**Title: Cross layer TCP implementation over IEEE 802.11 WLAN**

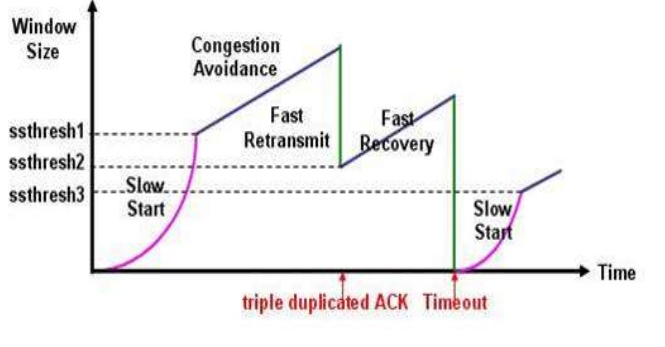
**Chapter 1: TCP congestion control**

* 1. **TCP mechanisms:**

Transmission control protocol is a connection oriented unicast transport protocol that offers the following feature: explicit and acknowledged connection initiation and termination; end-to-end reliability; in-order, and not duplicated data delivery; flow control and congestion avoidance. These multifold characteristic of TCP make it by far the most used transport protocol in the internet application. Flow control mechanism is mainly used to decide the flow at which the sender should transmit the data by changing the size of its congestion window. Congestion avoidance mechanism is employed by the TCP when the network encounters congestion. In congestion avoidance mechanism the network will undergo the fast recovery/fast retransmit.

* + 1. **Flow control:**

Flow control mainly defines the amount of data sender can send without waiting for an ack. TCP uses sliding window protocol for sending the amount of data. Window size is determined by the min(receiver window (*rwnd*),congestion window(*cwnd)*), *rwnd*  depend on capacity of receiver and *cwnd*  depends on capacity of network. Slow start and congestion avoidance are two phase of basic TCP congestion control.

****

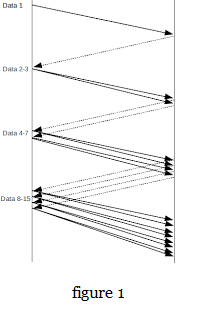
The above diagram describes slow start, congestion avoidance, fast retransmit and fast recovery. Here its clearly observed that TCP undergoes slow start until threshold and then continues with congestion avoidance until it encounters the congestion. After congestion encountered it undergoes fast recovery/fast retransmit.

**1.1.1.1 Slow start**

**Slow start (exponential increase)**

The slow start algorithm is based on the idea that the size of congestion window starts with one maximum segment size (MSS). The size of the window increase by one MSS each time one acknowledgement arrives. Thus, this algorithm starts slowly but grows exponentially as per figure1.

**cwnd = cwnd+1** (for every acknowledgement arrival)

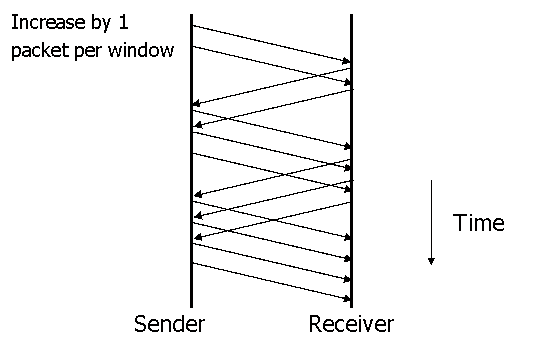
****

**1.1.1.2 CONGESTION AVOIDENCE**

Slow start growth in the network continues until it reaches ssthresh. ssthresh **-**  ssthresh is a threshold value up to which network follows the slow start growth. After ssthresh value the network enters into the congestion avoidance mode.

When the size of the congestion window reaches the slow start threshold then slow start phase stops and additive phase begins. In this algorithm, each time the whole “window” of segment is acknowledged, the size of the congestion window is increased by one.

**Equation** cwnd=cwnd+(1/cwnd)

****

* + 1. **Loss Recovery:**

If the congestion is detected in the network then rate of the data should be reduced. There are mainly two methods for detecting the congestion:

1. Three DUPAck (Duplicate Ack)

In the network when the sender sends the data and do not receive the acknowledgement and the retransmission time out occurs then the data will be send by the sender again. Thus, it may sometime happen that the acknowledgement of previously send frame may arrive after time out. As a result more than one ACK frame may be created on the sender side. This is known as duplicate ACK

1. Time out

Sender starts the timer and waits for the acknowledgement from the reciver. If the acknowledgement arrives within the given interval of time then it restart the timer. If the acknowledgement does not arrive within given interval of time then time out occurs.

