INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD • INDIA

A Memory-Aided Broadcast Mechanism for Enabling a Rural Community Radio on an Ad-Hoc Peer-To-Peer Mobile Network

Kavitha Ranganathan
Sonia Arora

W.P. No. 2013-03-04
March 2013

INDIAN INSTITUTE OF MANAGEMENT
AHMEDABAD-380 015
INDIA

A Memory-Aided Broadcast Mechanism for Enabling a Rural Community Radio on an Ad-Hoc Peer-To-Peer Mobile Network

Kavitha Ranganathan
Assistant Professor, Indian Institute of Management Ahmedabad
Email: kavitha@iimahd.ernet.in

Sonia Arora
Research Associate, Indian Institute of Management Ahmedabad

Abstract

This paper investigates deploying a village level community-radio application on top of a MANET comprising completely of basic mobile phones. We envision a system, where any user in the network is equally empowered to generate and distribute audio content to the entire network, using his or her mobile phone. The paper focuses on the study of suitable broadcast algorithms for the network. In this context, we propose a novel broadcast scheme where nodes bank on their memory to decide whether to forward packets of an audio stream – capitalizing on their past behavior to stabilize on fixed routes for the entire stream. In our scheme called Environs Aware Broadcast Mechanism (EABA), a node gauges the local mobility around itself, and uses that to decide which broadcast mechanism to use. When mobility is high, it uses SBA (Scalable Broadcast Algorithm), a popular neighbor -knowledge broadcast algorithm with high overheads, but when mobility is low, it switches to MaBA (Memory-aided Broadcast Algorithm). Extensive simulations on a village-level MANET, confirm that EABA is successful in substantially reducing jitter, latency and packet loss: all critical metrics for an audio application. At the same time, EABA does not incur other overheads and maintains the same levels of reachability and efficiency as SBA.

Keywords

Rural community radio, Ad hoc mobile network, peer-to-peer, Memory-aided Broadcast Algorithm

A Memory-Aided Broadcast Mechanism for Enabling a Rural Community Radio on an Ad-Hoc Peer-To-Peer Mobile Network

1. INTRODUCTION

Mobile ad hoc networks (MANETs) consist of wireless-enabled mobile nodes that not only communicate with each other, but also act as intermediaries for nodes that are out of direct range from one another. A message from a source node might travel through multiple hops before reaching the destination node. MANETs can be perceived as wireless peer-to-peer networks that are devoid of any centralized decision-making entity. Such networks have typically been studied for applications related to military operations in hostile regions or disaster recovery when traditional communication infrastructure has failed. More recent applications include mobile telephony for remote, rural regions which cannot afford traditional communication infrastructure [1][2][3].

Large parts of the developing world are devoid of traditional communication infrastructure, mainly due to economic reasons. MANETS comprising entirely of basic, affordable mobile phones are a promising alternative to create local telephony, without requiring phone towers or other expensive supporting infrastructure [1]. Other collaborative applications like a community radio service can be deployed above this substrate. Community radio has been seen as a powerful means for empowerment by enabling local content creation and broadcasting [4]. However, the traditional community radio model entails centralized content filtering and dissemination. A true peer-to-peer model empowers any user to create and broadcast content to the entire network, thus democratizing knowledge creation and dissemination and avoiding censorship. This decentralized model also ensures that the content consumers are also the content creators and providers.

Enabling such a community radio service on a phone-based MANET entails that each device in the network be able to reliably broadcast sound-bytes to all other nodes in the network. Recent work has proposed using mobile ad-hoc networks for exactly such a community radio system and reviewed the suitability of existing MANET broadcast algorithms for this application [5]. However, there are limitations to this approach, as explained shortly.

1.1 The Broadcasting Problem

There is extensive literature on broadcasting schemes for MANETS, where the role of the broadcast packets are chiefly to aid the routing protocols [6] [7][8]. These existing broadcast schemes can be classified into four broad categories, as proposed by William and Camp [9]: (i) Simple Flooding (ii) Probabilistic Schemes (iii) Position Based Methods and (iv) Neighbor Knowledge Schemes. Of the different methods, Neighbour Knowledge schemes have been found to be most promising for networks with dynamic topologies [9]. Flooding and probabilistic schemes are too inefficient and position based methods need special technology like GPS enabled devices. Of the different neighbor-knowledge algorithms, past work [9] [5] has identified SBA (Scalable Broadcast Algorithm) and SBA variants [10] as one of the best options for generic broadcasting in a mobile ad-hoc network.

The SBA protocol [11] works as follows: every node maintains a partial network map of all nodes within a two-hop radius of itself, using periodic “hello messages”. Suppose a node (say N1) receives a broadcast message m from node N0. N1 can find out all common neighbors between itself and N0 which would have already received m from N0. If there are additional neighbors of N1 which were not covered by N0, then m would be scheduled for a rebroadcast, after a random delay (called RAD). During the RAD, if a duplicate m is received from some other node, N1 would again determine if any new nodes can be reached by a re-broadcast. At the end of the RAD, if some neighbors of N1 have not yet received m, then the message is rebroadcast.

However, we have identified at-least three shortcomings should SBA (or similar neighbor-knowledge schemes) be directly used for our envisaged community radio service:

1) The rural ad-hoc network we envision is expected to be semi-static or static for large parts of the day, and highly dynamic only during certain times of the day. SBA has been shown to work well for dynamic ad-hoc networks but is probably an over-kill for static or semi-static networks, where the topology is not constantly changing. The reason SBA works well in a dynamic topology is because each node maintains up-to-date 2-hop knowledge about its neighbourhood via hello messages. These hello messages can be a significant overhead on the network, especially if they are not needed all the time (that is when the network is semi-static or static).

2) SBA is also known to incur significant amounts of latency in the message transfer, introduced due to the RAD component of the algorithm [12]. The RAD (Random Assessment Delay) component is essential for SBA's functioning. However, any additional latency or jitter could detrimentally affect the quality of the audio application in question, and hence algorithms that do not use a RAD component could succeed in lowering the end-to-end latency and variability in latency (jitter) of the packets in the broadcast.

3) Given that an entire sound-cast (expected to last for a few minutes) will comprise of a stream of messages emitted from the same source , there is potential for using past information of the behaviour of a node to stabilize on a particular set of routes that the message takes to reach the entire network. This "learning from past actions" can be especially useful for periods when the network is static. Neither SBA nor any other broadcasting algorithm we came across in the past literature exploits this kind of memory-aided optimization of node behavior.

Given the above observations, we propose and test the following hypothesis in this paper:

When the network is static or semi-static, SBA is unsuitable as the broadcasting algorithm for a community-radio service. An altogether different and far simpler algorithm with lesser overheads can be as or more effective than SBA. In this new algorithm (which we call Memory-aided Broadcast Algorithm –MaBA) , a node remembers its past behaviour and uses that to decide whether or not to rebroadcast a message. Potential advantages of MaBA include (1) lesser network congestion due to lesser number of hello messages, (2) lower latency and jitter because of not using the RAD component and (3) more efficiency because of exploiting a node's memory of its past behavior.

Hence, in our proposed mechanism called Environs Aware Broadcasting Algorithm (EABA), each node independently decides to use either SBA or MaBA as the broadcasting algorithm, depending on network conditions and the position of the current packet in the stream. The position of the packet in the stream is important as a node's memory of past behavior can only be used for that particular stream. With every new source, a new set of routes will have to be discovered (using a neighbor-knowledge scheme), stored and used (by MaBA).

