FINAL PROJECT REPORT

TCP Combined With Network Coding

Wang tianyi
5100309326
F1003006

***Abstract*—Under lossy channel condition, current TCP usually can not work very well due to constantly packet loss,which causes three-duplicated ACK or time-out conditions, consequently throughput is very low.So when TCP is applied to wireless network, we should try to find a better implementation, then the concept of coding emerges. Compared to the current TCP, using coding can make high throughput and high transmission efficiency. Especially, when it comes to multicast scenarios, coding performs even better.However, some coding mechanisms are not compatible with current TCP sliding window mechanism. Thus we provide our new protocol which has little modification called TCP/NC to current TCP.Its core idea is using a linear combination of packets to encode and meanwhile, it has a higher throughput. *KEY WORDS*: Wireless lossy network, ACK, Coding, TCP, Congestion control.**

I. INTRODUCTION

With the rise of wireless network, people find that current TCP is not suitable for lossy channel condition, we all know that current TCP uses sliding window management mechanism and TCP-Reno method when it comes across loss.Such mechanism leads to low throughput when there is frequently packet loss,which means traditional TCP no longer meets the need of wireless network, traditional TCP needs to be modified to adapt to different network environment.

The concept of coding across data has been put to extensive use in today's communication systems at the link level. We don't use it in link layer and we move the idea to a new layer called network coding layer, which is between TCP layer and network layer.In our algorism TCP/NC essentially masks losses from congestion control algorithm and allows TCP/NC to react smoothly to losses.Although maybe it needs more time to spend in coding and decoding, we avoid the time-out case or duplicated ACK, so in the end the throughput will not decrease and to the contrary it will increase by 30 percent or more.

I have to admit that I did't work out anything novel, at the beginning I choose SDN for my research, but it needs a lot of basic knowledge of network infrastructure which I lack seriously, I read many materials and it didn't work, I felt puzzled so I gave up, I asked another topic at week thirteen and just configured how it works. I will research it deeply in the future.In this paper I just write down what I understand and make some assumptions.

The following part is organized like this, in section II the benefit of coding. In section III why traditional TCP are not suitable any more in lossy channels is detailly discussed. In section IV, our new mechanism are presented and in section V some other TCP coding mechanism are introduced. At last in section VI I will tell some limitation of our protocol and extend it to some other application.

II. BENEFITS OF CODING

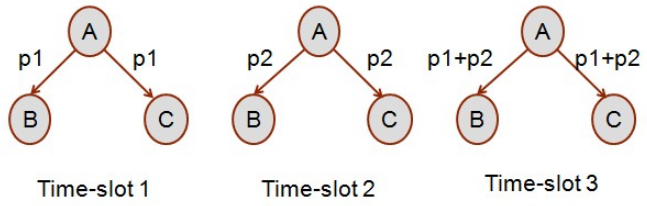

Fig. 1. examples using coding method

Assume such a scenario in figure 1.In the first time slot, A sends packet p1 to both B and C and this packet lost.In the second time slot, A sends packet p2 to both B and C and this packet lost.As the traditional TCP, in the third time slot A will multicast p1 to B and C again and in the forth time slot sends p2.But if we encode the two packet by XOR them and send the encoded packet p1*p2 at the third time slot. Then both B and C will receive the lost packet in one time slot.

So we can conclude that when using coding method, we can save one-time slot in our scenario and don't cause any other cost. This is a simple example, so if we use coding mechanism in more complicated scenario or in practice, we will make a higher efficiency.

III. CURRENT TCP DISADVANTAGES

Figure 2 shows a example of the condition when we come across three duplicated ACK.So firstly I will elaborate how data is sent between two host. Host A receives data from application layer ,then segment it in TCP layer and sends them to the IP layer. the segment goes through some procedure and then become packet for IP layer to transmit.It is passed through link layer and physical layer to IP layer of Host B,when host B on the destination computer receives them, the TCP layer of B reassembles the individual segments and ensures they are correctly ordered and error free as it streams them to an application.

As the picture shows, host A sends 5 segments to host B, and due to the traffic congestion, the second segment lost

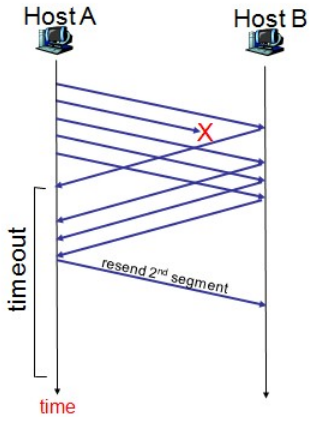

Fig. 2. Condition:Three duplicated ACK

so host B doesn't receive it. After a while the 3rd,4th,5th segments arrive, but host B needs the 2nd segment so it sends three ACK which sequence number is the second segment sequence number and buffer these segments which sequence number is larger than the 2nd's.According to TCP-Reno, this is the condition of three-duplicated ACK. When host A receive three-duplicated ACK, it knows the 2nd segment lost, then it resend the 2nd segment,and adjust the sending rate lower because it thinks it sends two much data ,the traffic is too busy to make a good transmitting environment. when host B receive the lost segment, it abstracts the segment in the buffer and then deliver them to application layer.