After detection of congestion TCP performs the loss recovery by fast retransmit and fast recovery.

* + - 1. **Fast Retransmit**

The purpose of the *dupack* is to let other end know that a segment was received out of order segment was received out of order, and to tell it what sequence number is expected.

TCP does not whether ack is caused by a lost segment or just by reordering of segments, it waits for a small number of *dupack* to be received. It is assumed that if there is reordering of the segment, there will be only one or two *dupack* before the reordered segment is processed, which generates a new *ack.*

If three or more *dupack* are received in a row , it is strong indication that the segment has been lost. TCP then performs retransmission of what appears to be the missing segment, without waiting for retransmission timer to expire. This retransmission is called fast retransmission.

* + - 1. **Fast Recovery**

After fast retransmission sends the missing segment, congestion window avoidance executes, but not slow start is preformed. This is the fast recovery algorithm. It is improvement that allows high throughput under reasonable congestion.

When the sender receives the *dupack* , it means the data is flowing through the network between two ends but the current packet has been dropped by the network. Thus, there is congestion in the network but it does not want to abruptly reduce the data flow in the network. Due to this reason TCP will undergo the fast recovery and fast retransmit together.

* When the third duplicate *ack* is arrived in a row is received, set ssthresh to one half the current *cwnd* . Retransmit the missing segment. Set *cwnd* to ssthresh plus 3 times the segment size. This inflates the congestion window by the number of segment that has left the network.
* Each time another duplicate *ack* arrives, increment cwnd by the segment size. This inflates the cwnd for the additional segment that has left the network. Transmit the packet, if allowed by the new value of *cwnd.*
* When the next *ack* arrives that acknowledges new data, set *cwnd* to ssthresh this *ack* should be the acknowledge of the retransmission after receive this first ack.
  1. **TCP in wireless network**

As the internet users requirement of flexibility and mobility has been increased, wireless communication have became the best solution to fulfill the requirement of the user. Nowadays due to increase in portable electronic gadgets and increment in the internet have raised the requirement of the wireless communication. Thus due to this reason the demand of wireless communication have increased.

**1.2.1 Wireless network characteristic**

TCP was mainly implemented for the wired network. In wired network where the congestion is the main reason for the packet loss. At the present time internet spans a very large base of heterogeneous network such as wired and wireless and TCP is required to provide efficient solution for both wired and wireless network.

The wireless network faces many problem such as frequent route failure caused by the high node mobility.

The bandwidth delay product of the network does not remain constant because the user is mobile and so the delay does not remain constant.

Compared to wired network, wireless network will have more error and which will increase the retransmission rate compared to wired network.

* + 1. **TCP Performance issue in Wireless N/W**

The main characteristic of a wireless network is high link BER. It violates the fundamental assumption of TCP that packet losses caused by the link error are negligible (much less than 1%) and that packet losses are mainly caused by the congestion.

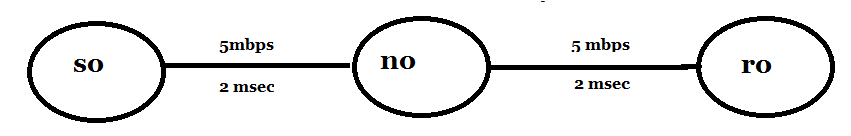
High BER in wireless network cause packet loss regardless of network congestion. This will cause TCP to unnecessarily reduce the transmission rate. The main cause for TCPs performance degradation in a mixed wireless/wired environment is its inability to detect the origin of the packet loss.

When the packet loss is detected, TCP employs congestion control algorithms to reduce the transmission rate. A single packet loss will cause duplicate ACKs and *cwnd* to be reduced by half according to the fast retransmit and fast recovery algorithm. TCP resolves the congestion in the network by lowering the transmission rate. However lowering transmission rate will degrade TCP performance if the packet is not caused by network congestion.

Thus, the main reason for degradation of TCP performance is mainly because is not able to detect if the packet loss is due to the error or it is due to the congestion. TCP consider all the packet loss is due to the congestion and it will packet loss due to error also due to the congestion. Thus, TCP efficiency reduce because of this reason.

