Collision-Free Medium Access Based on Traffic Forecasting

Vladislav Petkov
School of Engineering
University of California, Santa Cruz
Email: vladi@soe.ucsc.edu

Katia Obraczka
School of Engineering
University of California, Santa Cruz
Email: katia@soe.ucsc.edu

***Abstract*—In this paper we introduce a novel collision-free, schedule-based medium access control protocol for wireless networks. Our protocol, TRANSFORMA (TRAffic FORecasting Medium Access) uses traffic forecasting to significantly reduce packet delivery delays for delay-sensitive applications while maintaining delivery ratios higher than those of contention-based protocols. TRANSFORMA's novel approach to channel access uses the forecast data rate of each application flow to perform distributed probabilistic channel scheduling.**

We show through simulations that, thanks to its per-application traffic forecasting capabilities, TRANSFORMA yields lower average delays when compared against DYNAMMA, an existing schedule-based MAC, and against 802.11 under high load. TRANSFORMA caters to emerging high data rate, real-time services that will likely be prevalent particularly at the edges of the Internet of the future. Such services which are currently represented by applications such as Skype, Google Talk, and iChat, exhibit traffic characteristics that are fairly predictable and thus well served by TRANSFORMA's traffic forecasting abilities.

We also present an implementation of TRANSFORMA and show some promising preliminary experiment results.

I. Introduction

Prior research has shown that schedule-based medium access control (MAC) approaches provide efficient channel utilization, lend themselves well to reducing energy consumption by eliminating idle-listening, and ensure a deterministic level of service to the users of the medium ([1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11]). However, one drawback of schedule-based MACs is that their average packet delivery delay is higher than that of their contention-based counterparts. This paper describes the TRAffic FORecasting Medium Access protocol, or TRANSFORMA, a novel schedule-based MAC protocol that employs per-application traffic forecasting to significantly reduce delivery delay. Through its traffic forecasting capability, TRANSFORMA provides each data flow transmission opportunities that are: (1) commensurate with the flow's data generation patterns and (2) exhibit adequate temporal distribution.

TRANSFORMA is motivated by the clear upward trend in the number of communication and entertainment applications available over the Internet. This trend is visible today with services like Skype, YouTube, Hulu, and Netflix, to name just a few. Teleconferencing and distance learning applications are also becoming more popular. With the proliferation of smart phones, ambient computing applications that have hitherto only existed in research circles are soon likely to become mainstream as well.

Several studies of residential broadband traffic have shown that the two largest bandwidth consumers are peer-to-peer traffic and HTTP traffic [12],[13]. However, HTTP is no longer a protocol used just to deliver Web pages with text and images. HTTP traffic is made up of many distinct types of application traffic including video and interactive Web applications. Thanks to Adobe Flash and the HTML 5 standard, the multimedia content of the Web is constantly increasing. To receive adequate medium access service, expecting the application (in this case the Web browser) to inform the MAC layer of the traffic characteristics of each of its flows is not realistic. Instead, the MAC layer should detect the properties of each flow transparently and adapt its level of service accordingly.

TRANSFORMA does that by observing an application flow, learning its pattern (if one exists), and "forecasting" the flow's future behavior based on the observed one. In its current implementation, TRANSFORMA's forecaster examines the packet arrival process of each application flow and determines the corresponding per-flow inter-packet arrival times. It will then use this information to establish the flow's medium access schedule. TRANSFORMA operates under the assumption that applications that place more stringent requirements, e.g., higher data rates and delay sensitivity have forecastable network usage patterns. It turns out that many current applications fall under this category: Skype, iChat, and Google Talk are VoIP and video-conferencing applications and naturally exhibit this kind of forecastable behavior. Additionally, non-real-time media streaming applications such as Hulu, Netflix, and iTunes do also.

The simulation results show that given a heterogeneous collection of flows TRANSFORMA can detect the periodicity of each flow and prioritize the channel access in such a way as to keep delays smaller than inter-packet times. The results also show that TRANSFORMA can use data rate forecasts of application flows and provide resources proportionally to every flow even as load increases. As a result, TRANSFORMA's delay performance is superior to DYNAMMA [11] whenever flows are non-homogenous and superior to 802.11 at higher network loads.

The rest of this paper is organized as follows: in Section III we describe the design of TRANSFORMA. In Section IV we evaluate TRANSFORMA's performance through simulation, and Section VI concludes the paper.

II. Related Work

The energy efficiency potential of scheduled medium access is the guiding motivation behind a number of MAC protocols, predominately in the area of wireless sensor networks [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11]. This motivated us to build on our previous work in developing TRAMA [9] and

This work was partially supported by NSF grant CCF-091694.

978-1-4673-1239-4/12/$31.00 ©2012 IEEE

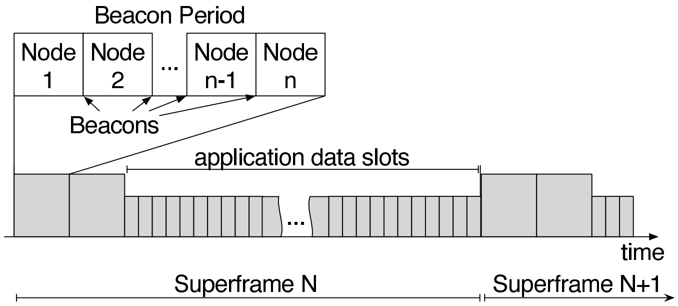


Fig. 1. TRANSFORMA's superframe structure

DYNAMMA [11] by improving the delay performance of the schedule-based approach while retaining its benefits.

Application awareness at the MAC layer has been advocated in the design of Rendezvous [14] and Sticky CSMA/CA [15]. Rendezvous is a MAC protocol that tries to take advantage of the periodic characteristics of "real-time" traffic. It is a contention based MAC in which the inter-arrival time of an application's packet is used to set up periodic reservations. In Sticky CSMA/CA [15], it is assumed that real-time flows are periodic by design or traffic-shaping. Once such flows acquire the wireless medium, they "stick" to a periodic schedule. Their neighbors can detect this schedule and avoid interfering with it.

Despite their application awareness, Rendezvous and Sticky CSMA/CA are contention-based MAC protocols and thus susceptible to the performance limitations that go with that class of protocols. In designing TRANSFORMA, our goal was to have a MAC that is schedule-based and thus possesses all the benefits thereof while leveraging application awareness to improve delay performance.

Although often dismissed as impractical because of their complexity, the existence of a variety of real implementations – such as Soft-TDMAC [16], MadMAC [17], FreeMAC [18], OverlayMAC [19], and now our TRANSFORMA implementation – show that schedule-based protocols can be realized in practice.

