Analysis of Routing Metrics for Wireless Mesh Networks

Vijay Gabale, Sagar Bijwe
Computer Science And Engineering Department,
Indian Institute of Technology, Bombay,India
CS 647 Class Project Instructor : Prof. Purushottam Kulkarni

April 11, 2008

Abstract

With respect to current trend of using off-the-shelf 802.11 devices to form wireless mesh network, there are several metrics proposed for routing in mult-hop networks. In this course project, we analyse the performance of two such metrics: ETX (Expected Transmissions) and ETT (Expected Transmission Time).

Our work is two fold. First, we characterize a link in multi-hop network by analysing the effects of packet size and transmisson rate over a link. We further observe how packet error rate varies with received signal strength and whether there is any correlation between the two. Second, we establish a 2-hop 4 node mesh network and quantify the behaviour of ETX and ETT in terms of delay and throughput.

Through this study, we find that there exist a considerable correlation between packet error rate and received signal strength. We attribute the high error rate to external interference. Also, since ETT takes into account bandwidth in its value, our experiment shows that, ETT performs better in terms of throughput than ETX. We also document our experiences in using Soekris board, installing operating system and madwifi drivers, turning a node into a router and the ‘things’ to consider while performing a link level wireless experiment.

1 Introduction

802.11 wireless networks have become ubiquitous owing to cheap wireless interfaces, unlicensed spectrum and inherent convenience of untethered computing. Until quite recent, most of the deployments have been realized through IEEE 802.11 Wireless LANs operating in infrastructure mode. An obvious problem is the location of the access points. No or improper site survey while placing access points often ends up creating RF holes (no coverage area). The installation of multiple access points is also expensive and is not convenient because of Ethernet wiring from access points to backhaul network. But there is way out. These 802.11 devices, in turn, can be connected to each other to form mesh network which are easy to deploy and maintain, scalable and reliable. Thus, a Wireless mesh network replaces the access points by wireless mesh routers with mesh connectivity established among them so that they can route each others traffic to provide seamless connectivity and comprehensive coverage. Low cost wifi nodes and low power requirements due to induction of multiple hops results in much cheaper network. Dead zones can also be eliminated by adding wireless mesh routers or changing locations of mesh routers with ease.

Recently, interesting application of multi-hop wireless networks have emerged which include commercial “community wireless network” and “Long distance mesh networks” for providing rural connectivity. In such a networks, most of the nodes are stationary or minimally mobile and hence certain parameters can be tuned to utilize the resource optimally. Thus, we can configure the transmit rates of nodes, size of packets and the next hop router to destination to make optimal use of network nodes. But because of the intrinsic unpredictability in wireless medium, the network frequently gets underutilised. Users experience lower throughput, packets get lost and nodes sometimes get trapped in intermittent connectivity. Considering these facts, in our study, we characterize a link in a multi-hop network and analyse two routing metrics ETT and ETX. The goal of any routing metric is to choose a high throughput path between source and destination. This path becomes critical to achieve given wireless mesh scenario. To be specific, the questions that we answer are as below

- Does there exist any correlation between RSSI and packet error rate?
- What are the reasons for high error rate given received signal strength?
- Based on these observations, how should we characterize a wireless link?
- How does ETT and ETX behave in a network in terms of delay and throughput?
- How does these metrics behave when there exist external interference?

We set up a 4 node mesh network in a 100m radius. The links are set up using high gain omni-directional antennas.

We used the Soekris Engineering 4826 boards as our nodes with Voyage operating system installed in it. We employed and modified 'olsrd' algorithm to add ETT. ETX was already present in the algorithm. Our main results are as follow:

- There exists a considerable correlation between packet error rate and recieved signal strength.
- We attribute the high error rate to external interference which we observe through our logs. Thus, in the absence of external interference, with respect of RSSI, we can consider abstracting a link as 'up' or 'down'.
- With increase in packet size, the probability of packet being corrupted increases.
- ETT outperforms ETX in terms of delay and throughput given the same networking scenario.
- In the presence of external interference, ETT sometimes ends up choosing interfering path. Unless, the bandwidth of a path compensates for the interference effect, ETT chooses the path as that of ETX.

Our results are close to the results we expect through theoretical study. The technical implications are that, there exists a threshold of received signal strength after which the probability of packet error rate increases sharply. Thus RSSI is a fair enough metric to characterize the link performance. In the absence of external interference, we can directly use the value of RSSI as the quality of link. Also, with increase in packet size, the probability of error increases. We should aim for an average packet size to strike the right balance between throughput and error rate. Any metric, that takes into account bandwidth of the link outperforms the metric which simply considers number of hops or expected transmissions.

Our contribution is in amending the 'olsrd' routing algorithm to incorporate the ETT metric, in quantifying the wireless link characteristics and in analysing these metrics in 'interference' and 'non-interferece' scenario thoroughly in terms of delay, throughput and paths taken by individual packets.

The rest of the report is organized as follows: In section 2, we describe our experimental set up, software and hardware used, methodologies employed befoe taking experiments and topology in detail. Subsequently, we give the analysis of data obtained from experiment thoroughly. Next, in section 4 we describe the olsrd algorithm along with how ETX and ETT metrics are calculated. In section 5, we elaborate, the configuration issues in setting up and deploying the Soekris boards. In section 6, we tabulate Dos and Donts while aiming for performing a wireless experiment. We hope that our experiences and in detail explanation will be most useful who plan to perform outdoor wireless experiments. Finally we give future scope and conclude in section 8 and 9 respectively.

