Capacity of Ad Hoc Wireless Networks

Ashish Agarwal and Chinmoy Dutta
Supervisor: Dr. Dheeraj Sanghi

Department Of Computer Science and Engineering,
Indian Institute of Technology, Kanpur, India
{ashag, chinmoy, dheeraj}@cse.iitk.ac.in

Abstract

Simple simulation topologies like chain and grid of nodes have been used to understand the behaviour of adhoc wireless networks. We study these models in greater detail. We start by considering end-to-end UDP and TCP traffic. We then study the link layer throughput achievable in a chain of nodes in more general scenarios. A detailed analysis is presented to estimate the average link utilization achievable by a chain and is verified by simulations. We then move to more general scenarios involving sources and sinks of data in the network. The maximum and average throughput in such scenarios is studied. We observe that as compared to the intuitive straight line routing of packets from source to destination, multipath routing can give much higher throughput.

We then study network throughput as a function of network load and transmission radius of nodes. The importance of rate control is highlighted. We also show the importance of assigning optimal transmission range values and question the feasibility of high rate UDP traffic in such networks. We finally end by suggesting two changes in the 802.11 protocol. The first one is aimed towards providing buffer congestion avoidance and congestion resolution in the network. It works by identifying useless transmissions in the network and reducing them. The second one tries to increase the link layer capacity by increasing parallelism in spatial spectral reuse.

1 Introduction

Mobile ad hoc networks (MANETS) is the cooperative engagement of a collection of mobile nodes without any requirement for intervention of a centralised coordinating point or centralised infrastructure. The links of the network are wireless and dynamic being purely based on the proximity of one node to another. The network topology is thus arbitrary and may change rapidly and unpredictably. We want ad hoc networks to be highly robust and scalable. Robustness is a much critical factor here compared to the case of wired networks because of the intrinsic nature of wireless networks characterised by lossy links, limited bandwidth and power constraints of the mobile nodes.

Piyush Gupta and P.R. Kumar in [1] gave a striking result that under optimal conditions, the expected throughput(bit-distance product) obtained by n identical nodes, each capable of transmitting at W bits/sec is $\theta(W/\sqrt{n})$. This has implications related to the scalability of the network since the bandwidth available to each node diminishes to zero with n. Further it has been shown that the capacity can scale provided the traffic patterns show locality property. However these results show only theoritical bounds without considering the effects of protocols and other relevant factors. In [3], for example, the authors observed that for a network of 100 nodes, the actual available capacity was about 50 times smaller than the apparent capacity. Various other studies have strengthened the demand for a detailed understanding of network capacity.

We consider ad hoc wireless scenarios with 802.11 protocol at the MAC layer. The study of the *capacity* of such networks has two different aspects . On one hand, there is a need to maximize the achievable throughput. On the other, there is a need to understand exactly how much can one expect from such networks and exactly how these networks can be characterized. There are various factors that need to be taken into account like the channel allocation policy, transmission radius of nodes, buffer scheduling policy, routing protocols, network load, congestion and others.

We start by trying to characterize these networks and to provide some understanding about them. We then suggest some optimizations based upon this understanding. The report is organised as follows. Section 2 presents a brief overview of the MAC 802.11 protocol. Section 3 deals with simple topolgies like chains and isolated sources and sinks. It tries to analyse the average link utilization for the chain and makes some observations about efficient routing of packets. Section 4 discusses the effect of transmission radius and network load on the end-to-end throughput of networks. Section 5 suggests some policies for buffer congetion avoidance. Section 6 provides suggestions about increasing parallelism in the network.

2 MAC 802.11 Overview

Ad hoc networks use the *Distributed Coordination Function* mode of 802.11 MAC protocol [12]. The DCF mode provides a best-effort delivery of data and its access scheme is quite similar to CSMA/CA (Carrier Sense Multiple access/ Collision Avoidance). Note that Collision Detection is not possible because a tranceiver cannot transmit and receive simultaneously. Thus Collision Avoidance is required. Also note that simple CSMA cannot avoid collisions effectively as it suffers from the *Hidden Node Problem*, where two transmitting nodes, too far apart to sense each others' transmissions, transmit to the same receiver who is close to both and thus at the receiver their transmissions collide. Thus a *Virtual Carrier Sensing* is applied using a four-way RTS/CTS/Data/ACK exchange.

In this protocol, Every node maintains a Network Allocation Vector (NAV) which is the remaining busy period of the shared channel. A node that wishes to send a data packet first senses the medium and if it is found idle, sends an RTS (Request To Send) packet to the destination. Otherwise it follows an exponential random backoff algorithm. If the destination believes that the network is idle, it responds with a CTS (Clear To Send) packet. The sender then transmits the data packet, and waits for an ACK (acknowledgement) from the receiver.

RTS and CTS packets include the amount of time the medium will be busy for the remainder of the exchange. Nodes use these times to update their Network Allocation Vector (NAV). The NAV value indicates the amount of time remaining before the network will become available. Upon successful receipt of an CTS packet not addressed to itself, a node updates its NAV to the maximum of the time carried in the CTS frame and its current NAV value. Upon receiving an RTS addressed to itself, a node returns a CTS frame only if its NAV value is zero, otherwise no CTS is sent. Hence a sender will see no CTS if its RTS packet has collided with another transmission at the receiver, or if the receiver's NAV indicates that the network is not available.

A node times out and re-sends the RTS if no CTS is received. 802.11 doubles its backoff window each time a timeout occurs; it resets the backoff to a minimum value after a packet is transmitted successfully or is dropped after reaching maximum retry limit.

3 Simple Topologies

Li et al [2] simulated a chain of nodes with a single end-to-end connection. They showed the effect of chain length and source rate on throughput. In particular it was shown that the chain can achieve a maximum of 1/4 of single hop throughput. The achieved throughput however was about 1/7 of single hop throughput.

In our simulations, the nodes are assumed stationary. Internode distance is approximately equal to the transmission radius unless otherwise stated. MAC protocol used is 802.11 with mandatory RTS-CTS exchange preceding all data transmissions. Link throughput is 2 Mbps. To avoid overheads and uncertainities of routing, we have used fixed routing policy. All the simulations have been done using the ns [13] simulator with CMU wireless extensions [14].