This, problem can be solved if we provide the correct information about the packet loss which has occurred due to error and the packet loss is not due to the congestion.

* 1. **Simulation based study:**
     1. **Wired topology**

****

With an objective to study network utilization by TCP (the most reliable transport protocol over Internet) we created simple script in ns-2. First of all we need to create a topology. After creating a topology, we assign the delay and channel to the link. The delay between node so and no is 2 ms and bit rate is 5mb/sec. The delay between node n0 and r0 is 2 ms and bit rate is 5 mb/sec

**1.3.1.1 Calculation of TCP efficiency:**

The above wired topology was simulated for 10 second. For obtaining the efficiency we have traced the max\_seqno variable in our script. The max\_seqno was traced in the trace link file to obtain the maximum number of packet transmitted in the network.

**Observation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Error Rate | Max\_seqno | Throughput | Rate(Megabytes/sec) | Efficiency |
| 0.0% | 6058 | 6058000 | 0.60 | 96% |
| 1% | 4785 | 4785000 | 0.47 | 75.5% |
| 5% | 1476 | 1476000 | 0.1476 | 23.5% |
| 10% | 125 | 125000 | 0.0125 | 2% |

**Formulae:**

**throughput = (max\_seqno)\*(pocketsize)**

**rate = (throughput)/time**

Now from the observation, as we increase the error rate total bytes transmitted reduces. The main reason for the reduction of total bytes transmitted is retransmission due to the error

As the retransmission increases so less sequence will be transmitted in the network. Thus the throughput of the system will also reduce.

**1.3.1.2 Impact of bandwidth delay product on sending rate:**

Bandwidth delay product (BDP) is mainly defined as the product of bandwidth and delay in the network. From the bandwidth delay product we can estimate the number of packets present in the network.

In above wired topology we can change the bandwidth delay product by changing the delay and bandwidth of the network. In our topology we changed the sending rate and obtain the following result.

**Observation**

|  |  |  |
| --- | --- | --- |
| Bit rate | Maximum seq. no. | Throughput(bytes) |
| 10mbps | 12131 | 11645760 |
| 5mbps | 6121 | 5876160 |

In the above topology, we have changed the sending rate in the topology.

On increasing the sending rate of the sender, the throughput of the network also increases. Thus, as the bandwidth delay product increases the throughput of the network also increases and as the bandwidth delay product reduce the throughput of the network also increases.

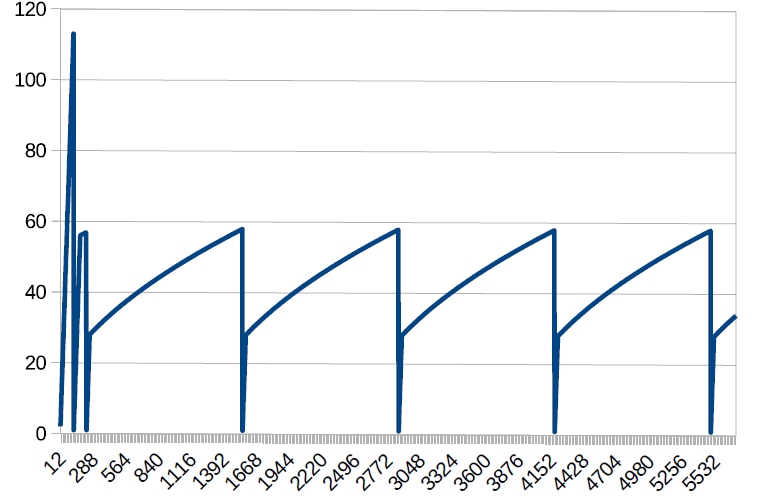
Thus, there is an impact of bandwidth delay product on the sending rate of the network.

**1.3.1.3 Impact of bandwidth delay product on cwnd**

In above wired topology we can change the bandwidth delay product by changing the delay and bandwidth of the network. In our topology we changed the sending rate and obtain the following result.

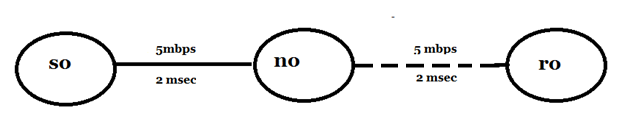
|  |  |
| --- | --- |
| Bit rate | Maximum cwnd |
| 5mbps | 56 |
| 10mbps | 61 |

Thus, as we increase the bit rate the maximum *cwnd* value also increases. The graph shown below was obtained by plotting the *cwnd* value obtained from trace file.