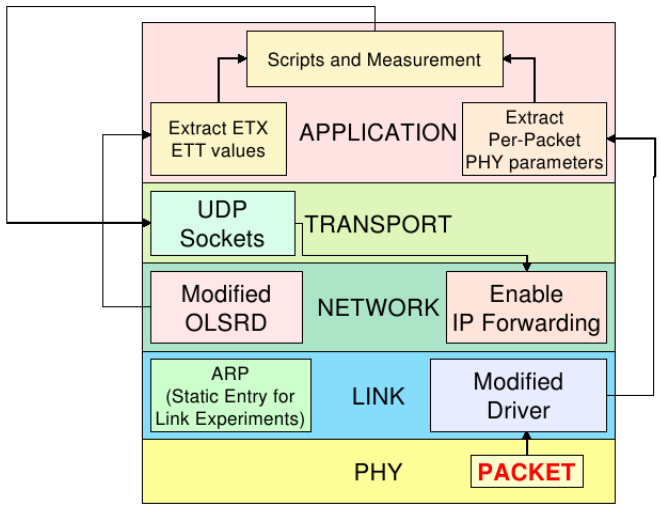

Figure 1: System Archietecture in terms of layers

2 Experimental Set up and Methodology

In this section, we describe the hardware and software used, the link used for experiments, mesh network topology. We also document the methodologies used before placing the nodes for an experiment.

2.1 System Architecture

Figure 1 shows our system architecture in terms of network layers. Our programs and scripts work at the application layer which avail physical layer values provided by modified madwifi drivers. The olsrd daemon runs as the routing algorithm. Every packet is thus routed according to this daemon. The algorithm calculated link probabilities by sending 10 packets and tracking their response. The detailed algorithm is explained in the next section. The olsrd algorithm is modified to incorporate ETT metric which considers bandwidth of the hop. For link characterization experiment, we added a static ARP(Address Resolution Protocol) entry in ARP table. This was necessary since receiver works in 'monitor' mode which does not respond to ARP queries of transmitter.

2.2 Software and Hardware Set up

Our nodes are Soekris Engineering 4826 boards. Figure 2 shows one such a board. It is actually a compact, low-power, low-cost, advanced communication computer and it is based on a 233 or 266 Mhz 586 class processor. It has one 10/100 Mbit ethernet ports, up to 256 Mbyte SDRAM main memory and uses a CompactFlash circuit soldered onboard for program and data storage. It can be expanded using up to two MiniPCI

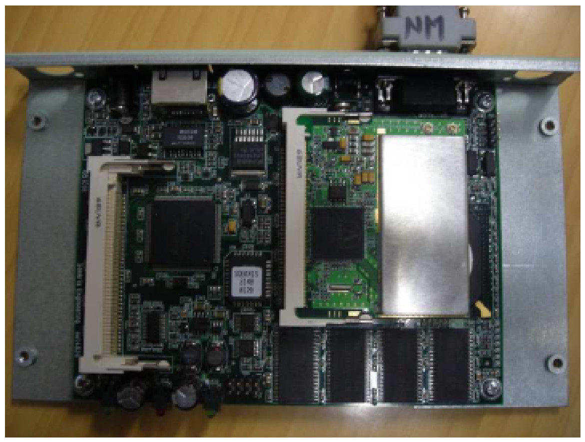

Figure 2: Soekris 4826

Figure 3: Experiment location, shows one of two face to face backyards

type III boards. It is economical in terms of form factor, cost and power as compared to conventionaly laptops. We powered these boards using a 12V battery or using POE (Power Over Ethernet) adapter.

We installed a debian flavour of linux, called Voyage (2.6.20.19 kernel) on Soekris board. We used modified madwifi drivers to pass per packet information to user level. This information includes per packet received signal strength, noise level, error checksum, MAC addresses and Sequence number with others. The Linux proc file-system is in turn used to read this information from kernel buffer. As a routing daemon, we used olsrd routing algorithm which had implementation of ETX metric. We modified the code to incorporate ETT metric. We also enabled the ip-forwarding of each node to make it act as a router.

We generate packets in bins. Each bin contains 100 packets spaced 20ms apart to allow external interference to happen, if any. The program actually acts a UDP sender which sends total of 6000 packets to server running at the other end.

Figure 3 shows backyard where we conducted our experiments. It had a similar backyard at the front. We get around 90 meter of end to end link of sight. Figure 4 shows our network configuration for this link. (a) shows the set up for link characterization experiment whereas (b) shows the network topology for routing metric analysis experiment. (b) shows that one of the node is deliberately kept operating in 5.5Mbps rate as compared other which is operating in 2Mbps rate.

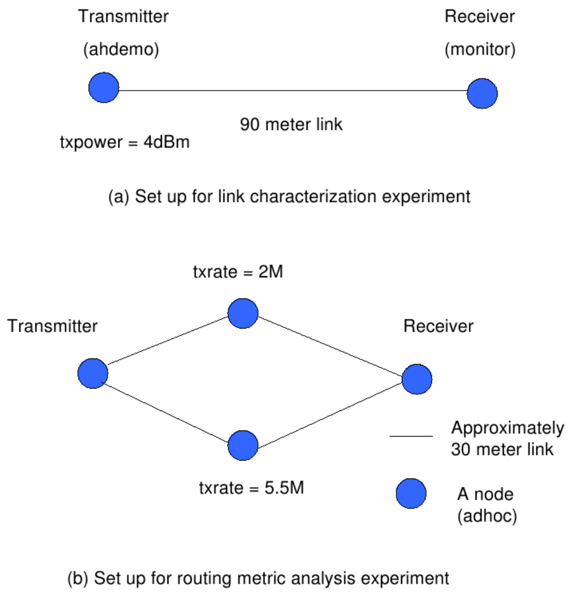

Figure 4: Network set up

2.3 Measurement Methodology

To perform experiments and to yield them fruitful results, we need to set certain parameters so as to analyse the behaviour we wanted.

