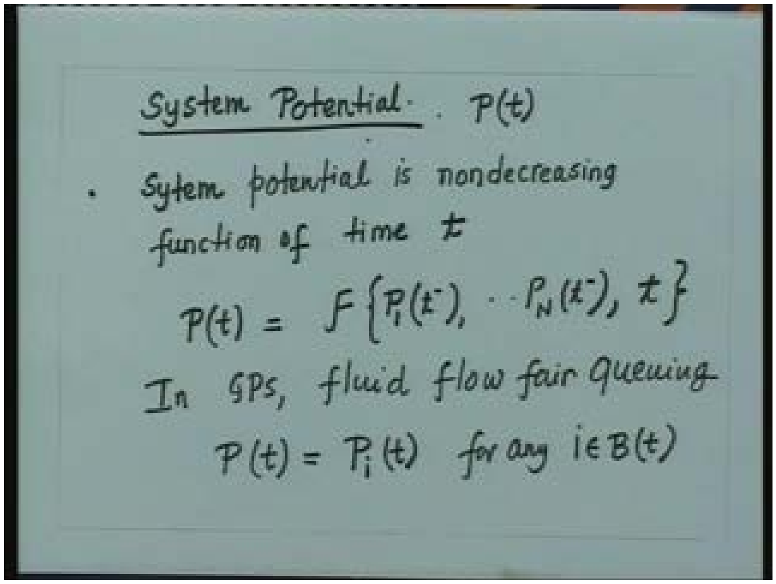
The question really is that when this session is backlogged, we know or we have just defined that the potential of this session is increasing in proportion to the normalized service that it is receiving. But we have not yet defined how will the potential of this session change when the session is ideal. It will not decrease; we have been already saying that the potential of the session will be a non decreasing function of time. So, it will not decrease when the session is ideal. But otherwise, how this is changing when the session is ideal.

So, the scheduling algorithms, various scheduling algorithms will differ in the way in which they update the potentials of the sessions when they are ideal. Now, towards that end, we therefore define some quantity which we call as the system potential. This is the potential of the session with something called as system potential.

Now, system potential is very similar to the systems virtual time that we have already defined. Now, this virtual time keeps track of the work which is already done by the system. So, this is the parameter which is related to the system. This is the system's property, the entire system's property. The potential of the session or the virtual time of the session is a specific property of that session.

So, now we define some quantity which we call as the system's potential – P (t). So, we define some quantity which we call as the systems potential.

(Refer Slide Time: 20:06)



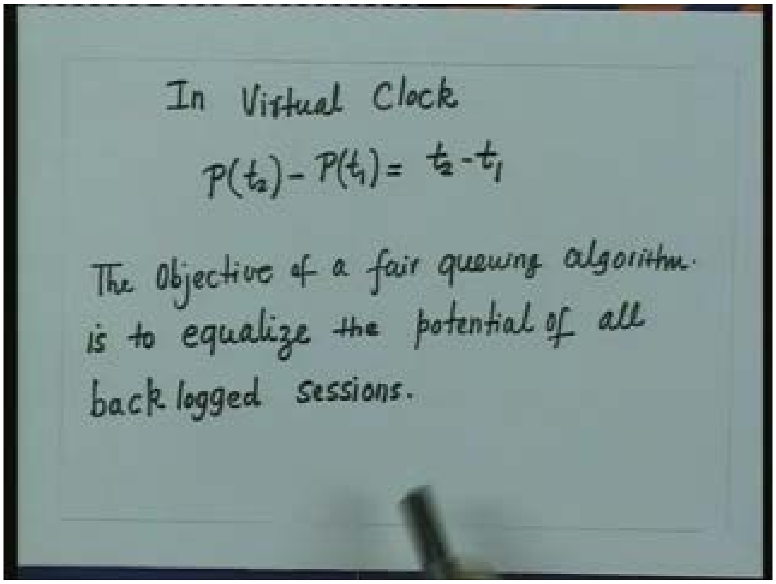
Now, we say that system's potential is a non decreasing function of time t. So, we define system potential P (t) is a non decreasing function of time t. So, P (t) is defined to be the function of P_1 t minus so on upto P_n t minus and the time t. So, this is a non decreasing function of the time at the same time it is a function of the potentials of the different sessions just before time t and also the real time t.

Now, if you look at in GPS or in GPS or in fluid flow fair queuing, the system potential which in the fluid flow fair queuing we are calling it to be the virtual time v (t) of the system; the system potential P (t) happens to be equal to P_i (t) all the time for any session i which is backlogged during the interval.