III. TRANSFORMA

A. Protocol Overview

TRANSFORMA provides collision-free medium access using a distributed scheduling algorithm based on forecasts of per-flow packet inter-arrival times. Consequently, TRANSFORMA's schedules not only adapt to application-level traffic but also do so in a proactive fashion. In other words, TRANSFORMA tries to anticipate the workload at each node and sets transmission schedules accordingly. This is in contrast to "traditional" scheduled access protocols (e.g., DYNAMMA [11]), which set schedules reactively and thus incur considerable delay. TRANSFORMA's proactive approach to scheduling is accomplished using a traffic forecaster which determines packet inter-arrival times for each application flow at every node. Nodes periodically exchange 2-hop traffic-forecast information. This information is used to select one or more non-conflicting transmitter-receiver pairs for each data transmission slot.

TRANSFORMA assumes that a single channel is shared between control and data packets. The channel is time slotted and slots are grouped into `superframes`, each of which starts with 2 `beacon-periods` (illustrated in Figure 1). During a beacon period every node transmits information about its own flows and those of its 1-hop neighbors. Information propagates one hop per beacon period, thus after two beacon periods 2-hop information reaches all nodes in a 2-hop neighborhood. Subsequent slots in the superframe are used for transmission of application data and are arbitrated by TRANSFORMA's Flow Selection Algorithm (detailed in Section III-D). More details on TRANSFORMA's time slot organization will be presented in Section III-B.

In order to address the specific bandwidth needs of each application and yet maintain network utilization at adequate levels, TRANSFORMA employs a novel approach to medium scheduling based on traffic forecasting. TRANSFORMA's forecaster examines each packet arrival of a flow to adjust its forecast of that flow's data rate. The resulting data rate forecast is then used to provide the corresponding application flow the right amount of slots. In TRANSFORMA, a flow is defined by its source and destination addresses as well as transport layer port numbers. Section III-C goes into more detail on how the forecaster works and how it is integrated into the operation of TRANSFORMA.

TRANSFORMA can be broken down into three main components: the **Control Plane**, the **Traffic Forecaster**, and the **Flow Selection Algorithm**. We will discuss these in detail in Sections III-B, III-C and III-D respectively. The **Control Plane** provides the infrastructure within which the other components of TRANSFORMA operate. It defines the layout of the superframe and the messaging that is used to disseminate traffic forecasts among the 2-hop neighborhood. The job of the **Traffic Forecaster** is to forecast the data rate of each flow and provide this forecast for dissemination to other nodes to be used for scheduling slots. The **Flow Selection Algorithm** is responsible for taking the per-flow forecasts and using them to provide a collision-free arbitration of the medium.

B. TRANSFORMA's Control Plane

TRANSFORMA defines basic rules for accessing the channel and uses one control message: the TRANSFORMA beacon. TRANSFORMA's channel access rules are:

- Nodes must send a beacon during beacon periods to be able to access the channel.
- Nodes must receive all beacons (barring errors) sent during beacon periods to be able to access the channel.
- Nodes must select a color not used by any other node in their 2-hop neighborhood and advertise it in their beacon (explained below).
- Nodes can only send a beacon during their assigned beacon slot in a beacon period.
- Nodes must execute the **Flow Selection Algorithm** during each data slot to determine whether they will receive, transmit, or sleep during that slot.
- Reception begins at the start of a slot.
- Transmission begins some guard time after the start of a slot.

TRANSFORMA uses a time-slotted channel access approach, the structure of which is illustrated in Figure 1. Slots are grouped into superframes, which repeat in time. At the beginning of each superframe are two beacon periods during each of which all nodes get an opportunity to send their beacon.

The purpose of the beacon, pictured in Figure 2, is to facilitate synchronization among nodes as well as topology– and traffic discovery. Like DYNAMMA [11], TRANSFORMA's use of beacons for topology management and synchronization is inspired by and emulates the WiMedia MAC [20]. In order to join the network, a node listens to the channel for at least two superframe durations or until hearing a full beacon period. It picks a random available beacon slot and transmits its beacon in that slot starting in the next beacon period. A node can detect a beacon collision by inspecting the beacons of its 1-hop neighbors, who should mention the node in their beacons. If a node does not appear in its neighbors' beacons, or if there is disagreement among them, then there has been a beacon collision and the node must choose a new beacon slot.

Nodes in TRANSFORMA have colors. Colors are chosen such that no two nodes in the same 2-hop neighborhood can have the same color. This means that nodes of the same color are free to transmit simultaneously without causing collisions and this fact is used in the scheduling algorithm. To keep color assignment optimal, each node must choose the lowest possible index color that is not already used in its two hop neighborhood.

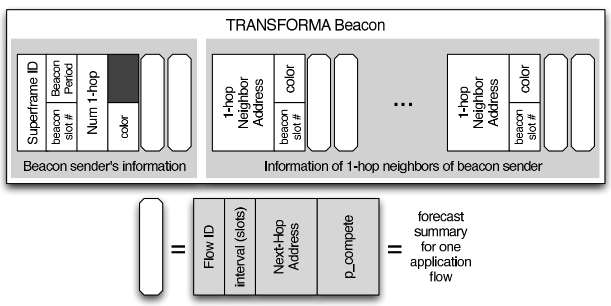


Fig. 2. The TRANSFORMA beacon holds neighborhood information of the sending node and forecast summaries that need to be known by all nodes within 2-hops of the sender.

When the network layer passes an outgoing packet down to TRANSFORMA, the corresponding application flow to which the packet belongs is identified using the packet's source address, source port number, destination address, and destination port number) 4-tuple and the traffic forecaster is notified of the new packet arrival. If the packet does not have a source and destination port (i.e. no TCP or UDP header), it belongs to a special flow that each node can have up to one instance of. The forecaster makes available the most recent data rate forecast for each application flow. Each node maintains a list of active outgoing application flows, $\mathbf{OF}[]$. Flows are removed from the list after a period of inactivity. Each entry in the $\mathbf{OF}[]$ list has these properties: *slots per superframe*, the *source node address*, the *source application port number*, the *destination address*, the *destination port*, the flow's *next hop*, and a unique *flow ID*. For protocol scalability, nodes include a limited number of flow advertisements in each beacon – in TRANSFORMA's current implementation we use 2 flow advertisements per node. For scheduling purposes, flow advertisements include *flow ID*, *slots per sf*, *competition probability* (Section III-D), and *next hop address*. When a node has more than 2 flows in its $\mathbf{OF}[]$ list, they are advertised in a round robin fashion.

Nodes maintain a second list of schedulable flows, $\mathbf{SF}[]$, which are populated by flow advertisements received from neighbors. The **Flow Selection Algorithm** described in Section III-D operates based on information in $\mathbf{SF}[]$.