3.1 Single Chain

This section deals with topologies involving a linear chain of nodes. The nodes are static and the internode distance is just enough to allow transmission.

3.1.1 End-to-End Traffic

Li et al [2] argued that in a chain of nodes, where internode distance, transmission radius and interference radius are roughly equal, the maximum achievable throughput is 1/3 of link capacity. This is because the nodes next to the sender and the receiver cannot transmit and hence at most one in every three links can be *active*. In case where interference radius of node is twice the transmission radius, one in every four links can be active and hence the limit is 1/4.

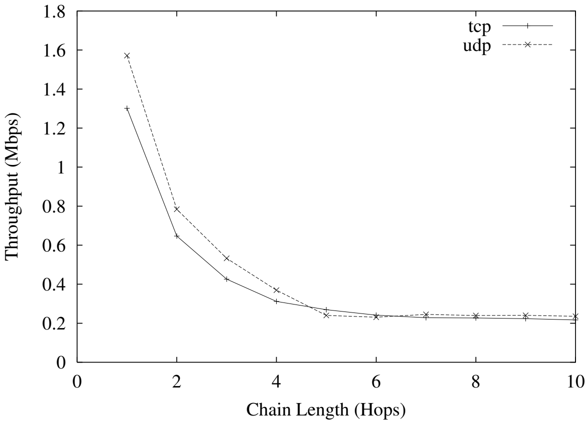

Figure 1: Calculated and simulated network throughput vs chain length

Figure 1 shows the result of simulating a chain of length n. There is a single high rate UDP connection from one end to another. End-to-end throughput is plotted against chain length. One immediately sees that for longer chains, the throughput is 1/7 of link throughput. If the source rate is controlled then one can achieve a higher throughput, even the maximum allowed limit of 1/4. However, once the source rate becomes higher than this maximum achieved limit, the throughput drops sharply to the limit of 1/7.

This seems to indicate that TCP should be able to perform better than UDP because TCP can find the appropriate rate to pump data by its congestion control policy and so can adjust itself to the network. However, simulation of a TCP connection in a chain revealed that TCP performed quite similar to UDP and throughput was 1/7 of the single hop throughput. Figure 1 shows the results of the simulation.

This seeming inefficiency can be explained by understanding bursty nature of TCP. TCP only adjusts the size of its window to reduce congestion and packet drops on the way. However, at the source, the traffic pattern remains bursty. This means that the source retains its aggressive behaviour. Once its buffer has been filled, the behaviour is like a high rate UDP source and the achieved throughput is about 1/7 of the single hop throughput.

3.1.2 Average Link Throughput

In section 3.1.1, we saw that when transmission radius, interference radius and internode distances are roughly equal, the maximum achievabe throughput is 1/3 of single hop throughput as only one in every 3 links can be active. For inteference radius twice of transmission radius, 1 in 4 links can transmit. In general, given values of transmission and interference radius and assuming unidirectional traffic, we can find a value m such that atmost 1 link in m links can be active and thus the maximum achievable throughput is $1/m$ of single hop throughput. This however is a very crude bound and the observed throughput is often much lower. This is because given that a link is active, the $(m+1)$th link need not be active. In fact, any one of the links from $m+1$ to $2m$ may be active provided the nodes have data to transmit. We will use this fact to provide a rough estimate of the number of active links in the chain.

For purpose of analysis, we consider a scenario where each node always has sufficient data to transmit to the next node. We assume that the least distance between two active edges is m and estimate the average number of active links in the chain. Let there be n links X_1 to X_n in the chain where n is sufficiently large. Let $f(n)$ denote the average number of active links in the chain given that X_1 is active. Let $g(n)$ denote the average number of active links in the chain. We now recursively calculate $f(n)$ and $g(n)$.

Suppose that X_1 is active. From our assumption, none of X_2 through X_m can be active. Also exactly one of X_{m+1} to X_{2m} must be active since every node always wants to send data and any node X_i can only be interfered by nodes X_{i+m-1} through X_{i+m-1}. Suppose X_{m+i} is active. Then the total number of active links in the chain is $1+f(n-m-i+1)$. Here we neglect the end effect of the edge, that is, the effect of lesser competition being offered to nodes on the edge, and assume that the number of active links in the chain from X_{m+i} to X_n, given that X_{m+i} is active, remains $f(n-m+i-1)$. If p_i, $1 \leq i \leq n$ denotes the probability that X_i is active, then the average value is $f(n)$ is

$$f(n) = 1 + \sum_{j=1}^{m} (p_j * f(n-m-j+1))$$

We now estimate $g(n)$. Let $q_i, 1 \leq i \leq m$, denote the probability that X_i is active. Then we get the following expression.

$$g(n) = \sum_{i=1}^{m} (q_i * f(n-i+1))$$

We now get a recursive equation for g(n) as follows.

$$
\begin{aligned}
&g(n)\\
&= \sum_{i=1}^{m}(q_i * f(n-i+1))\\
&= \sum_{i=1}^{m} q_i \sum_{j=1}^{m}(p_j * (1 + f(n-i+1-m-j+1)))\\
&= \sum_{i=1}^{m}\sum_{j=1}^{m} q_i * p_j + \sum_{j=1}^{m} p_j \sum_{i=1}^{m}(q_i * f(n-i+1-m-j+1))\\
&= 1 + \sum_{j=1}^{m} p_j * g(n-m-j+1)
\end{aligned}
$$

The expression obtained is thus similar to that of $f(n)$. To solve this recursive equation, we need the value of $g(n)$ at m consecutive points. One solution of $g(n)$ is $an+b$. This will satisfy the equation provided the initial points also lie on this line. In that case, the value of a, assuming $p_i = 1/m$, $1 \le i \le n$, is $2/(3m-1)$. If $g(n)$ converges to this solution, then one can estimate the average link throughput of the chain as $T * g(n)/n \simeq T * a = 2T/(3m-1)$ where T is the maximum single hop throughput, taking into account the various RTS-CTS overheads.

