Delay Models for Single Hop and Multihop HIPERLAN

Vinod Khandkar
Sasken Technologies Ltd
Bangalore INDIA
vinodk@sasken.com

D. Manjunath
Deptt of Elecl Engg, IIT Bombay
Powai Mumbai 400 076 INDIA
dmanju@ee.ac.in

Abstract— **Currently two standards are widely used for the channel access protocol in multihop radio networks over small geographic areas – HIPERLAN and IEEE 802.11. In this paper we consider the HIPERLAN protocol and use the CAC overhead analysis of Vukovic [1] to obtain a simple throughput-delay model for a HIPERLAN network. In this model we first compute the mean CAC overhead for various N, the number of contending nodes at the beginning of a channel access cycle, and show that this varies over a small range. Then, using simulation we show that the N has a small range and variance for moderate packet arrivals rates, up to 75% of the capacity. This allows us to provide two approximations for the mean throughput-delay model - an M/D/1 approximation and an M/D/1 with exceptional first service approximation. These are compare against simulation models. Further, we also use these models to study the throughput delay behaviour of multihop HIPERLANs.**

I. Introduction

HIPERLAN [4] is a variable packet length radio LAN standard designed to support asynchronous and isochronous traffic on fixed topology networks. This protocol can also be used in low mobility ad hoc networks. The HIPERLAN standard specifies only Medium Access Control (MAC) and the Physical (PHY) layer protocols. Our interest here is in the MAC layer performance of the HIPERLAN protocol. The MAC layer of HIPERLAN protocol has two sublayers - Channel Access Control (CAC) and the actual MAC. The HIPERLAN MAC sublayer defines features for power conservation, security, multihop routing, etc. The CAC sublayer defines the channel access process. A hierarchically independent channel access mechanism is provided for the isochronous traffic in which traffic of lower priorities do not affect the performance of traffic at higher priorities. We will consider the case of a single priority HIPERLAN network and we will not be accounting for the overheads of the prioritisation process. Thus it is the CAC that determines the performance of a HIPERLAN based radio network. Our goal is to model the performance of the CAC layer of HIPERLAN.

II. Channel Access Control in HIPERLAN

The CAC sublayer in a HIPERLAN is a collision avoidance based protocol called the Elimination-Yield, Non Preemptive Multiple Access (EY-NPMA) protocol. Elimination and Yield are two phases of the contention process on which one among the many contending nodes is chosen to transmit on the channel. The channel access is non preemptive in the sense that only transmission attempts ready at the start of a channel access cycle can contend for the channel in that cycle and no new data transmission attempts are allowed to interfere during that channel access cycle. Channel access occurs in cycles called "channel access cycles". There are two kinds of channel access cycles - channel free channel access (CFCA) and synchronised channel access (SCA), based on the status of the channel at the beginning of the cycle. A channel access cycle consists of transmission of packets, the associated overhead (of elimination of low priority traffic and contention of the equal priority traffic and its structure depends on the transmission mode) and subsequent transmission of acknowledgment.

At the beginning of a channel access cycle, if the node with packet to transmit finds the channel to be idle for a period called the channel free interval (CFI), it transmits the packet. If more than one node has a packet at the end of this interval the channel goes into the synchronised channel access mode. There are three activity phases defined by NPMA for the synchronised channel access mode - Prioritisation Phase, Elimination Phase and Yield Phase. Since we consider a single priority network, we will not discuss the prioritisation phase and assume that its duration is always zero. In our single priority network all nodes with a packet to transmit compete for channel access during the Elimination phase. In this phase the competing nodes generate an elimination burst (EB) period in which a node transmits for a random duration of K_{EB} elimination-slots with the random number chosen according to the probability mass function,

$$P(K_{EB}=k)=\begin{cases} P_E^k(1-P_E) & 0\le k< m_{ES} \\ P_E^{m_{ES}} & k=m_{ES}\end{cases} \qquad (1)$$

where m_{ES} is the maximum number of slots in the elimination burst and P_E, $0<P_E<1.0$, is the probability constant. Nodes having the longest elimination burst are said to have won this phase and will compete for access in the Yield phase. Nodes that selected shorter length EBs will sense the channel to be busy at the end of the transmission of their Elimination Burst and will compete in the next access cycle. In the Yield phase, the competing nodes listen to the channel for a random period, YL slots, and then transmit their packet if the channel has been idle for the entire duration. The period for which the node listens is called the Yield Listen period. The number of yield slots in the the YL interval is selected according to the probability mass function,

$$P[K_{YL}=k]=\frac{1}{(m_{YL}+1)} \qquad (2)$$

where m_{YL} is the maximum number of slots for the YL period. Thus, a contending node survives the yield phase if and only if it senses the channel to be idle during its Yield Listen Interval. Observe that the node(s) with the least yield period survive this phase. As we have seen, elimination and yield phases are