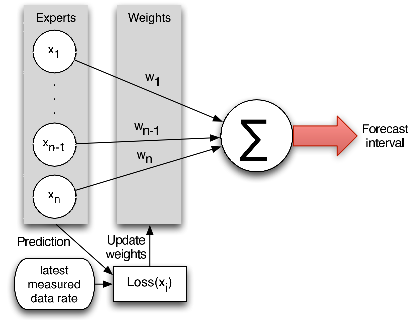


Fig. 3. The **share algorithm** used by TRANSFORMA

C. Traffic Forecaster

Channel access in TRANSFORMA is scheduled based on the disseminated per-flow data rate forecasts. The TRANSFORMA protocol, however, is independent of the particular forecasting algorithm used. In our current implementation we use the well known *share algorithm* [21] which has shown excellent performance in a variety of on-line problems [22], [23], [24], [25]. One direction of future work we plan to explore is to investigate alternate forecasting approaches.

The objective of the *share algorithm* is to pick from a set of experts, $\{x_1, x_2, ..., x_n\}$, the one whose output gives the smallest *loss*. The algorithm maintains a weight for each expert, $\{w_1, w_2, ..., w_n\}$, the value of which determines the impact that each expert's output has on the global output of the algorithm. The *share algorithm* redistributes the weight of experts whose *loss* is high to those experts with low *loss*. As a result, the algorithm quickly adapts to changes in the input (Figure 3).

In TRANSFORMA, the output of both the individual experts and the *share algorithm* represent a data rate forecast expressed in units of slots per superframe. In our implementation of the forecaster, the values of the experts, $\{x_1, x_2, ..., x_n\}$, are distributed linearly between a data rate of one slot per superframe and the maximum forecastable rate and consequently the output of the forecaster also falls within that range. The *loss* function penalizes experts proportionally to the difference between the measured data rate, λ, and their value, x_i.

TRANSFORMA maintains for each application flow its own forecast interval, $\hat{\lambda}$, and experts' weights, $\{w_1, w_2, ..., w_n\}$. When a packet arrives from the network layer, TRANSFORMA matches the packet's identification, i.e., (source address, source port number, destination address, and destination port number) tuple to the application flow it belongs to and performs the following actions:

1) Computes the latest frame's data rate, λ:

$$\tau = t_{\text{(latest packet arrival)}} - t_{\text{(previous packet arrival)}}$$

$$\lambda = \frac{\text{frame size}}{\tau}$$

2) Calculates the *loss* of each expert, x_i:

$$Loss(x_i) = \begin{cases} \left(\frac{0.75(\lambda - x_i)}{\text{max rate}}\right)^2 & \text{if } \lambda \leq x_i \\ \left(\frac{(\lambda - x_i)}{\text{max rate}}\right)^2 & \text{if } \lambda > x_i \end{cases}$$

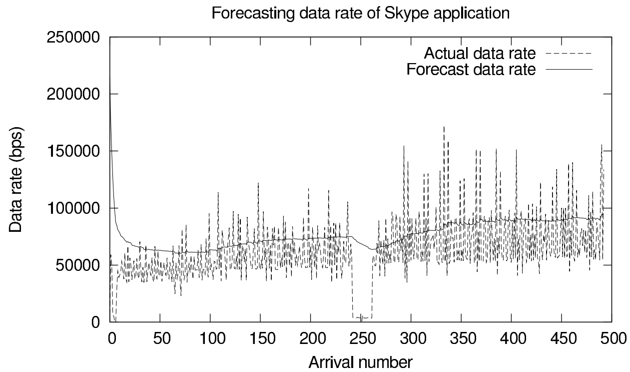


Fig. 4. The data rate forecast compared to the measured instantaneous data rate of a Skype flow.

3) Reduces weights of poorly performing experts:

$$w_i' = w_i e^{-\eta Loss(x_i)}$$

4) Shares some of the remaining weights:

$$pool = \sum_{i=1}^{n} w_i'(1-(1-\alpha)^{Loss(x_i)})$$

$$w_i'' = (1-\alpha)^{Loss(x_i)} w_i' + \frac{1}{n} pool$$

5) Calculates new forecaster output:

$$\hat{\lambda} = \frac{\sum_{i=0}^{n} w_i x_i}{\sum_{i=0}^{n} w_i}$$

The 0.75 constant in the loss function is an empirically determined value whose purpose is to penalize an expert that estimates too low more than an expert that estimates an equal amount too high. The reasoning is that allocating a few extra slots will help to absorb any bursty behavior whereas allocating too few slots can only cause buffers to grow. Figure 4 shows how the forecast data rate compares to the actual data rate of a Skype flow. The Skype trace was captured using `tcpdump` [26]; we then extracted the packet arrival times and packet sizes from the trace. In the plot, the forecast intentionally does not follow the instantaneous data rate of the flow; we can only use one forecast per superframe so it's not beneficial to have a more rapidly changing forecast. The η parameter controls the rate at which the algorithm adapts to the input and a value of 10 has been found empirically to provide a good balance between a smooth yet responsive forecast. The α parameter controls how much of the losing experts' weight is shared with the winners. Setting this parameter too high causes weight to quickly shift from one expert to another and makes the forecast less smooth, but sharing is beneficial to the responsiveness of the forecaster, so there is a balance. We found that a value of 0.04 works well.

D. Flow Selection: Scheduling Medium Access

Once per-flow rate forecast information has been distributed around the network, the next challenge is how to use it to most effectively schedule the medium in a distributed manner – each node uses only local information to make decisions that do not contradict those of its neighbors. The scheduling algorithm in TRANSFORMA is designed with the goal of providing as many slots to each flow as its traffic forecast requires. When the load on the network makes this impossible, the scheduler shares the slots in a fair manner among all competing flows.

1) Flow Selection Algorithm: After a beacon exchange, each node in the network has a list of schedulable flows, $\mathbf{SF}[\,]$, in its 2-hop neighborhood. For each flow, the node knows the *source*, *flow ID*, *rate forecast*, and *competition probability*, P_c. Computation of P_c is discussed below. Every node knows the colors of its 2-hop neighbors and consequently the color of all the flows (a flow has the color of its source).

The first step is to compute for each flow in $\mathbf{SF}[\,]$ two pseudo random numbers based on a hash of the flow's information. The first, *rand1*, represents a probability and is a number in the range $[0,1]$. The second, *rand2*, represents a ranking and is an integer in the range $[0, 2^{32}-1]$.

$$\text{rand1} = \text{hash1}(\#_{sf} \oplus \#_{slot} \oplus \text{flow color} \oplus \text{flow ID} \oplus \text{seed})$$

$$\text{rand2} = \text{hash2}(\#_{sf} \oplus \#_{slot} \oplus \text{flow color} \oplus \text{flow ID} \oplus \text{seed})$$

As the hashes are being computed, any flow with $P_c \geq$ rand1 is placed in the set of *competing flows*, $\mathbf{CF}[\,]$.