When this situation happens, the window at the sender stop slicing because it doesn't receive corresponding ACK, which leads to low transmission efficiency and the throughput are badly affected due to the unadvisable congestion control. In lossy channels, packet loss is not only caused by traffic congestion, but also influenced by channel condition.Obviously current TCP will recognize all these situations as traffic congestion. We can see that in such a scenario the time-out condition does't happen. But if a channel is too lossy, we will constantly meets the time-out situation which has a worse influence on the throughput.

IV. Some other coding mechanisms

Current approaches that use coding across packets are not readily compatible with TCP's retransmission and sliding-window mechanism.For example, the digital fountain codes, but the question is that it operates on a batch of packet, which means before a whole batch of packets is decoded, it can not be delivered to higher layer.Some already decoded part must wait, a lot of time is wasted on waiting.It is evident this kind of mechanism performs rather bad when it comes to some extreme situations.Meanwhile, current TCP uses sliding-window mechanism,such a mechanism doesn't know when a batch starts and when it finish,so it is contradictory to batch of packets.Apparently, Batch of packets is not suitable to current ACK mechanism either.

Some other coding mechanisms are mostly like the introduced one, when it comes into practice using, a lot of troubles appears so they all don't have practical application value.We need to find a appropriate way do implement the combination with TCP and network coding, which is applicable to the current TCP mechanism and needs subtle changes to TCP, it makes more significance and has much practical value.

V. TCP/NC mechanism

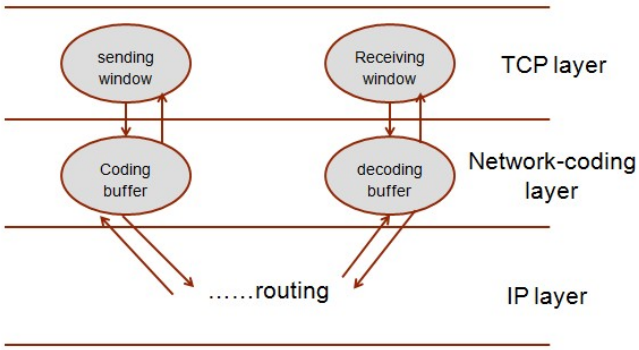

Fig. 3. TCP/NC implementation

We implement our coding mechanism by inserting a new layer called network-coding layer between TCP layer and IP layer,which is shown in figure 3.In our scheme, traditional TCP layer stays almost the same as it used to be, only a few modification is applied to it, such as RTT and way of ACK.Network coding-layer provide coding/decoding procedure. IP layer do some routing and doesn't have any changes, so do the lower layers.

A. Some Basic Concept

Some basic concepts are explained first, some of them will frequently appear next and this will help to understand our scheme better. We will use packet here and in practice, the operation object is indeed segment.

1) linear of combinations: If we have the number of k packets to form a linear combination, the linear combination is shown in the following equation.

$$\sum_{i=1}^{k} \alpha_i p_i$$

Where, α_i is the linear combination coefficient for packet p_i,chosen randomly from a specific area called field q.We should make the size of q larger enough in case of choosing dependent coefficients combinations.

2) See a packet: A terminal is said to have seen a packet p_k if it has enough information to compute a linear combination of the form $(p_k + q)$,where q is the combination of the packets,$\sum_{l>k} \alpha_l p_l$,which serial number l is larger than k, α_l is the linear combination coefficient for p_l.

For example, the sender sends a linear combination of p_1 to p_4 ,when the first combination arrives at the receiver, according to the definition, p_1 is seen, then after a while p_2 arrives at the receiver,multiplying the first one by a specific

coefficient and then using the second combination subtracts the result to eliminate p_1 p_2 is seen, and so on.But we should look out that if the current arriving combination coefficients must be independent from all the past combinations.If all packets in a linear combination are seen, then they can be decoded and then delivered to upper layer.