So, what we are saying? In the fluid flow fair queuing, the system potential or as it is called in the fluid fair flow queuing - system virtual time; the system potential or the system virtual time increases in the same way as the potential of any sessions. So, the system virtual time is always equal to the session's virtual time if that session is backlogged. So, it is always equal. But it is not necessary.

In fact, we can see that in the packet based scheduling algorithms, we can define this system potential to keep track of the work done in the actually packet based scheduling algorithms, remember, we were defining the weighted fair queuing etcetera scheduling algorithm by doing emulation of the fluid versions of the algorithms. So, here we are actually now generalizing the conception trying to see how different scheduling algorithms can be derived by defining this generalized quantity which we are calling it to be the system potential. It is not necessary that system potential must always be equal to the potential of the backlogged sessions. In GPS, it is.

(Refer Slide Time: 23:46)

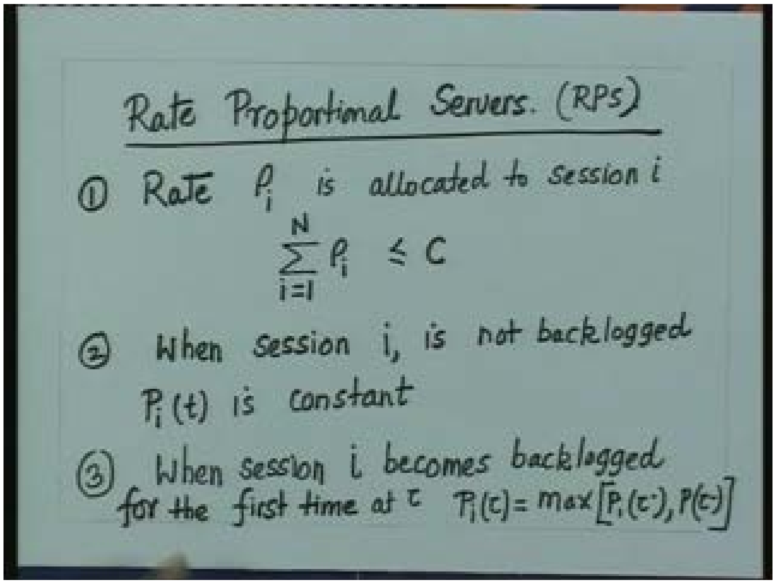


For example, in the virtual clock, this was a earlier scheduling algorithms, this was defined that in virtual clock, the potential of the system actually increases in proportion to the real time itself. Now, what we are saying is that in a pack in a scheduling algorithm, session's virtual time is increasing in proportion to the normalized service it receives when the session is backlogged. When the session becomes backlogged for the first time, its potential is updated depending upon what is the system potential.

The sessions, all the sessions whose potentials are minimum; those sessions only receive the service and objective of the scheduling algorithm should be to equalize the potentials of all sessions which are backlogged. So, I should write that the objective of a fair queuing algorithm should be to equalize the potential of all backlogged sessions. So, this is how we define a fair queuing algorithm.

Now, with this we define, we provide a general framework of the fair queuing algorithms which we call as the rate proportional servers or rate proportional fair queuing algorithms. So, let us see what are the rate proportional fair queuing algorithm is?

(Refer Slide Time: 26:10)

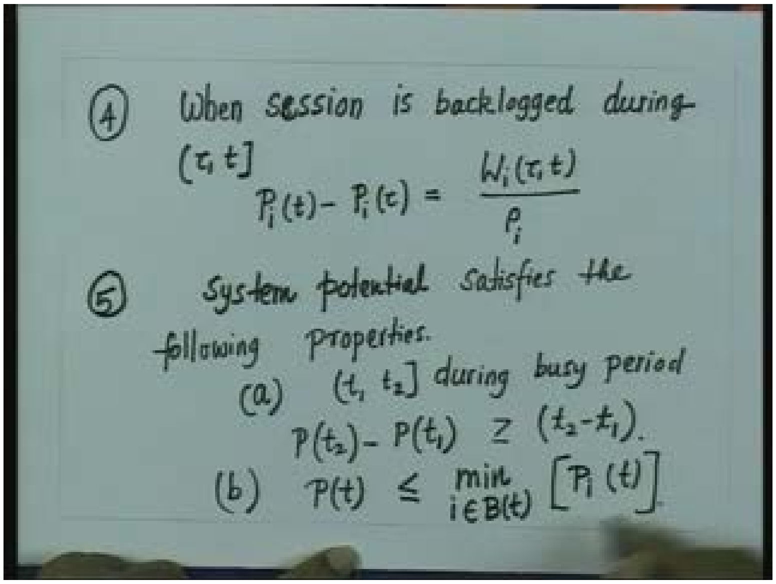


So, we define to be as something called as rate proportional servers or RPS. Now, in this for rate rho $_i$ is allocated let us say to session i. So, for i is equal to 1 to n rho i should be less than or equal to c where c is the total link rate of the systems where c is service rate of the system. Now, we define, of course we have defined the system potential which is P $_i$ (t) and we have also defined the system potential which is rho i(t).