The second step is to select from $\mathbf{CF}[\,]$ the flow(s) with the largest *rand2*. All of these flows must have the same color. If not, only those with the smallest color index are kept. The coloring scheme ensures that no two nodes in the same 2-hop neighborhood will have the same color, therefore the transmitters of these flows can safely transmit concurrently without causing collisions. If the node running this instance of the algorithm is the sender or receiver of a winning flow, it puts its radio in transmit or receive respectively. Otherwise it can sleep for the duration of the slot.

2) Computation of Competing Probability: We express the likelihood of a flow, f_i, winning a slot using the following equation:

$$P(f_i \text{ wins}) = P(f_i \text{ competes}) \cdot P(f_i \text{ wins} \mid f_i \text{ competes}) \tag{1}$$

The goal is to compute the value of $P(f_i \text{ competes})$ that would give us a $P(f_i \text{ wins})$ which corresponds to the forecast data rate for the flow. This value is dependent on all the flows that are trying to share the medium.

If we define the random variable $\mathbf{X}$ to represent the number of flows competing for the slot, the conditional probability on the right side of Equation 1 can be expressed as

$$P(f_i \text{ wins}|f_i \text{ competes}) = \frac{1}{E[X|\ f_i \text{ competes}]} \tag{2}$$

The random variable $\mathbf{X}$ can be expressed as a sum of random variables, $\mathbf{I}_i$, each representing the contribution of flow f_i to $\mathbf{X}$.

$$E[I_j|f_i \text{ competes}] = \begin{cases} P(f_i \text{ competes}) & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} \tag{3}$$

If there are n flows, knowing $E[X|f_i \text{ competes}] = \sum_{j=1}^{n} E[I_j|f_i \text{ competes}]$, Equation 2 becomes:

$$P(f_i \text{ wins} \mid f_i \text{ competes}) = \frac{1}{\left(\sum_{j=1}^{n} P(f_j \text{ competes})\right) - P(f_i \text{ competes}) + 1} \tag{4}$$

Now we rewrite Equation 1 as:

$$P(f_i \text{ wins}) = P(f_i \text{ competes}) \cdot \frac{1}{\varepsilon - P(f_i \text{ competes}) + 1} \quad (5)$$

$$\varepsilon = \sum_{j=1}^{n} P(f_j \text{ competes})$$

One final constraint enables us to solve for P(f_i competes): we find the flow with the greatest $P(f_i \text{ wins})$, call it f_{max} and set $P(f_{max} \text{ competes}) = 1$. Then we solve for ε:

$$\varepsilon = \frac{1}{P(f_{max} \text{ wins})} \quad (6)$$

Rearranging Equation 5, and having ε allows us to solve for P(f_i competes):

$$P(f_i \text{ competes}) = \frac{P(f_i \text{ wins})(1 + \varepsilon)}{1 + P(f_i \text{ wins})} \quad (7)$$

Each node uses Equations 7 and 6 to compute the $P(f_i \text{ competes})$ of its flows once per superframe.

3) Scaling P(win): Although we could directly compute $P(f_i \text{ wins})$ from the data rate forecast of a flow and use that to compute the P(f_i competes), this wouldn't give the desired result in situations where the sum of the $P(f_i \text{ wins})$ of all the flows exceeds 1. To keep slot allocation working fairly under such circumstances, we always scale the probability such that $\sum_{i=1}^{n} P(f_i \text{ wins})_{scaled} = 1$.

IV. Performance Evaluation

A. Experimental Setup

We evaluate the performance of TRANSFORMA using version 4.0 of the Qualnet [27] network simulator. As performance baseline, we chose one schedule-based and one contention-based MAC protocol to compare against TRANSFORMA. We selected DYNAMMA to serve the role of the schedule-based baseline protocol because it stands out as a general-purpose MAC protocol that has been shown to perform competitively against other schedule-based protocols [11]. IEEE 802.11 DCF [28] has been extensively used and studied and was therefore an obvious choice as the contention-based baseline.

When designing TRANSFORMA we targeted the niche of local area and enterprise networks such as those found in the home, office buildings and hospitals. For example, today's wireless home networks most often have an access point that serves as an internet gateway and all wireless communication goes through this gateway. It is not unlikely that as the number of devices in homes increase, a less centralized topology may serve them better. As the amount of multimedia and the ways in which to access it increase, the home network will increasingly be used to move data between devices in the home rather than just to and from the internet. With this in mind we devised two network topologies for our experiments (Figure 7). The first topology reflects a traditional hotspot network – we have 15 nodes distributed in a circle around a central node. We have spaced the nodes such that hidden terminals exist in the network. The second topology is a multihop grid with diagonal spacing set to the radio range. This topology represents a decentralized network and is large enough to provide some possibilities of spatial channel reuse. The grid topology also has multiple 2-hop neighborhoods, which is good for the purposes of stressing the schedule-based protocols.

In our simulations, all MAC protocols use the 802.11a physical layer operating in the 2.4GHz range with a data rate of 6.0Mbps. Qualnet does not have an 802.11g physical layer implementation. Instead using the 802.11a PHY in the 2.4GHz band approximates 802.11g. The 802.11 MAC is configured with all the default settings. DYNAMMA and TRANSFORMA are both configured with a 1024byte slot size and as close as possible to 1s superframe duration. Small differences in header sizes between DYNAMMA and TRANSFORMA mean that the slot duration is 1.422ms for TRANSFORMA compared to 1.458ms for DYNAMMA and that TRANSFORMA has 703 slots per superframe while DYNAMMA has 700. Both DYNAMMA and TRANSFORMA are configured to fit as many packets as possible into each slot, provided they belong to the same flow.

All experiments shown in the paper use the UDP transport protocol. To represent a real-time flow such as Skype, UDP is the appropriate transport to use. For the background traffic and heterogeneous flows using UDP gives us control over how heavily we load the network (TCP would back off under heavy load). Despite the feedback loop TCP uses in its congestion control function it operates without problems on top of TRANSFORMA. TRANSFORMA's traffic forecaster is tuned to ignore oscillations in the data rate and is slower to reduce the forecast data rate than it is to increase it. This prevents the forecaster from starving a TCP flow that has gone into congestion avoidance.

B. Hot Spot Topology

1) Heterogeneous flows: The network traffic in the first experiment consists of a number of heterogeneous CBR flows. We vary the load on the network by adding flows one by one. All the flows have a packet size of 450 bytes, but each flow has a different packet arrival interval. The first flow's packet arrival interval is 6ms and each successive flow has an interval 1ms larger than the previous one. To get all the nodes in the network involved, each node in the ring is the source of a single flow, and as flows are added they are distributed uniformly around the ring. The central node is the destination for all the flows.

