

# A Comparison of RaDiO and CoDiO packet scheduling algorithms over IEEE 802.11 WLANs

*Jeonghun Noh and Deepesh Jain*

Information Systems Lab, Stanford University  
{jhnoh, jaindee} @stanford.edu

## ABSTRACT

Video streaming in the Internet has been a huge success since H.263 was introduced. However, due to the fact that Internet offers a best-effort service with no guarantee of reliable packet transmission and bounded delay variation, video frame scheduling is crucial in video streaming. In this paper, we compare two video packet scheduling optimization schemes called RaDiO and CoDiO over IEEE 802.11 wireless networks. Then we propose a more practical CoDiO by using Loss-Delay based Algorithm (LDA) to adapt to the varying bottleneck link shared by multiple users. Simulation experiment results show that CoDiO performs similar to RaDiO in terms of Rate-Distortion, but reduces congestion over WLANs. Also, we show that CoDiO with LDA is a more practical approach for WLANs with varying traffic.

## 1. INTRODUCTION

So far, there has been lot of work to support streaming of pre-coded and stored video at the media server over the Internet. Unlike store-and-playback scheme, video streaming requires small amount of pre-roll delay due to interactivity between the server and the receiver such as user playback control, as well as hardware limitation of the buffer size at the receiver. In order to satisfy this requirement, active packet scheduling and other retransmission algorithm have been devised. Recently, designing an application that is aware of the network has become more attractive because it requires no advanced service from the underlying networks.

In [1], the authors suggest a Rate-Distortion Optimized scheduling algorithm that is called RaDiO. It determines optimal transmission policy for each packet in the transmission window iteratively by minimizing the Lagrangian cost function. The results show that it outperforms other heuristic packet scheduling schemes. When the last hop to the receiver is a bottleneck link, transmitting more traffic than the link can support will harm users sharing it. To reduce congestion in the network, authors in [2] suggest that Congestion (end-to-end delay) be used as a metric instead of transmission rate, which is called CoDiO (Congestion-Distortion Optimized scheduling). They show that CoDiO scheme outperforms RaDiO over one-to-one dedicated link with

limited bandwidth in terms of end-to-end delay, while achieving the same Rate-Distortion performance.

In this paper, we compare CoDiO with RaDiO over a shared medium such as an IEEE 802.11 WLAN. After analyzing the difference between a dedicated link and a shared medium, we propose a new algorithm which adapts to varying network traffic. Our method is based on the link capacity measurement by probe packets sent by the server and the average transmission rate estimated by the capacity measurement and the video packet loss rate.

This paper is organized as follows: we explain two channel estimation techniques. Then, we propose our new scheme of CoDiO with LDA [3]. In Section 3, the network simulation configuration is described and we provide the simulation results under various scenarios in section 4. In section 5, we conclude the paper and suggest future work.

## 2. ESTIMATION TECHNIQUES

Unlike RaDiO, CoDiO requires an accurate bandwidth estimate in order to calculate the best scheduling policy. In most cases the last hop link to end-users is a bottleneck, and the physical capacity varies from dozens of kbps (dialup modem) to several thousands of kbps (cable modem, xDSL) in case of point-to-point dedicated connections. In such a case, the bottleneck link capacity can be obtained by using back-to-back packets, details of which are described below. However, we need to estimate the amount of bandwidth available for the video stream as the last hop is a shared medium such as Ethernet or WLAN. We adopt LDA algorithm to estimate the current bandwidth. It tries to increase the transmission rate until it experiences a packet loss over some threshold. If packet loss rate exceeds the threshold, it drops the rate rapidly to avoid further congestion. This is considered as a TCP-friendly flow control of non-TCP flows, like audio or video traffic. The following sections describe how we implement capacity and bandwidth estimation.

### (1) Capacity Estimation

We define Capacity as the maximum instantaneous rate, i.e. the rate seen between two nodes in absence of

any traffic. Capacity can be approximated by sending several back-to-back packets as shown in figure 1. In most cases these back-to-back packets (burst of packets) are received at the receiver in the same order with no other packets intervening them [4]. In that case, we can estimate the capacity using the following formula,

$$C = S_{packet\_size} * (N_{packets} - 1) * \Delta t$$

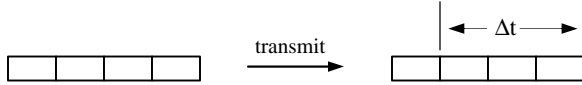


Fig 1. capacity estimation using a train of probe packets

where  $C$  denotes the link capacity.  $S_{packet\_size}$  denotes the size of each probe packet,  $N_{packets}$  the number of back-to-back packets sent and  $\Delta t$  the time difference between arrival of first and last packet.

The fact that wireless networks have much more overhead in sending packets compared to wireline networks, capacity estimation greatly depends on the size of burst packets. To get a more accurate estimate, we need to add packet overhead to  $S_{packet\_size}$  in the above calculation. The overhead can be theoretically calculated or estimated by simulations. If any of these burst packets are lost, we keep the past estimate.

