VoIP on Wireless Meshes: Models, Algorithms and Evaluation

Anand Kashyap[1], Samrat Ganguly[2], Samir R. Das[1], Suman Banerjee[3]
[1]Computer Science Department, Stony Brook University, Stony Brook, NY, USA
[2]NEC Laboratories, Princeton, NJ, USA
[3]Department of Computer Science, University of Wisconsin-Madison, Madison, WI, USA

***Abstract*— We study the problem of supporting VoIP calls in a wireless mesh network. Specifically, we propose solutions for call admission control (CAC) and route selection for VoIP calls. Call admission decisions must evaluate how the capacity of the mesh network is utilized by the existing calls. We address this issue via a measurement-based modeling effort to model mutual interference between wireless links. The modeling approach evaluates whether capacity constraints (or, required QoS metrics) will be satisfied if a new call is admitted with a given route. Evaluations with a 6-node 802.11a testbed demonstrate excellent accuracy of the model and thus also the CAC performance.**

We address the issue of route selection by also using a modeling approach that considers models of transmission and interference ranges to develop a polynomial-time algorithm to search for feasible routes. This problem takes exponential time for wireless networks without such modeling. In addition to studying feasibility, we study several routing metrics such as shortest feasible path and maximum residual feasible path. Finally, we develop a new method for routing using call statistics that uses prior calling patterns to avoid potentially critical links. We evaluate the performance of these route selection techniques via extensive simulations and demonstrate the superiority of using max residual feasible path over simply shortest feasible path, and routing using call statistics over max residual feasible path.

I. Introduction

Voice over IP (VoIP) applications have seen tremendous growth in the recent years. This has led to the emergence of VoIP applications, e.g., Skype, and service providers, e.g., Vonage, Packet8, etc. Recently, with the advent of dual interface cellphones with WiFi interfaces and ubiquitous availability of wireless LANs, the VoIP over WLAN is gaining popularity. In typical scenarios, the footprint of each Access Point (AP) in a WLAN is limited to 250 meters outdoors, and up to 100 meters indoors. For providing wide area coverage such as in a shopping mall or campus area, the deployment and maintenance of this wired backplane required for connecting a large number of APs becomes a fairly arduous task. This is where the emerging *wireless mesh networks* [1] can be useful. Mesh networks add routing functionalities to the APs of WLAN, thus eliminating the wired backplane, making them easier to deploy. Because of their potential wide-spread use, it is of paramount importance to study methods to implement VoIP services on wireless mesh networks.

Supporting VoIP over meshes

In this paper, we focus on quality of service (QoS) provisioning issues for supporting VoIP over mesh networks. Specifically, we address two related questions: a) How can we maintain the QoS of VoIP calls over a mesh network and b) How can we improve the capacity of the mesh network in terms of the number of VoIP calls that can be supported? We answer the above questions by solving the *call admission control* (CAC) and *route selection* problems for VoIP calls in the mesh network.

The role of CAC is to determine whether to accept or reject an incoming VoIP call based on the available capacity of the mesh network. CAC is a necessary component of a VoIP service in order to maintain QoS of the ongoing calls while ensuring that calls are not rejected when network capacity is available to accommodate the call. Accuracy of the CAC depends upon how well the mesh network capacity is inferred. This is inherently difficult because of wireless interference: two wireless links in the vicinity interfere to some extent. Interference also leads to MAC protocol inefficiency: two interfering links when active simultaneously often provides less aggregate throughput than when only one of them is active. A result of all these is that any new VoIP call can reduce throughputs (and hence QoS) of many existing calls even without directly sharing any link with them in the chosen route. Thus, call admission decisions must somehow model wireless interferences accurately and must be able to predict the available capacity. This is a hard problem.

Further, routing decisions are closely coupled with admission control. For efficient route selection, one not only has to look for a feasible route (i.e., one that has enough available capacity), but also one must ensure that the choice of routes still leaves enough residual capacity to be able to admit future calls for a given calling pattern statistics. Because of wireless interference, checking for feasibility itself can be computationally intensive. This is because there are exponentially many paths between a source-destination pair, and because of wireless interference each one of them must be checked in its entirety for feasibility. Thus, practical and effective heuristics are desired for the route selection problem.

Contributions

With this background we make the following contributions in this paper.

A. In order to develop an effective call admission decision, we develop a *measurement-based* capacity utilization

model for 802.11-based mesh network. This model provides the current view of every node's available capacity that needs to be met when admitting a new VoIP call. Using a 802.11a-based testbed, we experimentally demonstrate the effectiveness of this modeling approach in making call admission decisions.

B. We address the problem of finding a feasible route for a VoIP call while meeting any capacity constraint. We develop a polynomial time solution that can always find a feasible route if one exists. We show via simulations that by discovering feasible routes, we can increase the call acceptance rate by 20% compared to the traditional shortest path routing using hop count metric.

C. Finally, we demonstrate that if a distribution of calling pattern is known, it is possible to find routes that can improve the VoIP call acceptance rate. We present an algorithm that creates routes by avoiding critical links and results in increasing the acceptance of future calls. We show via simulations that using this routing approach, we can achieve up to 40% increase in call acceptance rate.

II. Related Work

Studies focussing specifically on VoIP over 802.11 have considered the delay and loss characteristics under the PCF and DCF modes [2], [3], [4]. A recent work on VoIP over WLAN [5] presents analytical studies on the number of calls that can be supported in a single hop WLAN. The study reports that increasing the payload per frame increases the number of supported calls. Various methods for improving the performance of VoIP in wireless mesh networks have been proposed in [6], [7]. These methods include using path diversity and packet aggregation. Our work addresses a more challenging problem of determining the capacity of a call along a path in a multi-hop mesh network.