We conduct extensive simulations to compare the performance of EABA to SBA and a variant called SBA-mob. Our results show that EABA is successful in measuring the amount of local mobility in the network and adapting to it. This results in significant savings in end-to-end packet latency and reduction in jitter : both crucial metrics for an audio application. There are considerable savings in bandwidth utilization as well, (in the form of saved hello messages), leading to EABA performing better in a congested network. Additionally, these performance gains do not affect the reachability or packet-delivery ratio of the broadcast, thus confirming that EABA is a good choice for a MANET enabled audio-broadcast application.

The rest of the paper is as follows: The next section details the workings of our proposed broadcasting mechanism: EABA. Section 3 contains a description of the simulation model and experiments used. The experimental results are discussed in section 4 and we conclude in section 5.

2. ENVIRONS AWARE BROADCASTING ALGORITHM

As mentioned above, in our proposed method, Environs Aware Broadcasting Algorithm (EABA), there are two modes in which a node can broadcast: SBA or MaBA, depending on the mobility in the network and position of the message in the broadcast stream. As part of EABA, four complementary strategies are deployed at each node (1) mobility detection (2) adapting the frequency of hello messages (3) maintaining and using the node's recent history of broadcasting behavior (4) deciding which mode to switch to: SBA or MaBa. We now explain each strategy in turn.

2.1 Mobility Detection

Since a node has to modify its behaviour depending on the degree of mobility in the network, the question that arises is, how does a node measure the mobility in the network? We propose that each node maintain a measure called the mobility factor (*mf*), which is its local view of the mobility or dynamicity in the network.

The mobility factor of a node is calculated by keeping track of changes to its neighbor table - the intuition being that rapid changes in the neighbor table of a node indicates a rapidly changing network topology and hence a high degree of mobility. Inversely, a stable neighbor table indicates that the network is static or at the very least, that locally there is relative stability in the network. By relative stability, we mean that a bunch of nodes could all move in the same direction at the same speed. In both scenarios (absolute static network vs relative stability of a node), a node can leverage the fact that it does not need to re-discover its neighborhood and the same broadcast paths discovered earlier can be re-used (the primary assumption in MaBA).

Hence for both possible scenarios: static and dynamic, comparing the current neighbor table to a snapshot of the table taken a little while ago should convey a fairly accurate picture of the relative mobility in the network.

A detailed description of how the mobility factor at each node is calculated, follows:

Recall that as part of the standard SBA protocol, each node periodically broadcasts *hello* messages to all its neighbors. These *hello* messages enable each node to build its neighbor-table.

The mobility factor (mf) at each node is calculated every T seconds , by comparing the current neighbor table, to a snapshot of the table that was recorded T seconds ago. All the common neighbors among the two versions of the table are discounted, and the count of the remaining nodes gives us the value of *mf*.

Mathematically:

$$mf(Node1, t+T) = Neighbors(Node1, t+T) \cup Neighbors(Node1, t) - Neighbors(Node1, t+T) \cap Neighbors(Node1, t)$$

where

mf(n, t) = mobility factor of node n at time t

Neighbors(n, t) = set of neighbors in the neighbor table of node n at time t.

If mf = 0 for a particular node, it denotes that the network around that node has been relatively stable between these two timestamps. The greater the value of *mf*, the network topology around that node can be expected to be more dynamic. Figure 1 illustrates two examples of how the mobility factor (mf) is calculated from neighbor table information.

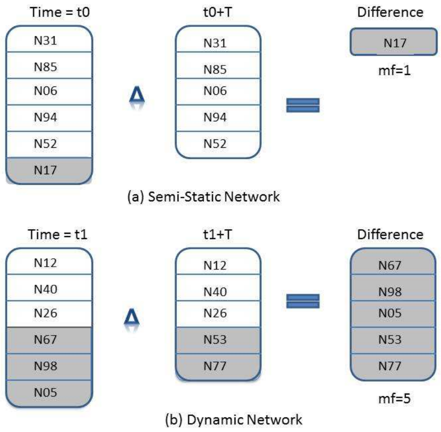

Figure 1: Calculation of mobility-factor (mf) from neighbor table information, for two scenarios (a) semi-static network and (b) dynamic network

Additions to a neighbor table happen whenever an unknown node's hello message is received by the node. Deletions from a neighbor table are facilitated periodically, using the procedure described below.

For each node in the neighbor table, the time of the latest hello message that was received from that neighbor is recorded. The neighbor table is refreshed every R seconds by executing the following check for each node in the table: if the difference in the current time and the last heard time from the neighbor node is greater than the interval of the refresh i.e. R, the node entry is deleted from the table.

Typically the refresh rate R should be set to a value at-least twice or thrice the value of the hello message interval. This ensures that a node is only deleted if it has not been heard from for at-least two or three hello cycles. This minimizes false deletions from the neighbor table caused by packet drops due to congestion in the network or collisions.

In our proposed broadcast algorithm (EABA), the frequency of hello messages (*f*) sent out by a node is dynamically adapted to the amount of mobility around the node. Since *R* (refresh rate of neighbor table) is intimately linked to *f*, *R* is also changed accordingly. The details of this adaptation along with concrete scenarios are explained in the following section.

2.2 Adaptation of hello message frequency

Note that SBA needs periodic hello messages so that nodes can decipher their local topology. However these messages can create a significant overhead in terms of network utilization. When the network is static or semi-static, the utility of these hello messages is limited as the neighbor tables will not change frequently. Hence in our approach, the frequency of hello messages is reduced, based on the calculated mobility factor (mf). For a mobility factor $\geq$ M (where M is a threshold value), a hello message is sent every 1 second (the default value in many neighbor-based broadcasting algorithms [9]). However, as mf decreases, the hello interval is gradually increased using the following formula:

If mf ≥ M then Hello message interval = 1 second

Else Hello message interval = M-*mf*

where mf is the mobility factor and M is the threshold mobility factor beyond which the network is considered highly dynamic.

Hence, for M = 5 and mf = 0 for node n (indicating a static network around n), hello messages emanate from node n only every 5 seconds. As the mobility factor (mf) increases to 5 or greater, the frequency of hello messages increases to 1 per second.

To avoid false deletions from a neighbor table, a node should not be expunged from the neighbor table, if it has not been heard from since the last hello cycle. Expunging a node if only one hello message is missing from it, might lead to unnecessary deletions, as the message might have been lost due to other reasons like congestion or collisions. Hence, the refresh interval (R) of a node's neighbor table should also be adapted to correspond to changes in the hello-message interval. The following formula is used to calculate R:

If mf ≥ M then R = M

Else R = 2M – mf

The above formula ensures that the refresh interval is at-least twice the hello message interval and steadily increases as network stability increases.

2.3 Recording and using history of node's behavior

Recall that each node maintains a recent history of its broadcasting behavior, and uses that in MaBA to influence its current broadcasting decision.

As discussed earlier, in SBA each node maintains 2-hop knowledge of its neighbors. In addition to this, in EABA each node maintains the following additional field for *each* node in its neighbor table - a list of the last *x* broadcasting decisions, where 0 denotes 'not broadcasting' and 1 denotes 'broadcasting'. The following example illustrates how this history is recorded.