- Transmit power: For 100m experiments we keep transmit power as one of the follows: 0dBm, 4dBm and 6dBm. Transmit power has strong influence on packet error rate and hence we experiment with above values so as to get -80dBm to -95dBm transmit power at the receiver location.

- Transmit rate: In future, we would like to analyse interesting behaviour with respect to transmit rate but this parameter is not varied when taking link level measurements for our objectives. We vary the parameter while taking ETT and ETX reading where one node is deliberately kept operating in 5.5Mbps mode while others are in 2Mbps.

- Packet Size: We vary packet size in terms of 256,512 and 1024 bytes. We compare the 256 and 1024 packet size data for analysis. Packet size is important since the bit error rate is affected by the received signal strength.

2.4 Experiments

We used above parameters to perform UDP experiments. For the link characterization experiment, we varied trasmit power as 0dBm, 4dBm and 6dBm. For each transmit power, we trasmit 6000 packets in a bin of 100 packets by varying packet size in 256, 512 and 1024 bytes. There is 20ms gap between successive packet transmissions and 6 second gap between starting transmission times of two bins. The experiment lasts for around an hour.

Initially, we observed that, during experiments, we were getting lot more management frames than intended data frames. Hence, we disabled the transmission of these frames by setting the transmitter in ahdemo mode. Also we want our receiver to capture all the packets to observe the effect of external interferece. Hence, we keep our receiver in monitor mode. This is actually promiscous mode operation where the card passes up errored packets and management frames up to the driver. The receiver logs the data for each experiment where a experiment involves setting of a particular transmission rate, for a particular packet size. We used flash memory of Soekris boards to store the logs. The approximate distance between the transmitter and receiver around 90m. We used high gain 8dbi omni-directional antennas to perform all our experiments.

For ETT and ETX measurements, we used a 4 node mesh network. We have source and destination at two ends (they are not in range of each other) with 2 intermediate nodes in the range of the two. We kept the packet size 512 bytes, transmit power 0dBm, transmission rate of one the intermediate node as 5.5Mbps with others operating in 2Mbps. The approximate distance between source and destination was 60m with distance from intermediate nodes as 30m which are spaced 20m apart. For observing the impact of interference on the metrics, we later kept a wifi source operating in the vicinity of node operating with 5.5Mbps rate.

3 Analysis of experiments

Here we elaborate on the data we obtained from these experiments. The primary characteristic of a link in mesh network is the packet error rate. Below we study the dependence of packet error rate on received signal strenght. The primary characteristic of a routing metric is the throughput and we analyse throughput for ETX and ETT metrics.

3.1 Dependence of error rate on received signal strength

As stated earlier, here we analyse packet error rate over a link. The receiver location is carefully chosen to get requisite variation in signal strength. For an experiment, the percent error rate and average RSSI is calculated for a bin comprising of 100 packets i.e. we only consider our data packets when calculating average RSSI for a bin. We say a packet as corrupted when

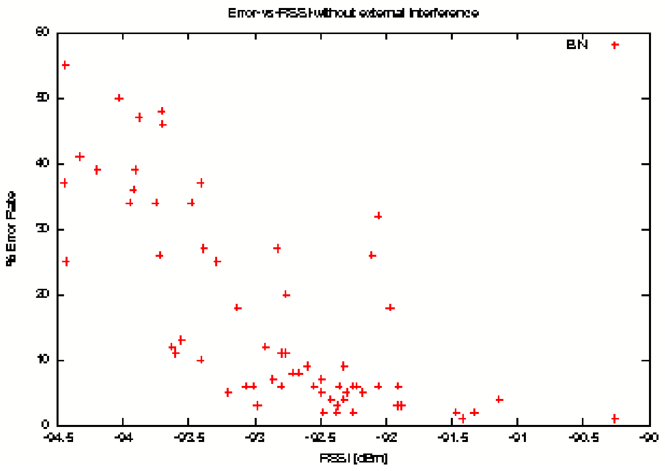

Figure 5: For an experiment, the percent error rate and average RSSI is calculated for a bin comprising of 100 packets.

the checksum of that is received as failed. The result is shown in figure 5. We observe that, as the received signal strength decreases below -92dBm, the percent error rate for a bin increases drastically to about 30 percent. There could be two reasons for this sudden shoot up in error rate: either these bins are experiencing external interference or the multipath fading effect is getting induced due to the building around experiment location. Thus, it is necessary for us to quantify at least one of the two reasons. We have receiver working in monitor mode which captures foreign packets. Thus we can quantify external interference effect and check whether it is the reason for high error rate.

Hence, we further quantify the effect of interference by analysing our traces. We found that, on an average around 1150 packets are getting induced in every bin. So we classify a bin as interference prone if it contains more than 1150 foreign packets else we consider a bin as interference free. Figure 6 show a different angle of figure 5 and indeed as the effect of interference increases, more packets get corrupted for a bin. Thus we attribute the high error rate to external interference.

We plot the percent error rate for a bin as a function of number of foreign packets that we receive while operating in monitor mode. We again see that, as the number of foreign packets for a bin increases, more errors get introduced. See figure 7.

3.2 Relation between packet size and error rate

Here, we repeat the above experiment for two different packet sizes: 256 and 1024. Figure 8 shows the results of this experiment. For this experiment of sending 6000 packets, for each received signal strength value, the percentage of errored packets is calculated and plotted. We observe that for larger packet size, the probability of packet getting corrupted increases.