## (2) Available Bandwidth Estimation

In order for CoDiO to work properly in a shared channel, it is necessary to estimate available bandwidth, which can vary with traffic. We use *Loss Delay Based Adjustment (LDA)* algorithm to estimate current bandwidth of the channel that can fluctuate due to various reasons including channel fading and sharing of the same channel by multiple users. Once the channel capacity is estimated, we use the following algorithm to obtain available bandwidth of the link.

First, the algorithm calculates the loss rate based on loss or late arrival of the packets at the receiver in a given loss window. Since CoDiO is not a constant bit rate (CBR) source, we modify the algorithm to set a loss threshold in order to tolerate a certain loss. When the loss is below the threshold, it increases the transmission rate by adding an AIR variable (additive increase rate) and decreases it in case loss is above some threshold. The details of the algorithm are explained in figure 2.

$$lossRate = \frac{\#lost / late \ packets}{\#transmitted \ packets} \quad (in \ loss \ window)$$

if ( $lossRate < loss\_threshold$ )

$$AIR(i) = AIR_{max} * (1 - R / C)$$

$$Rate(i) = Rate(i-1) + AIR(i) * If$$

else

$$Rate(i) = Rate(i-1) * (1 - lossRate * Rf)$$

endif

Fig 2. Loss-Delay Based Adjustment Algorithm

with  $Rf$  and  $If$  being the reduction and increment rate respectively. These values are determined by trial and error to get the best performance.  $C$  is the capacity, and  $R$  is the rate at which we have been transmitting the packets so far.

## 3. EXPERIMENTS SETUP

The experiments were carried out in *NS-2 network simulator* [5]. A High bandwidth link of 47.5Mbps is established between Node 1 and Node 2. An IEEE 802.11 WLAN is shared by two mobile nodes  $n2$  and  $n3$ .  $nx$  in the figure 3 represents use of network by other users. Conventionally WLANs have high bandwidth and will send data at higher rate, but for our experiment we use video stream encoded at 64kbps and the capacity of the link is set at 130Kbps<sup>1</sup>. We have kept capacity higher than 64Kbps to simulate scenarios with multiple users generating traffic.

First, we test RaDiO and CoDiO assuming that they know the bandwidth. Then we use LDA to estimate current bandwidth, because in real-life scenarios bandwidth can change due to traffic on the link.

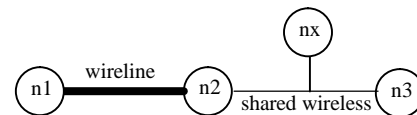


Fig 3. Node n1 and n2 are connected by a 47.5Mbps of high speed link. Node n2 is an access point, and n3 and nx are mobile nodes.

The video sequence used for this simulation is generated using a H.263 encoder for luminance component of the *foreman* sequence with layered coding

<sup>1</sup> When the link speed and the source rate are shrunk proportionally, the queuing delay and the packet loss rate remain the same.[6]

of one base and one enhancement layer.<sup>2</sup> Both layers have the same rate of 32Kbps. Frames are predictively encoded within a GOP and a packet contains exactly one frame. Therefore, once a precedent frame is lost, the rest of the following frames are no more useful. The video is displayed after pre-roll time of 600ms at the receiver.

For wireline networks, delay was measured by sending small probe packets and averaging the total delay. In case of wireless networks, however, the overhead of packet transmission is high. Thus, network congestion is measured by the queue length at node  $n2$  that serves as an access point of the WLAN.

#### 4. SIMULATION RESULTS

##### (1) Dynamic Wireline

As a preliminary study, we perform comparisons of RaDiO and CoDiO over a wire link that changes in capacity over time (switches between 50Kbps and 34Kbps every second). This helps us understand how capacity change affects performance. The last hop is dedicated to the video traffic alone. Thus, deterministic delay estimation of CoDiO is valid yet requires adaptation to the capacity change.

Back-to-back packet method is used to estimate capacity at the last hop assuming it is the bottleneck link. In this experiment, the channel capacity is the same as available bandwidth. Figure 4 and 5 show R-D (rate-distortion) and R- $\Delta$  (rate-delay) curves for optimistic and pessimistic cases, respectively. In the optimistic case, the two scheduling methods use capacity as measured, whereas in the pessimistic case the measured capacity is decreased by some factor before using it as an input to the scheduling algorithms.

In the pessimistic case, for most part RaDiO & CoDiO have similar R-D curves, but the R- $\Delta$  curves favor CoDiO. Whereas, for the optimistic case, we can easily see that RaDiO outperforms CoDiO, but has a much higher end-to-end delay.

From these two basic scenarios, we learn that CoDiO is sensitive to link capacity estimation. Also, if one wants to reduce end-to-end delay then a pessimistic way with CoDiO is better.