Several models have been proposed to compute the capacity of a wireless network. Bianchi proposed a model for determining the capacity of 802.11 in a single cell [8] which has been followed by other similar models as in [9]. In multi-hop wireless mesh networks with given interference and traffic models, the works in [10], [11], [12] formulate a linear program to solve the joint routing and scheduling problem to maximize the capacity of the network. The work in [13] addresses capacity issues specifically for VoIP calls, but it assumes a TDMA based MAC layer. All these works assume some form of scheduling at the MAC layer that is not available with 802.11. An analytical model to compute the end-to-end throughput capacity of a multi-hop path in 802.11-based network has been proposed in [14]. The capacity models proposed above use a somewhat abstract and idealized model of interference, that assumes that interference is binary and is between link pairs only. Further, interference is assumed to be based on physical distances between the transmitters and receivers, simplified radio propagation models, idealized transmitter and receiver characteristics, and so on. In practice, interference is a complex phenomenon as demonstrated in experimental studies in [15], [16], where practical, measurement-based methods are promoted to estimate the interference between 802.11 links.

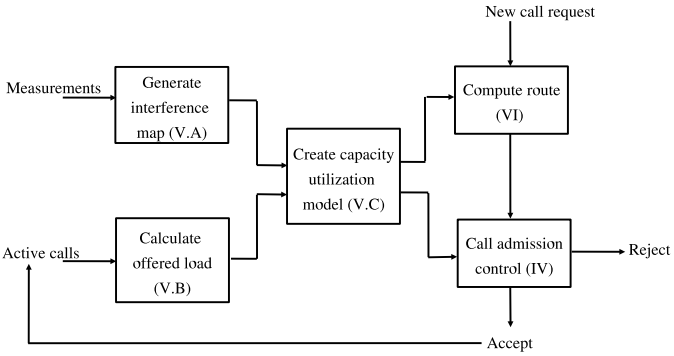

Fig. 1. Architectural block diagram for the approaches developed in the paper. The number in the parantheses in each block indicates the section where it is discussed.

For multihop wireless networks, several modifications to on-demand routing protocols have been proposed to support QoS for real-time applications [17], [18], [19], [20]. In spirit these techniques propose or modify an on-demand routing protocol to support QoS. These techniques cannot guarantee that a feasible path will be found if one exists as the proposed protocols perform only neighborhood checks to verify capacity constraints. Furthermore, the above on-demand protocols require exchange of multihop messages to find the route and result in significant call set up time.

III. Architecture overview

In a typical mesh network deployment for supporting VoIP services, a person can make VoIP calls using WiFi enabled phones. Any "internal call" (calls made between clients inside the network), or "external call" (calls made to or from clients outside the network) goes through a central Session Initiation Protocol (SIP) server. The SIP server authorizes the calls and resolves IP addresses and phone numbers, and then the route is established between the clients.

Figure 1 shows various system components in our architecture. An interference map is created based on the measurements reported by each mesh node. The interference map models the interference for the given mesh topology. A list of active calls along with their respective routes is maintained. From the list, the current offered load at each node is obtained. The capacity utilization model is constructed from per-node traffic load along with the interference map. This model is updated every time a call arrives or departs, or a new measurement report is received. Using this capacity utilization model, a route for the new arriving call is computed, and the call admission decision is made depending on whether a feasible route is found.

Upon call arrival, the call setup must be done within a few seconds. The call setup consists of SIP authorization, route computation, call admission control, and the actual route setup. SIP authorization and route setup are not studied in this paper.

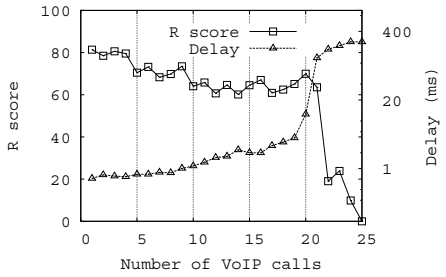

Fig. 2. On a 2-hop path, the graph shows how R score depends on capacity. Once the capacity threshold is reached delay increases due to queuing delay and R score goes down.

The entire system apart from the node-based measurements is deployed on a central server. Such a *centralized* architecture has several benefits: a) a central server can easily interact with the SIP server, b) minimum functionality at mesh nodes paves the way for deployment of commodity hardware and software platforms that is also easy to upgrade. Many commercial WLAN and mesh network platforms [21], [22] already have a central manager node where the CAC and routing software can be deployed. The centralized view is also consistent with one of the deployment scenarios currently being standardized by the CAPWAP working group [23].

IV. VoIP CALL ADMISSION CONTROL

A good call admission control design must look at a VoIP specific QoS metric and understand the effect of the network performance parameters such as delay and loss on this QoS metric. A capacity model can then be built based on this understanding.

VoIP QoS Measure: For G.729a encoder, VoIP sends 50 packets per second of 20 bytes each. The R-factor, or R-score, proposed in [24] is a popularly used QoS metric for VoIP calls. R-score takes into account one-way delay, loss rate, and the type of the encoder. For example, for the G.729a encoder [25],

$$\begin{aligned} R &= 94.2 - 0.024d - 0.11(d-177.3)H(d-177.3) \\ &\quad -11 - 40\log(1+10e), \end{aligned}$$

where

- $d = 25 + d_{jitter\_buffer} + d_{network}$ is the total one-way delay in ms comprising of 25 ms voice encoder delay, delay in the de-jitter buffer (50ms), and network delay;
- $e = e_{network} + (1 - e_{network})e_{jitter}$ is the total loss rate including network and jitter losses;
- $H(x) = 1$ if $x \geq 0$, else 0. R-score should be larger than 70 for acceptable call quality.

Meeting the target VoIP QoS: The quality of a VoIP call is sensitive to delay and loss. The exact dependence is non-linear as given by the R-score formulation above. In order to maintain a good call quality ($R \geq 70$), the one-way delay should be less than 200ms and the packet loss rate along the path should be less than 5%. Packet loss rate can be reduced by choosing a path consisting of links with a high delivery ratio. Packet transmission delays at each hop is typically within a few millisec. However, queueing delays can add up.