To measure the performance of TRANSFORMA and the other MACs, we use four metrics: average delay, per-flow delay, delivery ratio, and total goodput[1]. One of our main objectives is to minimize delay, so that metric is self evident. Delivery ratio is a strong indicator of the ability of a given MAC to cope with the traffic load. Goodput is a metric that reflects both delivery ratio and delay so it is very useful in evaluating performance. We included a per-flow metric because the flows are heterogeneous and we want to compare how TRANSFORMA, DYNAMMA, and 802.11 deal with each of them. The total goodput is an indirect way of measuring channel utilization. In this scenario the maximum theoretical goodput is 6.0Mbps (the physical data rate) because flows are one hop and successful simultaneous transmissions are impossible. Transport, network and MAC layer headers make it impossible to reach this maximum, but we can still draw conclusions based on how close each MAC gets.

We ran the experiment with 10 seeds and present the averaged results in Figures 5–6. TRANSFORMA's average delay (Figure 5a) is one order of magnitude less that DYNAMMA's

[1]We define goodput as the number of bytes successfully received at the application layer divided by the time between the first and last packet.

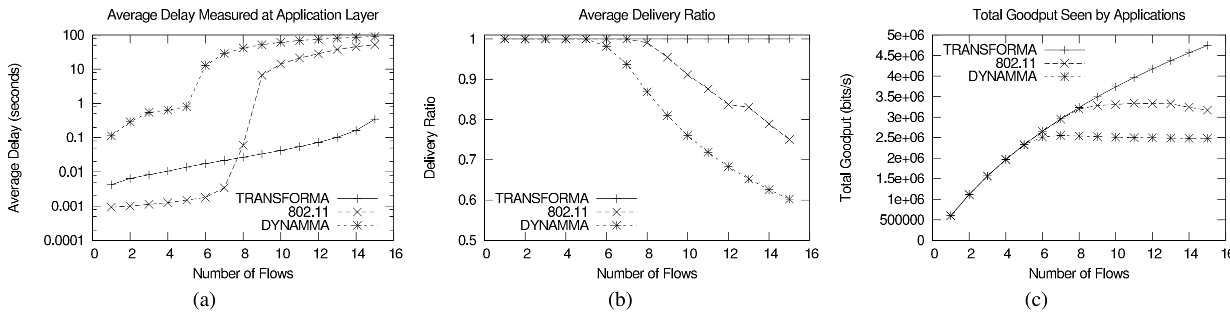


Fig. 5. Average delay (a), packet delivery ratio (b), and total goodput (c) for heterogeneous flows in hot spot topology

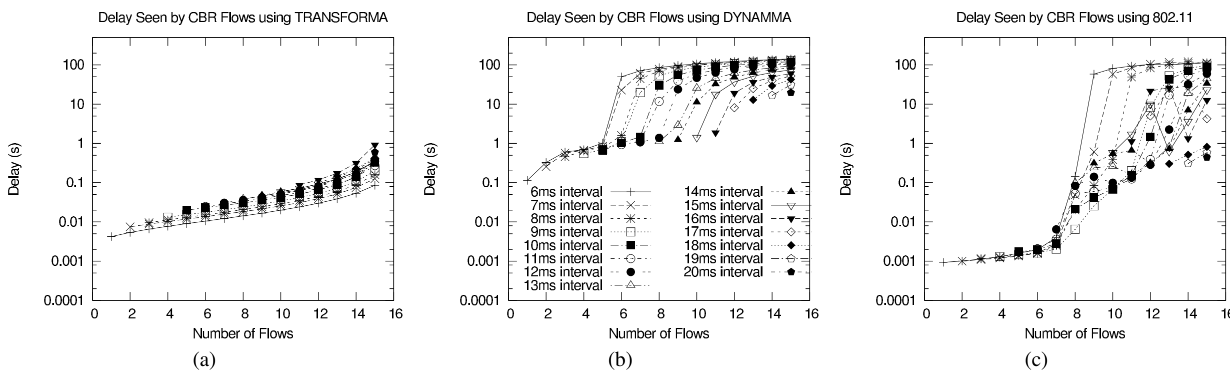


Fig. 6. Per flow delays using TRANSFORMA (a), DYNAMMA (b), and 802.11 DCF (c) in hot spot topology

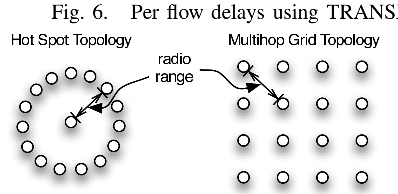


Fig. 7. Two topologies used in our experiments.

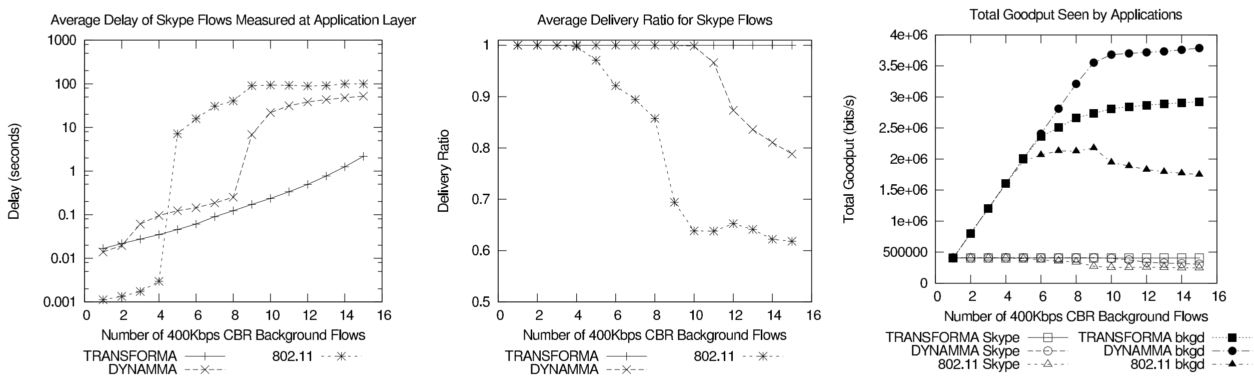
under low load and around two orders of magnitude lower as the load increases. TRANSFORMA's delay is higher than that of 802.11 until the 8th flow is added, at which point 802.11 begins to struggle with the load. Figure 6 shows that as the load increases, the flows that begin to suffer the most when using both 802.11 and DYNAMMA are the ones with the highest data rate. TRANSFORMA, on the other hand, scales back the number of slots each flow wins in equal proportion. This leads us to believe that in DYNAMMA, despite the 3 traffic classes, as the load increases all the flows end up in the highest class where they all win slots with equal likelyhood. This means that flows with more traffic will end up suffering first. In TRANSFORMA, the forecast of the flow's data rate and the total load in the network determine how many slots the flow will win each superframe; the more slots a flow wins, the closer together the slots will be on average and the lower the delay.