<sup>2</sup> When the network can specifically route packets that belong to different flows, Multiple Description coding can be a good choice. In this paper, we use one link between the server and the receiver where Layered Coding performs better.[7]

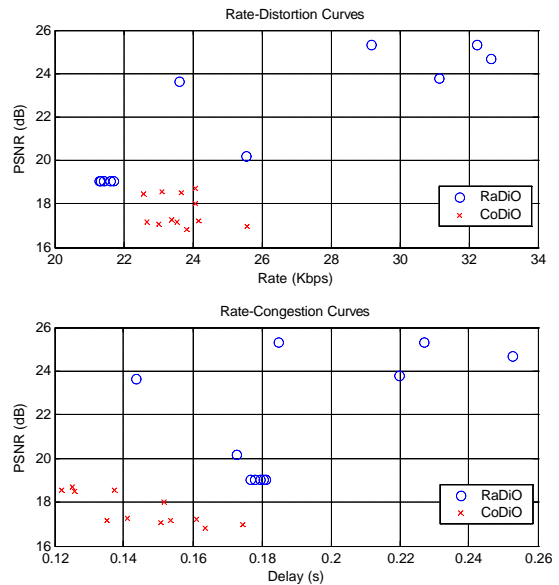


Fig 4. RaDiO & CoDiO over congested wireline (Optimistic)

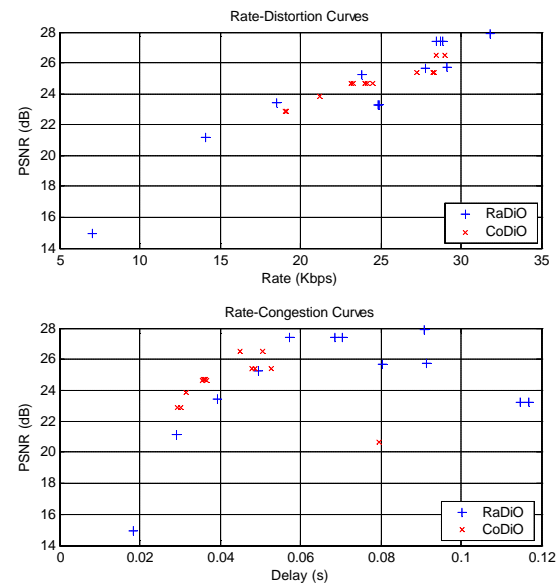


Fig 5. RaDiO & CoDiO over congested wireline (Pessimistic)

##### (2) WLAN – available bandwidth known

In the next step, we compare RaDiO & CoDiO over WLANs. The capacity was fixed at 130Kbps. We assumed that the server was informed of the available bandwidth before the transmission, 60kbps. When there was no other traffic, the wireless link plays a role similar to the dedicated point-to-point wireline. Next, we introduce a background traffic that does not congest the

link. A 64Kbps constant bit rate traffic source was added to the wireless link from the base station to another mobile node. Simulations were performed for RaDiO and CoDiO with and without this CBR traffic on the link.

Similar results to that of wireline network were observed, where the R-D curve for both RaDiO and CoDiO were similar, but CoDiO outperformed RaDiO as far as delay was concerned. Fig. 6 and 7 show the results of this experiment. The only major difference in the two figures is congestion (queue length), which can be attributed to the introduction of background traffic.

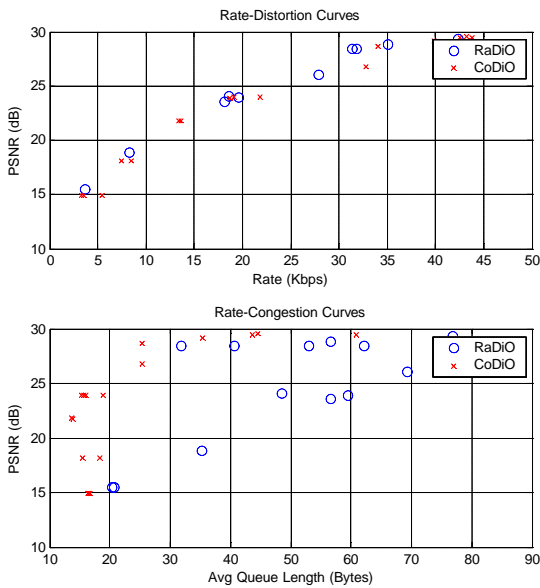


Fig 6. WLAN – Fixed Capacity, No Traffic

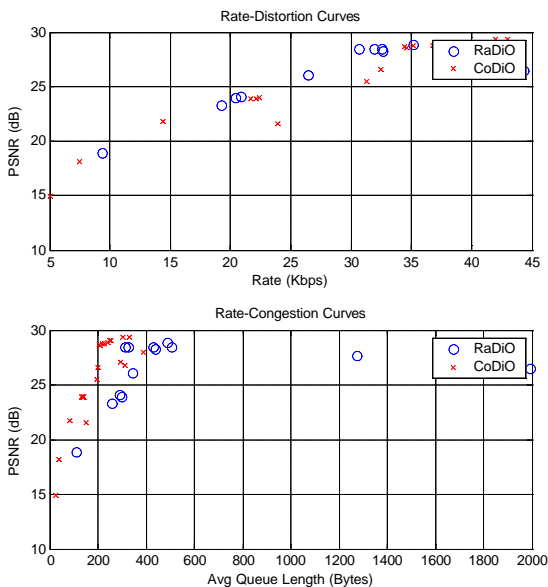


Fig 7. WLAN – Fixed Capacity, 64Kbps Traffic

### (3) WLAN – real-life scenario

In the previous setup, we assumed the video server could somehow know available bandwidth at the bottleneck link. However, this is not the case in general and the server has to estimate it. The wireless link was the same 130kbps as before but a CBR source with peak rate of 80kbps was induced to the wireless link. It was turned on for a second and turned off for the next second, repeating this till the end of simulation. We then tested the performance of CoDiO with LDA.