So, when the session is backlogged, the … is constant. So, if it is not backlogged, the P $_i$ (t) is not… when backlogged, when session becomes backlogged, when session i is not backlogged; the session potential i is constant and when the session is not backlogged, when the session becomes backlogged for the first time, when session i becomes backlogged for the first time, then let us say a tau, then the session is updated as P $_i$ tau to be equal to max of P $_i$ tau minus and P $_i$ tau minus here.

So, when the session becomes backlogged at time tau; then when the session is not backlogged, then its session potential is constant and when the session becomes backlogged for the time for the first time let us say tau, then its session is updated to the system potential or maximum of its own potential just before the time. That is what we were looking at it. So, when we say that when session i becomes backlogged for first time at tau, then its potential is updated to the maximum of the system potential or its own potential whichever is greater.

(Refer Slide Time: 29:55)



However, when the session is backlogged, when during the time tau to t, so when session i is backlogged during the interval of tau to t; then its potential is updated as we are already saying to the normalized service it receives. Now, what we have not yet seen is how is the system potential is changing. So, we have seen that system potential is the function of the session potential in the real time. But how exactly it is changing?

So, we say that the system potential again is a non decreasing function of time. So, it is again a non decreasing function of time system potential and we said that the system potential will satisfy the following properties. So, the system potential satisfies the following properties. What are those properties? One is during the interval of t_1 to t_2, during the busy period, system busy period. P (t_2) minus P (t_1) is greater than or equal to t_2 minus t_1 and b…

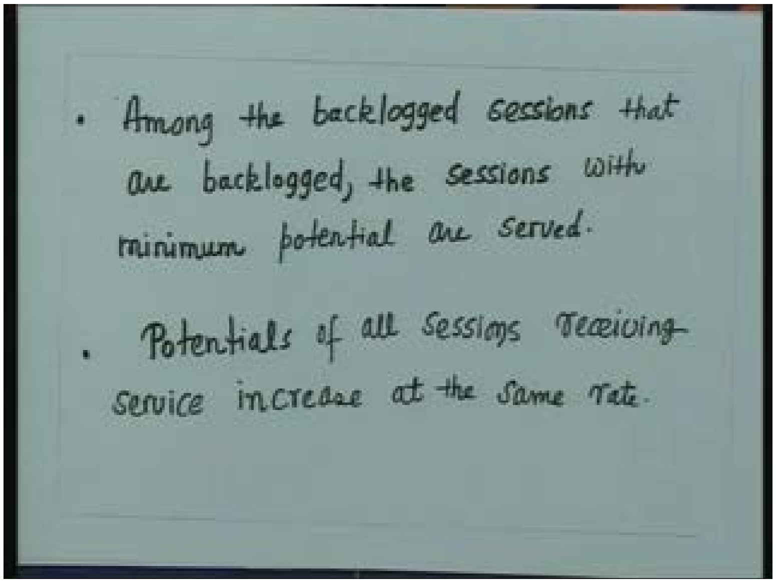
So, what we are saying is that the system potential during an interval and this interval happens to be a sub interval of the system busy period. System potential is greater than or equal to the increase in the real time. Other thing is that the system potential is always less than or equal to the minimum session potential of the backlogged session.

So, what we are saying is that the system potential is always maintained below the session potential, it is always maintained below, it is less than or equal to; it could be equal to the system potential which is backlogged and this is what we had seen that this is equal to the potential of the sessions which are backlogged in the fluid flow version of the fair queuing algorithms or in the GPS - generalized processor sharing algorithm. So, it is always equal to the potential of the sessions that is backlogged.