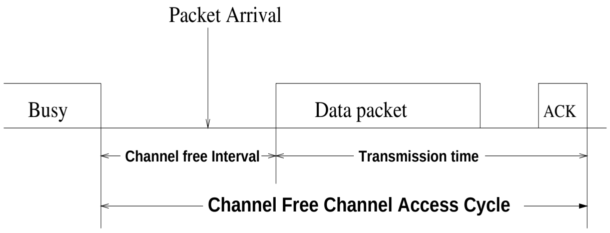

Fig. 1. Channel Free Channel Access

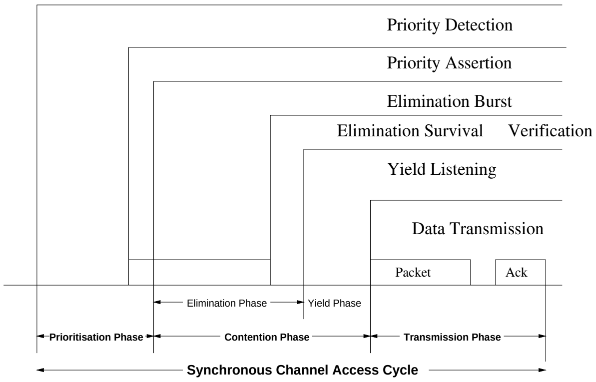

Fig. 2. Synchronised Channel Access Cycle

complementary to each other. Also, observe that in EY-NPMA mechanism at least one node is always allowed to transmit thus making the CAC a work conserving protocol. After a packet transmission, it has to be acknowledged by the destination node and this the last part of the channel access cycle.

Figures 1 and 2 show the structure of the channel-free-channel access and the synchronised channel access cycles.

III. Throughput Delay Model for a Single Hop HIPERLAN

Consider a HIPERLAN consisting of M nodes such that every node is within the radio range of every other node. Each node in the HIPERLAN generates packets according to a Poisson process with rate λ/M. We assume that each node is equally likely to be a destination, i.e., uniform routing of the packets. Since the network is a multiple access network, we can model it as a single server distributed queue with arbitrary service order. Note that the service order is not random because if a node has more than one packet to transmit, only one will contend for transmission. The combined arrival rate into the single server queue that models the networks will be λ. For analysis we will assume fixed length packets and infinite buffers at each node. Further, we will also assume that there is no packet loss due to channel error and that there is no capture effect in the network.

A. The Synchronised Channel Access Cycle

We first analyse a single synchronised channel access cycle. The transmission from a node is successful, if it transmits the longest elimination burst during elimination burst (EB) phase **and** senses the channel for the least amount of time in the yield listen (YL) phase. In the following we characterise these two periods. This analysis is from [1].

We first calculate the probability that in a channel access cycle, exactly one packet succeeds after the Elimination and the Yield $(EB + YL)$ phases. The analysis here is conditioned on there being N nodes with packets to transmit. The probability of successful transmission in the cycle is given by,

$$\begin{aligned}&\Pr[\text{succ}|N\text{nodes contending}] =\\ &\sum_{k=1}^{N} \Pr[\text{1 node wins YL} \mid k \text{ survive EB}, N] \times\\ &\qquad\qquad \times \Pr[k \text{ survive EB} \mid N]\end{aligned}$$

For k nodes out of N to survive the elimination phase, k nodes should transmit elimination burst of same length, say T, and the remaining $(N-k)$ nodes should transmit elimination bursts shorter than T. And for only one node out of k contending nodes to be successful in yield phase, it has to sense the channel for less amount of time than other nodes.