3) ACK: ACK mechanism is not as usual because coding operation, the new ACK mechanism is shown in the illustration,figure 4.Receiver no longer send TCP ACK to the sender, and the receiver TCP ACK is just used to inform the decoding part its information and condition.The new ACK is sent by the decoding part, when a new linear combination arrives at the receiver and causes a new packet to be "seen", then it sends an ACK for next packet, this ACK resembles traditional ACK, when sender encoding part receives the ACK, it adjusts its coding buffer by sequence number in the ACK and then delivers it to the sender TCP, and then sender TCP adjust its sliding window.

4) RTT: The start time for p_k is when the linear combination containing it is first sent, the end time is when the sender receives the ACK which means p_k is seen at the receiver.RTT is the time between the start and the end.

B. Encoding

After sender TCP do some preprocessing ,divide data into segments and then deliver them to the sender module, the core part of the sender module is coding buffer and coding window.When several segments are encoded, the coding part will add a coding header before the encoded segment, it contains all the essential messages about encoding.Then the sender encoding module sends the encoded segment R times to IP layer and then transmit it to the destination as it used to do. We should notice that source and destination address will not be encoded because these information needs to be checked when they are transmitted, these parts will be separated from its TCP header and added to the coding header.

1) coding buffer: Coding buffer stores segments which has not been "seen" yet,which is delivered from the TCP layer.it should observe a rule to add/remove segments in order to make sure dynamic space balance.

Add: When TCP segment comes, if the buffer is not full, then do some examination and then add it to the buffer.

Delete:

1 If buffer is full, drop the new coming segment.

2 If the sequence number of a segment in the buffer is less than the sequence number in ACK(this ACK is sent by decoder), then drop it.

2) coding window: it contains sub-set of the segments in coding buffer and use segments in this window to encode. Segments in this window is always the oldest W(window size) which has the oldest sequence number.This mechanism makes sure that a segment which has not been "seen" at the decoder will not be dropped.If we choose all of segments in the coding buffer to encode, we will suffer large overhead on the coding header, which will decrease rate of data part in a segment. So we use coding window to encode in order to make high efficiency and throughput.

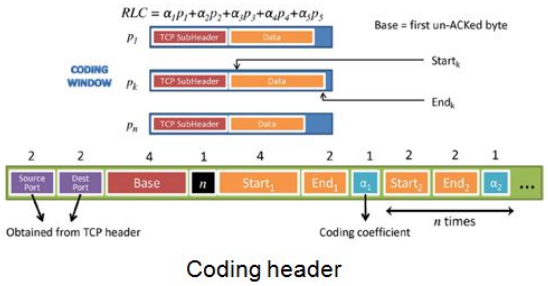

Fig. 4. Coding header

3) coding header: The constitution of coding header is show as figure 4, the meaning of elements in the header are listed below.

1.source and destination port:the port information is needed for the receiver to identify the coded packet's session.It is taken out of the TCP header and included in the network coding header.

2.Base:The TCP byte sequence number of the first byte that has not been acknowledged. The field is used by intermediate nodes or the decoder to decide which packets can be safely dropped from their buffers without affecting reliability.

3.n:the number of segments involved in the linear combination.

4:$start_i$:the starting byte of the ith packet involved in the linear combination.

5:End_i:The last byte of the ith packet involved in the linear combination.

6:α_i:The coefficiend used for the ith packet involved in the linear combination.

We can see that there is a start and end field for each segment, this is because when TCP layer divide data into segments, the data part length of a segment may be not the same. According to encoding rule, we should make the length consistent so we add some zeros after the end of the data part to segments which has a shorter data part.The start and end information are of great value when receiver decode the segments, receiver uses these messages to derive the original segments..

Each number above each field are the number of bytes it occupies, we can see that the more segments we encoded, the larger coding head we will get, for the number of n, we will have $5n + 7$ additional overhead.

4) Redundancy factor: R is short for redundancy factor, it's of vital important. We choose to transmit the encoded segment more times in order to mask loss from TCP layer. R is related to the channel condition, if we have more lossy channels ,we should choose larger R. On the basis of some other people's theory, if the losing rate of a channel is α, the theoretical value of R is $\frac{1}{1-\alpha}$, in the real world, we should choose more larger R in case some continuous loss happens.

We can see that choosing proper R is key to guarantee better performance.If We choose R too large, that will makes low throughput because you send a lot of redundant packet. But if it is too little, we cant mask the loss, which will lead to timeout and much lower throughput.

C. Decoding

When a encoding packet comes,abstract the combination coefficient, add it to a matrix(which has combination coefficient of other encoding segment).Using linear algebra solution to compute this matrix, if the new one is independent from each other, then add the segment to the decoding buffer and send a ACK to the sender for the oldest packet is "seen".If all the packets in the encoding packet is seen, then the encoding packet is decoded then deliver it to higher layer.