From elementary queueing theory, the average queueing delay increases with load, but really becomes large when the average load reaches close to the capacity. We demonstrate this in connection with VoIP and mesh networks using an experiment with a 2-hop segment of a 802.11a-based mesh testbed (testbed described later in Section VII). The 802.11a links are operated at 6 Mbps (and thus in theory each link individually can support 42 calls using calculations of [5]). In the experiments we keep adding VoIP calls to this 2-hop segment and record average R-score and total one-way delay. See Figure 2. Notice that at the point the queueing delay starts increasing abruptly (around 20 calls), R-score also falls rapidly from around 60-70.[1] This experiment demonstrates that the 2-hop segment can support a maximum of about 20 calls,[2] a limit admission control protocols must understand a priori.

Capacity utilization: Based on the above observation, we conclude that in order to meet the QoS for a given set of active calls, the load on each node in the mesh network must meet a capacity bound. In order to ensure the above condition, we determine the capacity utilization at every node for a given set of active calls. Specifically, we solve the following problem: *For a mesh network of n nodes, modeled as a graph $G = (V, E)$, and a set of paths for k active calls, $P = \{p_1, p_2, ..., p_k\}$, find the normalized capacity utilization $c_i, 0 < c_i < 1$, for each node i in the network.* Here, a path is defined as a sequence of nodes. Normalized capacity utilization of a node is the total bits/sec traffic transmitted, received or heard by the node (i.e., the total busy time for the radio medium as perceived by the node), normalized to the nominal link capacity.

Call admission decision: The call admission controller is invoked once route computation is attempted for a new call. The route computation is described in the section VI. The route must ensure that the capacity constraint at all the nodes in the network is satisfied (i.e., $c_i < 1$) with this new call admitted on this route, i.e., the route is *feasible*. If no such route is found, the call is rejected. If a feasible route is found, the call is added to the set of active calls and the capacity utilization is recomputed for future use.

V. MODELING CAPACITY UTILIZATION

Capacity utilization is modeled by first measuring the amount of interference between nodes and creating an interference map. The individual VoIP call's contribution to any node's capacity utilization can be inferred from the interference map.

A. Interference Map

Following our recent work on measurement-based interference modeling [26], we characterize interference between a

[1] We relax the acceptable R-score limit to 60 to account for unavoidable errors caused due to random losses on the wireless link.

[2] Ideally, we should get half of the theoretical capacity of a single link, which is 42 calls.

node pair in terms of the *carrier sense factor* or *csf*. For two nodes x and y, csf_x^y (carrier sense factor of "x" with respect to "y") is defined as the ratio of the actual transmission rate of x when both x and y attempt to transmit at the maximum possible rate, to the transmission rate of x, when x transmits alone at its maximum possible rate. csf_x^y thus denotes the "normalized transmission rate" for x in presence of y.

Typically, csf_x^y takes values between 0.5 and 1. A value of 0.5 implies that x and y are perfectly within carrier sensing range of each other. On the other hand, a value of 1 implies that x and y cannot hear each other. csf between a pair of nodes can be indirectly estimated by using the correlation with the delivery ratio of the link y to x, as well as the signal strength and noise of received packets on the link y to x [16], [26]. This requires just $O(n)$ measurements in a mesh network, where each node takes turn in broadcasting at the maximum possible rate, and the other nodes measure the required parameters, i.e., delivery ratio and signal strength from received packets. csf can also be explicitly measured by doing pairwise experiments for all pairs of nodes in the network. In each experiment, two nodes broadcast at the maximum possible rate, and the number of packets sent out can be measured at each node to get the csf values. This, however, requires $O(n^2)$ measurements. This strategy is somewhat similar to techniques described in [15].

csf estimates (or measurements) between all node pairs define the interference map for the network.

B. Current Offered Load

The offered load (l) at each node is normalized with respect to the link capacity of the node. As pointed out before, the maximum number of calls that can be supported on an 802.11a link at 6Mbps is 42. Thus, for a single two-way call on a link, the offered load on each node is 1/84. For a two-hop call on path A-B-C, the offered load on A and C is 1/84, while the offered load on the middle node is 2/84, because it has to forward traffic in both directions. Thus, for any call with a given path, the offered load on the source and destination of the call is 1/84, while at the intermediate nodes, it is 2/84. For each node i, the offered load due to all active calls can be added to compute the total offered load l_i, $0 < l_i < 1$.

C. Capacity Utilization at Nodes

The normalized capacity utilization at any node has been defined in the previous section. For brevity, we may not always use the term "normalized." Note that if capacity utilization of a node is c_i, it means that the unutilized (or residual) capacity of the node is $1 - c_i$. Capacity utilization can be modeled by the following factors.

Actual traffic load on the node: One of the components of the capacity utilization of a node is the actual traffic the node is transmitting. Actual traffic load (t_i) at a node (i) is greater than the offered load (l_i). It is equal to the offered load plus the extra traffic the node has to transmit due to retransmissions caused by packet collisions. Packets will collide at the receiver if the receiver has another transmitter in its carrier sensing range that is outside the carrier sensing range of the transmitter (hidden terminal phenomenon).[3] Thus, a node j is a hidden terminal for i, if for the receiver k, $csf_i^j = 1$ and $csf_k^j < 1$. The fraction of $j's$ traffic reaching k is $2 \cdot (1 - csf_k^j)$. Let the amount of traffic sent by i to k be deonted as l_{ik}, such that $\sum_k l_{ik} = l_i$. Since the packet transmissions are uniformly distributed over time, the probability of collision of a packet at the receiver can be approximated as $l_{ik} \cdot l_j \cdot 2 \cdot (1 - csf_k^j)$. This amount of traffic must be retransmitted by i. The retransmitted packets may collide again with a smaller probability. So, if we approximate the number of MAC layer retransmissions to just one, and add the extra traffic generated due to all such hidden terminals (j) and all the neighbors k at i, we get the expression, $t_i = \sum_k l_{ik} \cdot (1 + \sum_j l_j \cdot 2 \cdot (1 - csf_k^j))$, if $csf_i^j = 1$ and $csf_k^j < 1$.