Suppose node B receives a broadcast data packet from node A, and using some algorithm (note that the algorithm could be SBA or MaBA depending on conditions described in the next subsection), decides to drop the packet. Hence, in the neighbor table, the last history of A would be updated to 0. Suppose, x = 5, that is the last five broadcast decisions of B (when the source was A) are maintained, then the neighbor-table of B might look as follows:

Figure 2: Partial illustration of Neighbor Table for node B

Neighbor	History
A	1 0 0 1 0
D	1 1 1 1 1
F	0 1 1 0 0

In the above table, the five bits for A denote the five most recent broadcasting decisions of B; when the packet was received from A. As can be seen from the table, B decided to rebroadcast twice and dropped the packet three times. Similar broadcasting history is maintained for each node in the neighbor table.

When a node receives a data packet, it first decides which of the two broadcasting algorithms (SBA or MaBA) to use : the details of this decision-making process can be found in the next sub-section.

Suppose it decides to use MaBA, it looks up the recorded history for that particular source node and uses the following algorithm to decide whether to broadcast or not.

Note that we use the following notation for the history : N(i)

where N is the source node from where the packet has been received and N(i) denotes the ith recorded history, where i ranges from 1 to x.

For the example in Figure 2, i ranges from 1 to 5 and A(1) = 1 and A(5) = 0. Note that A(5) in this case is the most recent history.

The following algorithm is used for the broadcasting decision:

If N(x) = 1 or sum (N(1) , …., (N(x-1)) > x/2) then broadcast packet, else drop packet.

The intuition behind this algorithm is that instead of relying only on one instance of past behavior, even if the latest decision was not to broadcast, we see if the node has broadcasted often in the recent past. This leads to a more conservative algorithm, which errs on the side of more nodes broadcasting. We noticed in our experiments that this leads to more reachability, and compensates for the randomness caused by SBA's RAD component.

Note that the value of x should be small enough to reflect recent history and not dated behavior. A value of x = 5 worked sufficiently well in our experiments.

2.4 Deciding the mode: SBA or MaBA

Finally, we describe how a node decides between SBA or MaBa as the broadcasting algorithm.

A node uses SBA when it perceives the network as dynamic and MaBA when the network is seen to be static or semi-static. In addition to the mobility in the network, another factor should determine if a node uses SBA or MaBA. Recall, that for our application, a stream of messages will be emitted from the same source, for a short to medium duration of time (a typical sound-cast could last from 2 to 7 minutes). When a new stream of messages starts out from a new source, nodes will not have any memory to bank upon for this new set of routes from the new source to every node in the network. The memory that a node has will be regarding its behaviour of the previous stream (where some other node was the source). Hence, we have to give all nodes a chance to discovery a good set of routes using a neighbor-knowledge algorithm (like SBA) for each new stream. Hence, even if the network is perceived as static, a node should use SBA for the first m messages in a stream and then switch to MaBA for the rest of that stream.

The 2* 2 matrix below covers the four possible situations and recommended node behavior.

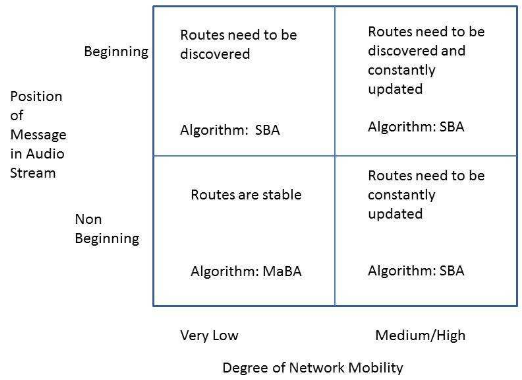

Figure 3: Recommended node behavior for different network conditions.

For our experiments, if the mobility factor (mf) of a node is greater than 0 or the packet belongs to the beginning of a stream (first 10 packets of a stream), then SBA is used, else MaBA is used. Variations to the threshold mobility were also explored in our experiments, and are reported in the results section.

3. SIMULATION MODEL AND DESIGN OF EXPERIMENTS

We model a village level peer-to-peer mobile adhoc network, where each peer in the network is a wifi-enabled mobile phone. Peers move around independently within the designated area according to realistic mobility patterns based on actual village scenarios. We use a popular network simulator called Glomosim [ref] to model the mobile ad-hoc network, and have implemented the broadcast algorithms that we evaluated, above the network layer provided by GlomoSim 2.03.

All mobile devices in our experiments are assumed to have a uniform range for communication and the same maximum possible throughput.

3.1 Modeling a rural village

Villages vary greatly in their population and spread and therefore it is challenging to model a typical village size. According to the Indian 2001 census for example, out of around 600,000 villages in India, more than 200,000 villages have a population under 500 persons, while around 4000 villages have a population greater than 10,000 persons. Since our application is targeted towards the more rural and remote villages where traditional communication infrastructure is non-existent; these villages typically tend to be small. We assume around 100 mobile phone users and an area of 4 square kilometers in our simulations.

To understand the mobility pattern of villagers, we visited two villages in Gujarat, India and also interviewed people who have lived in villages. The following patterns emerged from our investigations:

The typical mobility pattern in a village can be divided into four phases as seen from the view of an adhoc network: (1) a short, highly dynamic phase in the morning when almost everyone is commuting to their place of work, (2) a relatively static, long phase for the bulk of the day when they are working, (3) a highly dynamic, short phase at dusk when most people are getting back home, and (4) a relatively stable/static night time phase.

The village population can roughly be divided into three categories : (a) those who live in or near the center of the village (typically, there is a main street of shops and adjoining residences) and commute to nearby fields at the periphery of the village (could be a couple of kilometers) in the morning and come back at dusk (b) those who live at the outskirts, close to their fields itself and (c) those who reside near the core of the village and work there as well : in and around their homes or in the village shops etc.

Existing human mobility models like SMOOTH [13], SLAW [14] and others [15] proved inadequate for our specific village network and its unique mobility pattern. Hence, we have created our own mobility model to capture the dynamics of village-level movements. This model is described below:

Node mobility is divided into distinct dynamic and static phases (recall that we want EABA to be able to distinguish between a dynamic and static network).

For the dynamic phase 30% of the nodes are placed randomly within a central square of size 750m * 750m and move to a randomly chosen location within the central square. This mimics category c. Another 30% also start off from the central square but move towards a randomly chosen location in the periphery of the network (figure *), to mimic category a. Another 30% of nodes are randomly placed in the periphery, and move to a random location in the periphery (in the same rectangle they belong to): these are category b. The last 10 % of nodes are randomly placed anywhere in the network and move to a randomly chosen location in the entire network area (to account for miscellaneous other activity). These are termed category d.

Once all the nodes reach their destinations, they stay there for the entire static phase. During the next dynamic phase, all nodes return to their original positions (to mimic the dusk-time commute).

In our investigations, people in remote villages typically commuted by foot, occasionally on cycles or bullock-carts and rarely on motorized vehicles. 1 m/s can be considered typical walking speed and a fast bicycle might travel at 8 m/s. In our simulations, the average node speed is set to 1.43 m/s.

3.2 Network modeling

As mentioned earlier we use GloMoSim v2.03 (a discrete-event network simulator) [16] developed at UCLA, for our experiments.

The protocol stack used is illustrated in Figure 4. As shown in the figure, IEEE 802.11 is used for the Physical and Mac layers and EABA for the network layer. The rest of the protocol stack is typical of a VoIP (Voice Over IP) stack, with UDP, RTP and an Audio Coder as the top three layers.