Simulation results show that when there is congestion and the available bandwidth varies, CoDiO with LDA performs better than CoDiO without LDA. Since CoDiO with LDA adapts to the current network state, it does not increase the transmission rate beyond a certain rate. Whereas, the original assumes the link capacity is 100% available and it sends as much as possible to exploit the free bandwidth, but loses packets (or are received after the deadline).

Also, when other traffic is sharing the same channel, a deterministic model to estimate the expected congestion by each transmission policy is no longer valid.

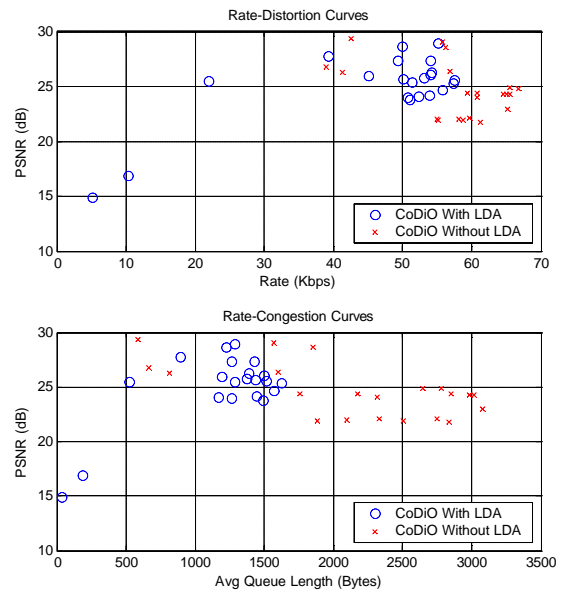


Fig 8. WLAN – CoDiO with & without LDA (Traffic)

## 5. CONCLUSION

In this paper, we compared CoDiO and RaDiO over IEEE 802.11 WLANs. From our results, we conclude that CoDiO performs better than RaDiO in reducing network

congestion when the available channel bandwidth is already known. RaDiO sends packets to the receiver regardless of network congestion; therefore it causes the queue at the last hop to be crowded frequently. Whereas, CoDiO is conservative in transmitting packets as it estimates the expected congestion while optimizing Delay-Distortion function. When multiple users share the bottleneck link, we observe that the original CoDiO no longer shows a good performance. We propose a modified CoDiO that incorporates packet-pair probe packets to estimate the maximum channel capacity and Loss-Delay Adjustment Algorithm to estimate available bandwidth. The simulation results show that CoDiO with LDA outperforms the original CoDiO when the shared link is congested by its own video traffic and other traffic.

For future work, a more accurate delay estimation model under a shared medium is necessary for CoDiO. Also, studying influence of collision caused by multiple requests from mobile users over WLANs is worthwhile. Lastly, CoDiO/LDA algorithm can be further optimized for a general situation.

## 6. REFERENCES

- [1] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," IEEE Transactions on Multimedia, February 2001. Submitted.
- [2] E. Setton and B. Girod, "Congestion-distortion optimized scheduling of video over a bottleneck link," submitted to MMSP 2004.
- [3] D. Sisalem, H. Schulzrinne, "The Loss-Delay Adjustment Algorithm: A TCP-friendly Adaptation Scheme", Network and Operating System Support for Digital Audio and Video (NOSSDAV), Cambridge, UK, July 8-10, 1998.
- [4] S. Keshav, "Congestion Control in Computer Networks" PhD Thesis, published as UC Berkeley TR-654, September 1991, Chapter 4: The Packet Pair Flow Control Protocol.
- [5] ns-2, network simulator, <http://www.isi.edu/nsnam/ns/>
- [6] K. Psounis, R. Pan, B. Prabhakar, D. Wischik, "The scaling hypothesis: Simplifying the prediction of network performance using scaled-down simulations," ACM Computer Communication Review, January 2003.
- [7] J. Chakareski, S. Han and B. Girod, "Layered Coding vs. Multiple Descriptions for Video Streaming Over Multiple Paths", ACM Intl. Conf. on Multimedia, Berkeley, CA, Nov. 2003.