The plots of delivery ratio (Figure 5b) and total goodput (Figure 5c) show that TRANSFORMA outperforms DYNAMMA and 802.11 under high load and is able to use use a larger percentage of the available bandwidth. TRANSFORMA outperforms DYNAMMA because its level of control over how many slots each flow gets is greater than that of DYNAMMA. It outperforms 802.11 because it is collision free and thus better able to deal with high load.

2) Skype flows: We designed the second experiment to observe the effect that background traffic has on real-time flows when using TRANSFORMA and the two other MAC protocols. Instead of using a synthetic traffic generator to model real-time traffic, we used a real trace of a Skype phone call captured using `tcpdump` [26]. We then extracted the packet arrival times and packet sizes from the trace and fed them into Qualnet using its trace-based traffic generator. TRANSFORMA's traffic forecaster was able to identify the modal packet inter-arrival times for each of the three applications.

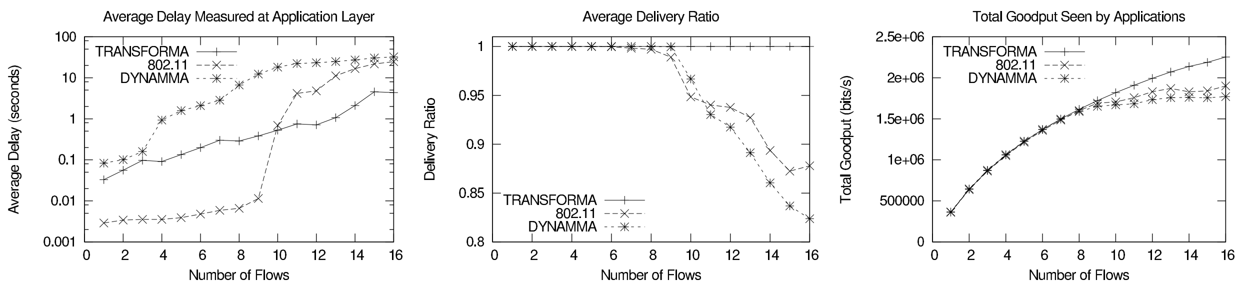
In this experiment there are 3 Skype calls that we term "foreground" traffic, and an increasing number of CBR "background" flows. Each Skype call is made up of two separate trace-based flows: one going into the center node and one going away from it. Here, as in the previous experiment, the center node plays the role of the internet gateway. Each background CBR flow has 200byte packets spaced at an interval of 4ms.

The delay plot in Figure 8a shows that TRANSFORMA is able to maintain a low delay for the foreground traffic while the delays of DYNAMMA and 802.11 increase sharply as the background load increases. Figure 8b shows that 802.11 drops packets due to high contention and DYNAMMA drops packets because it's buffers begin to overflow, whereas TRANSFORMA is able to keep its buffers from overflowing by allocating slots to each flow commensurately to its forecast; in other words, by allocating the right amount of resources to the right flows and being collision-free, TRANSFORMA's is able to outperform the other MACs.



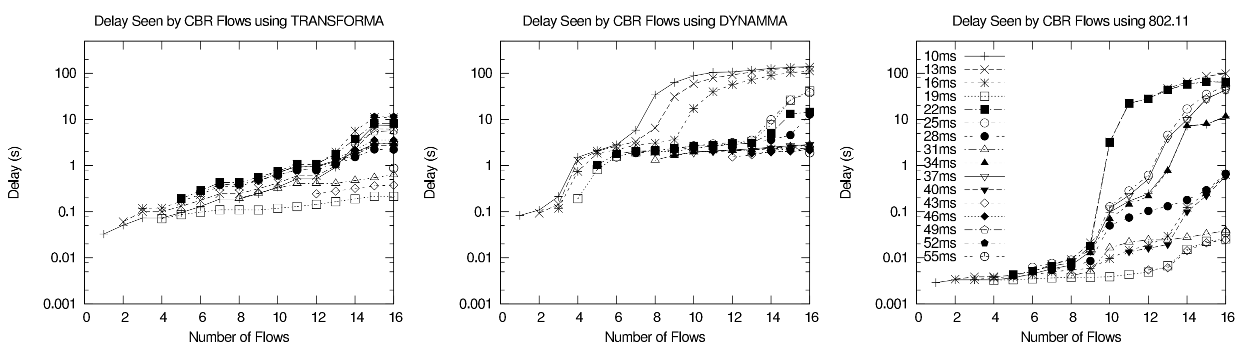
(a) (b) (c)

Fig. 8. Traffic delay (a) and delivery ratio (b) for Skype foreground traffic. Goodput (c) for Skype (foreground) flows and background flows.



(a) (b) (c)

Fig. 9. Average delay (a), packet delivery ratio (b), and total good put (c) of all heterogeneous flows using grid topology



(a) (b) (c)

Fig. 10. Per flow delays using TRANSFORMA (a), DYNAMMA (b), and 802.11 DCF (c) in multihop grid topology

C. Multihop Grid Topology

In this topology we decided to again use the heterogeneous flows traffic scenario. The challenge with the grid was adding flows in a systematic fashion so that the addition of a single flow didn't suddenly change the dynamics. Each application flow in this experiment traverses 3 hops to get from one side of the 4 by 4 grid to the other. The flows were added in the order shown in Figure 11. Similarly to the hot spot topology version of this experiment, we ran the simulation with 10 seeds and evaluated the three MACs based on average delay, per-flow delay, delivery ratio, and total goodput (Figures 9–10). In this scenario, the first flow had a packet interval of 10ms and each successive flow had a 3ms larger interval. Taking into account the fact that each flow traverses 3 hops, the total load offered to the network is 6.8Mbps. This load is manageable because simultaneous transmissions are possible in this topology.

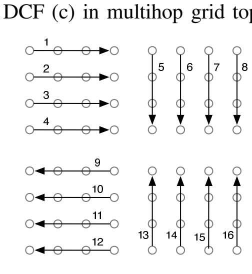


Fig. 11. The order in which flows are added to the 4 by 4 grid

The results show that in the multihop grid topology TRANSFORMA once again outperforms DYNAMMA by an order of magnitude at almost all loads. The delay metrics shown in these graphs are application-layer delays, so all protocols will see their delays increased due to the multi hop nature of these flows: 802.11 will have to contend three separate times to get a packet from source to destination and DYNAMMA and TRANSFORMA have to schedule slots at each node along the way. When the load is low, contending has lower overhead than scheduling, so 802.11 does considerably better than either schedule-based approach. As the load increases, however, the overhead for contention surpasses that of scheduling. Figure 10b clearly shows that with DYNAMMA the flows with highest data rate feel the effects of the higher data rate first. TRANSFORMA once again more appropriately allocates slots so that all the flows share the effects of the increasing load. The relationship between flow rate and delay that was so clearly evident in TRANSFORMA's curves in the hot spot topology (Figure 6a) has been obscured in this topology by the interplay between flows. TRANSFORMA's use of node color when computing competition probabilities for each flow and subsequently computing flow winners for each slot has the side effect that low rate flows with the same color as high rate flows can sometimes experience lower than expected delays.