We now characterise the duration of the synchronised channel access period. Specifically, we will characterise the length of the contention overhead in a channel access cycle of the synchronised channel access mode. The CAC overhead comprises of two random variables, namely elimination burst length and yield period. The actual length of the EB period is the maximum of all EB lengths whereas the actual length of YL period is the minimum of all YL periods drawn by each contending node. The joint probability mass function (pmf) of EB and YL period which is also the distribution function of CAC overhead is given by,

$$\begin{aligned}&P[L_{EB} = i, L_{YL} = j|N]\\ &= \sum_{k=1}^{N} P[L_{YL} = j|L_{EB} = i, N_{EB} = k, N] \qquad (3)\\ &P[L_{EB} = i, N_{EB} = k|N]\end{aligned}$$

In deriving the above we use the fact that k nodes get successful in through elimination phase, with length of elimination phase equal to i slots, if and only if all k nodes transmit the elimination burst of length i slots and all other $N-k$ nodes should transmit the elimination burst of length less than i slots. Likewise, for l nodes out of k survived nodes to get successful in yield phase with length of yield phase equal to j slots, all l nodes should sense the channel to be idle for j slot and all other $k-l$ nodes should sense the channel for more than j slots. Eqn. 3 gives the conditional distribution of contention period (CAC overhead) conditioned on the number of active nodes in the beginning of access cycle. Denoting i_{ES} and i_{YL} to be the Elimination and Yield slot lengths respectively, the average length of CAC overhead can now be written as,

$$\begin{aligned}&E[\text{CACOVHD}|N]\\ &= \sum_{i=0}^{M_{ES}} \sum_{j=0}^{M_{YL}} (i_{ES} \cdot i + i_{YL} \cdot j) P[L_{EB} = i, L_{YL} = j|N] \quad (4)\end{aligned}$$

The second moment of the CAC overhead can be similarly written down.

Total nodes = 10	
λ_{norm}	N
0.1	1.057
0.2	1.128
0.3	1.227
0.4	1.380
0.5	1.586
0.6	2.087
0.7	3.097

N	CAC Overhead
1	0.1960
2	0.2104
3	0.2303
4	0.2468
5	0.2598
6	0.2704
8	0.2872
10	0.3006

TABLE I

VARIATION OF CAC OVERHEAD WITH NUMBER OF CONTENDING NODES

We have already found the probability that a node in HIPERLAN will transmit its packet successfully. If more than one node survives after YL phase, then the access cycle repeats. In such a case the total CAC overhead (CAC_{total}) associated with the packet transmitted will be the accumulation of all such CAC overheads with packet failure at their end. We will call this accumulated overhead plus the packet length as service time of the single server distributed queue (SSDQ) representing HIPERLAN. If P_{succ} be the probability that a node will transmit successfully in the synchronised access mode, then the mean of the total CAC overhead, $\overline{CAC_{total}}$ conditioned on N nodes starting the access cycle is given by

$$\overline{CAC_{total}} = \frac{E[\text{CACOVHD}|N]}{P_{succ}} \qquad (5)$$

B. Approximation for the HIPERLAN Network

In this section we will use numerical results from the equations derived above and also simulation results to argue that a single hop HIPERLAN network can be approximated by an $M/D/1$ queue where the service time of a packet is equal to the length of the access cycle and the arrival rate is λ. Our simulation model assumes a network of 10 nodes. The other parameters of the simulation are as follows: P_E=0.5, m_{ES}=12, m_{YL}=9, i_{ES} = 212 bit times, i_{YL} = 168 bit times, CFI =1800 bit times, packet length of 5000 bits. The arrival time and the CAC overhead is normalised to the packet length.

First, consider the variation of N, the number of contending nodes at the beginning of an access cycle as a function of λ, the arrival rate. Table III-B shows this variation from our simulation model of the HIPERLAN. Observe that even at a reasonably high load of 0.7, the average number of contending nodes at the beginning of an access cycle is 2.5. Now consider the variation of the CAC overhead as a function of the number of contending nodes in the access cycle. Table I shows this variation evaluated using Eqn. 5. Observe that the CAC overhead shows no significant variation with N, and for N =1–3, the overhead is in the range 0.20–0.23. Thus for various values of λ, we can assume that the CAC overhead is a constant in every synchronised channel access. Thus the "service time" for a successful packet in the synchronised channel access cycle can be approximated to be a constant equal to 1.23.