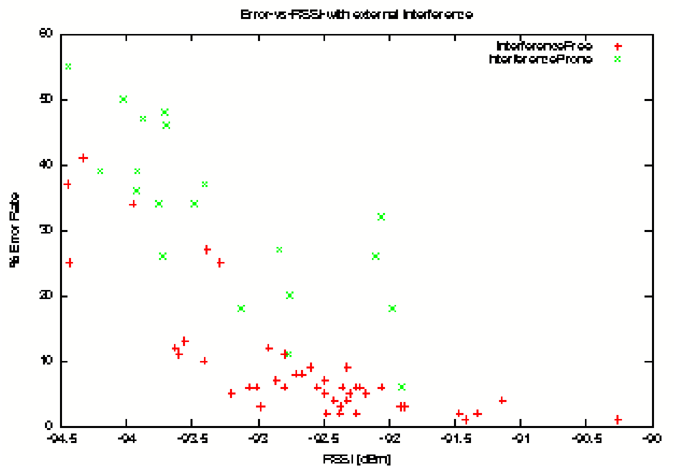

Figure 6: For an experiment, the percent error rate and average RSSI is calculated for a bin comprising of 100 packets. We classify a bin as interference prone if it contains more than 1150 foreign packets.

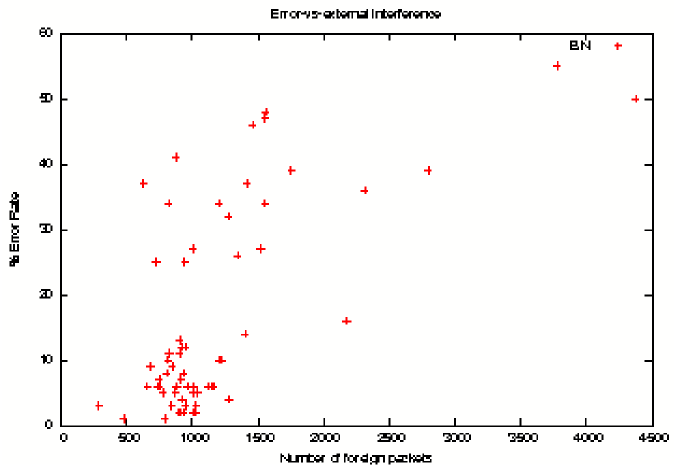

Figure 7: For each bin, we calculate the number of foreign packets that are interfering with our transmissions. This plot shows that as the number of foreign packets increases, more packets for a bin get corrupted.

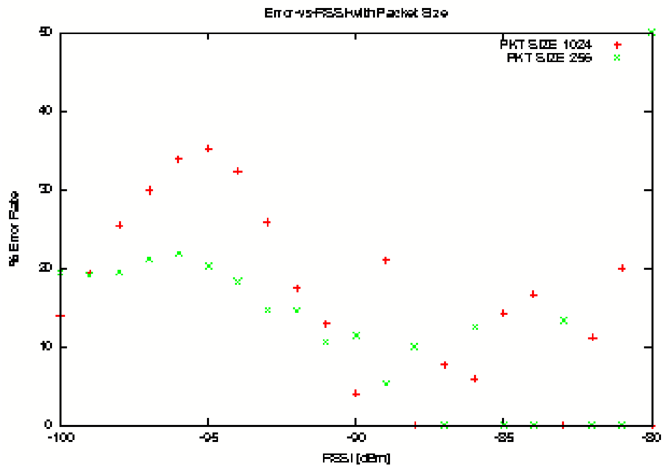

Figure 8: For an experiment of 6000 packets, for each received signal strength value, the percentage of errored packets is calculated and plotted. We observe that for larger packet size, the probability of packet getting corrupted increases.

3.3 Analysis of ETX and ETT metrics

As mentioned earlier, we establish a 2 hop 4 node mesh network. The source sends 10000 packets to destination. The transmission rate is varied as 2M and 5.5M. And for each transmit rate, we ran ETX and ETT metrics. Each experiment lasts for around 20 minutes and we perform 4 such experiments. We deliberately keep on intermediate node operating for all the 4 experiements with 5.5Mbps.

Throughput and delay are the two metrics that we wish to measure to compare the two metrics. Delay component actually comprises of the time required to transmit, propagation delay over the path and queuing delay at the intermediate node. We keep the packet size constant as 512 bytes. So for every packet the transmission time is constant. To quantify the propagation time and queuing delay, we transmit 10 packets at the UDP level and make the receiver send acknowledgment immediately. This give us round trip time (RTT) over the path. Thus we calculate end to end one way delay over the path by dividing RTT by 2. This delay is then conveyed to the receiver through the first packet that transmitter sends to receiver. The throughput is calculated as the packet size divided by the time difference between reception of two successive packets, per packet.

Figures 9 to 12 show the results of above experiments. The figures for delay show that, for both transmit rate, the delay required for packets is more in case of ETX as compared to ETT. Also, the throughtput is more for the case which uses ETT metric. This behaviour is as per our expectations since ETT takes into account the bandwidth of the path while choosing the next hop while ETX just considers the expected number of transmissions. Thus, when routing algorithm runs with ETT, the source is almost always expected to choose the node which employs 5.5Mbps transmit rate.