## Division of labor for the project

### Jeonghun

- ns-2 implementation of LDA
- supporting tcl and perl scripts
- IEEE802 WLAN setup for ns-2
- change of delay estimation from end-to-end probe packets to queue length probing
- simulations on dynamic wirelines
- Project Report / Presentation

### Deepesh

- implementation of back-to-back packets for channel estimation
- simulations on WLANs
- analysis of the effect of the packet overhead added by 802.11 MAC layer
- Project Report / Presentation

---

Project work journal written by Jeonghun

May 24, 2004 08:50-11:50 AM

- analyze the results
- wrote the part of LDA
- collected the references
- ran more sims w/ modified LDA.

May 24, 2004 12:25 - 12:59:30 AM

- prepare the final network simulation setup.
- making notes.

May 24, 2004 11:30-12:25:09 AM

- wrote a part of the results section.

May 23, 2004 10:30-11:21:51 PM

- implemented on/off source.
  - ; verified.

May 23, 2004 06:56:46-10:20 PM

- cowork w/ Deepesh.
  - ; tune LDA together
  - ; began to write the report.

May 23, 2004 04:10-6:00 PM

- worked with Deepesh together.
- discussed the results
- talked out the outline of the report.

May 23, 2004 12:14:13 - 2:14PM

- check results.
  - ; CODIO w/ LDA seems to work well over dynamic wireline.
  - ; what about WLANs?
  - ; plotted the results.
  - ; talked to Deepesh over phone

May 23, 2004 02:30-03:49:08 AM

- TUNING for LDA.
  - ; modified algorithm in a detail part.
  - ; collect data for wireline. looks good.

May 22, 2004 08:19:32-01:30 AM

- CODING for LDA
  - ; scratch 30mins
  - ; coding 70mins.
  - ; syntax debugging 4mins! miraculous.
  - ; logic debugging 3hours!
  - ; files : codiovideotrace.cc, RLS.h, RLS.cc

May 22, 2004 03:40-06:10 PM

- kept thinking about how to measure congestion, not instantaneous capacity
  - ; impossible to it only looking at end-to-end AVG time.

- `; also, we should not rely on absolute time. hard to know one way trip time.`
- borrow ideas from TCP or TCP-friendly.
- `; study a paper called LDA.`

May 22, 2004 12:10-03:32:50 PM

- implement code for queue probing and printing into a file.
- `; drop-tail.cc, run.pl, count.pl, codiovideo.tcl`
- collecting data on wireline (pessimistic)
- Note
- `; from now on, we use the unit of BYTE for queue congestion for WLAN`
- `; end-to-end ms is still valid for wireline (just for simplicity)`

May 22, 2004 09:30:37-10:50 AM

- checked the sim results.
- plot results
- meeting w/ Deepesh

May 22, 2004 12:00-4:10 AM

- queue/drop-tail.cc
- tcl/lib/ns-mobilenode.tcl
- implement queue length measurement
- `; add a new command`
- `; use a default command`
- collecting data for plots (CODIO, no shared medium)

May 21, 2004 08:20-11:43:54 PM

- revised count.pl, run.pl, codiovideol.pl, filter\_flow.pl.
- to consider the first packet drop, calculate PSNR only on the second PLAYOUT.
- `; in playoutprocess.cc, RLS.h (WLAN flag introduced)`
- removed id 701(size 1000) packet transmission in codiovideotrace.cc
- check that RADIO is worse than CODIO when no channel sharing allowed.
- collecting data for plots (RADIO, no shared medium)

May 21, 2004 12:50-04:55:11 PM

- limit is 130kbps. if we go below it, I frame will be dropped.
- let's fill up traffic sharing the last wireless hop.
- wasted time.. need rest.

May 21, 2004 11:00-12:10 PM

- Meeting w/ Prof. Girod and Deepesh
- `; prepare meeting`
- `; talk to Prof.`
- `; discuss next step`

May 21, 2004 09:00-10:45:09 AM

- check the results obtained during the last night.
- `; concluded that there's a threshold to a transmission rate below which we cannot go`
- `; probably, due to a maximum propagation time of a packet on MAC.`
- `; the packet drop only tells where packets are dropped, not the cause.`
- contrived a way to adjust capacity.
- `; instead of manipulating transmission rates dynamically as we did on wireline`

---

```
; try to share the link with others.  
; CBR over UDP will do it.
```

May 21, 2004 12:07:07-2:24 AM

- help Deepesh get out of his bugs
- continue to setup WLANs
- do sims on RADIO over WLANs. w/ fixed 1500kbps.
  - ; it works very well.
- now with congested links. to 55kbps.

May 20, 2004 9:50-11:42:11 PM

- write a code to plot actual delay and estimated delay.
- Instructions:
  1. when you run './run.pl 1 > ScreenOut', a new count.pl, "actual\_delay###" will be created.
  2. Then, run 'grep updating ScreenOut | ./getDelayEstimate.pl 1'.

This will create 'delay\_estimation\_1'. When you run them several times, you'll see how to use them. It's very good for understanding situations!

  3. Then plot A and B arrays in matlab.
- summarize the results on dynamic wireline.
- setup WLANs

May 20, 2004 08:10-09:30 PM

- doing sims.
- write a perl code to extract end-to-end estimate of CoDiO.