Amount of traffic overheard: The amount of traffic the node can overhear is also included in the capacity utilization of the node. This is because the node is unable to transmit during that time, and hence the capacity is utilized for that moment. A node i can hear the traffic of all the nodes j, where $csf_i^j < 1$. If $csf_i^j = 0.5$, the node can listen to all the packets from j. When $0.5 < csf_i^j < 1$, the amount of traffic the node can listen to is $2 \cdot (1 - csf_i^j)$. Thus the amount of traffic overheard at node i, say o_i, can be approximated as $o_i = \sum_{j \neq i} t_j \cdot 2 \cdot (1 - csf_i^j)$.

Consideration of residual capacity: The residual capacity is given by $1 - t_i - o_i$. The residual "usable" capacity is in fact less than this because of possible collision and retransmissions due to hidden terminals. We model this effect indirectly. Assume that the residual capacity is $1 - c_i$. Thus, the node can generate an extra $1 - c_i$ amount of traffic. The extra traffic generated due to retransmissions of this traffic, say r_i, can be given as $r_i =$ maximum of $(1 - c_i) \cdot (1 + \sum_j l_j \cdot 2 \cdot (1 - csf_k^j))$ if $csf_i^j = 1$ and $csf_k^j < 1$, over all neighbors k. This is similar to the method in the first step.

To determine c_i, we add up all these components and equate it to 1. This gives the equation: $t_i + o_i + r_i = 1$. This equation is solved for c_i to get the capacity utilization at each node in the network.

VI. Route computation

For a new call, if a feasible path is found that meets the capacity constraint for all nodes (i.e., $c_i < 1, \forall i$), we can accept the call and use the path to route the call. A question of routing metric arises if there are more than one feasible paths. Conventional link quality based metrics like ETX [27] is not appropriate in this context. The assumption is that only good links are chosen and the interference map-based approach in Section V has already modeled the effect of interference. Instead our goal here is to choose feasible paths that increases the number of supported calls and minimize future call rejections. We first focus on studying the feasibility aspect.

[3]Note that the RTS/CTS is not useful in VoIP. This is because VoIP payloads are small (20 bytes), and relatively RTS/CTS would be a significant overhead.

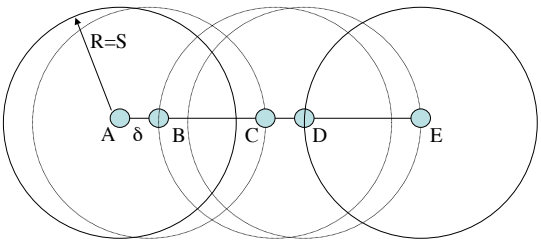

Fig. 3. Transmission range (*R*) = carrier sense range (*S*). In the worst case scenario, two nodes A and E 4 hops apart in a path do not share any node in their carrier sense ranges.

Typically, a feasible route can be constructed by incrementally including links from the network graph and forming a path that connects the source to the destination (note that we are not trying to optimize any path metric here). Any incremental strategy usually results in a fast polynomial time algorithm for discovering a feasible path. However, such an incremental strategy works only if the following condition is true: *when more than one links are determined to be feasible in isolation to carry a certain amount of traffic, they remain feasible when considered together.* In case of wired networks, the above condition is true. Thus, if links are determined feasible, any path in the subgraph containing only the feasible links is also feasible. However, in wireless networks, the above condition is generally not true.

For call admission, we need a fast heuristic to discover a path. We cannot exhaustively explore all paths and check for feasibility, as there are exponentially many of them. We take a different approach. Instead of checking for feasibility on a link basis, we check for feasibility on a sequence of links (path segment) and then string these path segments together. A sequence of links is able to capture the capacity utilization of a larger area that reflects the interference region of the intermediate nodes in the sequence. Essentially, our goal is to find the length of the path segments such that if individual path segments are determined to be feasible, so is the path comprising of these path segments. If we consider a unit disk model, we can show that this length depends upon the ratio of carrier sense range (S) to the transmission range (R). We assume that S and R are circular regions around a node, which define the area where a packet transmission by the node can be sensed and received respectively.

We start with the simplest case, where $S = R$. See Figure 3. It can be argued using geometry that the nodes such as A and E that are 4 hop away must be at least $2R = 2S$ distance away from each other in the worst case, when δ (distance of A-B) tends to 0. If they are any closer than this, the number of hops will decrease as well.

Nodes A and E that are at least 4 hops apart are guaranteed not to share any node that are within both of their carrier sense ranges. Clearly, this implies that if a path segment $S_1 : (AB, BC, CD)$ is feasible and path segment $S_2 : (BC, CD, DE)$ is feasible, so is $S_1 \cup S_2$. Therefore, if one just considers feasible 3 hop path segments and finds a path using these segments, one can discover a feasible path. In order to use this approach, we need to mark all infeasible path segments from the original graph. Any path using the unmarked path segments is then feasible. Checking for feasibility of each path segment can be done in $O(k+1)$ time, where $k+1$ is the number of nodes in the path segment. The number of such k hop segments in the network can be estimated as $O(nd^k)$, where n is the number of nodes in the network and d the average node degree.

Feasible_Route(G, a, b)

/* First compute the edge graph containing only feasible edge pairs. */
Compute G_E, where $V[G_E] = E[G]$ and
$E[G_E] = \{((a,b),(x,y)) : (a,b) \in E[G],$
$(x,y) \in E[G]$, $b = x$, and
$Check\_Feasibility((a,b),(x,y)) = TRUE\}$.
Find set $X \subset V[G_E]$ where $x \in X$ is edge incident from a.
Find set $Y \subset V[G_E]$ where $y \in Y$ is edge incident on b.
Add node s to $V[G_E]$ and edges s to X in $E[G_E]$.
Add node d to $V[G_E]$ and edges Y to d in $E[G_E]$.
Find shortest path P from s to d in G_E.
Convert $P = \{s, (a,b_1), (b_1,b_2)...(b_j,b), d\}$ to
$P' = \{a, b_1, b_2, ...b\}$.
Return P'.

Fig. 4. Fast heuristic algorithm to find a feasible path in the graph G.

We can similarly show that for the case of carrier sense range twice the transmission range (i.e., $S = 2R$), the hop-wise length of the path segments to be considered is 7. We note that with higher length, the marking of the path segments can become a slower process as the number of k-hop path segments grows as $O(nd^k)$. However, it still remains polynomial time.