* + 1. **Wireless network**

**Topology**

****

**1.3.2.1 Calculation of efficiency**

Consider the above topology shown in the figure. In this topology there is wired connection between nodes s0 and n0. There is wireless link between the node n0 and r0.Simulation is performed by keeping the same parameter which were kept in the wired script so that we can understand the impact of TCP performance on wireless link.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Error Rate | Max\_seqno | Throughput | Rate(Megabit/sec) | Efficiency |
| 0.0% | 59276 | 455239680 | 2.68 | 53.60 |

From the above result it was obtained that at 0% error rate the efficiency is almost 50% in wireless topology. But in wired topology we observed above 95% efficiency and from this we can predict that the efficiency in wireless network reduces in the TCP.

**1.3.2.2 Impact of bandwidth delay product on sending rate**

In above wired topology we can change the bandwidth delay product by changing the delay and bandwidth of the network. In our topology we changed the sending rate and obtain the following result.

|  |  |  |
| --- | --- | --- |
| Bit rate | Maximum seq. no. | Throughput(bytes) |
| 5mbps | 3928 | 3770880 |
| 10mbps | 4046 | 3884160 |

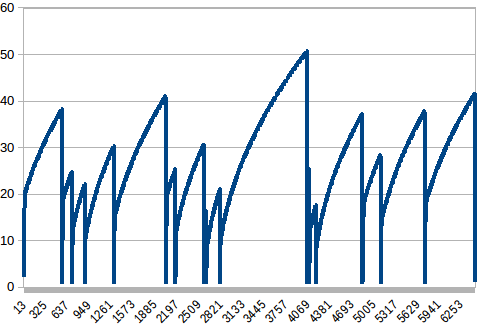
In the above topology, we have changed the sending rate in the topology.

It was observed that in wireless network we don’t obtain linear increment in

throughput as it was obtained in the wired network. In wired network the throughput almost became double when sending rate was doubled. But this result were not observed in the wireless network.

**1.3.2.3 Response of cwnd over wireless link**

In above wired topology we can change the bandwidth delay product by changing the delay and bandwidth of the network. In our topology we traced the *cwnd* variable in the trace file and we have obtained the following result.



The following graph was obtained by tracing the *cwnd* variable in the trace file. As per the graph, the value of the *cwnd*  do not remain constant.

**Reason:**

The main reason for the obtaining fluctuation in the maximum cwnd value is because the TCP do not differentiate between packet loss and congestion. In wireless communication there is more packet loss compared to wired communication. Thus, it will consider the packet loss as a congestion and reduce its data rate.

* 1. **Discussion**

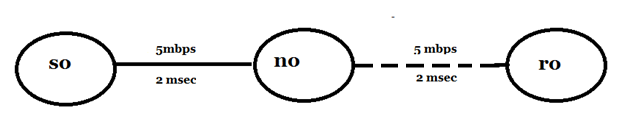
TCP was initially implemented for the wired network. But due to the increasing internet user and also increase in portable device raise the need for the wireless communication. But, when TCP was implemented for the wireless network the efficiency of the network has reduced greatly. TCP consider all the packet loss due to the congestion. But in wireless link there is normally bit error rate of 1% and it will consider all this packet loss due to the congestion. Due to this reason sender will reduce its data rate. Thus, it may not be able to utilize its network completely and efficiency of the network will be reduced.

**Chapter 2**

**Requirement of cross layer scheme**

**Theory**

**Topology**

****

Consider the above topology shown in the figure. In this topology there is wired connection between nodes s0 and n0. There is wireless link between the node n0 and r0.Now link delay variation in wireless link is more compared to the wired link. Thus due to this reason there is huge fluctuation in RTT measurement due to the presence of wireless link.

**2.1 Reason for variation in RTT**

**2.1.1 Link failure and reestablishment**

In stable wired networks, route failure occurs very rarely. The node mobility is the main source of frequent topology change and route failure in mobile ad-hoc network. Moreover, the link failure due to the contention on wireless channel may lead to route failure in both static and ad-hoc network. Due to this link failure bandwidth delay product change and it results into frequent buffer overflow because new link which establish has more delay with respect to old delay.