We tested our analysis using simulation. We considered the case when transmission radius, interference radius and internode distance are roughly equal. This corresponds to $m = 3$. We simulated a chain of length n, with each node having a high rate connection with its next neighbour. The length n was varied from 1 to 20. Total network throughput was recorded in each case and average link throughput was computed. We then computed the average number of active links using the recursive equation derived above. The initial values can be easily computed as $g(1) = g(2) = g(3) = 1$, $g(4) = 1.33$ and $g(5) = 1.67$. Single hop throughput T was found to be 1.57 Mbps by simulating an isolated pair of nodes. The average link throughput was calculated as $T * g(n)/n$. Average link throughput was plotted against n for both the simulated and the calculated case. The throughput of a single chain with a single end-to-end 2 Mbps connection was also plotted. Figure 1 shows the results.

From the figure we can see that the simulation setup provides a rough estimate of the end-to-end throughput as well which improves over the initial estimate of 1/3. The calculated graph reasonably approximates the simulated one. The calculated graph coverges towards 0.405 Mbps whereas the simulated one converges to 0.36 Mbps. Besides, the value $2T/(3m-1)$ derived above comes to 0.392 Mbps. This shows that

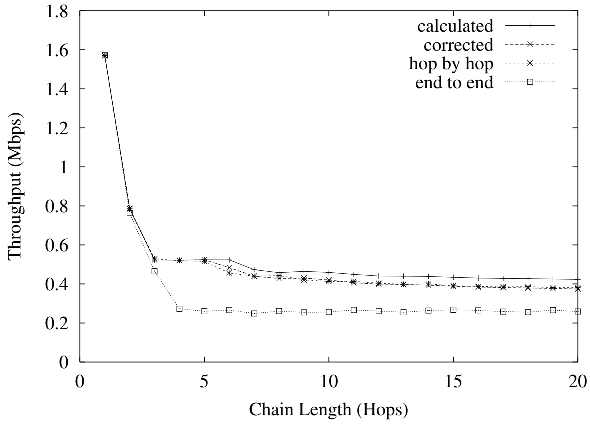

Figure 2: Calculated and simulated network throughput vs chain length

the value of $g(n)$ infacts coverges to the linear solution with parameter a as estimated and is also a good estimation of the simulation setup.

The small discrepancy between the calculated and simulated values can also be explained. In our analysis, we assumed that exactly one of m consecutive edges must be active. However, this is not true in practice even if the nodes always have something to send. This is because of the RTS-CTS mechanism which causes the nodes to backoff in case of collision of control packets. Due to this, all the m consecutive nodes may sometimes remain backed off. This translates to

$$\sum_{j=1}^{m} p_j = 1 - \delta < 1$$
$$\sum_{i=1}^{m} q_i = 1 - \epsilon < 1$$

If $\delta \neq 0$, $g(n)$ is no longer linear. If we neglect δ, then the equation that we get for $g(n)$ is

$$g(n) = 1 - \epsilon + \sum_{j=1}^{m} p_j * g(n-m-j+1)$$

The value of a is then $2T(1-\epsilon)/(3m-1)$. An experimental value of 0.15 for ϵ gave very good results. As can be seen from figure 1 the $\epsilon-$corrected and simulated values are almost identical.

3.2 Intersecting chains

In the previous subsection we studied and analysed the case of a chain of nodes. This is an important first step towards understanding the behavior and functioning of the MAC protocol. Now we present our study of scenarios with intersecting chains (figure 3). Here we had two or four chains with a common end point. Two important cases were identified, with the common node behaving as the source or sink for the connections. The former one models the case of a common source for multiple connections with the central node sending packets in different directions. On the other hand, the latter models the case of a common sink for multiple connections with the central node receiving packets from different directions.

3.2.1 Traffic Sources

In the first set of studies, we simulated three scenarios. In the first (henceforth called multidirectional scenario), there were four udp connections running along four different chains but with a common end-point as the source for all. CBR application at the same rate (2 MB) were running on all of them. In the second (henceforth called the bidirectional scenario) and the third (henceforth called the skewed scenario), we simulated two udp connections along two different chains with a common end point as the source for both of them. We had two CBR applications running at the same rate (2 MB) in the bidirectional scenario, while in the skewed scenario, we ran two CBR applications with diffent rates (2 MB and 0.3 MB). In our simulations, the internode distance, transmission range and interference range were 200, 250 and 550 meters respectively.

Refering to figure 3 , we find that when node 1 is receiving a packet from node 0, it cannot transmit to node 2, since a node cannot receive and transmit simultaneously. Also, 0 to 1 transmission and 2 to 3 transmission cannot go on simultaneously as transmission from 2 would disturb 1. Similarly, simulataneous 0 to 1 and 3 to 4 transmissions is ruled out since node 1 is within interference range of node 3 as well. But the interesting point to note here is that node 0 can transmit to nodes 1', 1" or 1"' when node 3 is transmitting to node 4, since nodes 0, 1', 1" and 1"' are all outside the interference range of node 3. Thus our analysis shows that achievable end-to-end network throughput is 1/3 of the single link capacity. We note that the achievable throughput here is larger than what can be achieved in a single chain. This is due to the possibility of sending successive packets in different directions to avoid mutual interference.

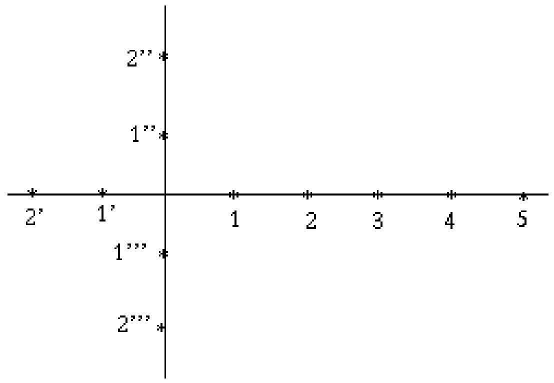

Figure 3: Network topology used for intersecting chain experiments

Figure 4 shows total end-to-end network throughput against chain length for each of the three cases. As can be seen from the figure, the multidirectional scenario performs the best followed by the bidirectional scenario. The skewed scenario performs worst of all. In trying to explain the observed results, we find that there are essentially two factors to consider here. The first is the source scheduling of packets of different directions. The higher the probability of successive packets being sent to different directions, the better it is due to the reasons cited in the preceding paragraph.