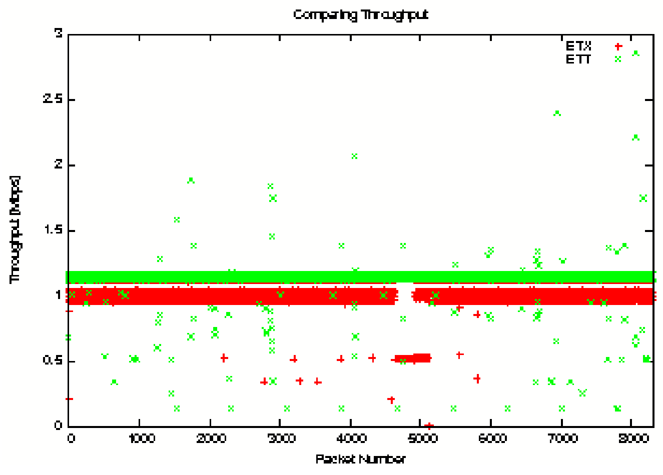

Figure 9: This graph shows the throughput variations over the packets. The higher throughput for ETT is due to selection of higher bandwidth path to destination from transmitter.

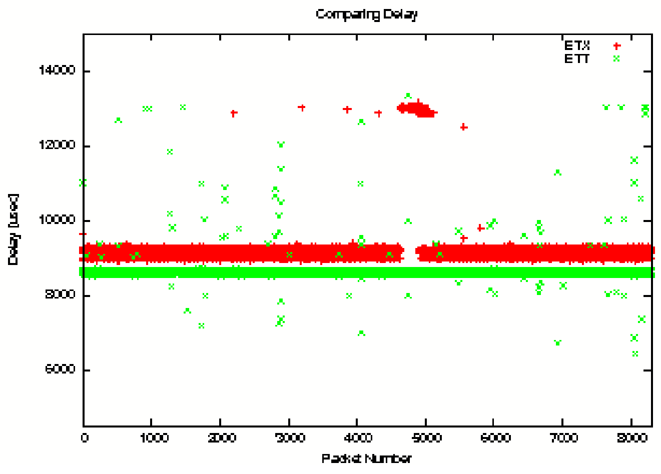

Figure 11: This graph shows the delay variations over the packets. The lower delay for ETT is due to selection of higher bandwidth path to destination from transmitter.

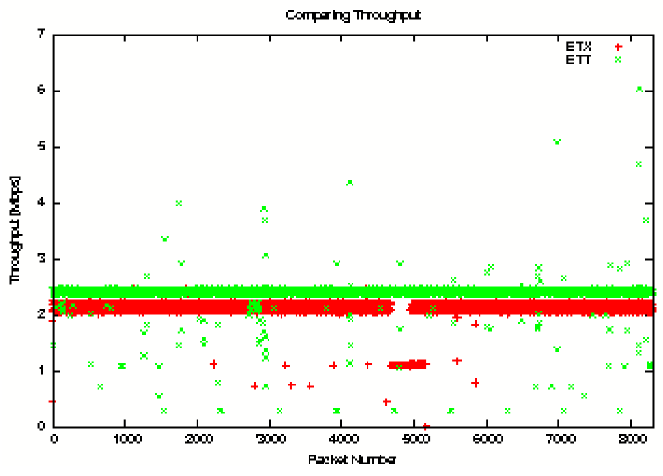

Figure 10: This graph shows the throughput variations over the packets. The higher throughput for ETT is due to selection of higher bandwidth path to destination from transmitter. The difference becomes apparent when transmitter is set to 5.5Mbps.

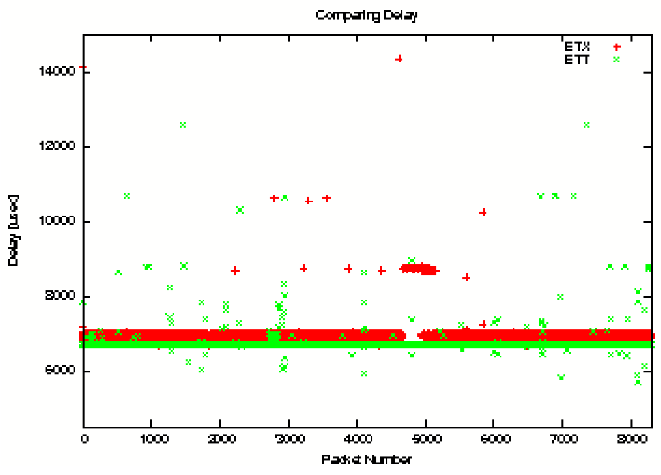

Figure 12: This graph shows the delay variations over the packets. The lower delay for ETT is due to selection of higher bandwidth path to destination from transmitter. The difference becomes apparent when transmitter is set to 5.5Mbps.

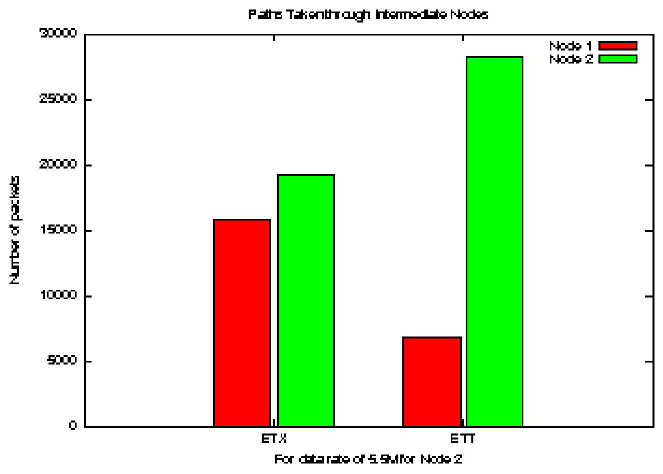

Figure 13: This graph shows that packets get routed through intermediate node operating at 5.5Mbps taking advantage of ETT which considers bandwidth to calculate metric value.