But in general, it can be less than or equal to the potential of the session that is backlogged. So, one thing is that the system potential is a non decreasing function of time, the system potential will always be greater than or equal to the difference between the real times and the third thing is

that the system potential will always be less than or equal to the minimum session potential of that session of those sessions which are backlogged.
Remember, there are sessions which are backlogged, the session potential is increasing in proportion to the normalized service received by the backlogged sessions and when the session is not backlogged, when it is ideal; then its own session potential is remaining constant and when it becomes backlogged, it is updated to the system potential, it is made equal to the system potentials.

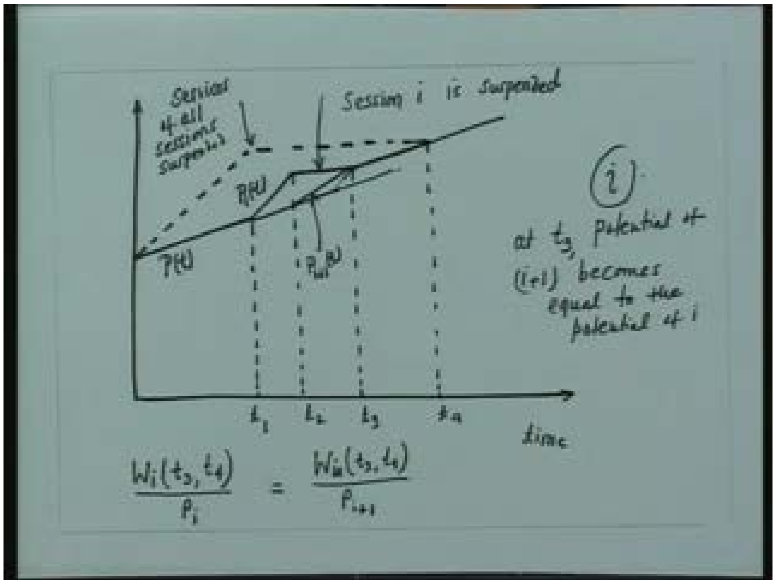
(Refer Slide Time: 34:22)



Now, how sessions are serviced? Sessions are serviced among all those backlogged sessions; so we say that among all the sessions which are backlogged among all the sessions which are backlogged, only those sessions whose potentials are minimum, they will be served. So, the sessions with the minimum potential are served. And of course, as you know that the potentials of all the sessions which are receiving service that will increase at the same time; so potentials of all sessions receiving service will increase at the same rate. So, this is what we define as the rate proportional server algorithm.

So, let me just illustrate in practice how it will happen.

(Refer Slide Time: 36:06)



Let us say in this graph, I define suppose, this is the potential of the sessions which are increasing; so this is I am denoting it to be the time and this is the potential of the sessions, how it is changing and t_3 and this time is t_4 and so now let us look at this diagram.

This is the So now, these are the potentials of the sessions which are backlogged. So, this is increasing and this is the system potential P (t). As you can see, at time t_1, some session i becomes backlogged. So, now it's potential P_i (t) actually increases and at this point the another session actually becomes backlogged. So, I should show it to be here; this is the another sessions which becomes backlogged and therefore this session may be equal to another session i plus 1. So, its potential is now increasing. But the session i is temporarily suspended.

So, at this point you can say that session i is temporarily suspended its service. At this point, remember at t_1, all the sessions have been suspended. So, at this point, session i is temporarily suspended and i plus 1 is increasing and at the point t_3, the potential of session i plus 1 becomes equal to the potential of session i and from this time onwards upto t_4, the potentials of both sessions i and i plus 1 increase in the same rates and at point t_4, the potential of all the sessions becomes equal and then it increases in the same fashion.

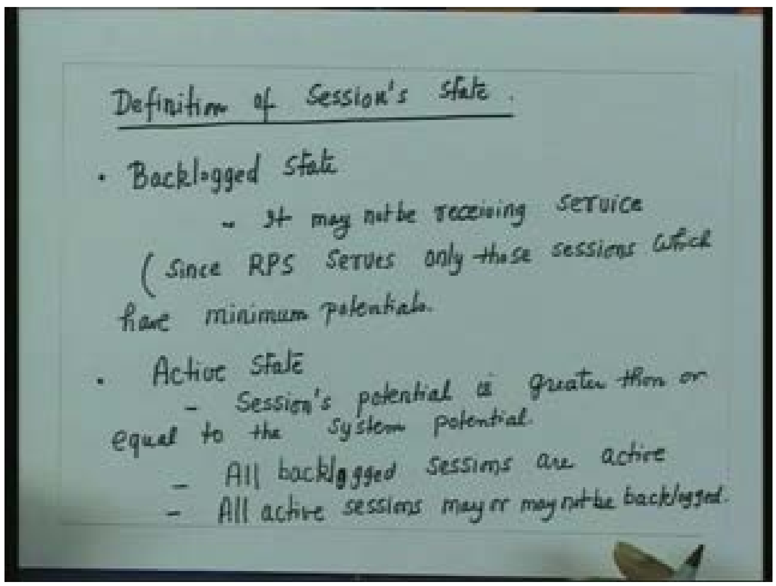
The potential, the system potentials may remain less than the potentials of the sessions which are which are backlogged. So, what we are saying is that in this diagram if you see, then during t_3 to t_4, the W_i (t_3, t_4) that is the normalized service received by the session i is equal to the normalized service received by this session i plus 1 (t_3, t_4) given by row$_i$ plus 1. So, during this interval t_3 to t_4, the session i and i plus 1's potentials are increasing with the same rate.