The second factor is more close to the behavior of the MAC protocol, namely the MAC scheduling. We know that 802.11, by itself, fails to achieve the optimal scheduling because a node's ability to send depends upon the competition it experiences. Thus, in the single chain case, the source experiences less competition than the nodes down the chain and network throughput equal to 1/7 of the link capacity is actually achieved although the achievable throughput is as high as 1/4 of the link capacity. So it can be said that more the rate control at the source, better it is.

It can be easily seen that the probability for successive packets to be sent in different directions is highest for the multidirectional case, because of the four equally likely directions. This is followed by the bidirectional case, where there are two equally possible directions. This probability is very low for the skewed case, as most of the traffic here is unidirectional. Both for the multidirectional and the bidirectional scenario, there is source rate control as the source now expe-

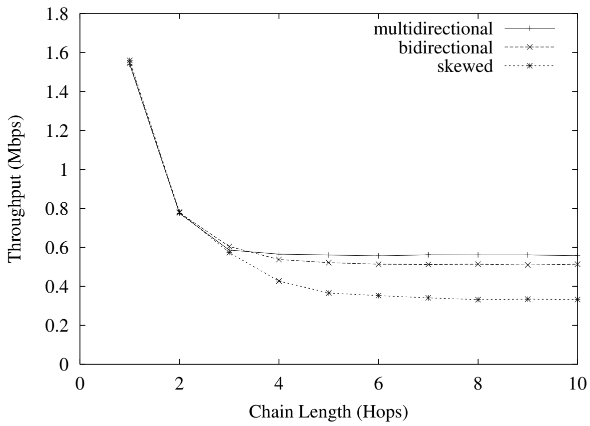

Figure 4: Network throughput vs chain length for the traffic source experiments

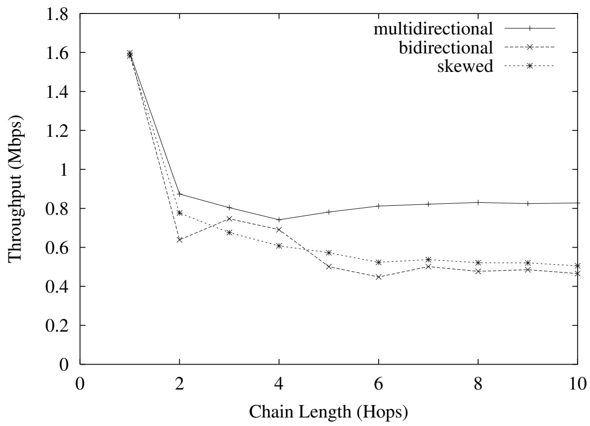

Figure 5: Network throughput vs chain length for the traffic sink experiments

riences similar or more competition than the other nodes. But in the skewed case, the source experiences less competition because the traffic in one direction has a very low rate. Thus these two factors taken together easily explain the observed results. We note that for the multidirectional case the achieved throughput is close to optimal. We also note that for the skewed scenario, the achieved throughput is less than even what can be optimally achieved in a single chain, and this highlights the importance of rate control for proper MAC scheduling in wireless adhoc networks.

3.2.2 Traffic Sinks

Our second set of studies involved simulation of similar scenarios as in the first. But this time the common node was the sink for the different connections. Packets were being sent from different directions towards this common sink. We simulated the multidirectional, bidirectional and skewed scenarios. The achieved total end-to-end network throughput is plotted against chain length in figure 5 .

The observed behavior can easily be explained. As we know the throughput achieved in a single chain is 1/7 of the link capacity. In the bidirectional scenario, the network throughput achieved was 2/7 of the link capacity which is the sum of the combinations from the two chains. For the multidirectional case, we would similarly expect a throughput of 4/7 of the link capacity, but the achieved throughput is slightly less than that. This is again due to the fact that 802.11 couldn't achieve the optimal scheduling. One interesting point to note is the better performance of the skewed scenario over the bidirectional scenario. This is actually due to the fact that performance along a chain with source rate control is better than without source rate control. Thus in the skewed scenario, the connection with low rate outperformed the connection with high rate giving an overall better network throughput compared to the bidirectional case. This again highlights the importance of rate control in wireless adhoc networks.

3.2.3 Implications of the Study

The studies presented here have a crucial implication for wireless adhoc networks. In such networks, packets are forwarded from one node to the other towards the destination. This packet forwarding leads to successive packets of the same connection interfering and competing with each other. This interference, in turn, implies that achieved throughput is significantly lower than the raw link capacity. This is quite different from the case of wireline networks. Consider, for example, a node A sending data to another node B. The path followed by these packets form a chain of nodes. Thus the optimal achievable throughput itself is about 1/4 of the raw link capacity and to make matters worse, the throughput achieved is about 1/7 of that. Our study suggests that we can do significantly better than that. If we allow the packets sent from node A destined for node B to travel along multiple paths, the optimal achievable throughput increases to about 1/3 of the raw link capacity. Moreover, the better source rate control achieved in this case also ensures that throughput is close to this optimal value.

4 Effect of Transmission range and Network Load on Capacity

In this section we present our study of the effect of the transmission range and network load on network throughput. The transmission range is determined by the power level at which the sender transmits. It is easy to see that the total network throughput depends upon the amount of spatial reuse of the spectrum possible, and this parallelism is determined by the floor area acquired by each transmission. Network load determines the amount of congestion in the network which is another important determinant of the network throughput.

4.1 Simulation environment

To study the effect of transmission range on network throughput, we simulated a network of 100 nodes in a 500m by 500m square area. The nodes were placed in a regular 10 x 10 grid, the internode distance being fixed to 50 meters. The nodes were kept static. We then varied the transmission radius which is assumed the same for all the nodes in the network and the number of randomly chosen source destination pairs (this reflects the network load). Both UDP (with constant bit rate application running on top) and TCP (with FTP running on top) were separately studied.