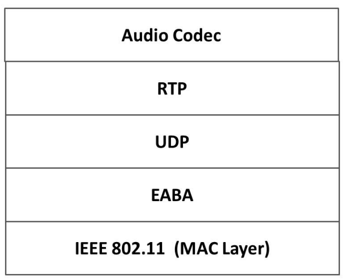

Figure 4: Proposed VoIP protocol stack

IEEE 802.11 with the distributed coordination function (DCF) is implemented at the MAC layer, as this is the default that works well for MANETS. We chose 802.11b over other versions of 802.11 since it is already available on some basic phone models and is known to work well for wireless LANS [17]. Recall that one of the goals of the project is to use off-the-shelf and affordable phones for the adhoc community network. IEEE 802.11b works at a few different range/throughput pairs: 400m/1 mbps , 250m/5.5 mbps and 150m/11 mbps among others.

We assume the first configuration (range of up-to 400m and a bit rate of 1 mbps) as the default for most of our scenarios. Results of experiments with the other two configurations are also reported.

Voice data typically requires a bit rate of 10 kb/s (kilobits/second) but can give adequate voice quality at up-to 8 kb/s (depending on the codec, for example G.729) [18] [19]. The default for voice is to send audio frames every 20 ms – to keep latency levels in check. This translates to a packet rate of 50 packets per second (pps) and a packet size of 20 bytes, which we model in our simulations. With a data payload of 20 bytes, an additional 32 bytes are required for RTP, UDP and EABA combined (12 for RTP, 8 for UDP and the rest for EABA). Also, since many VoIP codecs send data at a constant bit rate, we assume a Constant Bit Rate (CBR) model to simulate VoIP in our MANET.

We assume that a particular source will be allowed a sound-cast of around four minutes, resulting in 12,000 packets per audio stream emanating from the same source. A random new node is chosen as the source for each sound-cast, and one simulation run consists of five such unique sources.

Discussion of issues related to other logistics of the application: how is the schedule and content for the broadcasts decided , who gets to broadcast and when etc. is deferred to the concluding section.

The specific parameter values used in our experiments are mentioned in Tables * and *. Table * contains the network parameters while table * contains the application and broadcast algorithm specific parameters. The first number denotes the default value and the latter denotes the range used for further experimentation.

We assume a threshold mobility factor of 5 and hence the hello message interval ranges from 1 second to 5 seconds, depending on the current mobility factor at a node. A maximum jitter of 250 ms is applied to the hello message interval - this is a common practice in MANETS to avoid packet loss resulting from collisions. The neighbor table refresh rate ranges from 5 seconds to 10 seconds, depending on the hello message interval (refer to section 5 for a detailed explanation of hello message interval adaptation and related issues). MaBA uses a small jitter of maximum 1 ms to counter possible collisions resulting from nodes simultaneously broadcasting a message. Note that SBA does not need an induced jitter because of its RAD component that introduces variability.

Table 1:Simulation Parameters for Network

Network Size	2000m * 2000m
Number of Nodes	100 (60 to 140)
Range	400 m (150 m -1000 m)
Throughput	1 mbps (5.5 mpbs, 11 mbps)
Average Speed of nodes	1.43 m/s
Simulation Time	1200 seconds

Table 2: Simulation Parameters for Application and Broadcast Algorithms

Data Payload	20 Bytes
Packet Rate	50 packets per second (pps)
Hello message interval	1 second – 5 seconds
Threshold Mobility Factor (M)	5
Hello message jitter	250 ms
Neighbor-table refresh rate	5 seconds – 10 seconds
Mobility Detection Interval (EABA)	10 seconds
Size of Audio stream	12,000 packets
Total number of audio packets in simulation run	60,000packets
RAD Tmax (SBA)	20 ms
Introduced Jitter (MaBA)	1 ms

3.3 Design of Experiments And Metrics for Evaluation

We design and conduct three distinct sets of experiments. The first set of experiments evaluate our proposed mobility detection scheme. The second set evaluates our proposed broadcast algorithm and compares its performance to other common broadcast algorithms, on a number or relevant dimensions. The third set of experiments looks at the effects of various parameters on the broadcast algorithms being evaluated.

Each experiment was repeated with ten different mobility files and five randomized runs for each mobility file. Thus, each value reported is the average of 50 simulation runs.

Experiment Set 1

Since the crux of our proposal relies on correctly detecting mobility in the network, we designed a set of experiments to evaluate how well the mobility detection scheme and broadcast algorithm adaptation in EABA works.

To simulate the two phases of peak mobility (at dawn and dusk) and their transitions to and from the static phase, the following mobility pattern is used: nodes are mobile for 500 seconds (according to the mobility pattern described in section 6.1 for the morning commute), followed by a static phase of 200 seconds and finally followed by another mobile phase of 500 seconds (the evening commute pattern as described in section 6.1). This mobility pattern allows us to see whether nodes using EABA can detect the changing mobility around them and adapt to it. We report two metrics for these experiments:

(1) the mobility factor across time for four selected nodes falling in the four different categories described in section 6.1

(2) the number of nodes using SBA versus MaBA across time

Experiment Set 2

The second set of experiments is aimed at evaluating the performance of our proposed broadcast scheme : EABA (Environs Aware Broadcast Algorithm), and to compare its performance to other broadcasting algorithms : specifically SBA, SBA-mob and Flood. SBA has already been explained in section 1. SBA-mob is a variant of

SBA, where the hello-message frequency is adapted according to the local mobility in the network, much like what happens in EABA. We show results for SBA-mob since it helps differentiate between performance gains in EABA due to reduced hello-messages versus other factors related to the nature of the broadcast algorithm. Flood (where every node re-broadcasts every message it receives) is included as a base case (worst case scenario).

The mobility pattern should follow what might be typically expected in the village network setting. We model a section of the day in the morning that comprises of a dynamic phase for 500 seconds as described in section 6.1 (to capture the morning commute) followed by a longer static phase of 700 seconds.

The algorithms were evaluated on the following metrics which can be divided into two groups: broadcast related metrics and audio application related metrics. Broadcast related metrics include efficiency, reachability and bandwidth overhead, while the audio application metrics include latency, jitter and packet loss.

While energy efficiency is an important concern in MANETs, the broadcast algorithm per say plays a small role in how much energy is consumed by a node. The overwhelming factor of energy consumption depends on the state of the node: idle or asleep [20] which is not relevant to a community-radio application, since a node need to be awake to receive the signal. However, the transmission range of a node could be a critical factor in energy consumption: this aspect is explored further in the next set of experiments.

A brief definition of each metric follows:

Broadcast related metrics

Efficiency: The percentage of nodes that re-broadcasted the message. Lesser re-broadcasts while not hampering reachability implies a more efficient algorithm.

Reachability**:** The percentage of nodes that received the message. A broadcast by definition implies that all nodes in the network should receive the message. However, network partitions and/or packet drops can cause a reachability value of less than 100%.

***Message overhead* :** Some broadcast mechanisms generate extra messages like “hello packets” that incur additional overhead in terms of bandwidth utilization, which could cause congestion in the network.

Audio application related

Latency**:** The packet transit delay measured as the difference between two timestamps: the time a packet was released at its source and the time the last node in the network received it. A high latency can adversely affect the quality of an audio stream.

Jitter: Jitter is the variation in packet transit delay. In general, higher levels of jitter are more likely to occur on either slow or heavily congested links. Some amount of jitter can be mitigated by the audio codec, but higher levels (above 30 ms or so) adversely impact audio quality. Jitter at a node is calculated by summing up the absolute differences in arrival time between each consecutive pair of packets in a stream and calculating the average of these differences.

Packet Loss: The percentage of packets in an audio stream that did not reach the receiver. Packet loss up to 5% is generally acceptable for audio applications.