May 20, 2004 04:40-06:50:54 PM

- doing sims.
- one OBSERVATION : when CODIO is optimistic, it sends much and it loses much.
- next GUESS : when CODIO is pessimistic, it sends less and it earns less.
- find out packet-pair flow control papers
- prepare basic results for PPT.

May 20, 2004 04:10-30 PM

- discussion w/ deepesh. future plan

May 20, 2004 02:00-02:28:56 PM

- confirmed the guess that when channel estimate is incorrect(too optimistic), the first part of a GOP gets lost, then that GOP gets lost.
  - ; when channel expands too fast, reduce it. otherwise CoDiO will excessively distribute packets and they will get lost.
  - ; conversely, when channel shrinks too fast, increase it. Otherwise, CoDiO won't send any packets.

May 20, 2004 07:45-10:30 AM

- check sim results

- remove background traffic
- run sims on skewed capacity change

May 19, 2004 09:50-02:00 PM/A

- keep running simulations
- think of reasons about CoDiO performs not well

May 19, 2004 06:50-09:42:04 PM

- do simulations
- read Eric's paper again.
- looked at the RLS codes.
  - ; found that FTT, RTT distribution is fixed. (to exponential?)

May 19, 2004 06:15-6:30 PM

- meeting w/ Deepesh

May 19, 2004 4:10-5:35 AM

- read Phillip's paper again. (p.16-20)

May 19, 2004 1:10-2:30 AM

- read Phillip's paper again. (p.8-15)

May 19, 2004 11:45-12:17:46 PM

- add debug code

May 19, 2004 10:50-11:42:03 AM

- read Phillip's paper again. (p.1-7)

May 18, 2004 10:07:30-10:30 PM

- added code to print odd frame info to playoutprocess.cc
- do sims.
  - ; for 1 sim, it takes about 50mins.
  - ex) 2sims -> 2856.610u 0.670s 1:37:22.77 48.9% 0+0k 0+0io 4507pf+0w

May 18, 2004 07:55-08:51:31 PM

- check the sim results.
  - ; observation :
    - at 4.0sec, delay estimate was 0sec. Channel est. was 50000bps
    - But, at 4.1sec, the actual capacity was set to 30000bps.
    - Packet ID 42 was sent at 2.6sec. CoDiO expected it to arrive at the receiver before 2800 ms, but due to the channel fluctuation, its estimation gets wrong.
    - Even pid 46 has been sent before pid 42 at 2500ms!! Which was a wrong decision also.
    - updating the vector of estimates time at 2600(ms), end-to-end delay estimate: 0.005040(s)
    - sending optimal packet 44 at time 2600+time shift ms, size=366Bytes
    - updating the vector of estimates time at 3800(ms), end-to-end delay estimate: 0.000000(s)
    - pid:40, packet Arrival :2358.527422 sec, deadline : 2600.000000 sec
    - pid:42, packet Arrival :2448.660612 sec, deadline : 2700.000000 sec
    - pid:44, packet Arrival :3023.594141 sec, deadline : 2800.000000 sec

Since the packet 44 is already late, the receiver ignores the following packets in the same GOP.

; remedy : leave a margin at capacity estimation!

- talked about this with Deepesh

May 18, 2004 04:20-06:55 PM

- modify codiovideotrace.cc to change 'NUM\_OF\_FRAMES' from 120 to 40 in RLS.h
- modify playoutprocess.cc for debug
- modify RLS.cc for debug. (DEFAULT\_CLIP\_LEN 120->40)
- > due to complexity, gave up and tried to recover the changes.
- do simulations

May 18, 2004 08:40:22-10:00 AM

- check the sim results
- talked to Deepesh, discussed our results.
  - ; he found that RaDiO performs better over WLAN even at high transmission speeds.
  - ; I found that incorrent capacity estimation of CoDiO causes performance degradation

May 17, 2004 09:00-11:29:37 PM

- set the reverse link speed of the wireline setup to 50KB. (it was 10000KB)
- examine the effect of capacity estimation skew
  - ; at each second the codio examines the capacity.
  - > to check how CoDiO works when the channel capacity changes after each second. for example, 100 ms after each second. 1.1, 2.1....

May 17, 2004 10:40-11:54:24 AM

- check WLAN overhead
- check the influence of cross traffic to channel estimation.

May 16, 2004 6:50-8:50 PM

- Meeting with Deepesh
  - ; discussed the topics I had when I had a meeting with Prof. Girod
  - ; explained in detail what has been changed for the last 3 days.
  - ; suggestions and comments, things to do for the next step.
  - ; do simulations and see an intermediate result. examined it together

May 16, 2004 04:30-05:42:37 PM

- Found a problem with packet drops in WLAN at the beginning.
  - ; It's due to #define DROP\_RTR\_MAC\_CALLBACK "CBK" // MAC callback
  - ; in trace/cmu-trace.h
  - ; each routing protocol such as AODV, DSDV, DSR, TORA asks the remote node to call off transmission.
  - ; this is an inherent property of WLAN. To reduce this influence, try to play video many times.