4.2 Observations

We obtained the total end-to-end throughput for each case. The results are presented in figures 6 and 7. The Y-axis shows the transport layer throughput obtained, while the X-axis shows the transmission range, in meters, of the nodes.

4.2.1 TCP

Following the graph for 10 connections we can infer the general trend. The graph first rises, reaches an optimal value and then starts falling. This gives a bell shaped graph. When the radius of transmission becomes equal to 500 meters, the network essentially becomes a single cell, and the throughput achieved is the single cell capacity. This value is more or less same with varying number of connections. The slight variation can be attributed to the increasing amount of time to resolve contention to access the channel with increase in network load.

With very small radius of transmission, the network loses its connectivity and the throughput is small. At the lower end of the transmission range values, there is enough possibility for parallelism in channel use, and hence increasing the load also increases the network

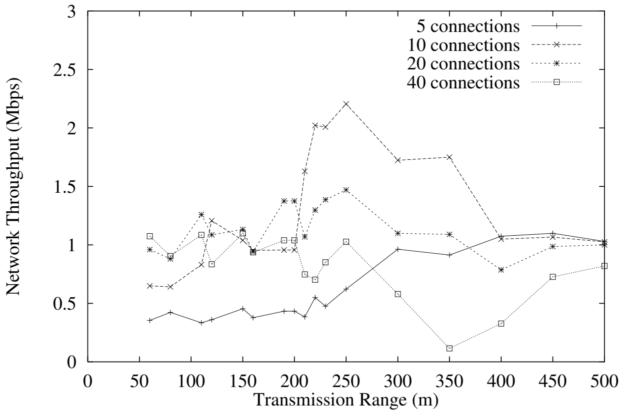

Figure 6: Network throughput vs transmission range for different number of TCP connections.

throughput. Thus we see that for radius of transmission equal to 50 meters, the network throughput increases with number of connections.

As the radius of transmission increases, the number of hops from source to destination for all the connections decreases. The number of hops determines the RTT (Round Trip Time) and also the drop probability of the packets. Thus increasing the radius of transmission increases the network throughput initially. In this region, the network throughput is TCP layer constrained. But our results clearly show the presence of optimal value for radius of transmssion. After reachingthis limit the throughput decreases with further increase in radius of transmission. This is due to the loss of parallelism in spatial channel use. In this region the throughput is MAC layer constrained.

The position of this optimal value for radius of transmission can also be tentatively explained. It can be argued that parallelism is more important in the case of high network load than that of low network load. And this is why the graphs for high network loads start falling much earlier than the graphs for low network loads.

Congestion in the network seems to be another important factor in determining the network throughput that can be achieved. The effects of congestion are more pronounced in scenarios involving high network load and high value of transmission radius. We note that the achieved network throughput is extremely small in these cases (see the dip in 40 connection graph around 350 meters range).

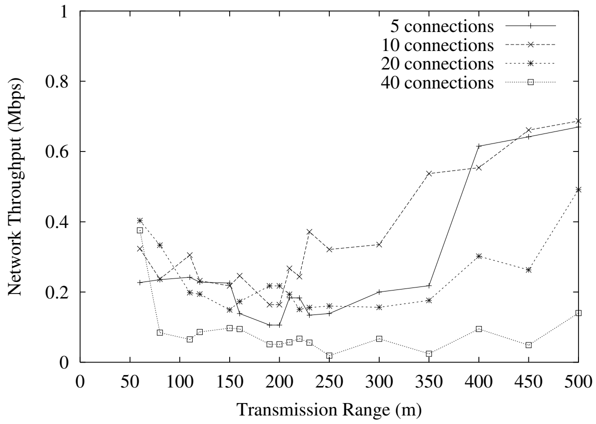

Figure 7: Network throughput vs transmission range for different number of UDP connections.

Finally we note the presence of an optimal value for network load too. While the case of 5 connections is one where the network capacity is underutilised, we find that the overall throughput achieved with 20 or 40 connections is lower than that achieved with 10 connections. This is because with increasing load beyond a limit, the effect of congestion in the network takes over. Also the effect of choosing the right value of transmission radius for this optimal network load seems to be dramatic. As can be seen from the graph, the throughput achieved with 10 connections and transmission range equal to 250 meters is significantly more than that achieved in other case. This strengthens the the need for understanding deeply, the significance of these factors and their effects on the performance of the network.

4.2.2 UDP

The trend shown by the UDP graphs is quite different from that shown by the TCP graphs. In general, the graphs are asymmetric inverted bell-shaped. We find that for a particular network load, as we go from lower values of transmission range to higher values, the throughput first decreases, reaches a minimum and then rises.

The initial decay in achieved throughput can be attributed to the loss of parallelism. We note that parallelism is much more important for the high load cases than for the low load cases. Thus the graph for 40 connections falls very steeply as we increase the transmission range. The effect of parallelism is less observed in the graphs for 5 and 10 connections.

After this initial fall, the throughput starts rising. This can be explained as follows. The interference experienced by the packets of a connection by other connections increases as the length of the connection increases. Close to the intersection points where multiple connections meet, there is high congestion and these junction nodes become bottleneck. It can be easily seen that along a connection the rate of packet arrival successively decreases. Thus at the junction points, the connections with higher rate kill the other connections. But as we go down the path, the rate decreases and this connection in turn gets killed by other connections. This significantly lowers the network throughput because the packets that consume useful bandwidth are dropped later. This effect reduces with reduction in the number of hops, and this is achieved with incresing transmission radius. Thus the throughput increases dramatically. This rise is more pronounced for the low load cases than for the high load cases.

Again we note the presence of an optimal value for the network load. While the network capacity remains underutilised with only 5 connections, having 20 or 40 connections is worse due to the effect of congestion. The performance of 10 connections case is in general better than the other cases.

Our study implies the importance of rate control in the case of UDP. Since UDP does not adaptively control its rate like TCP, its performance can suffer significantly in cases of high network load with high congestion. It also seems that with lower network loads it is better to have higher values of transmission range. On the other hand, if the network load is high, it might be better to choose lower values for transmission range.