**2.1.2 Mobility of node**

In wireless network the node are continuously moving then it affects the RTT. The situation arrives mainly in satellite network as satellite is moving arbitrary on earth. When it goes away from the transmitter then RTT is increased and when it comes nearer to transmitter then it results into sudden decrement in RTT. Out of order packet reaches to receiver. Receiver discards those out of order packets. This situation cause false retransmission and timeout ultimately it degrades TCP performance. Mobility of node causes link failure. Packets are drop by router so retransmission will reduce the transmission rate.

**2.1.3 Random wireless losses**

The medium transmission errors in wired network so called bit error rate are negligible in compared to wireless network. Channel errors of wireless usually occurs randomly and in bursts due to the channel fading and interference and fading which cause it to be error prone in nature. High BER introduce two challenges to the reliable TCP protocol.

1. Packet loss due to corruption violates the assumption of TCP congestion control mechanism in that congestion is the main reason for most losses. In this case whenever the packet loss is detected the TCP will mistakenly take this as an indication of network congestion and reduce its sending rate.
2. In the presence of severe wireless channel contention, where the BER is high and the link layer local recovery mechanism is unable to recover the lost packet, TCP will face large number of consecutive packet loss. This may fluctuate the RTT.

The above are main reason for the RTT variation which are not due to th e congestion.

Above reason falsely trigger the congestion measured due to the variation in RTT.

**2.2 Impact of link retransmission on RTT**

The sending rate of TCP is decided from the RTT. Thus, RTT variation has the impact on the cwnd (congestion window). Now, cwnd variable mainly decides the number of packets to be sent in the network. As the packet size changes throughput of the network will also change. Now link retransmission will artificially inflate the RTT and it will have the direct impact on the throughput. Link retransmission adds the delay to the RTT and this additional delay will be interpreted by the network as congestion in the network. TCP at sender side will reduce the sending rate by reducing the *cwnd*  variable in the network. By this phenomenon the overall throughput will be reduced in the network. Link retransmission is responsible parameter for reducing the overall throughput and the efficiency of the network. In wireless communication there are more packet loss due to the error. As a result there will be more retransmission link delay in the network and it will reduce the efficiency of the network

**2.3 Approach for improvement of TCP efficiency**

The sending rate of the network mainly depends on the RTT. In our project, we are required to improve the RTT of the network by correcting the RTT. For the correction of RTT we have two approaches: single layer and cross layer.

In single layer approach we are required to make changes at only single layer and in cross layer scheme we are required to communicate between two layers of OSI model. Techniques used for the improvement of TCP efficiency improvement are given below:

**TCP-F(2002)**  TCP-F mechanism includes feedback based approach for the improvement of the TCP efficiency. Ad-hoc network are completely wireless network of mobile host, in which the topology rapidly changes due to the movement of mobile host. Now TCP protocols are mainly made for the fixed and reliable network. Hence, when used in ad-hoc network, TCP will misinterpret this as the congestion and reduce the sending rate and thereby reduce its throughput. To solve this problem feedback mechanism has been implemented so the source can distinguish between route failure and network congestion.

Whenever there is a route failure then route failure notification has been sent, allowing it to freeze the packet and stop the sending packet .Route re-establishment notification packet, upon which it resumes by unfreezing timer and continuing packet transmission.

**RR-TCP (2003)** TCP sender using the mechanism of Reordering Robust TCP(RR-TCP) to distinguish between reordering and loss of the packet. In the network, TCP may consider the packet disorder as the congestion and the throughput may be reduced. Thus, the above method can be employed to improve the TCP efficiency.

**RN-TCP (2006)** TCP sender using mechanism of Reorder Notifying TCP (RN-TCP) to distinguish whether the packet has been lost or reorder .The above method can be used to notify the sender that the packet has been reorder because the network may consider it has a congestion in the network.

**TCP DOOR (2012)** Detection of out of order TCP(TCP DOOR) is based on route failure notification and it resumes transmission upon the route restoration. The resumed transmission rate will not be always appropriate always, if new route has additional capacity, then a linear increase in *cwnd* does not react quickly enough to use capacity.

**Adaptive – TCP (2015)** In wireless network measuring the accurately the available share of network bandwidth by a TCP sender is difficult. Thus, in adaptive TCP adjust the loss window and ssthresh to suitable values from data analysis of obtained RTT and by concluding the uninterrupted evolution of TCPs sending rate.