For fast computation of route, we consider only 2 hop path segments (i.e., edge pairs) as a heuristic in our evaluations. The penalty we pay for this simplicity is that, occasionally routing may determine routes that are actually infeasible. However the call admission controller determines the infeasibility of such routes and rejects them. In experiments (as reported in the next section), we have found that the chance of finding infeasible routes using this 2 hop technique is negligible. We next present the algorithm for computing feasible paths using this approach.

A. Edge-pair Algorithm

From the given original graph $G = (V, E)$, we construct an edge graph $G_E = (V_E, E_E)$ where an edge in G is represented as a unique node in G_E. There is an edge between two nodes (x, y) in G_E only if (x, y) represents a feasible edge pair in the original graph G. For example, if $(a \rightarrow b), (b \rightarrow c)$ are two edges in G forming a feasible edge pair, then the corresponding nodes $x : (a \rightarrow b)$ and $y : (b \rightarrow c)$ in G_E have an edge between them.

In order to find a feasible route between nodes a and b in G, we consider the node set X and Y in G_E such that $x \in X, y \in Y$ represent edges incident from a and to b in G respectively. The set of paths P from $x \in X$ to $y \in Y$ for all

x, y forms the feasible path set from a to b. The algorithm is presented in Figure 4.

In the above algorithm, the feasibility of an edge pair is determined by using the technique described for CAC in Section IV and V. The current offered load on the nodes in the edge pair is increased assuming the edge pair will lie on a path of an incoming call. The increase in offered load depends on the position of the edge pair in an end-to-end path, because end points generate traffic only in one direction and do not relay traffic. The capacity utilization at all nodes is then recomputed and if the capacity constraint at any node is violated, the edge pair is marked infeasible.

In order to select a path in G_E, we add two virtual nodes s and d to V_E with edges from s to X and edges from Y to d. We compute shortest path from s to d in G_E. This gives us the *shortest feasible path* from a to b in original graph G. In order to choose less loaded paths, we can also assign a weight to each link that is, for example, proportional to the current load along that link and use this metric to compute the shortest path. In the evaluation section, we refer to this extension as *max residual feasible path.*

Note that the above algorithm can be extended to 3 hop or longer path segments. For example, in case of 3 hops, we need to first create G_E with 2 hop segments or edge pairs. From G_E, we construct G_E^2 (a graph with edge pairs as vertices and links between feasible edge triplets as edges) by repeating the same process as used to get G_E from G. This technique is quite efficient, because, before considering the feasibility of a 3 hop path segment, we check the feasibility of 2 hop segments and thus reduce the number of 3 hop segments to be checked.

B. Routing using Call Statistics

So far, we have restricted our attention to finding feasible paths efficiently and focused less on which of the many feasible paths that might exist should be selected for routing the incoming call. We indeed have provided two simple methods for selection – shortest feasible path and max residual feasible path. However, a potentially better approach for path selection could be to allow more calls to be supported in future. Such an approach is important to VoIP service providers that are interested in supporting as large a call volume as possible while maintaining call quality. The exact sequence of future call arrivals may be unknown; however, an approach can be designed simply based on long-term call statistics, specified in terms of the probability $p(a, b)$ of a mesh node pair (a, b) to be the source and destination of a new call. Such statistics may be available to the service providers collected via long term measurements. Hot-spot nature of certain mesh routers or regions of mesh networks can generate quite skewed distributions that can be exploited in this approach.

A similar idea called *Minimum interference routing algorithm (MIRA)* [28] has been proposed for traffic engineering work in wireline networks. The basic principle behind MIRA is to define a notion of criticality for a given link and select a route that best avoids critical links. For a given source a and destination b, a link is critical if it belongs to the min-cut [29] between a and b. The level of criticality is determined by $p(a, b)$.

Loosely based on MIRA, we propose a route computation algorithm for VoIP calls over mesh network using call statistics. A weight is assigned to each link based on link criticality. The notion of criticality is explained below. Weights are defined such that a route computation becomes as simple as finding a shortest path on the weighted graph after the feasibility has been ascertained. In order to capture the interference properties in a wireless mesh network, we initialize all link weights to zero, and then develop the following weight assignment rule when a new call arrives between nodes s and d.

- *Assign weights to links based on their criticality* – In the first phase, the set of critical links for each node pair except (s, d) is determined. In wired networks, a critical link for a node pair is one which belongs to any one of the min-cuts for that node pair. All the critical links for a node pair can be found by running the Ford-Fulkerson max-flow algorithm [29] just once. This constitutes a critical link set for a node pair (a, b) and is denoted as $C(a, b)$. Since we have a wireless medium, any link which interferes with the critical link should also be a critical link, because adding traffic on that link reduces the maximum flow between the node pair as well. So, we add all the links which have any node which interferes with any of the nodes in the critical links, $(csf < 1)$, to the critical link set $C(a, b)$.
  Then at this stage, the weight of each link l is given as
  $$w_0(l) = \sum_{(a,b):(a,b)\neq(s,d), l\in C(a,b)} p(a, b).$$
- *Add capacity utilization constraint* – In the next phase, the weight of each link calculated in the first phase is multiplied with the capacity utilization at the link. Capacity utilization of a link is the maximum of the capacity utilization at the nodes of the link. The revised weights are
  $$w_1(l) = w_0(l) \times max(c_u, c_v), \text{where } l = (u, v).$$
- *Make weights non zero* – In the final phase, all links which still have a zero weight are assigned a very small weight, ϵ, such that this weight is not significant enough to make the weight of the link comparable to a critical link, or a link with some capacity utilization. A non zero weight is required because the path weight is the sum of link weights, and smaller paths are desired. We chose the value of 0.001 for our experiments. The final link weights are
  $$w(l) = w_1(l), \text{if } w_1(l) > 0; \text{otherwise } w(l) = \epsilon.$$

With the above link weight assignment, we compute the shortest path on the edge graph (G_E) proposed in the previous subsection. Formal description of the algorithm is given in Figure 5.