4.3 Summary of our observations

We thus conclude that transmission range and the network load are two very important determinants of network throughput. We find the presence of an optimal value for transmission range in the case of TCP. On the other hand, in case of UDP, working at the extremes of transmission range gives better performance.

Network load is a very crucial factor and dramatically effects the network throughput. While very low loads can mean underutilisation of the network capacity, very high rate leads to congestion and the network throughput again suffers. Some intermediate value for network load seems to perform the best and this, in some sense, represents the capacity of the network. It is clear that an optimal choice of these two factors can significantly improve the utilisation of the network capacity.

5 Buffer Congestion Avoidance

We have seen that UDP performs very badly in adhoc networks. Scenarios with high load and substantial node density yield very low throughput. The primary reason for this seems to be buffer congestion. Any attempt to avoid or resolve this congestion can provide significant improvement in network capacity.

5.1 Observed Inefficiencies

A close look at a congested network yields an important observation. Consider a network with a large number of high rate sources. A typical picture of this network reveals that the buffers of most of the nodes in the netwok remain close to full. Especially, the buffers of the sources are almost always full. As these nodes are routers as well, such a congestion leads to many inefficiencies. One significant one is the fact that nodes often transmit packets to other nodes whose buffers are full. This packet meets its obvious fate - being dropped at the other end. This should not be seen as the loss of one packet only. Any packet transmission causes the inhibition of a large number of nodes in the range of transmission. Since nodes are congested, most of these nodes were trying to send their packets. Thus there is a lot of competition in the network for transmitting packets which are to be ultimately dropped. Congestion resolution in the network is also very slow. Once a buffer gets full, it takes very long to clear up, if ever, by transmitting packets to other nodes. All this leads to poor performance in the network.

5.2 Suggested Schemes

We want to avoid packet transmissions in cases when the buffer at the other end is full. One way of doing this is for nodes to regularly advertize their buffer sizes so that other nodes may decide when to send their packets. However there is a problem. In case a source is greedy, it may try to maximize its throughput by ignoring such advertizements and hence defeat the purpose. However, if we don't advertize the buffer size, the source cannot know when the other end is free and hence it cannot make the decision. Therefore it is the destination that should make this decision. A simple way of doing this is for a node to just ignore the RTS requests when its buffer is full. We can extend this to have some kind of buffer congestion avoidance. A node can probabilistically ignore RTS requests when its buffer is close to full. The node thus tries to save its buffer from getting congested.

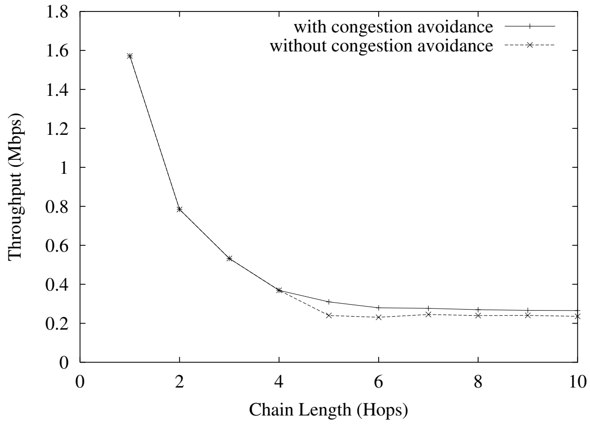

Figure 8: Network throughput vs chain length for a single end-to-end UDP connection with and without congestion avoidance

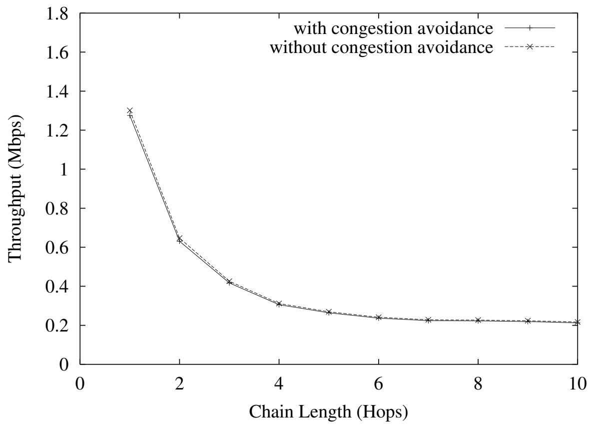

Figure 9: Network throughput vs chain length for a single end-to-end TCP connection with and without congestion avoidance

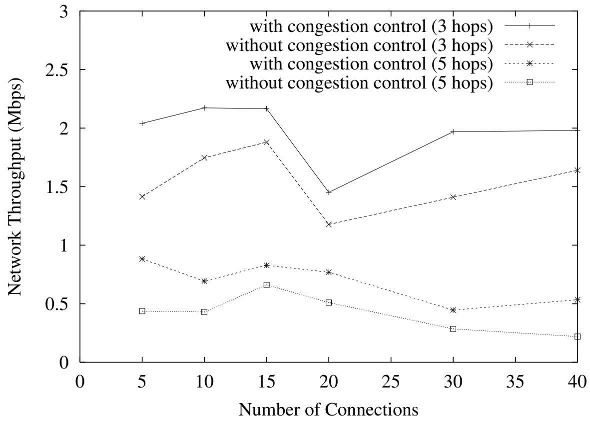

Figure 10: Network throughput vs number of connections for different length of UDP connections in a grid topology. Rate is 500 Kbps.

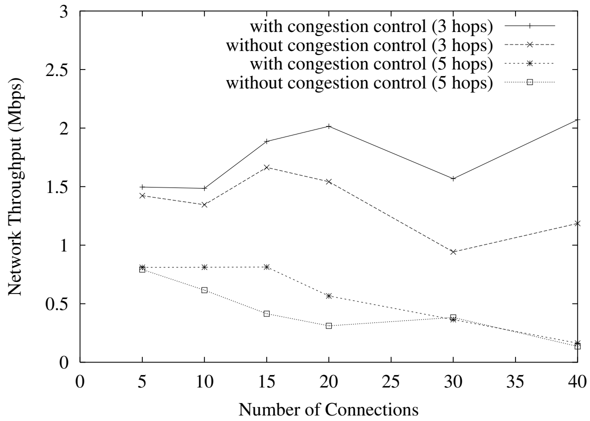

Figure 11: Network throughput vs number of connections for different length of UDP connections in a grid topology. Rate is 1500 Kbps.