**2.4 Discussion on the approaches**

In wireless link there is high BER compared to the wired network. Also in wireless network there occurs frequent change in topology as the hosts are mobile and there is frequent route failure which results in the packet drop.

Due to the above reason, the efficiency of the TCP is reduced in the wireless network. Due to the above mentioned problem in wireless network there is huge variation in the measurement of RTT.

The solution to all above problem is the correction of RTT. Normally RTT is measured as an end to end delay and it will include all contention and retransmission delay. But by correcting the RTT, we can solve our problem and improve the TCP efficiency.

For correcting the RTT, it is required to calculate the retransmission delay and link delay. Information about the link delay and retransmission can only be obtain at link layer. Now TCP is implemented at the transport layer and information from the link layer cannot be transferred from link layer to transport.

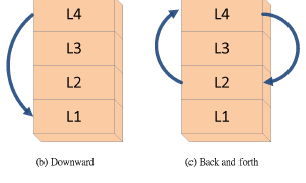
In order to transfer the information from the link layer to the transport layer, we are required to implement the cross layer scheme. By implementation of cross layer scheme we can transfer the information of the link layer to the transport layer. By obtaining the information of the link delay and retransmission delay, we can correct the value of RTT. The correction RTT can help us to improve the TCP efficiency by improving the throughput of the network.

**Chapter 3**

**Implementation of cross layer scheme in ns2**

**3.1 Cross layer**

Cross layer scheme is mainly used to communicate between two layer. Normally upper layer can obtain information from only layer below it. In cross layer scheme this rule is violated and we communicate between two layers to improve the network performance. As shown in the figure, there is cross layer implementation between two layers.



In my project, we need to communicate between two layers one is MAC and second is TCP. Direct communication between these two layers is not possible according to OSI model. Thus, we require implementing cross layer scheme for communication between transport layer and link layer.

**3.1.1 Two layer required for cross layer implementation**

In my project it was required to implement the cross layer between two layer transport layer and data link layer. The requirement of implementing the cross layer between this two layer is to transport the information from the data link layer to the transport layer.

Basically the transport layer has been implemented between two end so we can obtain the information of the end to end delay. The information of the retransmission and total time required for the retransmission required for the packet transporting are with the data link layer. This information of the total time required in the retransmission is at data link layer. In my project this information has to transmitted to the transport so network can correct it’s RTT.

For transporting the information from data link layer to the transport, we are required to implement the cross layer scheme. For implementing the cross layer we need to make changes in tcp and mac file.

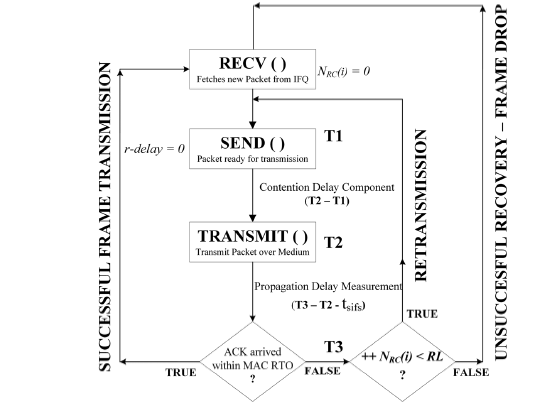
**3.2 Cross layer implementation in ns2**

For correcting the RTT, we are required to calculate the retransmission delay and contention delay. The RTT can be corrected by subtracting retransmission delay and contention delay from the original RTT. Contention delay and retransmission delay can be calculated from the mac file. After calculating the delay, we can transfer the information of mac to the tcp using the cross layer.

TCP is implemented at the transport layer and tcp.cc file contains the program for tcp file .This file contains the function which are responsible for deciding the sending rate of the application, a message segmentation and message reassembly. At transport layer the message is divided into segments; each segment contains sequence number, which enables this layer in reassembling the message. Message is reassembled correctly upon arrival at the destination and replaces packets which were lost in transmission.

**3.2.1 Logic flow of the mac\_802.11.cc file**

Below flow chart describes the flow of the program in the mac file



As per the above flow chart, first recv() function will fetch the packet from the application. Now the packet will move from recv() to send() function. Now send function will ready the packet for the transmission. The transmit() function will transmit the packet over the wireless channel.