A busy period in a HIPERLAN begins with the CFCA mode and the service time of first packet will be 1.0 units. During the rest of the busy period, the channel will be accessed in SCA mode and the service time will be 1.23 units as derived earlier. Thus we have a queue in which the first customer of a busy period receives extraordinary service. Such a queue has been analysed in [5] from where it can be shown that the mean delay (T_s) is given by,

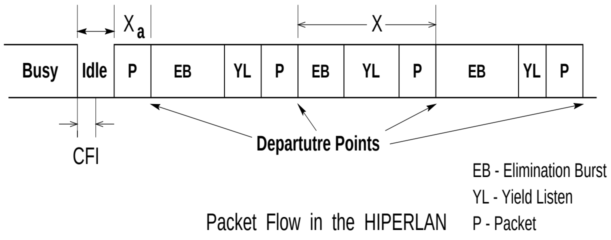

Fig. 3. Packet Flow in HIPERLAN

$$T_s = \frac{\overline{X_a}}{1-\lambda\overline{X}+\lambda\overline{X_a}} + \frac{\lambda\overline{X^2}}{2(1-\lambda\overline{X})} + \frac{\lambda(\overline{X_a^2}-\overline{X^2})}{2(1-\lambda\overline{X}+\lambda\overline{X_a})} \qquad (6)$$

Here X_a is the service time of first packet of the busy period and X is the service time of remaining packets in the busy period. We will neglect the effect of channel free interval on X_a. The channel will be busy for time equal to packet length if packet is transmitted in the channel free access mode and will be equal to packet length plus CAC overhead if it is transmitted in the synchronised access mode (refer Figure 3). We will call this busy period associated with the transmission of every packet as service time ($\hat{X}$) of Single Server Distributed Queue representing HIPERLAN. So $\hat{X}$ is given by,

$$\hat{X} = \begin{cases} 1 & \text{Channel Free Channel Access Mode} \\ 1+\overline{CAC_{total}} & \text{Synchronous Channel Access Mode} \end{cases}$$

So the mean values of X_a and X are,

$$\begin{aligned} \overline{X_a} &= 1; \\ \overline{X} &= 1+\overline{CAC_{total}} \end{aligned} \qquad (7)$$

The second moments of X_a and X can be found to be,

$$\begin{aligned} \overline{E_a^2} &= 1 \\ \overline{E^2} &= 1+2\cdot\overline{CAC_{total}}+\overline{CAC_{total}^2} \end{aligned} \qquad (8)$$

By substituting Eqn. 7 and Eqn.8 in Eqn. 6, we can calculate the delay experienced by the packet in the single hop HIPERLAN.

Alternatively, we could use a simple M/D/1 queueing model with mean service time ($\overline{X}$) dependent on the load. The delay of packet (T_s)in this will be given by,

$$T_s = \frac{\lambda\overline{X^2}}{2(1-\lambda\overline{X})} + \overline{X} \qquad (9)$$

At low loads, since the channel access is predominantly be in CFCA mode, we could use the service time to be equal to 1.0. At moderate loads, since there will be a significant number of SCA cycles, we could assume that the service time is 1.2–1.23.

λ_{norm}	$M/D/1$ $\overline{X}$	T_s	$T_s(Simulation)$	$T_s(M/G/1)$
0.1	1.00	1.055	1.088	1.084
0.2	1.00	1.125	1.201	1.200
0.3	1.10	1.370	1.358	1.364
0.4	1.20	1.661	1.603	1.615
0.5	1.23	2.212	2.193	2.033
0.6	1.23	2.962	2.910	2.857
0.7	1.23	5.039	5.574	5.168

TABLE II

SYSTEM DELAY AS A FUNCTION OF λ

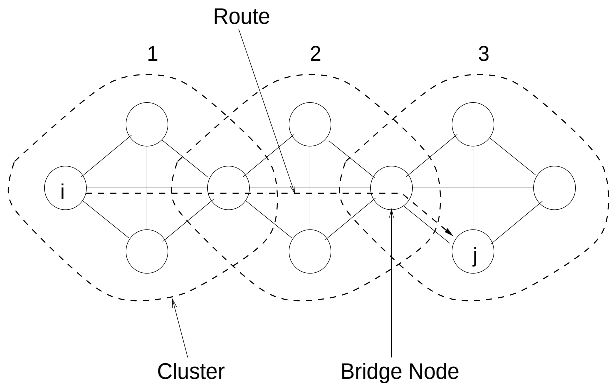

Fig. 4. Ad hoc network partitioned into clusters

The numerical results from these two models are tabulated in Table II. Observe that both the extraordinary first service model and the M/D/1 approximation do reasonably well. At high loads, the channel is always in the SCA mode and the capacity is limited by the CAC overhead.