Experiment Set 3

The third set of experiments looks at the performance of each protocol as various conditions in the network change: the number of nodes, the range of the transmission and range vs. bit rate for wifi. The last parameter is relevant for energy efficiency mechanisms, since battery power consumption is a major concern in rural MANETS with erratic power supplies [ref]. With wi-fi constantly on, most current phone batteries can be expected to deplete themselves in a few hours. This might tempt people to switch off their phones very often. However, only if a sufficiently large number of users keep their wi-fi on, will adequate network connectivity be maintained. To minimize battery drainage, the community radio could operate at fixed times in the day: for example for two hours in the morning and evening each.

Other common energy saving techniques in MANETS include (1) switching a percentage of nodes to their sleep state (2) decreasing the range of the transmission [20]. By decreasing the transmission range, less energy is required by an individual node. However, the fall-out is that more hops might be needed for the message to reach a recipient – involving more nodes in the process. Thus, energy consumption gets better distributed in the network, leading to an overall increase in the network's life.

For a radio application, switching nodes off might not make sense. However, we do explore option 2 (decreasing the transmission range) in our experiments.

The next section contains experimental results for all three sets of experiments and related discussion.

4. RESULTS AND DISCUSSION

As mentioned earlier, each experiment was repeated with ten different mobility files and five randomized runs for each mobility file. Thus, each value reported is the average of 50 simulation runs. Error-bars are included in graphs where possible. In certain graphs, the error bars hamper readability and hence their values are shown separately in tables in the Appendix.

Results of Experimental Set 1

As described earlier, the mobility pattern used for these experiments consists of 500 seconds of a dynamic network followed by 200 seconds of static nodes followed by another 500 seconds of mobility. The network consists of 100 nodes and each node employs EABA as the broadcasting algorithm.

The first question to answer is whether the mobility factor that each node calculates depending on its local view reflects the actual state of the network. Figure 5 plots the average mobility factor for four nodes over the entire length of the simulation. These four nodes are chosen from across the four categories of users described in section 6.1. Though the mobility factor at a node is a whole number, the graph plots the average across 50 runs and hence the values plotted are not whole numbers. As **Figure 5** shows, the mobility factor that nodes calculate when using EABA, clearly reflects the mobility in the network. The mobility factor dips to almost zero for all four categories, during the static phase and is significantly higher during the two dynamic phases. The nodes in Category b have a lower mobility factor even in the dynamic phase. Recall that Category b consists of those nodes that are at the periphery to begin with and move within the periphery zone they are in. This explains why their neighbor tables do not change drastically and hence their low mobility factor.

The graph in **Figure 5** also helps decide the threshold value of the mobility factor (mf), that should be used in EABA to switch between SBA and MaBA. A threshold value of mf > 0 is used for the switch, to correspond to the correlation between the state of the network and the average mobility factor that nodes calculate. We also experimented with mf > 1, but found that in this case SBA does not get sufficient time to run and thus the neighbor tables and memory at nodes do not sufficiently stabilize to be useful for MaBA.

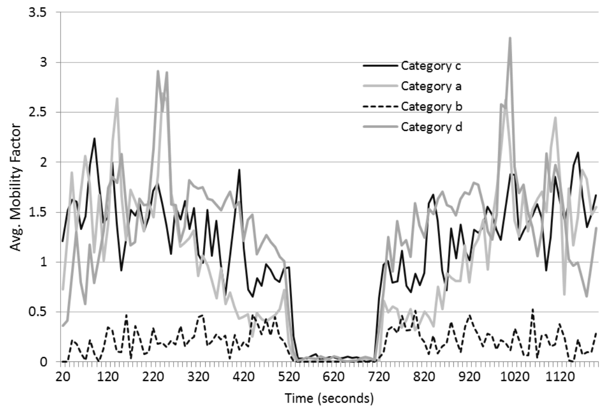

Figure 5: Average mobility factor for a node from each category, across time.

The next question to answer is wether EABA manages to switch between SBA and MaBA according to the value of mf and position of the packet in the stream.. **Figure 6** shows the number of nodes that use SBA versus MaBA at any given point in time. As can be seen, initially all broadcasting nodes start with SBA and then some nodes switch to MaBA. Some nFigure 8Figure 8Figure 8odes use MaBA even in the dynamic phase as there are pockets where nodes might not see changes in their neighbor table - if the mobile nodes are all moving together for example. In the static phase, there is a sharp drop in the number of nodes using SBA as almost all nodes switch to MaBA. Again in the dynamic phase, many nodes switch back to SBA, though some use MaBA as well.

This graph shows that EABA is quite successful in adapting to the mobility in the network and switching between SBA and MaBA. Note that though all nodes switch to SBA and the beginning of each stream, this is only for the first 10 packets (200 ms) of a stream, and does not show up in our recording granularity of every 10 seconds.

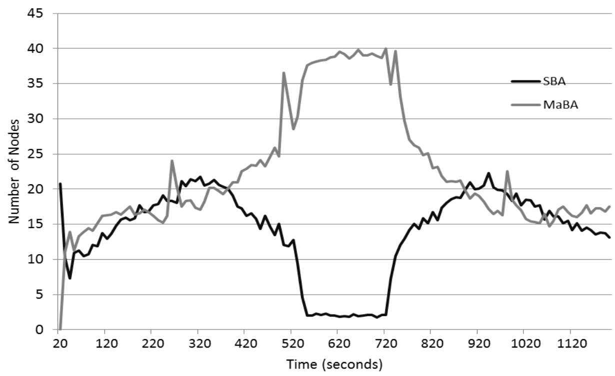

Figure 6: Number of nodes using SBA versus MaBA across time.

Results of Experiment Set 2

This set of experiments evaluates EABA (and compares its performance to three other broadcasting algorithms) on six different metrics described in the previous section. As mentioned earlier, the experiments were run for 10 different mobility files and 5 variations within each mobility file. The values reported are the average of these 50 simulation runs.

We report the results for reachability, efficiency, packet overhead, jitter, latency and packet loss.

Figure 7 plots the reachability for all the algorithms, for 100 nodes, for the mobility pattern described in the earlier section. As can be seen, EABA has reachability close to 90%, which means on average around 90% of the nodes receive a sound-cast. The reachability of SBA and SBA-mob is similar, while Flood has a slightly lower reachability. Closer investigation reveals that 100% reachability is not achieved because of network partitions. Some nodes, specifically the ones at the periphery are not connected to the network at many times. This is especially true in the beginning of the simulation when around 60% of the nodes are clustered at the center. As nodes move outwards, the network connectivity improves, as the periphery nodes can now use intermediaries to connect to the larger network. What this implies is that, users in category b might have to wait for the morning commute to be well underway before they can catch the sound-cast.

Flood exhibits lower reachability because of congestion: indiscriminate re-broadcasting cause congestion and packet-drops leading to many nodes not receiving adequate packets.

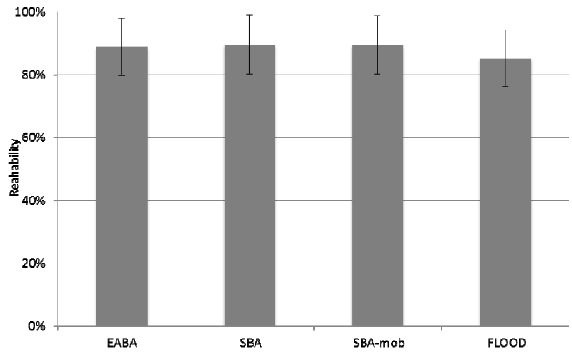

Figure 7: Reachability (delivery ratio) for the various broadcast algorithms, for 100 nodes.