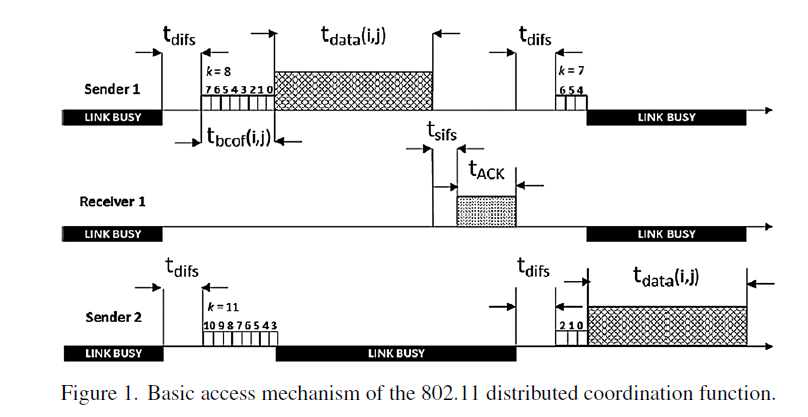
Now sender will wait for the acknowledgement till the retransmission time out. After the retransmission time out, the sender will send the packet again.

When the receiver receives the packet, it will verify if there is an error in the packet or not. If there is an error in the packet then it will demand the sender to retransmit the packet. The retransmission of the packet by the sender will be possible till its retry limit. If the packet has arrived error free then it will send the acknowledgement to sender and will demand for transmitting for new packet.

The above flow of the diagram has been implemented in mac\_802.11.cc for the wireless script. The above process has been implemented in mac\_802.11.cc for the transmission of the packet.

**3.2.2 Calculation of contention delay**

**Contention delay** is defined as the time taken for the transmitter from fetching of the packet till the transmission of the packet over the channel. Consider the below diagram for the transmission of the packet over the wireless channel.

****

As per the figure, the sender will first check if the channel is busy or idle. If the channel is sensed idle by the sender then the sender will wait for the DIFS duration.

After waiting for DIFS interval of time, if the channel is still sensed idle then the sender will wait for back off interval of time. After waiting for back off interval off time the sender will transmit the data.

According to figure, the sender first wait until the channel is busy. After waiting for DIFS duration and back off period the sender transmit the data frame. While waiting for DIFS interval of time if the channel is still busy than it will reset the timer and wait for DIFS interval of time. But while waiting for back off time if the channel is found busy then it will freeze the timer and restart the timer after the channel is sensed idle.

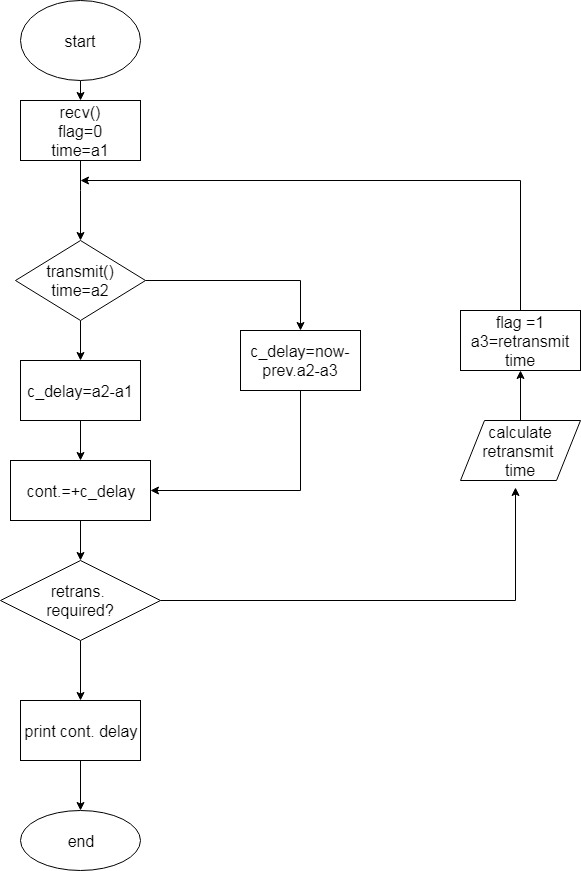
**3.2.3 Flow chart for calculating contention delay**

Consider the above flow chart for the implementation of the above algorithm. For the implementation of above program, we need to make changes in tcp.cc, mac\_802.11.cc file.