May 16, 2004 02:30-04:20 PM

- Doing SIMULs.
  - ; with dynamicWireline.
  - ; check that time shift of starting time has no problem.
  - ; found a problem in RLS. (frame 215 repeated 5 times, which makes no sense)
  - ; added a code to change # of probe packets to CVT.cc

May 16, 2004 07:50-08:50:09 AM

- back to dynamic wireline version.

May 16, 2004 02:50-04:10:03 AM

- to improve count.pl  
; wrote separator.pl  
; modified count.pl, run.pl  
; debug  
; due to ns bugs, compromise performance  
- sometimes, trace results are missing. ex) +,-,r

May 16, 2004 12:10-01:02:23 AM

- to improve count.pl (speedup of matching algorithm)  
- do simulations with changed channel capacity

May 15, 2004 10:50-11:37:17 PM

- minor change.  
- the results seem wrong.  
; PSNR doesn't change with different lambda values.

May 15, 2004 08:20-09:34:30 PM

- A big improvement!  
; found a problem in codiovideotrace.cc and rls.cc  
# HUN clever trick!  
# May 15, 2004  
# problem: a WLAN needs time to be stable.  
# solution : we start codio video server after some time.  
# for instance after 2.0 sec  
#  
# another problem : codiovideotrace and RLS assumes it starts at 0.0!  
# solution : 1) modify code <- impossible (requires a great amount of time + debug  
)  
# 2) deceive the server by recording a time-shifted info to the ACK  
packets.  
# of course, we adopt 2) option.  
; now it works. I worked for 7 hours straight. need time for dinner.....

May 15, 2004 06:50-08:20 PM

- Simulations on WLAN.  
; no UDP packetization  
; probe packet size 500B. 11 train packets.  
; channel Tx speed fluctuation ny manual setup  
; # of loops : 2 (27 sec simul time)

May 15, 2004 02:50-6:40 PM

- knew that codiovideotrace doesn't support an arbitrary start time.  
; change of t\_now dynamically needed.

- variable binding in Tcl and C++ code.

; found a bug  
; modified tcl/lib/ns-default.tcl (added initialization of variables)

- initialize of t\_now in CVT.cc

; t\_now = 0; -> commented out. ( 3 locations )  
instead, binding and initialization at codiovideol.tcl

---

- NOTES:

for dynamically changing wireline links:  
use count.pl\_original, filter\_flow.pl\_original, codiovideo1.tcl\_dynamic

for WLAN:

use count.pl\_WLAN, filter\_flow.pl\_WLAN, codiovideo1.tcl\_WLAN

May 15, 2004 12:20-02:30 AM

- in order to reduce packet drop at the early stage due to ARP table setup, start sending probe packets(to measure end-to-end delay) 1 second early, so that actual video packet won't be dropped.
- debug of perl script continued..

May 15, 2004 6:00-7:18:36 AM

- debugging perl script
- found that WLAN trace has different format.  
; had to change perl scripts(count.pl, filter\_flow.pl)
- change the queue size of IFQ(interface queue) between MAC and LLC from 50 to 100.

May 14, 2004 15:10-16:30 AM

- studying Perl to modify perl scripts.

May 14, 2004 13:10-14:10 AM

- meeting with Prof. Girod
- making a note after the meeting.

May 14, 2004 08:40-10:10 AM

- debug perl code.  
; out.trace file was not written. why?  
; since we changed the network topology, the format of the trace file changed too.  
; count.pl, flow\_filter.pl changed accordingly.

May 13, 2004 08:10-9:54 PM

- debug WLAN code. Deepesh pointed that ACK is no longer produced.  
; figured that out. in NS, to use WLAN, we need to wait for over 1sec in order for the network to exchange info  
; using ARP. -> start codiovideo server at 1.2 sec.

May 12, 2004 03:47:30-6:55 PM

- resume designing a functionality to update bandwidth change dynamically
- modified /mac/mac-802\_11.cc  
added command "datarate" // for now, input numbers without a unit, such as Kb or Mb
- learned that,  
; if any tcl files in /tcl/lib are changed, ns-tcl.c is newly created and compiled again for ns and nse.
- finished designing 'wireless link with dynamic bandwidth.'

May 12, 2004 12:40:26-2:35 PM

- resuming WLAN setup
- modified /tcl/lib/ns-mobilenode.tcl
  - ; added procedure 'bandwidth'
  - ; added "WOW" to see if a modified otcl file needs to be recompiled.
  - > yes. do 'make' if you want ns to reflect the change.

May 12, 2004 10:55:13-12:10 AM

- starting WLAN setup for dynamic capacity change

May 12, 2004 10:30-10:54:36 AM

- reviewed Eric's presentation ppt slides.

May 12, 2004 12:34:54-1:45 AM

- began coding Tcl for WLAN.
- checking points
  - ; generate a node movement
  - ; generate a nam trace for wireless node
  - ; adjusting data rate(bandwidth)
- understanding of MAC layer
- understanding of a new trace for wireless channel (MAC)

May 11, 2004 11:10-11:45 PM

- studied how to change bandwidth of WLAN.