Simulation of this simple policy doesn't give very good results. There are many reasons for this. One is that a node may be rejecting packets that were meant for it. This means that in a network, no packet delivery will occur till the congestion has been resolved. Secondly, this also causes control and routing packets to be dropped more vigorously. A simple way to resolve these issues is for the sender node to keep a flag in the RTS to signal that the data is *urgent* or is meant for the receiver node itself. The receiving node may then decide what it wants to do.

However, even with this change, the improvement is not significant. The reason for this is that we have only attempted to do congestion avoidance here and not cared about what happens if the network does get congested. There should be some method of congestion recovery which allows the nodes to empty their full buffers. In the original case, this method was for the nodes to transmit their packets to other nodes in the hope that these packet would finally reach their destination or would get dropped somewhere, and that congestion would finally reduce.

With our policy, we avoid the transmission of packets. Thus the only method for the nodes is to keep on retransmitting their RTS till they decide to drop the packet. Thus all the node remain backed off for most of the time and keep on ignoring the RTS of other nodes. This continues till the buffers become clear again when transmission continues. Thus the duration of congestion recovery is larger and with lesser packet transmissions.

To take care of this problem, we use a simple idea. Instead of ignoring the RTS of other nodes, the congested node sends back the CTS with an indication that it doesn't want to receive a packet. It sets the time in the CTS packet to a small value so that the nodes around it don't wait for the transmission. The other node on receiving the CTS figures out that it is not supposed to transmit and can just drop the packet. This saves it from useless RTS retransmissions and backoffs and also doesn't impose any burden on the neighbouring nodes. The node may actually decide not to drop the packet if it has too little data to send. A threshold buffer size limit can be chosen. The node will drop its packet only if its buffer size is greater than this threshold.

5.3 Results

We implemented our policy in the ns simulator. First we repeated the chain experiments mentioned in section 3.1.1. Figure 8 and 9 show the results of the simulation with UDP and TCP traffic respectively. It can be seen that congestion avoidance provides some performance improvement in case of UDP traffic. In case of TCP the performance was almost identical.

Figures 10, 11 and 12 show simulation results for a grid of static nodes. Internode distance is roughly equal to the transmission range. Transmission radius and interference radius are 250m and 550m respectively to model the standard lucent Wavelan card behaviour. Connections are chosen at random. Length of

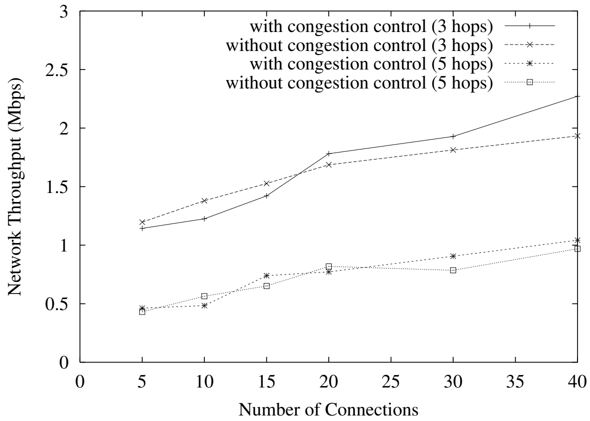

Figure 12: Network throughput vs number of connections for different length of TCP connections in a grid topology.

connection is kept same for all the connections and this value is varied. Number of connections is also varied to allow congestion to set in. Total network throughput is plotted against the number of connections. The different graphs in each figure show the results for different connection lengths for both the original and the new policy. Figure 10 and 11 are for constant bit rate UDP connections with the source rate being 0.5 Mbps and 1.5 Mbps per connection respectively. Figure 12 is for FTP connections over TCP.

The graphs show that congestion avoidance provides significant performance improvement in case of UDP traffic. The increase can be enormous, even by a factor of 2 in some cases. In case of TCP traffic, the performance was somewhat comparable.

6 Improving Link Layer Throughput

In section 4.2, we noted that in case of high radius situations, the throughput suffers mainly due to lack of parallelism. Another factor was the increased RTS resolution time and the greater chances of control packet collisions. However there is one more inefficiency that is especially significant in case of high density or large radius scenarios. In such situations, the network may not utilize the capacity allowed by MAC. We suggest reason why the network may not be able to achieve the maximum possible parallelism.

6.1 Observed Inefficiency

Consider a particular scenario. Node A sends an RTS to node B, but fails to get a CTS from it as its RTS has collided with some packet transmission reachable at node B. Node A thus times out and increases its backoff window. If the packetsize is large this can happen many times leading to large increase in the backoff window of node A. This has two disadvantages. Firstly, node A can remain backed off even when the packet transmission ends. Secondly, node A might have packets to send to other destinations that are currently idle and can communicate with it. Irrespective of this, node A remains backed off. In case of high contention for the shared medium, this idle time can be large and wasteful.

6.2 Suggested Schemes

We tried some ways of solving this problem by allowing useful packet transmissions in cases mentioned. What we desire is that once a node fails to receive a CTS, then instead of retransmitting the RTS, it tries other packets in its queue. However we cannot completely do away with the backoff mechanism as it is necessary to resolve contention. Hence one has to allow the node to search for other idle nodes without too much backoff. At the same time one has to ensure that the node doesn't become too aggressive to disrupt the contention resolution procedure. We have suggested two schemes to solve this problem. Although we have not been able to study and explain them extensively, initial experimentation has indicated that they can give significant performance improvement.