So, let me just explain again. This is the system potential P (t), this is below the potentials of all other sessions which are backlogged and have been serviced. So, this potential is of all other sessions which are backlogged and increasing. Now, at time t_1, some session i let us say that becomes backlogged. So, when that becomes backlogged, services of all other sessions are suspended. So, at this point, services of all sessions are suspended and only session i receives exclusive service. So therefore, its potential increases.

At time t_2, another session i plus 1 becomes backlogged. So therefore, session i service is suspended and i plus 1 potential increases. At point t_3, the potential of so, at t_3, potential of i plus 1 becomes equal to the potential of i at t_3 and after that the potentials of both i and i plus 1is increasing and at point t_4, the potential of all the backlogged sessions becomes equal and after that they increase in the same manner.

So, what we have tried to see is that the objective of the fair queuing algorithm is always to equalize the potentials of the sessions that are backlogged. That is how the rate proportional scheduling algorithm is trying to do. It is trying to equalize the potentials of all the sessions that are backlogged.

(Refer Slide Time: 42:07)



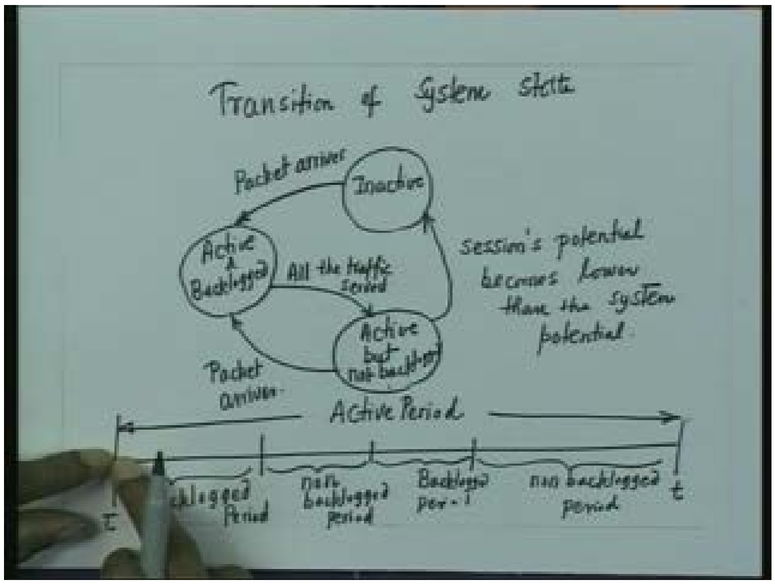
Now, we define some session's states. So, let us have some definitions of session's state. So, backlogged - we have already defined. The session will be in the backlogged states if it has some amount of the traffic in its buffers. It is not necessary that it may not be necessary that session may be receiving service. So, session may not be receiving service in general. Why? because, the rate proportional server serves only those sessions which have minimum potential. So, it might be quite possible that a session may be backlogged. But its potential may not be the minimum and as a result, it may not be receiving any service.

So, we define then active state: session could be in the active state when the session's potential when the session's potential is greater than or equal to the system potential. That means the session potential is not below, the system potential could be in the active state. Now, note that all the sessions which are backlogged all the sessions which are backlogged, they are always active. So, all backlogged sessions will be considered as active, all backlogged sessions are always active. But all active sessions may or may not be backlogged, it is not necessary that all active sessions have to be backlogged.

So, I will say that all active sessions, so all active sessions may or may not backlogged; but all backlogged sessions will always be considered as active and then we have a non backlogged state and in the non backlogged state, the queue is empty which is ideal state.

So, let us see how the transition of these backlogged to active state and active to backlog state will take place.