Route_Using_Call_Statistics(G, a, b)

Collect call statistics to get $p(x, y)$, $\forall x, y \in V[G]$.
$\forall l = (u, v) \in E[G]$,
$w_0(l) = \sum_{(x,y):(x,y)\neq(a,b), l \in C(x,y)} p(x, y)$, where
$C(x, y)$ is the set of critical links for node pair (x, y).
$w_1(l) = w_0(l) * max(c_u, c_v)$.
$w(l) = w_1(l)$, if $w_1(l) > 0$, ϵ, if $w_1(l) = 0$.
Compute G_E from G as in Figure 4.
Assign weights to links in $E[G_E]$,
$w_E((u, v), (v, w)) = w(u, v) + w(v, w)$.
Find shortest path P in G_E and convert it to P' in G.
Return P'.

Fig. 5. Algorithm for routing using call statistics in the network graph G for a call between nodes a and b.

VII. Performance Evaluation

Here, we present the results of the evaluation of the capacity utilization model and routing. The capacity utilization model and call admission decisions are evaluated on an experimental testbed. Routing is evaluated on the ns-2 simulator [30], as this evaluation requires a large number of nodes.

A. Experimental Testbed

We evaluate our capacity model using a testbed consisting of six identical Dell laptops runing Linux 2.6.15, and using Atheros 802.11a/b/g cards and madwifi driver [31]. The laptops are located at different locations in one floor of the NEC Labs building (150' X 120') to create various topologies. 802.11a is used at 6 Mbps for all experiments. Using 802.11a provides us with shorter links so that interesting topologies can be created within a small area. As indicated before, each such link can support 42 VoIP calls. For each topology, an initial experiment is run to generate the interference map by measuring csf values, and to find the delivery ratio on each link of the network graph. Static routes are setup between nodes using only those links which have a close to 100% delivery ratio. Calls are generated as a Poisson process with a mean rate of λ calls/sec. The average duration of a call is exponentially distributed with the mean rate μ sec. Calls originate between random source-destination pairs. Since there is no waiting time for the calls, the system can be modeled like an $M/M/\infty$ queue, and the average number of calls in the system at any time is given by λ/μ. In our experiments, we fix λ to 0.2 calls/sec, and vary μ to increase the load in the system. Also, we check the R-score of all the active calls for 2 sec intervals and record it for later analysis.

We use three different mesh topologies for the experiments, as shown in Figure 6. The solid lines indicate the good links which are used in routing VoIP calls. "Topology 1" is a linear chain where every node can hear nodes only one-hop away. Thus, a node 2-hop away would be a hidden terminal for a node. "Topology 2" is a dense mesh network, while "Topology 3" is a sparse mesh network.

B. Evaluation of Capacity Utilization Model

We compare our model of creating the capacity utilization graph with a naive model that simply reserves capacity based on the traffic the node generates and the traffic it receives or overhears. In the experiment, we use "Topology 1" (linear chain) that measures the number of calls that can be supported on a path as we increase the path length from 1 to 5. We then predict the number of calls that can be supported using our capacity model as well as the naive capacity model. Table I shows that our model estimates the number of calls that can be supported much better. The supported number of calls is separately determined by observing beyond how many calls the average R-score drops below 70. As the path length increases, the naive model keeps overestimating the capacity because it does not consider collisions due to hidden terminals. Our model is much more accurate and predicts the capacity within 10% of the obtained capacity.

Path length	#Calls predicted by naive capacity model	#Calls predicted by our capacity model	Actual #calls supported
1	42	42	42
2	21	19	18
3	17	14	14
4	14	11	10
5	14	10	9

TABLE I
Number of calls supported in a linear network

C. Evaluation of Call Admission Control

We use the random calling patterns as described in subsection *A* above to evaluate the effectiveness of CAC. All three topologies are considered. 100 calls are used for each experiment with a) CAC enabled, and b) CAC disabled. We increase the load (λ/μ) on the network from 5 calls to 30 calls. This range includes very low load to very high load (much beyond network capacity). The results for the three topologies are shown in Figure 6. For all topologies, median R-scores for all calls and the number of calls that are rejected are plotted against network load (λ/μ).

Note that for all cases, in absence of CAC, the median R-score gets poorer for higher load. However, in the presence of CAC, the R-score is relatively stable independent of load. Also, note that since "Topology 2" is relatively denser, CAC really kicks in at a higher load and rejects less number of calls. Also, for any topology CAC does kick pretty much at the same load where we would have R-score degradation without CAC. It does, however, seem that the CAC is slightly aggressive. The reason for this is that for the current model link delivery fractions are assumed to be ideal (100%). While we indeed chose very good links to route packets in these experiments, we still had to cover for less than perfect link qualities by being slightly conservative in accepting calls.

D. Simulation Setup

The simulation experiments are performed on ns-2 [30] using 802.11b 11Mbps links. The radio propagation model uses the two-ray ground reflection path loss model for the large-scale propagation model, augmented by a small-scale Ricean fading model [32]. We also patched ns-2 with a realistic packet capture model.

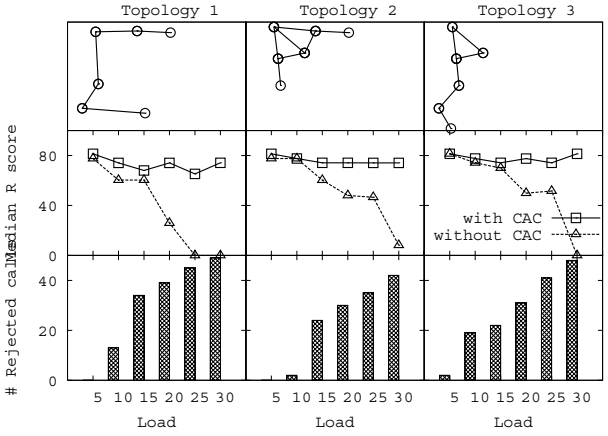

Fig. 6. Evaluation of CAC for various topologies.