But we inspected the traces so as to know what are and how many packets get routed through what intermediate hop. Figure 13 shows the results of parsing the traces collected at the receiver side. The graph shows that packets get routed through intermediate node operating at 5.5Mbps taking advantage of ETT which considers bandwidth to calculate metric value. For ETX, we see that both the hops get chosen with almost equal probability. But the node operating in 5.5Mbps shows its presence with slightly higher transmissions going through it when we employ ETX. But the we see 4 fold more packets being routed through the hop running with 5.5Mbps than the one running with 2Mbps.

We also tracked how the values of ETX and ETT converge. Figure 14 and 15 show results of this inspection. We see that there is an initial random stage after which the ETX and ETT valus get into the steady state. These values are what are seen by the receiver with respect to source. What we observe is that, the ETX values remain almost the same for both intermediate nodes (lines overlapping with each other) as shown in figure 10. But ETT value for 5.5Mbps is considerably lower than the other node. This explains why most of the paths get routed through this node.

We were further curious regarding how these metrics would behave under the effect of external interference. To answer this, we arranged an interfering source in the vicinity of the node operating at 5.5Mbps rate. From the figure 16, we see that, as compared to figure 15, packets now favour 2Mbps link more than 5.5Mbps in case of ETX. For some packets, the effect of interferene sometimes outperforms that of bandwidth. Hence, in case of ETT, the bar of paths taken through 5.5Mbps node is shrinked as compared to previous figure 15. For some packets, though, the effect of larger bandwidth still persists, and they get routed through the 5.5Mbps node.

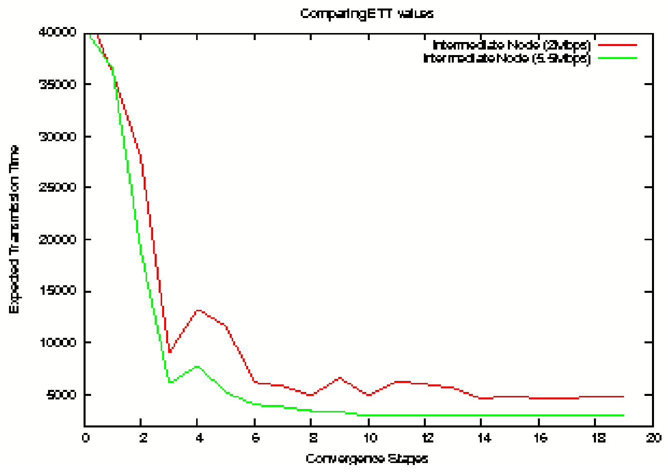

Figure 14: This graph shows the convergence of values of ETT metric for the two intermediate nodes. The node with 5.5M rate has lower ETT as compared to node with 2M rate.

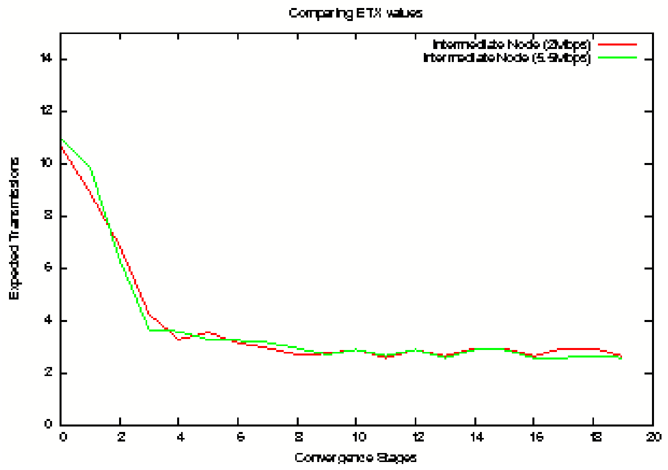

Figure 15: This graph shows the convergence of values of ETT metric for the two intermediate nodes. Both the nodes have same ETX irrespective of their rates.

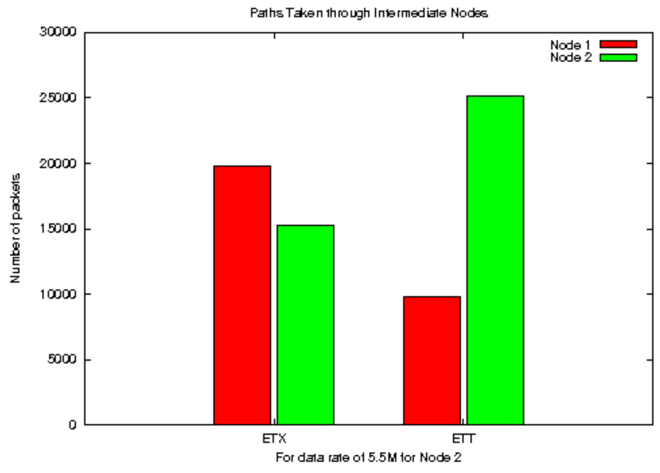

Figure 16: This graph shows Inteference effect on routing metrics. Unless, bandwidth compensates of interference ETT ends up choosing the same hop as ETX.

4 OLSR Protocol and Computation of Routing Metric for a Path

4.1 OLSR:

The Optimized Link State Routing Protocol (OLSR) is developed for mobile ad hoc networks. It operates as a table driven and proactive protocol, thus exchanges topology information with other nodes of the network regularly. The nodes which are selected as a multipoint relay (MPR) by some neighbor nodes announce this information periodically in their control messages. Thereby, a node announces to the network, that it has reachability to the nodes which have selected it as MPR. In route calculation, the MPRs are used to form the route from a given node to any destination in the network. The protocol uses the MPRs to facilitate efficient flooding of control messages in the network. The detailed list can be found at http://hipercom.inria.fr/olsr/. We built on the implementation of UniK University[1] to also take into account the bandwidth of the channel.