(Refer Slide Time: 45:58)



So, let me just draw the transitions of the various transitions of system state. So, we draw the transition of the system states as let us say that you are in an inactive state and a packet arrives; so, if you are in inactive state and a packet arrives, your potential will be updated to the system potential and therefore you will become both active and backlogged.

Now, at this point, if this potential is the minimum potential; then it may start receiving service and if it starts receiving service and if all the traffic has been served; now it may start receiving service and if it start receiving service, its potential will start increasing and system potential will now become lower than the session potential. So, its potential will start increasing but the system potential will be lower.

See remember, when the session was inactive for all the time, when the session was inactive and when it became and the packet arrives, it became backlogged for the first time. When it became backlogged for the first time, its potential is updated to the system potential. Now, this session when it starts receiving service, then its potential will now keep on increasing in proportion to the normalized service it receives and the system potential will act behind.

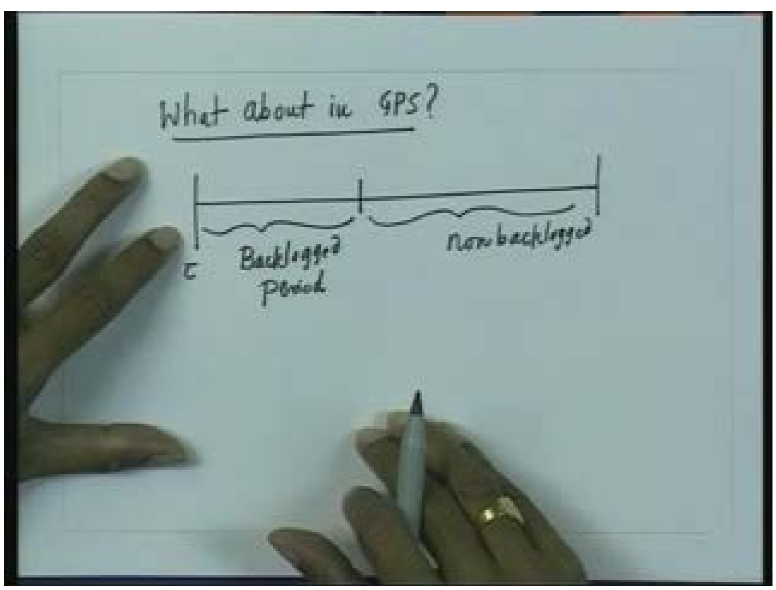
When all the traffic has been served, note that since its session potential is more than the system potential; it will continue to be act active but not backlogged. So, it will go in a state where it will be active but not backlogged. Now, what will happen is that suppose if it is in the active state and a packet arrives, then obviously it will move into the active and the backlogged state. But suppose, when it moves from here to here in the active state, but suppose no traffic arrives for a long time; then what will happen? The system potential will keep on increasing, remember, but the session's potential will be constant. It will not be increasing.

So, as a result what will happen? A time will come when the system potential crosses the session's potential. When that happens, the session will move into the inactive states. At this point, session's potential becomes lower than the system potential. So, session will actually have the periodic cycle, following cycles. So, it will be like this that tau, the session becomes backlogged for… the session's inactive period ends here and let us say that this is a backlogged period.

So, if it is a backlogged period, then again it enters into the non backlogged but active period. The session may enter into the backlogged period again and may go into a non backlogged period and at this time, its potential may become lower than the system potential and then this is the total active period. During this active period, the session's potential was always greater than or equal to the system's potential - during the active period and here is the inactive. So, afterwards, inactive period will start where the session's potential will always be lower than the system's potential.

Note that in the GPS, the fluid flow fair queuing what happens, how the situation is different than the fluid flow fair queuing algorithms? What about in fluid flow? So, if you ask this question that what about in gbs or the fluid flow fair queuing algorithm; then in the fluid flow fair queuing algorithm, we have backlogged and the non backlogged period only.

(Refer Slide Time: 51:59)



So, when the session becomes backlogged for the first time, its potential is made equal to its system's potentials and then the session starts receiving service and at that time, its potential is always equal to the system potentials, the backlogged period and when all the traffic is served here, the session becomes non backlogged. So, the session becomes non backlogged and when the session becomes backlogged for the first time, again its potential is made equal to the system potential.

So therefore, in the GPS you have these backlogged periods and non backlogged periods. During backlogged periods, the session's potential is always equal to the system potentials. So, the system potential is never less than the session's potential.

So, we will continue with our discussion of the rate proportional servers in the next lectures.

(Refer Slide Time: 53:30)