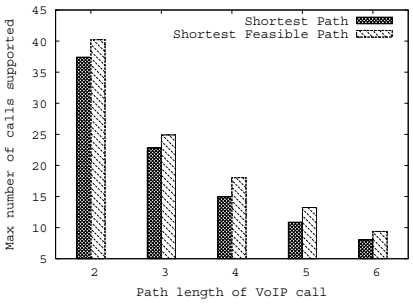

Fig. 7. Evaluating feasibility improves performance.

We use two separate topologies for our evaluation. The first is a 13×13 grid in a 4000×4000 square meter area. The radio models are such that the transmission range is about 250 meters and the carrier sense range is set to 550 meters. Thus, every non-boundary node has four neighbors at a distance of 250 meters in each direction. Each vertical or horizontal edge in the grid represents a link and every node can listen upto its two-hop neighbors. The links have no network losses and the csf between nodes is either 0.5 or 1. The second topology contains 169 nodes randomly placed in a 2000×2000 square meter area. For call pattern, we consider two scenarios: uniform and skewed. For uniform, the source and destination pair for a call is selected with a fixed probability. For skewed scenario, the source destination pair is chosen based on a weight following the zipf distribution.

A centralized program runs various routing algorithms and determines the routes for the calls. These routes are then fed as static routes in the simulator. If a feasible route is not found, the call is rejected. Calls arrive as a Poisson process λ = 1/6 calls/sec, and we vary μ to increase the load in the system. Also, we check the R- score of all the active calls every 5 seconds, and drop the calls for which the R score is less than 70. We run a single long simulation for every scenario, which stops when 2000 calls have been completed.

E. Feasible Route Calculation Evaluation

We first show that using feasible routes, we can support more calls in the network. Figure 7 shows the maximum

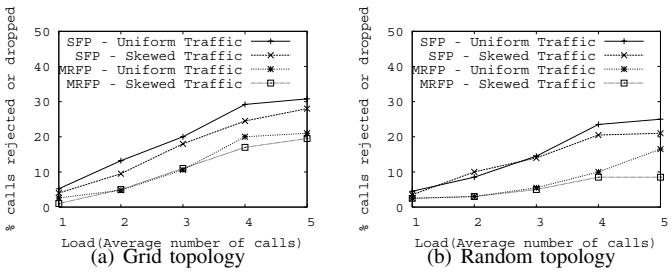

Fig. 8. Comparison of shortest path (SPF) with max residual feasible path (MRFP) for different load.

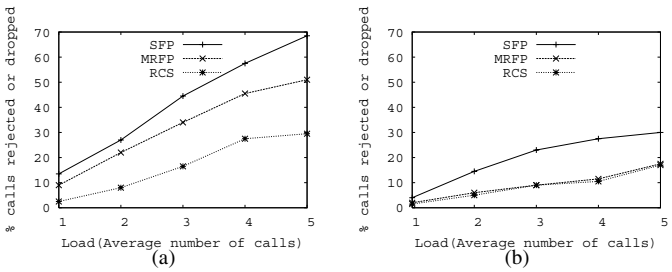

Fig. 9. Comparison of routing using call statistics (RCS) with shortest feasible path (SFP) and max residual feasible path (MRFP) for a) heavily skewed load (2% nodes are callers), b) lightly skewed load (10% nodes are callers) in Random topology.

number of calls that can be supported in the grid as we choose node pairs further away from each other. The metrics used are shortest path (SP) and shortest feasible path (SFP) . We observe that while lesser number of calls can be supported for calls with larger path length, there are more opportunities for finding non-interfering paths, and hence the network can support 10-15% extra calls at saturation.

Next, we evaluate two routing strategies: a) shortest feasible path (SFP) routing and b) max residual feasible path (MRFP) routing. Figure 8(a) and Figure 8(b) show the percentage of calls rejected or dropped for each routing scheme in grid and random topologies. For grid topology, drops or rejections by using MRFP reduces by about 30% for large loads when compared to SFP. For random topology, this factor can be upto about 50%.

F. Evaluating Routing Using Call Statistics

Here, we compare routing using call statistics (RCS) with SFP and MRFP routing. To generate interesting calling patterns where the RCS technique could be beneficial, we assume calls are generated only at hot-spot routers and only a fraction of routers in the network are such hot-spots. This information is provided to the RCS protocol. It is intuitive to see that RCS should perform better when the VoIP traffic is heavily skewed. When the traffic is fairly balanced, all the node pairs have the same weight and if the network topology is also uniform, all the links in the graph get similar weights based on the criticality. RCS degenerates to MRFP routing in such cases.

We present results on a random topology. A grid is not favorable to RCS due to its uniformity. We present results for two cases – 2% and 10% hot-spots, representing a heavily

skewed and a lightly skewed traffic pattern shown in Figures 9(a) and 9(b). As expected, RCS drops lesser number of calls in a heavily skewed traffic pattern, but similar to MRFP routing in lightly skewed traffic.

VIII. Conclusions

We have addressed two important questions in running VoIP on wireless mesh networks. First, maintaining QoS means that call admission control must be performed. However, without any reasonable model of multihop capacity of the network, the admission decisions cannot be taken. We have shown how a simple, measurement-based model can fairly accurately model the available capacity and thus can guide call admission decisions. Second, because of the wireless interference, looking for a feasible route to accommodate an incoming call can be computationally hard. We have simplified this issue by introducing the assumption of the knowledge of the ratio of interference and carrier sensing ranges. This ensures that path segments of constant length can be evaluated separately to determine feasibility in polynomial time. We have also introduced routing metrics such as max residual feasible path and new strategies like routing using call statistics. Both improve performance significantly compared to naive methods.

Our modeling work is general enough that it can be extended for newer architectures such as directional antenna or multi-radio/multi-channel system – something that we will address in our future work. We have not explicitly accounted for data traffic in our evaluation, as our methods can always be used to set aside some amount of capacity for data traffic.