Figure 8 plots the efficiency of the algorithms – how many nodes are required to re-broadcast the message for network wide reception. As is expected, Flood has the worst efficiency by definition: all nodes (those that are connected to the network) rebroadcast the message. Both SBA and SBA-mob have an efficiency less than 40% whereas EABA has a slightly higher percentage of nodes that rebroadcast (around 40%). This is because MaBA (the memory-aided part of EABA) has been tuned to be slightly more conservative than SBA (refer to the earlier section where MaBA is described), to ensure that reachability levels do not drop. Though its efficiency is slightly lower than SBA, EABA outperforms SBA in other dimensions that are critical for audio applications (as shall be seen shortly). In general, a slightly higher number of nodes rebroadcasting does not have a detrimental effect (in terms of congestion) on EABA's performance, since bandwidth is saved by lowering other overheads like hello packets.

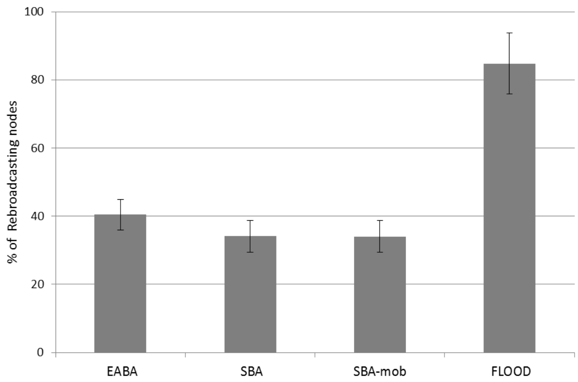

Figure 8: Efficiency of the various broadcast algorithms, for 100 nodes.

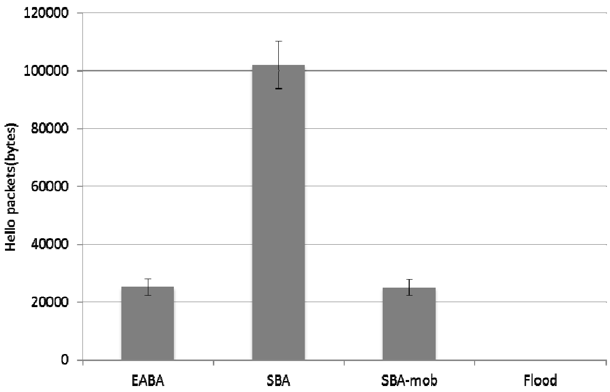

Figure 9: Packet overhead for the various broadcast algorithms, for 100 nodes.

Figure 9 contains the hello packet overhead for each algorithm. Recall that both EABA and SBA-mob adapt the frequency of hello-packets to the mobility in the network As can be seen, SBA has high packet overheads where as both EABA and SBA-mob have managed to significantly lower the usage of hello packets without significantly affecting their performance in other dimensions.. Flood of-course does not use any hello packets.

We see that on the metrics related to the network and broadcasting, all three algorithms (except flood) exhibit reasonably good performance: reachability is adequately high, and efficiency is significantly low. SBA does incur larger overheads in terms of bandwidth consumption, which do not affect its performance in this scenario of 100 nodes and no additional traffic in the network. However, as we shall see in the next set of experiments, in a bandwidth constrained environment, SBA's packet overheads play a detrimental role in its performance on the other dimensions. Assuming that our MANET might be used for other applications like phone calls in addition to the community radio, we can quickly see that algorithms that do not conserve bandwidth (especially in a bandwidth constrained WLAN environment like our scenario) would have a significant disadvantage.

We now discuss the metrics related to audio quality: latency, jitter and packet loss. Due to high levels of congestion, flood performs very poorly on all three dimensions and is not included in the graphs below.

Figure 10 shows the end-to-end latency incurred for the different algorithms. As seen, EABA exhibits far lower latency (around 23 ms) than both SBA and SBA-mob which are close to 70 ms: leading to a reduction of around one-third

According to our observation, for EABA, SBA and SBA-mob, the latency figures are in the absence of any congestion. The primary cause of latency for SBA and SBA-mob is the RAD component. Recall that in SBA, a node waits for a certain duration before deciding whether or not to rebroadcast a packet, called the RAD (Random Assessment Delay). Thus the RAD component introduces additional delay at each hop the packet traverses. Since MaBA does not require a RAD component, there are significant savings in latency when compared to SBA and SBA variants.

In general, a latency of up-to 80 ms is considered acceptable for VOIP data while higher values could substantially hamper the quality of the connection [21]. With other traffic on the network or slight delays caused by congestion, SBA easily crosses the 80 ms level as discussed shortly. EABA manages to keep latency well in check for most scenarios.

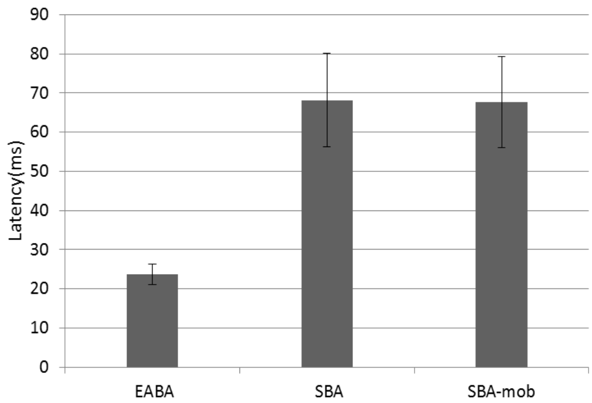

Figure 10: Latency incurred for the various broadcast algorithms, for 100 nodes.

It can be argued that more than latency, jitter plays a crucial role in audio quality, especially when the transmission is one-way (as in a broadcast). Figure 11 plots the jitter values for all three algorithms. We calculate the jitter at each node for each audio-stream and find the maximum jitter value across all nodes (the node with the worst jitter). This value is then averaged out for multiple streams across multiple simulation runs.

As can be seen from Figure 11, jitter is substantially less for EABA; 10 ms compared to more than 30 ms for the other two. Also, the error bars in this case plot the highest and lowest values in each case. As can be seen, when SBA and SBA-mob are employeed, some nodes experience very high jitter (in the tune of 75 ms). EABA experiences a maximum of 15 ms of jitter, which is well within acceptable limits. In general, values upto 30 ms are considered tolerable for VoIP data as most audio-codecs are cabable of handing some amount of jitter [21]. Clearly, the jitter caused by SBA would be challenging for an audio codec to smooth out.

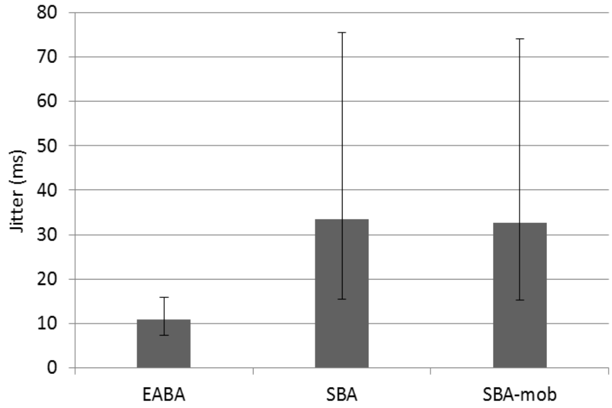

Figure 11: Jitter incurred when different broadcast algorithms are used.

EABA manages to keep jitter in check because of two primary features of its mechanism; does not use a RAD component that introduces variability and brings down congestion in the network by minimizing hello packets in the network.