In this experiment TRANSFORMA once again retains its high delivery ratio while 802.11 drops packets due to contention and DYNAMMA drops them due to buffer overflow of the high rate flows. Consequently, the maximum achievable total goodput of TRANSFORMA is the highest of the three. Because all the flows in this experiment traverse 3 hops, the total goodput can be multiplied by 3 to get a lower bound on how much data has to be transmitted to achieve that goodput.

D. What about 802.11e?

802.11 EDCA provides quality of service enhancements to the standard 802.11 DCF. These enhancements allow 802.11 EDCA to prioritize contention based on 4 access categories, each of which has a different priority. It is the responsibility of the higher layers to select the access category for each packet and place it in the corresponding queue. We assert that 802.11e will perform similarly to 802.11 under high loads and further, a fair comparison with TRANSFORMA was not possible given that 802.11e does not determine the access category of traffic on its own.

V. Implementation

We implemented TRANSFORMA using the STX1201 wireless modem development platform from Starix Technology [29]. The Starix platform contains a processor interfaced with an ultra wide-band radio and several peripherals including Universal Serial Bus (USB) module that enables the board to be connected to a USB host. Out of the box, the board implements the WiMedia MAC [20], in large part in firmware. The platform comes with a software development kit that allows customization of the firmware that runs on the processor and allows us to customize the behavior of the MAC. A lot of the infrastructure of the WiMedia MAC such as beaconing, joining the network and synchronizing node clocks is directly usable by TRANSFORMA and makes this platform very suitable for our implementation.

Our goal with this implementation was to make a fully usable network interface that runs TRANSFORMA, enabling experimentation with TRANSFORMA on real network applications. The computation and memory resources of the STX1201 platform are insufficient to implement TRANSFORMA entirely in firmware, so our implementation is a combination of firmware, a Linux driver and a user-space

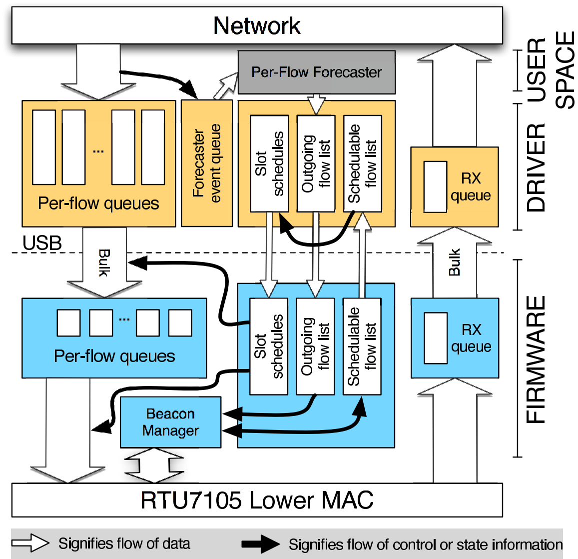


Fig. 12. Implementation block diagram

daemon (Fig 12). The role of the driver is to maintain the per-flow queues of TRANSFORMA and carry out most of the computation required to elect a winner for each slot. The user-space daemon is needed to perform the floating point computations of the forecaster that cannot be done in kernel-space. The firmware's main task, aside from sending and receiving data, is to advertise outgoing flow information in its beacon and collect neighborhood flow information from neighbors' beacons to build the schedulable flow list, which is used by the driver to compute the slot schedules.

We put together a preliminary experiment with the two STX1201 boards we had at the time of this writing. Using the iperf tool [30] for bandwidth performance testing we set up an experiment similar to the one in Section IV-B1. In this experiment we systematically add heterogeneous flows one by one and observe the performance seen by each flow under varying loads. Although the physical layer of the STX1201 was running at 80Mbps, limitations of the current version of our implementation put the maximum sustainable load at 20Mbps. The rates of the heterogeneous flows in this experiment were chosen so that the total load would approach this practical maximum when all the flows were running. The superframe duration in the implementation was 64ms and each superframe was made up of 256 slots, 10 of which were reserved for beaconing. The results in Fig 14 show that the delay performance of the TRANSFORMA implementation is quite good and the delivery ratio and goodputs are unaffected by the increase in the load. Packet traces at the TRANSFORMA network interfaces on both computers were used to extract the performance metrics. We could merge these two traces thanks to the fact that the system clocks of the two computers were synchronized to within 1ms using Network Time Protocol. For future work we plan to improve the performance of the implementation and run more intricate experiments.

VI. Conclusion

In this paper we presented TRANSFORMA, a collision-free, scheduled-based medium access control protocol that employs a novel approach to medium access based on traffic forecasting. TRANSFORMA's traffic forecaster identifies

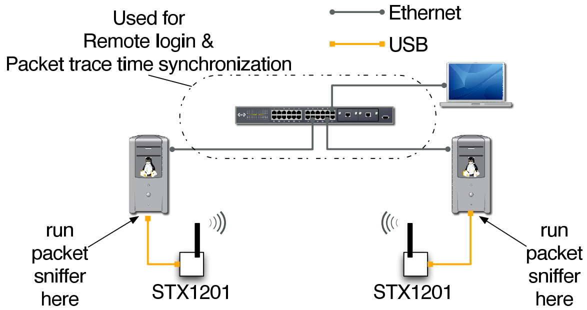


Fig. 13. Implementation experiment setup

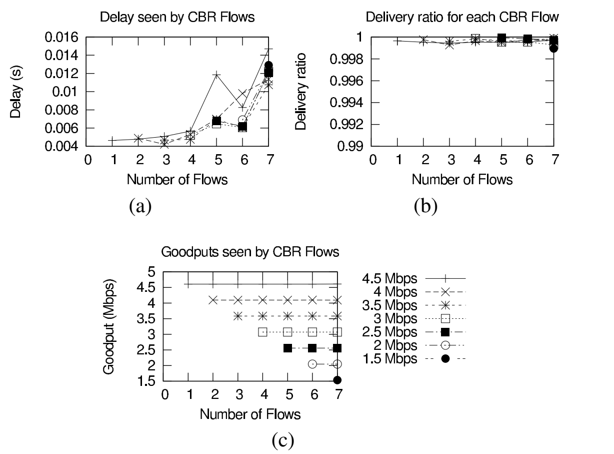


Fig. 14. Delay (a), delivery ratio (b), and goodput (c) of 7 heterogeneous flows going over our implementation's TRANSFORMA link.

patterns in application flows and uses this information to schedule access to the medium most effectively. By doing so, TRANSFORMA tries to anticipate the workload at each node and sets transmission schedules accordingly. This is in contrast to "traditional" scheduled access protocols (e.g., DYNAMMA [11], which set schedules reactively and thus incur considerably higher delays.