1) Decoding buffer: Decoding buffer stores the packets which has not been decoded yet.Its rule to add/delete packets is shown as follows.
1.Add: If the new coming packet cause a new packet to be seen then add this packet to the decoding buffer.
2.Delete:
(1)The packet must be decoded.
(2)If the number of base in the coding header is larger than this packet sequence number.

2) Decoding algorithm: It is done by using Gauss-Jordan elimination on the coefficient matrix.

VI. Some limitations

A. Checksum

We all know that traditional TCP uses checksum to check whether the packet arriving at the receiver is correct, if not, it will ask the sender to retransmit.But when using this mechanism, a packet cant be checked before it is decoded, so if it is wrong, the sender has already recognized it correct and the sliding window slide, which lead to wrong packet. So we should find a proper way do deal with it.

B. Proper R

We know that the redundancy factor R is a very important factor, If We choose R too large, that will makes low throughput because you send a lot of redundant packet. But if it is too little, we cant mask the loss, which will lead to timeout and much lower throughput. So how to choose the right R in different channels is a concerned thing, especially in multicast scenario, where different path has different losing rate.

C. Proper W

If W is too small, when we come across continuous loss, we cant mask loss and will suffer a timeout, but if we have too large W, the coding header will be long and decrease the data part proportion, which will also lead to a low throughput.

D. re-encoding cost

We know that our mechanism allows inter-mediate node to re-encode, but this will cause a unavoidable problem, when we decode the combination at the receiver decoding buffer, the more times we encode, the longer time we will need to recover the original packet.In the meantime,re-encode the packet also increase the length of coding header, if the upper limit packet length which the link layer can endure is just as the length of current combinations, and if we encode this for another time, the length will exceed the maximum. This maybe cause link layer to divide the segment or drop it.

VII. Some work in the future

It's evident that TCP/NC works well when there is a great deal of loss in transmission channel.Meanwhile,as for the natural properties, it can make high efficiency on the condition of multi-hop channels.So we mainly focus on its application on wireless or multi-hop situation.

A. TCP/NC with opportunistic multi-hop routing

Multi-hop wireless networks typically use routing techniques similar to those in wired networks,these traditional routing protocols choose the best sequence of nodes between the source and destination, and for ward each packet through that sequence, it's not convenient because some of the packets maybe transmitted to more distant node, which is nearer to the destination, but this kind of routing protocol regular which nodes to transmit, so these packets are wasted.Similarly, if the channel condition is bad, most of the packets fail to arrive at the specified node, but a nearer node ,they are dropped too and after a while receiver asks for the sender to retransmit, it's low efficient.If we can make use of the unexpected packets which arrives at other nodes, it can make things better.Then the opportunistic multi-hop routing protocol emerges, its abbreviation is EXOR. EXOR chooses each hop of a packet's route after after the transmission for that hop, so that the choice can reflect which intermediate nodes actually received the transmission. This deferred choice gives each transmission multiple opportunities to make progress,as a result EXOR can adapt to rather lossy radio links.

But EXOR meets two enormous challenges. Firstly,it operates on batch of packets to make sure the protocol works in order, but it's not compatible with the current sliding window mechanism.At the same time,operating on a batch of packets means you can't deliver the packets to the higher layer until the whole batch arrives.Secongly, because the randomly transmitting condition, we will suffer packets reordering and can easily cause 3-duplicated ACK or time-out.

Using TCP/NC combined with EXOR can deal it with both of the challenges well,since the receiver does not have to wail to decode a packet but can send a TCP ACK for every combination which cause a new packet to be "seen". To the reordering issue, the combination don't have this problem inherently, so our mechanism is of great help to EXOR.

B. intelligent R chosen in wireless multi-hop routing

Different pathes have different channel losing rate,which means when a node multicast packets to several nodes, we should choose different R according different channel condition in order to make high transmitting efficiency. To achieve this, we should make some modifications in our coding part. In wireless network, packets are broadcasted on radio wave, Consequently how to choose intelligent R remains a problem.I think using directional transmitting combined with time-divided transmitting method can work.

REFERENCES

[1] Jay Kumar Sundararajan, Devavrat Shah, *Network coding meets TCP:Theory and implementation*

[2] R.Koetter and M.Medard, *An algebraic approach to network coding* IEEE/ACM Trans.Netw.vol.11,no.5,pp. 782-795 Oct. 2003

[3] L.S.Sundararajan, D.Shah, and M.Medard, *TCP Vegas: New techniques for congestion detection and avoidance* in Proc.SIGCOMM symp.,pp.24-35,Aug.1994

[4] S.biswas and R.Morris, *EXOR:Opportunistic multi-hop routing for wireless networks* in Proc.ACM SIGCOMM,2005,pp.133-144