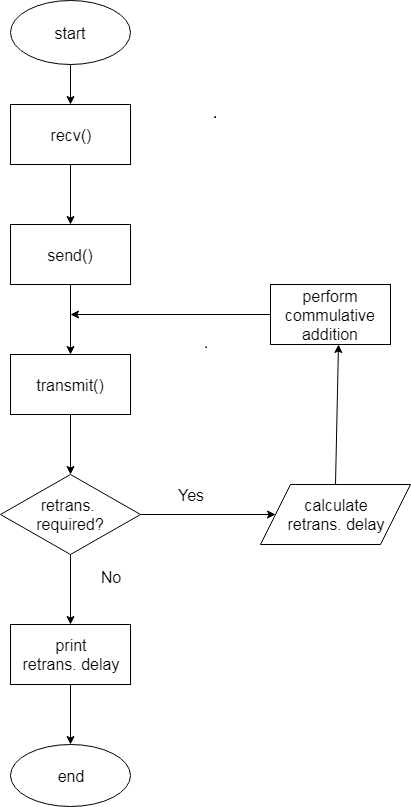
Contention time is given as the time required for the packet from the ready of packet till the transmission of the packet. Contention time is normally can be calculated as the difference of recv() and transmit() function.

If there is a retransmission in the given topology than again the packet has to be transmitted so now for calculating retransmission we need to remove the retransmission time from the given period of the time to obtain the actual difference of time.

By applying above method, we can calculate the contention delay for every transmission of the packet. We need to do cumulative summation of all the contention time obtained and we need to print this variable by implementation of the cross layer scheme.

****

**3.2.4 Flow chart for calculation of retransmission delay**

****

Consider the above flow chart for the calculation of link retransmission delay. In wireless communication if the sender transmit the packet and there is an error in the packet then the sender will undergoes the retransmission of the packet. It will continue to the retransmit the packet until the retry limit. In our project we are required to calculate the above mentioned retransmission time.

For calculation of retransmission time, we are required to make changes in the TCP and mac file. Now as per the flow chart, first recv() function will be called to fetch the new packet from the application. After recv() function, the sender will call the send() function to ready the packet for the transmission of the packet.

After the packet is ready for the transmission, the transmit() function will be called to transmit the packet over the wireless channel. Now, the receiver will verify if the packet is arrived at sender without error or not. If there is error then sender will retransmit the data over the channel.

In our program we will calculate the retransmission delay for the packet transmission and do the cumulative addition of the retransmission delay. Thus we can calculate the retransmission by subtracting the transmission time from the retransmission time of the network.

After calculating the cumulative delay of the retransmission, we are required to transfer this information to the transport layer. For transporting the information of the retransmission delay from link layer to transport, we are required to implement the cross layer scheme.

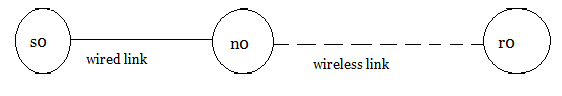
For implementing the cross layer we are require to define the variable in the header of the TCP. Now after defining the header variable in TCP header, we are required to store the variable in the TCP header.

When the packet is decoded at the transport layer, it will extract the delay from the transport layer. Now, it will subtract this delay from the original obtain RTT to correct the RTT.

By implementing the above process, we can correct the RTT of the network.

The obtain result of retransmission delay and contention delay are important in our project. Total link delay can be obtained as the addition of the retransmission time and contention delay. Now this delay can be used to correct the RTT of the network. We can improve the throughput of the network by correcting the RTT of the network.

**3.3 Creation of wireless script in ns2**



Consider the above topology for implementation in our script. In the above script there is wired link between s0 and n0. There is a wireless link between n0 and r0.

**Syntax** (creation of node):

set s0[$ns node]

**Description**

The above syntax will create the node s0

Using the above syntax, we are required to create node s0, n0 and r0.

**Syntax**(create a link)

$ns duplex-link $s0 $n0 100Mb 1ms DropTail

**Description**

The above syntax will create the link between node n0-s0 with 100Mb and 1 msec delay

Using the above syntax, we are required to set bandwidth and delay between the two wired nodes.

After creating the node and assigning the bandwidth between the nodes, we are required to define the packet size and the value of the ssthresh. The packet size defines the maximum value of the