We showed through simulations that TRANSFORMA is able to identify the traffic patterns of various kinds of flows and use that information to schedule them, assuring each flow a packet delay on the order of its packet inter-arrival time. Our results also showed that TRANSFORMA is able to schedule real-time flows alongside background traffic with less adverse effects on the real time flows' delay than 802.11 and DYNAMMA.

Our implementation of TRANSFORMA showed that TRANSFORMA can be implemented on real hardware and that its delay performance on such hardware can be expected to be on the order of 10s of milliseconds.

REFERENCES

[1] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, 2002, pp. 1567 – 1576 vol.3.

[2] L. van Hoesel, T. Nieberg, H. Kip, and P. Havinga, "Advantages of a tdma based, energy-efficient, self-organizing mac protocol for wsns," in *Vehicular Technology Conference, 2004. VTC 2004-Spring. 2004 IEEE 59th*, vol. 3, 17-19 2004, pp. 1598 – 1602 Vol.3.

[3] L. Shi and A. Fapojuwo, "Tdma scheduling with optimized energy efficiency and minimum delay in clustered wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 9, no. 7, pp. 927 –940, july 2010.

[4] L. Bing, Z. Lin, and Z. Huimin, "An adaptive schedule medium access control for wireless sensor networks," in *Networking, 2007. ICN '07. Sixth International Conference on*, 22-28 2007, pp. 12 –12.

[5] M. Sekine, S. Takeuchi, and K. Sezaki, "An energy-efficient mac protocol with lightweight and adaptive scheduling for wireless sensor networks," in *Radio and Wireless Symposium, 2007 IEEE*, 9-11 2007, pp. 161 –164.

[6] Z. Karakehayov and N. Andersen, "Energy-efficient medium access for data intensive wireless sensor networks," in *Mobile Data Management, 2007 International Conference on*, 1-1 2007, pp. 361 –365.

[7] Y. Liu, I. Elhanany, and H. Qi, "An energy-efficient qos-aware media access control protocol for wireless sensor networks," in *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, 7-7 2005, pp. 3 pp. –191.

[8] J. Cai, K.-H. Liu, X. Shen, J. Mark, and T. Todd, "Power allocation and scheduling for mac layer design in uwb networks," in *Quality of Service in Heterogeneous Wired/Wireless Networks, 2005. Second International Conference on*, 24-24 2005, pp. 8 pp. –2.

[9] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves, "Energy-efficient collision-free medium access control for wireless sensor networks," *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, Nov 2003.

[10] V. Rajendran, J. Garcia-Luna-Aceves, and K. Obraczka, "Energy-efficient, application-aware medium access for sensor networks," *IEEE Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, Jan 2005.

[11] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves, "Dynamma: A dynamic multi-channel medium access framework for wireless ad hoc networks," *Mobile Adhoc and Sensor Systems*, Jan 2007.

[12] G. Maier, A. Feldmann, V. Paxson, and M. Allman, "On dominant characteristics of residential broadband internet traffic," in *IMC '09: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. New York, NY, USA: ACM, 2009, pp. 90–102.

[13] R. Pries, F. Wamser, D. Staehle, K. Heck, and P. Tran-Gia, "On traffic characteristics of a broadband wireless internet access," in *Next Generation Internet Networks, 2009. NGI '09*, 1-3 2009, pp. 1 –7.

[14] A. Sharma and E. M. Belding, "A case for application aware channel access in wireless networks," in *HotMobile '09: Proceedings of the 10th workshop on Mobile Computing Systems and Applications*. New York, NY, USA: ACM, 2009, pp. 1–6.

[15] S. Singh, P. Acharya, U. Madhow, and E. Belding-Royer, "Sticky csma/ca: Implicit synchronization and real-time qos in mesh networks," *Ad Hoc Networks*, Jan 2007. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1570870507000121

[16] P. Djukic and P. Mohapatra, "Soft-tdmac: A software tdma-based mac over commodity 802.11 hardware," in *INFOCOM 2009, IEEE*, 19-25 2009, pp. 1836 –1844.

[17] A. Sharma, M. Tiwari, and H. Zheng, "Madmac: Building a reconfiguration radio testbed using commodity 802.11 hardware," in *Networking Technologies for Software Defined Radio Networks, 2006. SDR '06.1st IEEE Workshop on*, 25-25 2006, pp. 78 –83.

[18] A. Sharma and E. M. Belding, "Freemac: framework for multi-channel mac development on 802.11 hardware," in *PRESTO '08: Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow*. New York, NY, USA: ACM, 2008, pp. 69–74.

[19] A. Rao and I. Stoica, "An overlay mac layer for 802.11 networks," in *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*. New York, NY, USA: ACM, 2005, pp. 135–148.

[20] "High rate ultra wideband phy and mac standard," ECMA Internal Standard ECMA-368, 2007.

[21] M. Herbster and M. Warmuth, "Tracking the best expert," in *Machine Learning*. Morgan Kaufmann, 1995, pp. 286–294.

[22] N. Littlestone, "Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm," in *Foundations of Computer Science, 1987., 28th Annual Symposium on*, oct. 1987, pp. 68 –77.

[23] N. Littlestone and M. Warmuth, "The weighted majority algorithm," in *Foundations of Computer Science, 1989., 30th Annual Symposium on*, oct-1 nov 1989, pp. 256 –261.

[24] Y. FREUND, D. HAUSSLER, D. HELMBOLD, R. SCHAPIRE, and et al., "How to use expert advice," *Journal of the ACM*, Jan 1997. [Online]. Available: http://mercurio.srv.dsi.unimi.it/~cesabian/Pubblicazioni/jacm-97a.pdf

[25] D. Helmbold, D. Long, T. Sconyers, and B. Sherrod, "Adaptive disk spin-down for mobile computers," *Mobile Networks and Applications*, vol. 5, p. 2000, 1998.

[26] Tcpdump/libpcap. [Online]. Available: http://www.tcpdump.org/

[27] Scalable networks. [Online]. Available: http://www.scalable-networks.com

[28] B. Crow, I. Widjaja, L. Kim, and P. Sakai, "Ieee 802.11 wireless local area networks," *Communications Magazine, IEEE*, vol. 35, no. 9, pp. 116 –126, sep 1997.

[29] Starix technology. [Online]. Available: http://www.starixtech.com

[30] Iperf. [Online]. Available: http://iperf.sourceforge.net/