C. Capacity Analysis

From the results in section III-A, the CAC overhead associated with the transmission of the packet in synchronous channel access mode is a increasing function of number of contending nodes present in the beginning of that access cycle for a given load. As the number of nodes in the network increases to very high value, the number of nodes that contends for the channel always remain very high irrespective of the load. So the CAC overhead also becomes very high leading to very low throughput (S) i.e., as $M \rightarrow \infty$, $S \rightarrow 0$. Consider the HIPERLAN in saturation – at least one of the node always has a packet to transmit and that it always operates in the synchronised channel access mode. It is easy to see that the maximum throughput is obtained when there is only one node waiting to transmit in each access cycle. Therefore, the capacity of the channel, G_{max} is given by

$$G_{max} = \frac{1}{1 + CAC(1)}$$

where $CAC(1)$ is the CAC overhead with $N = 1$ and can be calculated from above. So the capacity of HIPERLAN for a given parameters will be $G_{max} = \frac{1}{1+0.196} = 0.836$.

The various CAC layer parameter used in the analysis are as before.

It has to be noted that the CAC overhead is independent of λ (after removing the condition of N). At very high load the CAC will be same as low load CAC i.e. HIPERLAN works at its capacity at high load.

IV. ANALYSIS OF AD HOC MULTIHOP HIPERLAN NETWORK

Having analysed a single hop HIPERLAN in the previous section, we now extend that analysis to the case of a multihop ad hoc HIPERLAN radio network. We analyse the multihop radio network by considering the network to be partitioned into clusters as shown in Fig.4. The clusters form a connected network in such a way that there is only one node, a bridge node, common to adjacent clusters. The arrival rate of packets is assumed to form a Poisson process with every node generating a fixed length packet at rate θ for every other node in the network. Thus the total packet arrival rate in the networks is $M(M-1)\theta$. A packet having a destination not in its own cluster has to pass through various clusters before received at the destination. Fig.4 shows the typical flow of a packet between a source and a destination. It can be seen that the total traffic on cluster c, $\hat{\Lambda}_c$, will be a sum of the traffic sourced at and destined for that cluster and the traffic that passes through it. Any node belonging to cluster c other than a bridge node will experience a load equal to $\hat{\Lambda}_c$, but as bridge node is connected to multiple clusters it will experience a load equal to load on the cluster in which it is transmitting the packet.

We illustrate our multihop network model using the following example. Consider the network shown in Fig.4 in which a packet from node i of cluster 1 has destination node j in cluster 3. Since the source and destination nodes are not in the same cluster, a packet has to leave its own cluster and has to pass through intermediate clusters before being received at node j in cluster 3. The end to end delay of the packet, $d_{i,j}$ in this case will be,

$$d_{i,j} = \hat{d}(i) + \hat{d}(b_1) + \hat{d}(b_2) \tag{10}$$

where, $\hat{d}(x)$ is the store and forward delay at node x which could be a source or a bridge node.

We generalise this as follows. For a packet from node i to node j, let $R(i,j)$ be the set of bridge nodes that will relay the packet towards the destination.

$$d(i,j) = \hat{d}(i) + \sum_{b \in R(i,j)} \hat{d}(b) \tag{11}$$

As derived in the previous section, the store and forward delay at a transmitting node (which could be either the source or the bridge) is a function of the number of nodes that can interfere with it and also the total load seen by the transmitter. So the terms on the right hand side of the Eqn.11 can be obtained from Eqn. 6 with the appropriate $\lambda = \Lambda(x)$ where $\Lambda(x)$ is the load seen by the transmitting node x. As has been observed earlier, $\Lambda(x)$ is equal to the load on the cluster if x is not a bridge node and is equal to the sum of the loads on the clusters that the bridge connects. The load on cluster c is derived as follows. It is the sum of the internal transmissions and the transit traffic. If N_c is the number of nodes in the cluster, $R(i, j$ is the set of bridge nodes on the path from i to j, B_c is the set of bridge nodes connected to cluster c,

θ	λ_1,λ_4		λ_2,λ_3		T_{e-e}	
	S	A	S	A	S	A
0.0005	0.08	0.08	0.14	0.14	2.73	2.39
0.0010	0.16	0.16	0.28	0.28	2.96	2.76
0.0015	0.24	0.24	0.43	0.43	3.19	3.41
0.0020	0.31	0.32	0.57	0.57	3.53	4.80

TABLE III
END TO END DELAY IN THE NETWORK AS A FUNCTION OF θ (EXAMPLE:1)