In the first scheme, we use the idea that once a node's backoff window has become so large that its expected idle time would be greater than the expected packet transmission time of the network, then instead of remaining backed off for this duration, the nodes should try to transmit a packet in this time. The hope is that this transmission will proceed independently of the earlier transmission. The node can then continue to poll its neighbour. To implement this policy, a flag bit is needed in the header of each packet. A node tries the head of the queue packet some m number of times increasing the backoff window each time as usual. If it succeeds in getting the CTS, the congestion window size is reset and the node proceeds with the next packet. Otherwise this packet is put some k places from the head of the queue and its flag bit is set to 1. The congestion window size is again reset. When this packet again comes to the head, we find its flag bit set. This time we try it for n number of times (n

> m) increasing the backoff window each time. If we fail to transmit the packet this time too, we drop the packet and set the window size to the minimum value. Priority packets like those containing routing information are directly tried n times. This is done to avoid large delays for these urgent packets. The parameter k can be adjusted to meet the delay requirements. The introduction of backoff prevents the node from getting aggressive and makes contention resolution possible. Excessive blocking and backoff is avoided.

In the second scheme, we make the backoff scheme independent of the node's policy to switch between different packets in its queue. The node will always backoff if it fails to receive a CTS. This ensures that we don't make the node aggressive. Also, if a node fails to get a CTS, it tries another packet in the queue as before.

6.3 Results

We simulated both these schemes and compared performance with the original implementation. The topology was a grid of static nodes with internode distance being 50m. Standard lucent parameters were used to adjust the transmission and interference radii. This produced a high density, high radius situation with a large probability of blocking and backoff. Each node had some probability of transmission which was set by adjusting its rate. Connections were one hop with constant bit rate traffic over UDP. Total link layer throughput was measured.

Results showed significant improvement in the total link layer capacity. The first policy worked quite well and even gave upto 2.5 times the throughput given by the original policy. The second policy also performed better than the actual protocol. However, the gain diminished on reducing the radius of transmission and throughput was similar with all the three policies.

7 Conclusions

Our work has focussed in the following directions - providing an understanding of the network through simulation and analysis of simple scenarios, study of different factors like transmission radius and network load, emphasizing the importance of congestion control in the network and highlighting of reasons causing inadequate spatial reuse of the spectrum.

We have utilized chain and source-sink models to provide an understanding of the capacity of networks. While the former allows a good estimation of link utilization, the latter guides us to the important observation that instead of the single straight line routing of packets, one may attempt multipath routing of pakets. This can result in tremendous increase in performance especially in presence of high rate sources.

We have shown the effect of transmission radius and network load on total throughput. For TCP traffic, results indicated the presence of optimum values for transmission radius and network load for maximum throughput. For UDP, overall performance was very poor in case of high network load. This fact is very crucial, especially with the high demands of multimedia traffic that UDP needs to support. This observation led us to explore the congestion patterns in the network. Consequently, some buffer congestion avoidance and congestion resolution strategies have been suggested which have given significant performance improvement.

Finally, we have suggested some policies for increasing the parallelism in the network by avoiding excessive and wasteful backoffs. Preliminary experimentation has suggested that these policies can provide greater throughput. However, there is also an issue of shared medium allocation and one needs a deeper understanding before a satisfactory solution is found.

Acknowledgements

We would like to sincerely thank our BTP guide, Dr. Dheeraj Sanghi for guiding and helping us all along and giving valuable advices. We would also like to thank Dr. Rajeev Shorey for introducing us to this fascinating problem and Dr. Pravin Bhagawat for his valuable suggestions and encouragement.

References

[1] P. Gupta and P.R. Kumar, *The Capacity of Wireless Networks*, IEEE transactions and Information Theory, March 2000.

[2] J. Li, et al, *Capacity of ad-hoc wireless networks*, Proceedings of ACM MOBICOM 01, July 200

[3] S. Das, C. Perkins and E. Royer, *Performance comparision of two On-demand routing Protocols for Ad hoc networks*, IEEE INFOCOM, March 2000.

[4] Archan Mishra, et al, *Energy Efficiency and throughput for TCP traffic in multi-hop wireless networks*, IEEE INFOCOM, 2002.

[5] Vaduvur Bharghavan, Alan Demers, Scott Shenker, Lixia Zhang, *MACAW: A Media Access Protocol for Wireless LANs*, IEEE SIGCOMM, 1994

[6] M. Grossglauser and David Tse, *Mobility increases the capacity of ad hoc networks*,IEEE INFOCOM, April 2001.

[7] Nishannt Gupta and Samir R. Das, *A Capacity and Utilization Study of Mobile Ad Hoc Networks*, Proceedings, IEEE LCN, 2001.

[8] L. Wang, L. Zhang, O. Yang, Y. Shu and M. Dong, *Multipath Source Routing in Wireless Ad Hoc Networks*, accepted for presentation at CCECE2000, Halifax, May 7-10, 2000

[9] Ogawa, T., Eisuke Kudoh, Suda, H., *Multi-routing schemes for AD-HOC wireless networks*, Circuits and Systems, 2001 ISCAS 2001. The 2001 IEEE International Symposium on , Volume: 4 , 6-9 May 2001 Page(s): 866 -869

[10] Sajal K. Das, A. Mukherjee, S. Bandyopadhyay, Krishna Paul, D. Saha, *Improving Quality-of-Service in Ad hoc Wireless Networks with Adaptive Multi-path Routing*, Accepted in GLOBECOM 2000, San Francisco, California, Nov. 27-Dec 1, 2000.

[11] Roy Leung, Jilei Liu, Edmond Poon, Charles Chan, Baochun Li, *MP-DSR: A QoS-aware Multipath Dynamic Source Routing Protocol for Wireless Ad-hoc Networks,*, 26th IEEE Annual Conference on Local Computer Networks (LCN 2001), Tampa, Florida, November 15-16, 2001

[12] IEEE 802.11 Working Group, *IEEE 802.11 Standards*, http://grouper.ieee.org/groups/802/11/.

[13] Kevin Fall and Kannan Varadhan, *ns Notes and Documentation*, Technical report, UC Berkeley, LBL, USC/LSI, and Xerox PARC, November 1997.

[14] CMU Monarch Group, *CMU monarch extensions to ns*, http://www.monarch.cs.cmu.edu/.