4.2 ETX and ETT :

The Expected Transmission Count (ETX) path metric is a simple, proven routing path metric that favors high-capacity, reliable links. The ETX metric is found from the proportion of beacons sent but not received in both directions on a wireless link. ETX is calculated as $1/(p_f * p_b)$,where p_f is forward delivery ratio and p_b backward delivery ratio of the link.The drawbacks of ETX are

1. It's Bidirectional, meaning it will be same for uplink and downlink.
2. It does not consider bandwidth.
3. It does not give preference channel diversity.

ETT of a link is a "bandwidth-adjusted ETX". In other words, the ETX is multiplied by the link bandwidth to obtain the time spent in transmitting the packet. This can be formalized as follows. Let S denote the size of the packet and B the bandwidth (raw data rate) of the link.Then $ETT = ETX * S/B$. To measure bandwidth, each node sends two back-to-back probe packets to each of its neighbors every minute. The first probe packet is small (137 bytes), while the second probe packet is large (1137 bytes). The neighbor measures the time difference between the receipt of the first and the second packet and communicates the value back to the sender. The sender estimates the bandwidth by dividing the size of the second probe packet by the time.

4.3 Metric over the path:

The ETX and ETT values, that are calculated for the links, can be used to find quality of the route by simply summing up the ETX (or ETT) over the links along the path. Then, simple Dijkstra's algorithm can be used to find out the best route. This is precisely done in olsrd.

5 Installation and Configuration

5.1 Soekris board and Voyage installation:

We used net4826 soekris board with following specifications:

- 233/266 Mhz AMD Geode SC1100
- 32-256 Mbyte SDRAM, soldered on board
- 4 Mbit BIOS/BOOT Flash
- Soldered CompactFLASH, 16 Mbyte to 128 Mbyte.
- 1 10/100 Mbit Ethernet ports, RJ-45
- 1 Serial port, DB9.
- Board size 4.0" x 5.2"
- Power using external power supply is 11-56V DC, max 12 Watt

Steps to install voyage:

- To enable the serial port we ran *setserial -g /devS0*. We used terminal emulator program called minicom for this. To configure minicom we edited /etc/minicom/minirc.dfl and set the baudrate to 19200.

- The Preboot eXecution Environment (PXE) was used to install the soekris with voyage. To host the boot server following services should be started:TFTP,DHCP. Voyage comes with the Live CD which can be used for the purpose of hosting such sever. We needed to boot from the CD and used 'root' and 'voyage' as username and password. Then we ran
  /etc/init.d/voyage-pxe start 19200
  command to start the services. The soekris needs to be instructed for the boot from the network interface. We created an interface to soekris using another PC, boot the soekris and then press CTRL-P to enter into BIOS. Then we connected, with the cross crunch cable, soekris to the server. After entering into monitor mode we used the command *boot f0*. Typically, after DHCP allocation, we saw a list of options, of which we used 4 for this procedure.

5.2 Compiling and installing kernel with madwifi drivers:

- We obtained modified madwifi drivers ($version 0.9.3$) which reports the signal strength, silence level, MAC type, MAC subtype, length of the header, sequence number, error check result, rate, source, destination addresses. (We first tried with capturing these parameters using 'radiotap' header with the help of 'libpcap' library, but that did not work out.) Most of the changes can be found in *if_ath.c*. The driver actually maintains a circular buffer of physical layer parameters per packet. The buffer can be read and flushed out using 'proc' filesystem in linux. Everytime, we read the file, some number of recent entries from last read, get read and can be used at user level. The speed of reading is greater that the speed at which the buffer gets full. So we never miss packets.
- To compile the modified drivers we need the kernel source code as well as headers. We downloaded the *2.6.18-5-686* kernel source and compiled using following commands
  make
  make install
  Note this compilation is done on other machine.
- Now we copied *config-<kernel_ver>, System.map-<kernel_ver>,vmlinuz<kernel_ver> and initrd.img<kernel_ver>* from /boot directory and /lib/modules/<kernel_ver> to corresponding directories on the soekris.
- The grub entries (*menu.list* file) were modified to show the new kernel.
- The modified madwifi drivers' files including those in *ath,ath_hal,ath_rate,net802.11* directories were copied onto */lib/modules/<kernel ver>/drivers/net/*
- Running *depmod -e* updates the modules available.
- The *modprobe ath_pci* command can be used to load the drivers.
- The *meas_log* file creation in */proc/net/madwifi/wlan0/* confirms correct operation.
- The interface wlan0 can be configured for adhoc mode as follows
  modprobe ath_pci autocreate=none
  wlanconfig wlan0 create wlandev wifi0 wlanmode adhoc
  iwconfig wlan0 essid iitb1 channel 1
  ifconfig wlan0 up
  ifconfig wlan0 10.6.0.6 netmask 255.255.0.0
  ifconfig wlan0 up

5.3 Running the programs at boot time:

We edited */etc/rc.lacal* file to start various programs like olsrd daemon, a program to read physical layer parameters of packets, client and server programs at the boot time. This is necessary since when we start experiments, it was hard to actually connect serial cable to get interface to execute these programs and especially when deployed outside.

5.4 Scripts:

We used the following scripts in our program

- start_if.sh to configure the interface
- set_params.sh to set all required parameters like bit-rate and tx_power
- main_script.sh to run perticular executables like client, server, program to read the physical layer parameters.

Apart from these the olsrdclient and olsrdserver programs were also run for OLSR experiments.