Acknowledgments

Anand Kashyap and Samir Das's work was partially supported by a research award from the NEC Labs, NSF grants CNS-0519734, OISE-0423460, CNS-0308631 and a grant from the SensorCAT center. Suman Banerjee was supported in part by the following NSF grants: CNS-0520152, CNS-0627102, CNS-0639434 and CNS-0627589.

References

[1] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless Mesh Networks: a Survey," *Computer Networks and ISDN Systems*, vol. 47, no. 4, pp. 445–487, 2005.

[2] M. Veeraraghavan, N. Cocker, and T. Moors, "Support of Voice Services in IEEE 802.11 Wireless LANs," in *Proceedings of IEEE INFOCOM*, 2001.

[3] D. Hole and F. Tobagi, "Capacity of an IEEE 802.11b Wireless LAN supporting VoIP," in *Proceedings of IEEE ICC*, 2004.

[4] H.-Y. Wei, K. Kim, A. Kashyap, and S. Ganguly, "On Admission of VoIP Calls over Wireless Mesh Network," in *Proceedings of IEEE ICC*, 2006.

[5] S. Garg and M. Kappes, "Can I add a VoIP call?" in *Proceedings of IEEE ICC*, 2003.

[6] D. Niculescu, S. Ganguly, K. Kim, and R. Izmailov, "Performance of VoIP in a 802.11 Wireless Mesh Network," in *Proc. of IEEE INFOCOM 2006*, Barcelona, 2006.

[7] S. Ganguly, V. Navda, K. Kim, A. Kashyap, D. Niculescu, R. Izmailov, S. Hong, and S. R. Das, "Performance Optimizations for Deploying VoIP Services in Mesh Networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2147–2158, Nov. 2006.

[8] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal of Selected Areas in Communications*, 2000.

[9] Y. C. Tay and K. C. Chua, "A Capacity Analysis for the IEEE 802.11 MAC Protocol," *Wireless Networks*, vol. 7, no. 2, pp. 159–171, 2001.

[10] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of Interference on Multi-hop Wireless Network Performance," in *Proceedings of ACM MobiCom*, 2003.

[11] M. Kodialam and T. Nandagopal, "Characterizing Achievable Rates in Multi-hop Wireless Networks: the Joint Routing and Scheduling Problem," in *Proceedings of ACM MobiCom*, 2003.

[12] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "Algorithmic Aspects of Capacity in Wireless Networks," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 1, pp. 133–144, 2005.

[13] S. Sriram, T. B. Reddy, B. S. Manoj, and C. S. R. Murthy, "On the End-to-end Call Acceptance and the Possibility of Deterministic QoS Guarantees in Ad hoc Wireless Networks," in *Proceedings of MobiHoc*, 2005.

[14] Y. Gao, D.-M. Chiu, and J. C. Lui, "Determining the End-to-end Throughput Capacity in Multi-hop Networks: Methodology and Applications," in *SIGMETRICS '06/Performance '06: Proceedings of the joint international conference on Measurement and modeling of computer systems*, 2006.

[15] J. Padhye, S. Agarwal, V. Padmanabhan, L. Qiu, A. Rao, and B. Zill, "Estimation of Link Interference in Static Multi-hop Wireless Networks," in *Proceedings of IMC*, 2005.

[16] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Measurement-based Models of Delivery and Interference in Static Wireless Networks," in *SIGCOMM*, 2006.

[17] S.-B. Lee, G.-S. Ahn, X. Zhang, and A. T. Campbell, "INSIGNIA: an IP-based Quality of Service Framework for Mobile Ad hoc Networks," *Journal of Parallel and Distributed Computing*, vol. 60, no. 4, pp. 374–406, 2000.

[18] Q. Xue and A. Ganz, "Ad hoc QoS On-demand Routing (AQOR) in Mobile Ad hoc Networks," *Journal of Parallel and Distributed Computing*, vol. 63, no. 2, pp. 154–165, 2003.

[19] L. Chen and W. Heinzelman, "Qos-aware Routing based on Bandwidth Estimation for Mobile Ad hoc Networks," *IEEE Journal of Selected Areas in Communications*, 2005.

[20] Y. Sun, E. Belding-Royer, X. Gao, and J. Kempf, "A Priority-based Distributed Call Admission Protocol for Multi-hop Wireless Ad hoc Networks," UCSB Technical Report, 2004.

[21] Firetide, "Wireless Instant Networks," http://www.firetide.com.

[22] Meru Networks, http://www.merunetworks.com.

[23] B. O'Hara, P. Calhoun, and J. Kempf, "Configuration and Provisioning for Wireless Access Points (CAPWAP)," RFC 3990, February 2005.

[24] R. Cle and J. Rosenbluth, "Voice over IP Performance Monitoring," *ACM Computer Communication Review*, vol. 31, no. 2, April 2001.

[25] I.-T. R. G.729a, "Coding of Speech at 8kbit/s using Conjugate-structure Algebraic-code-excited Linear-prediction (SC-ACELP)," March 1996.

[26] A. Kashyap, S. Ganguly, and S. R. Das, "Characterizing Interference in 802.11-based wireless Mesh Networks," http://www.wings.cs.sunysb.edu/~anand/interference.pdf, 2006.

[27] D. De Couto and D. Aguayo and J. Bicket and R. Morris, "A High-throughput Path metric for Multi-hop Wireless Routing," in *Proc. of ACM MobiCom*, 2003.

[28] M. Kodialam and T. V. Lakshman, "Minimum Interference Routing with Applications to MPLS Traffic Engineering," in *Proc. of IEEE INFOCOM*, 2000.

[29] L. R. Ford and D. R. Fulkerson, "Flows in Networks," *Princeton University Press*, 1962.

[30] K. Fall and K. Varadhan, "ns Notes and Documentation," http://www.isi.edu/nsnam/ns, Nov 1997.

[31] "Multiband Atheros Driver for WiFi (MADWIFI)," http://sourceforge.net/projects/madwifi/.

[32] R. Punnoose, P. Nikitin, and D. Stancil, "Efficient Simulation of Ricean Fading within a Packet Simulator," in *Proceedings of IEEE Vehicular Technology Conference (VTC 2000)*, 2000, pp. 764–767.