θ	λ_1, λ_2,λ_3,λ_4		T_{e-e}	
	S	A	S	A
0.0005	0.09	0.09	1.94	1.82
0.0010	0.18	0.18	2.04	2.00
0.0015	0.27	0.27	2.14	2.27
0.0020	0.37	0.37	2.26	2.65
0.0025	0.46	0.46	2.40	3.28
0.0030	0.55	0.56	2.54	4.41

TABLE IV
END TO END DELAY IN THE NETWORK AS A FUNCTION OF θ (EXAMPLE:2)

$$\hat{\Lambda}_c = \lambda[N_c(N_c-1) + \sum_{b\in B_c} \sum_{i,j: i \text{ or } j \text{ or both} \notin B_c} \mathcal{I}(b \in R(i,j))]$$

where $\mathcal{I}(b \in R(i,j))]$ is the indicator variable that the bridge b is on the route from node i to j. The mean end-to-end delay ($\overline{T_{e-e}}$) of a packet in the network will be

$$\overline{T_{e-e}} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} d(i,j)}{N(N-1)} \tag{12}$$

A. Numerical Examples

In the previous section we have derived the formula for the mean end to end delay of a packet in the general network. Now we will use the same formula to find the mean end to end delay of a packet in the network having special topologies. First we will consider the linear arrangement of clusters forming chain like structure and then the arrangemet of clusters forming loop.

A.1 Example:1

In linear arrangement of clusters every cluster has two bridge nodes associated with it except end clusters. Since the traffic is flowing through these end clusters in only one direction, the total load on these cluster will be minimum. At the same time the traffic on the central cluster will be maximum as traffic going to and from the either side of the cluster has to pass through it.

Table III shows the mean end to end delay and also the variation of traffic on the clusters of the packet in the network as a function of mean traffic rate between any two nodes in the network.

A.2 Example:2

Now consider a topology in which the clusters form a ring rather than the linear arrangement of the previous example. In this topology there are two paths available for a packet to reach its destination. We will assume the shortest path routing in which a packet selects the shortest of two paths available to it. The traffic on each cluster will be same as clusters are connected to form a loop. Table IV shows the analytical and simulation results for the end to end delay in the network and the traffic on each cluster.

Note that at low utilisations, the analytical and simulation results for end to end delay match very well but differ slightly for higher utilisations. It should be noted that the traffic rate into the end cluster is much lower than the intermediate clusters. Also, observe that for a given traffic rate between any two nodes in the network, the clusters in the second example always experience a low load than first one because of increased connectivity in the network which facilitates the multiple path between source and destination.

B. Capacity Analysis

In a multihop HIPERLAN with network partitioned into interconnected clusters, the capacity of the network is limited by the heavily loaded bridge nodes. Consider a randomly connected network of HIPERLAN. The network is still partitioned into the interconnected clusters but clusters are placed arbitrarily. The load on each link in the network will be decided by the routing protocol in operation. Suppose ρ_j is the load on the link j in the network. The average load, λ_i, on the bridge node i is $\lambda_i = \sum_{j\in B_i} \rho_j$, where B_i is the set of all links connected to node i.

V. DISCUSSION

In this paper we have developed simple approximate models for the capacity and delay of single priority HIPERLAN by showing that the CAC overhead is approximately a constant. The results from the analytical model compares well with the simulation model. We have also developed tractable analytical models for the delay and capacity of a multihop HIPERLAN which work reasonably well at low to medium utilisations.

REFERENCES

[1] I. Vukovic, "Hiperlan type1:Performance analysis of the channel access control protocol", *http://www.motorola.com/NSS/r&d/ext/pub.html*.

[2] G. Anastasi, L. Lenzini, E. Mingozzi, "Stability and Performance Analysis Of HIPERLAN", *http://citeseer.nj.nec.com/pdf/260597*

[3] I. Vukovic, "Analysis Of Hidden Node Problem in HIPERLAN", *http://www.motorola.com/NSS/r&d/ext/pub.html*

[4] "Broadband Radio Access Networks (BRAN) High Performance Radio Local Area Network (HIPERLAN) Type 1;Functional specification", *http://www.etsi.org/*, July 1998.

[5] " Advanced Queuing, Variations of the M/G/1 Queue, Vacations, Priorities", Lecture Notes, *http://home.iitk.ac.in/ skb/ee679/ee679.html*