We also measured packet-loss for all three algorithms; the results are shown in Figure 12. Packet delivery was measured as the percentage of packets in a stream that arrived successfully at each node. Note that nodes that did not receive packets because they were out of the network (out of reach of the source node: either directly or through hops), were disregarded for the packet-loss calculations.

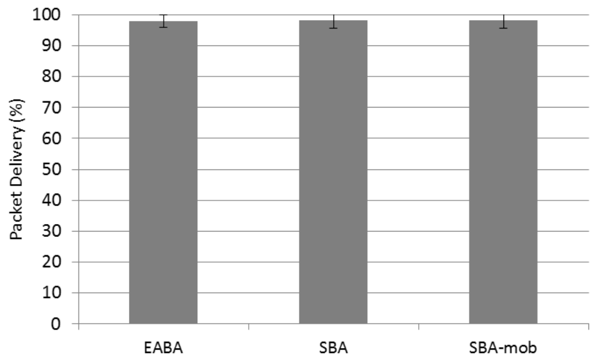

Figure 12: Packet Delivery Ratio for various broadcast algorithms.

Our experiments show that packet loss is sufficiently low (around 2%) for all three algorithms. This was expected as the current scenario does not incur any noticeable congestion, which is the main cause of packet loss. In general packet loss less than 5% is not noticeable and is considered acceptable for audio quality [19].

To summarize the results of this set of experiments, we find that EABA manages to substantially reduce jitter and latency in the audio-stream in comparison to SBA and its variants, while maintaining similar levels of reachability, efficiency and packet delivery in the network wide broadcast.

The next set of experiments test the three algorithms under varying conditions of network density, transmission range and congestion.

Results of Experiment Set 3

As mentioned earlier, the three broadcast algorithms were tested under varying conditions. The first change is to the density of the network : this is achieved by changing the number of nodes in the same 2000m X 2000m area. **Figure 13** and Figure 14 show the reachability and latency respectively, for the different networks, where nodes range from 60 to 140. As expected with only 60 or 80 nodes, the network is sparse and highly partitioned and has very low reachability (around 60%). As the number of nodes increase the reachability improves for all three algorithms. However, the latency incurred also increases as the network density increases. AS seen in Figure 14, the latency for SBA and SBA-mob more than doubles when the number of nodes is changed from 100 to 120. EABA on the other hand exhibits a slower increase in latency. However, for a network of very high density (140 nodes) all three algorithms incur very high latency. The primary explanation for this behavior is that after a certain point, as density increases, the congestion in the network takes over. EABA performs best in curbing latency as the decreased hello-messages and absence of RAD both help its performance. SBA-mob performs better than SBA, as the reduction in hello messages is still significant (even though both use the RAD component). However, for 140 nodes, the congestion in the network is too high even for EABA, even though it still incurs substantially lower latency than the other two.

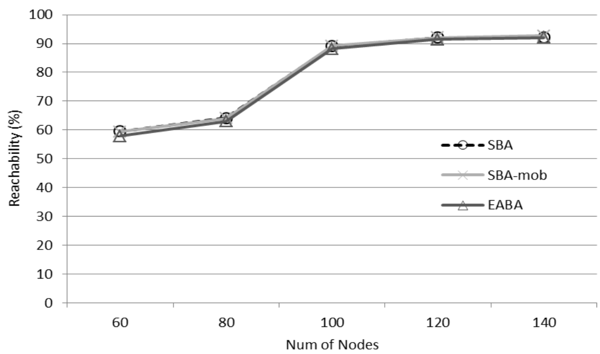

Figure 13: Reachability of the various broadcast algorithms for networks with varying density.

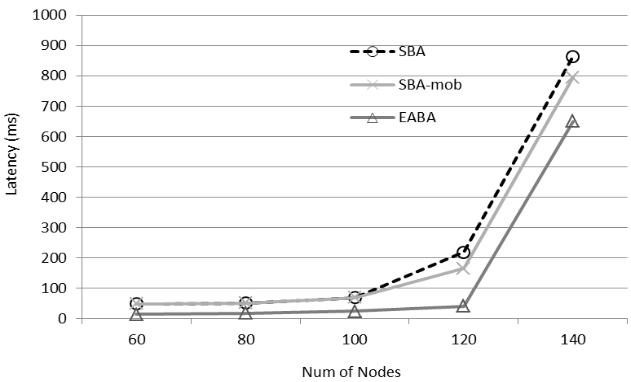

Figure 14: Latency for the various broadcast algorithms for networks with varying density

To test whether mechanisms to decrease power consumption would work with EABA, we tested the following: decreasing the transmission range of nodes to 250 meters, in a bid to decrease the power consumed at each node. This has two fallouts with 802.11b: for a range of 250 meters, we get a higher bit rate of 5.5 mpbs. Also, if the range decreases, then for 100 nodes, the network becomes very partitioned. Thus we test this range for more nodes in the network : 150, 200 and 250, as it is clear that it cannot be implemented for only 100 nodes in our scenario of 4 Km square. Figures 15-17 contain results for this scenario.

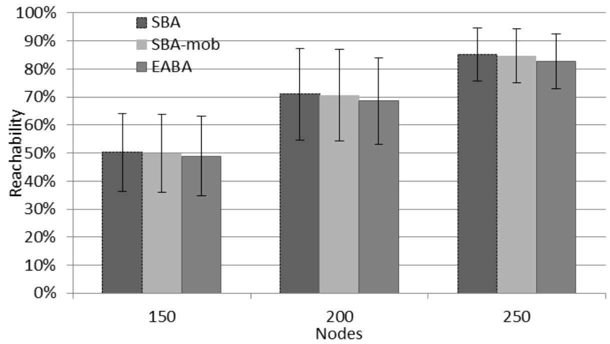

Figure 15: Reachability for 5.5 mbps and 250 m transmission range.

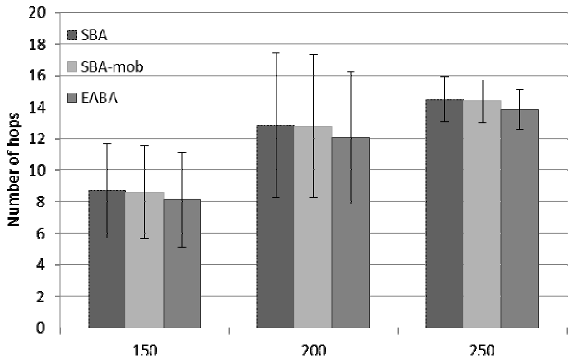

Figure 16: Average number of hops for 5.5 mbps and 250 m transmission range.

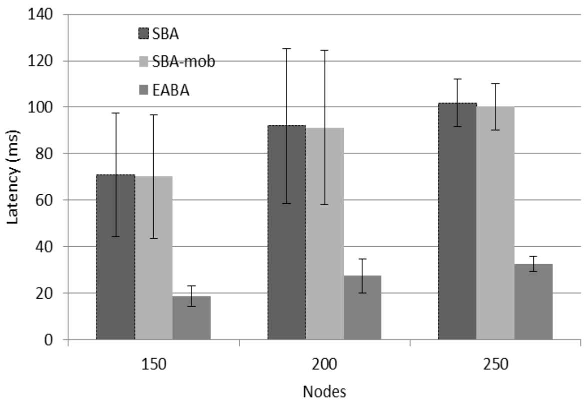

Figure 17: Latency incurred for 5.5 mbps and 250 m transmission range.