May 11, 2004 07:26:21-10:30 PM

- setup the first round of simulations.
  - ; according to Eric's suggestion.
  - ; prepare the last hop whose link capacity is changed manually.
  - ; hard to change the wireline link capacity on the fly.
  - ; found a good way to do this.

May 11, 2004 04:10-6:00 PM

- meeting with Eric.
  - ; the presentation about CoDiO, RaDiO in a lot detail.
  - ; discussion regarding them.

May 11, 2004 9:30-10:46:51 AM

- work on ErrorModel
- work on MAC / Wireless-PHY(NetIf) / Channel structure for mobile nodes

May 10, 7:10-11:19pm

- review codiovideo.tcl
  - ; connection between UDP - LossMonitor
  - ; need to make a design decision
    - > a new application VS modified UDP VS LossMonitor
- learn how to add a new agent(transport layer protocol) to ns-2.
  - ; to continue to use Eric's code.
- learn NAM animator.
- learn a wired-cum-wireless network (wired nodes, base-station gateway nodes, wireless nodes)

- learn a new trace for wireless nodes including their movement information
- learn how to create a new agent such as ping (for a new application or a new agent)
- read the chapter 16 of mobile networking in ns. (ns-manual)
- help Deepesh design a capacity estimating apps.

=====

<< imported from my PDA >>

05/10/2004	1:00 PM-	05/10/2004	2:00 PM	1	Using nam. Create a wired cum wireless network
05/10/2004	7:20 PM-	05/10/2004	11:20 PM	4	Ns, create a new agent, codiovideo.tcl, udp.cc, loss-montor.cc
05/09/2004	10:20PM-	05/09/2004	11:20 PM	1	Ns. Nam
05/09/2004	8:10 PM-	05/10/2004	12:10 AM	4	Ns, wlan, udp, phy, wls phy
05/08/2004	7:10 PM-	05/08/2004	10:10 PM	3	Ns. Eric's code, wlan in NS
05/07/2004	1:00 PM-	05/07/2004	2:00 PM	1	Deepesh, eric's code
05/03/2004	7:50 PM-	05/03/2004	9:50 PM	2	Study ns
05/01/2004	10:00 PM-	05/02/2004	12:20 AM	2.33	
04/30/2004	12:40 AM-	04/30/2004	1:40 AM	1	NS
04/30/2004	2:00 PM-	04/30/2004	3:20 PM	1.33	Meeting w/ eric
04/30/2004	1:20 PM-	04/30/2004	2:00 PM	0.66	Review eric's manual
04/30/2004	3:20 PM-	04/30/2004	6:20 PM	3	Ns installation for deepesh
04/30/2004	8:40 PM-	04/30/2004	9:30 PM	0.83	Ns installation for deepesh
04/29/2004	5:20 PM-	04/29/2004	7:50 PM	2.5	Play with ns
04/28/2004	1:00 PM-	04/28/2004	2:20 PM	1.33	Help deepesh install ns
04/27/2004	8:40 AM-	04/27/2004	9:00 AM	0.33	Final review
04/26/2004	1:00 PM-	04/26/2004	2:20 PM	1.33	Meeting w/Eric, deepesh.
04/26/2004	3:50 PM-	04/26/2004	5:40 PM	1.83	ns2 install, reading
04/26/2004	6:10 PM-	04/26/2004	6:50 PM	0.66	Install
04/26/2004	8:30 PM-	04/26/2004	9:40 PM	1.16	Install ns-2.26 success! At firebird
04/26/2004	10:30 PM-	04/27/2004	2:20 AM	3.83	Writing proposal
04/25/2004	12:10 PM-	04/25/2004	1:00 PM	0.83	Paper
04/25/2004	3:00 PM-	04/25/2004	4:20 PM	1.33	Discussion w/ deepesh.
04/24/2004	1:00 AM-	04/24/2004	3:00 AM	2	
04/24/2004	10:20 AM-	04/24/2004	2:10 PM	3.83	Try to install ns. Failed
04/24/2004	8:30 PM-	04/24/2004	9:50 PM	1.33	Paper
04/24/2004	10:30 PM-	04/24/2004	11:30 PM	1	Paper
04/24/2004	11:50 PM-	04/25/2004	1:10 AM	1.33	
04/23/2004	12:20 AM-	04/23/2004	1:10 AM	0.83	Print paper
04/23/2004	11:10 AM-	04/23/2004	12:10 PM	1	paper, phil
04/23/2004	1:50 PM-	04/23/2004	2:50 PM	1	Meeting w/ deepesh.
04/22/2004	12:30 AM-	04/22/2004	2:30 AM	2	ise website surfing.
04/22/2004	9:30 PM-	04/22/2004	11:30 PM	2	Paper survey
04/20/2004	9:30 PM-	04/20/2004	11:30 PM	2	Paper reading