6 Experiences

6.1 Equipments:

We required 4 net4826 soekris boards for the experiments. Each soekris board can be run on 12V battery or POE adapter. To connect soekris to computer, we need the computer to have serial port and a serial cable. For PoE, we need a cross cable. A straight cable can also be connected to soekris to have wired access to the terminal of the board. We also had a laptop for the experiments which saved lot of our efforts. We could actually test the signal level at perticular spot using laptop. We recommend this exercise for any wireless measurement, which gives you information regarding, at first place whether the remote node is working or not, or what is signal strength am I getting here, or what are the external wifi source in the vicinity. We used 'iwlist scan' command to see the external wifi source. Interfacing soekris to terminal to install operating system, gave us troubles initially. Specifically, you should choose the right version of Voyage which has PXE which we did not in first attempt, thus ended up wondering for some time.

6.2 Experiments :

One should always look for the recieved signal strength value at the receiver location. Also, output of the scripts, that would run at boot time, must be directed to a file and run in background to have shell access once system boots up. Otherwise, if experiment is supposed to run for say 30 minute, you might have to wait closing your hands haplessly. One should also make sure that there is enough space on soekris to store log data. Once we ended up taking 1 hour long experiment, only to find that, there was no space on soekris to store data for this experiment because of previous experiments data storage.

6.3 Configuring minicom:

Setting the right parameters of minicom is very essential since if you fail to do that, you may not be able to see the normal booting sequence and may see a wierd set of characters appearing on the screen.

6.4 Wireless Tools and Radiotap header:

We found that The wireless tools (e.g. iwlist) read the proc file system only after a particular fixed time interval like 100 ms. We wanted per packet low level parameters, thus these tools did not cater our need. Though the Radiotap header of the 802.11 provides the information at per packet level but the some of the attributes like signal level were absent. Before using modified madwifi drivers, we explored this option but could only catch normal 802.11 header information through libpcap library. We could not obtain the required per-packet information. We would further like to explore this stream in detail though.

6.5 IP forwarding:

When we sent packets, the packet carries source and destination address. But if the destination is not directly reachable and if appropriate routing entries are not provided, then one can not expect to receive them at the receiver. This is what happened with us initially. We initially expected the packets to follow the olsrd daemon. But for packets to consult olsrd daemon and take the path through intermediate hops, we had to enable ip forwarding. To enable ip forwarding on the intermediate nodes, one can just type the command "echo 1 > /proc/sys/net/ipv4/ip_forward". This will enable ip forwarding.

7 Lessons Learnt

This project has exposed to us to magical field of wireless and measuring and quantifying unpredictable parameters. We got hands on experience of handling mini computer in the form of soekris board. We also played with madwifi driver and physical layer headers and also with radiotap headers. It has become our belief that taking measurements in wireless networks is non-trivial job, which should be executed with utmost care. There could be many factors affecting the ongoing experiment which one needs to consider while making conclusions.

8 Future work

Given the ground work that we have done, we would further like to test few metrics. ExclusiveETT and MIC (Metric of interference and channel switch) [2] are the metrics which take into account the external interference. WCETT (Weighted Cumulative ETT) actually considers the channel diversity given multi-radio node. We are keen in analysing their performance.

We would also like to deploy a campus wide wireless mesh network. This was one of our primary aims which due to timing constraints, we could not achieve. Such a network would actually be useful in running various applications like video conferencing or real time monitoring.

9 Conclusion

We tested a link for possible reasons of packet error rate and analysed two routing metrics ETT and ETX by setting up a 4 node 2 hop mesh network. We modified 'olsrd' algorithm to realize the ETT metric. We see that there exists a considerable correlation between packet error rate and recieved signal strength. We attribute the high error rate to external interference which we observe through our logs. Thus, in the absence of external interference, with respect of RSSI, we can consider abstracting a link as 'up' or 'down'. We further observe that, with increase in packet size, the probability of packet being corrupted increases. ETT outperforms ETX in terms of delay and throughput given the same networking scenario. In the presence of external interference, ETT sometimes ends up choosing interfering path. Unless, the bandwidth of a path compensates for the interference effect, ETT chooses the path as that of ETX. Through a small set up, we are able to characterize a wireless link and analyse the routing metrics. We plan to cover WCETT and EETT metrics in future. We hope that our study would help, in general, in motivating wireless measurements and routing metrics and can be seen as a ground level reference of experiences with equipments used.

Acknowledgement

We are grateful to Nithin Dara for his timely suggestions and being generous in helping us out. We also thank Advait Mishra and Pradeep Gopulani for cooperating with us in sharing the soekris boards and batteries. We would also like to gratefully acknowledge our course instructor Prof. Purushottam Kulkarni for several useful and insightful discussion we had.

References

[1] Olsrd, an adhoc wireless mesh routing daemon.

[2] Yun Zhu Weirong Jiang, Shuping Liu and Zhiming Zhang. Optimizing routing metrics for large-scale multi-radio mesh networks. In *IEEE journal*, 2007.

[3] Kameswari Chebrolu, Bhaskaran Raman, and Sayandeep Sen. Long-distance 802.11b Links: Performance Measurements and Experience. In *MOBICOM*, 2006.

[4] Kameswari Chebrolu Dattatraya Gokhale, Sayandeep Sen and Bhaskaran Raman. On the feasibility of the link abstraction in (rural) mesh networks. In *INFOCOM '04:The 25th Annual Conference on Computer Communications*, To appear in April 2008.

[5] Roofnet: http://pdos.csail.mit.edu/roofnet/doku.php.

[6] Richard Draves, Jitendra Padhye, and Brian Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 114–128, New York, NY, USA, 2004. ACM.

[7] I. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: A survey, 2005.

[8] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. *SIGCOMM Comput. Commun. Rev.*, 34(4):121–132, 2004.