As can be seen, with a decreased range, the reachability is low (around 70%) even with 200 nodes in the network. At least 250 nodes are needed so that 80% reachability is achieved and the network is not highly partitioned. One fall-out of decreased range is that the number of hops a message has to go through increases substantially. In the case of 250 nodes, the average hop count is 14 for all three algorithms. While this does not substantially impact the performance of EABA whose latency remains around 30 ms, it has a large impact on the latency incurred by SBA and SBA-mob (from less than 70 ms to 100 ms, an increase of 43%). Similar increases in jitter was also observed for SBA and SBa-mob. The reason the increase in hop-count effects SBA and SBA-mob is because each hop adds an additional RAD delay leading to a substantial increase in end-to-end latency. EABA does not use RAD and hence its performance is largely unaffected.

These results point to the conclusion that decreasing the transmission range to 250 m (and simultaneously increasing the bit rate to 5.5 mbps) as an energy-saving option might work well for EABA, provided there are enough nodes for overall network connectivity.

5. CONCLUSIONS AND FUTURE WORK

Community radio has been seen as a powerful medium not only for broadcasting information but also for empowerment via the creation and dissemination of local content. However, lack of traditional communication infrastructure in remote and poor regions, mainly due to economic reasons, suggests looking at alternative approaches that might be easier to deploy and more affordable. MANETS (mobile ad-hoc networks) comprising entirely of low-cost phone sets provide an affordable and simple alternative. We envision a truly decentralized service, where the content consumers are also the producers, without the need for any centralized control/censorship on the content that is broadcast.

This paper explores the possibility of deploying such a rural community radio service on top of a peer-to-peer mobile based ad-hoc network. Since only one user can broadcast at a time, issues related to who broadcasts when etc. do naturally arise. While not trivial to solve, they are beyond the scope of the current work. We suggest that scheduling and usage norms be decided at the village level via weekly user meetings, but other solutions might be worth investigating as well.

This paper concentrates on designing and evaluating a suitable network-wide broadcast algorithm for this application. While the literature is awash with broadcasting algorithms for MANETS, they are ill-suited for the audio application in question. The paper details these design insufficiencies of the best of existing algorithms and goes on to describe a novel approach in which nodes use their past behavior to decide what to do with a packet. The basic idea is that the broadcast algorithm can be made more efficient as a stream of packets can potentially follow the same routes from the source to all nodes in the network

We propose EABA (Environs Aware Broadcasting Algorithm), in which each node builds recent memory of its own behavior. Each node also measures local network mobility around it, by gauging changes in its neighbor table. When the network is dynamic, a node uses SBA (an existing algorithm for broadcasting, which is considered one of the best but incurs high overheads), but once the local neighborhood is static or semi-static, the node switches to MaBa (Memory-aided Broadcasting Algorithm). Hence EABA has two modes: SBA is used to discover the neighborhood and update a nodes memory and MaBA uses that memory to decide on future actions.

Using extensive discrete-event simulations on a village mobility model we compare EABA's performance to SBA. Our results show that EABA is successful is substantially reducing the latency and jitter in the broadcast - critical elements for voice-quality in an audio-application. At the same time, EABA does not compromise on the reachability of the broadcast or the packet delivery ratio.

Our results also indicate that EABA can deal well with higher levels of network congestion than SBA, and is also more suitable for energy saving interventions like decreasing the transmission range and hence power consumption at nodes.

As future work, we plan to look at mechanisms to control power (transmission range) on a per node basis, depending on local network density. Hence a lesser range in dense part and longer range in sparse parts of the network, will maintain network connectivity while simultaneously conserving energy.

REFERENCES

[1] “The Serval Project.” [Online]. Available: www.servalproject.org. [Accessed: 11-Sep-2011].

[2] M. Adeyeye and P. Gardner-Stephen, “The Village Telco project: a reliable and practical wireless mesh telephony infrastructure,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, no. 1, p. 78, Aug. 2011.

[3] “TerraNet Project.” [Online]. Available: www.terranet.se. [Accessed: 11-Sep-2011].

[4] B. Girard, “A Passion for Radio: Radio Waves and Community.” [Online]. Available: http://comunica.org/passion/contents.htm. [Accessed: 16-Apr-2012].

[5] K. Ranganathan and A. Sarin, “A Voice for the Voiceless Peer-to-Peer Mobile Phone Networks for a Community Radio Service,” *Information Development*, vol. 28, no. 1, pp. 68–79, Feb. 2012.

[6] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, “The broadcast storm problem in a mobile ad hoc network,” *Wirel. Netw.*, vol. 8, pp. 153–167, Mar. 2002.

[7] E. Vollset, “A survey of reliable broadcast protocols for mobile ad-hoc networks,” 2003.

[8] “Efficient Broadcasting in Mobile Ad Hoc Networks,” *Mobile Computing, IEEE Transactions on*, vol. 8, no. 2, pp. 231–245, Feb. 2009.

[9] B. Williams and T. Camp, “Comparison of broadcasting techniques for mobile ad hoc networks,” in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing - MobiHoc ’02*, Lausanne, Switzerland, 2002, p. 194.

[10] M. Khabbazian and V. K. Bhargava, “Efficient Broadcasting in Mobile Ad Hoc Networks,” *IEEE Transactions on Mobile Computing*, vol. 8, no. 2, pp. 231–245, 2009.

[11] Wei Peng and Xi-Cheng Lu, “On the reduction of broadcast redundancy in mobile ad hoc networks,” in *2000 First Annual Workshop on Mobile and Ad Hoc Networking and Computing, 2000. MobiHOC*, 2000, pp. 129–130.

[12] Q. Zhang and D. P. Agrawal, “Dynamic probabilistic broadcasting in MANETs,” *Journal of Parallel and Distributed Computing*, vol. 65, no. 2, pp. 220–233, Feb. 2005.

[13] A. Munjal, T. Camp, and W. Navidi, “{SMOOTH}: a simple way to model human mobility,” in *Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, Miami, FL, USA, 2011, pp. 351–360.

[14] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, “SLAW: A New Mobility Model for Human Walks,” in *INFOCOM 2009, IEEE*, 2009, pp. 855 –863.

[15] F. Bai and A. Helmy, “A survey of mobility models,” *Wireless Adhoc Networks. University of Southern California, USA*, vol. 206, 2004.

[16] X. Zeng, R. Bagrodia, and M. Gerla, “GloMoSim: a library for parallel simulation of large-scale wireless networks,” in *Twelfth Workshop on Parallel and Distributed Simulation, 1998. PADS 98. Proceedings*, 1998, pp. 154–161.

[17] “Mesh Potato.” [Online]. Available: villagetelco.org/mesh-potato. [Accessed: 11-Sep-2011].

[18] H. Dong, I. D. Chakares, C.-H. Lin, A. Gersho, E. Belding-Royer, U. Madhow, and J. D. Gibson, “Speech coding for mobile ad hoc networks,” in *Conference Record of the Thirty-Seventh Asilomar Conference on Signals, Systems and Computers, 2004*, 2003, vol. 1, pp. 280–284 Vol.1.

[19] K. Kubik, G. Einicke, H. Zhang, and J. Homer, "Performance Comparison and Analysis of Voice Communication over Ad Hoc Network," Mar. 2007.
[20] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, *Mobile Ad Hoc Networking*. John Wiley & Sons, 2004.
[21] "Voice Over Internet Protocol (VoIP) : Telecompute Corporation." [Online]. Available: http://www.telecompute.com/voip.asp. [Accessed: 01-Apr-2013].