Delay-tolerant Adaptive Real-time Communication: A Case Study for Voice

Nagasai Panchakarla and Jörg Ott
Aalto University, School of Electrical Engineering
Department of Communications and Networking, Finland

ABSTRACT

Delay-tolerant networking allows dealing with temporary disconnections and non-available end-to-end paths. While many DTN system designs focus on asynchronous applications (such as messaging or content sharing), DTN communication may also be a suitable fallback for adaptive real-time applications if end-to-end communication fails. In this paper, we explore this idea to enhance the adaptivity of real-time voice for which we design a system that switches between RTP/UDP-based and RTP/DTN-based voice as a function of the observed RTT and loss rate.

1. INTRODUCTION

In the Internet with its best-effort service for packet delivery, applications are expected to continuously observe networking conditions and adapt to them so that the available resources (link and path capacity, buffer space) are shared "fairly". While the notion of fairness and especially the appropriateness of flow-level fairness have been subject to quite some debate (e.g., [5]), applications usually benefit from being able to adapt irrespective of fairness as they may reduce loss and latency.

Adaptation is well-explored and implemented for *elastic applications* that do not have bounds for data delivery and can thus easily delegate adaptation to transport protocols such as TCP or even less aggressive protocols that just feed on the residual capacity, as has been suggested for background traffic of peer-to-peer applications [43]. *Inelastic applications*, i.e., those with some time-bounded delivery requirements such as (interactive) audio and video, face more difficulties to adapt to changing network conditions.

While rate adaptation for video with its potentially substantial data rates has received quite some attention, low bit rate conversational audio (often at data rates of 16 kbit/s or less) appears to be below the congestion radar and considered to not matter much in today's networks.[1] In addition to error concealment techniques built right into the codecs, simple redundancy [33] or FEC schemes [10] may be added to compensate for losses at modest overhead.

[1]Audio streaming of Internet radio stations uses higher data rates, but commonly uses TCP as a transport and thus inherits adaptation, compensating temporary reductions with extended buffering.

ExtremeCom '12, March 10-14, 2012, Zürich, Switzerland.

This non-adaptation works apparently fine as long as the users have a continuous end-to-end path—e.g., a sufficiently stable access link to the network—so that occasional (congestion) losses remain the main reason for a perceived degradation in path capacity or latency. However, this does not necessarily hold for mobile nodes where interference, attenuation, or coverage gaps may impact path characteristics more heavily and less predictably. In such cases, physical connectivity may quickly become the bottleneck even for low bitrate audio, leading to large instant loss rates or substantial delay, which again results in losses when packets are dropped on the receiver side because they miss their playout deadline. When such situations occur, scaling back the transmission rate of individual audio flows, even if implemented, simply would not help. Other mechanisms for adaptation are required instead.

We build our voice adaptation idea on three observations: 1) Studies of skype users showed that users can tolerate more than the delay expressed by ITU-T [17], indicating that even for voice communication delay may be less important than voice quality [8]. The two-way alternate communication style of walkie-talkies (and their modern emulation over cellular networks as *push-to-talk*) hint that *expected* reduced interactivity may be perfectly acceptable. 2) TCP can deliver voice packets timely for a certain operational range of packet loss and RTT combinations, indicating that reliable transmission under reasonable conditions are not at odds with real-time requirements [6], even more so if the latter can be relaxed as per 1). 3) In the presence of errors—such as frequent packet losses, e.g., due to bad wireless link quality or temporary outages—trading off delay for intelligible speech appears sensible for many cases [18], especially since voice messaging has a history as a fallback for interactive voice if the peer cannot be reached.

Quite a few different voice communication systems were developed that relax the real-time requirement, yielding walkie-talkie-style or voice messaging-style conversation modes, as we will discuss in section 2. However, those offer clearly distinct modes of operation: interactive *or* walkie-talkie *or* voice messaging. In this paper, we take the idea of adaptive real-time communication one step further and integrate interactive and message-based communication into a continuum across which we move as function of observed loss rates and round-trip times. Based upon our conceptual outline [29], we present a system and algorithm design (section 3) and evaluate its performance compared to conventional real-time voice in two scenarios (section 4).

2. RELATED WORK

Multimedia applications are often referred to as *inelastic* because of their delay and capacity constraints imposed on the media path. Traditionally, real-time applications have avoided TCP and defined their own protocol functions on top of UDP, thus remaining in full control of scheduling, rate adaptation, and error repair. A common set of protocol functions is defined in the Real-time Transport Protocol (RTP) [40] in conjunction with many payload formats (see [13]). Packet losses can be mitigated by interleaving [35] and repaired by, e.g., applying FEC [33, 10, 3] or retransmissions [37, 21, 45]. While no complete rate adaptation protocol has been standardized so far,[2] numerous mechanisms have been explored in the past such as switching codecs [4], changing codec parameters as in multi-rate codecs such as AMR [42], and adjusting packet sizes. The above mechanisms may be combined (e.g., [25]).

The necessary feedback loops for observing network conditions to adapt transmission behavior may be based upon RTCP reporting [40, 30, 11]. Such feedback loop may also use information of transport protocols such as DCCP [34, 1] (or SCTP) and simply observe the net receive data rate over TCP as in adaptive HTTP streaming.

Thus, real-time applications are, in theory, able to adapt to network congestion and packet losses to some extent but, in practice, especially audio applications often simply don't. The key constraint seems to be the perceivably acceptable delay when trading off delay, loss, and data rate in rate control, interleaving, FEC, and retransmission schemes. According to ITU-T G.114 [17], the one-way delay for interactive should not exceed 150 ms, 150–400 ms are potentially tolerable, and delays above 400 ms are not acceptable. This limits the range of network conditions across which these applications can operate *if a strict notion of interactive voice shall be preserved*. As noted above, we believe that this strict notion can be relaxed if the user gets something in return: better quality and no or fewer outages or dropped calls. We have in the past experimented with disconnection tolerance for SIP-based voice calls, but this work was limited to buffering speech during outages and possibly automatically re-establishing calls [31].

Less interactive variants of voice conversation are known from walkie-talkies and the (cellular) push-to-talk (PTT) service, where conversations are two-way alternate, using a button to switch between transmission and reception. Examples include particularly PTT-over-Cellular (PoC) systems [44, 32, 20, 36, 9] and their evaluation [28]; extensions to multimedia content [2] or context awareness [16]; and distributed (server-less) systems (in ad-hoc networks) [38, 24, 7, 12]. They share their reliance on end-to-end data paths and focus by design on the PTT-style operation.

Finally, numerous systems were designed for asynchronous voice communications. Honicky et al. [15] propose using a mobile phone primarily for voice messaging, focusing on asynchronous communication; they outline the potential benefits of an asynchronous model, even with infrastructure. Heimerl et al. [14] extend this idea [15] by developing a prototype cellphone system with voice messaging and explore its value for Ugandan users via trial deployments. Scholl et al. [39] present rural Telemedicine networks based upon store-and-forward VoIP and advocate building such networks on the basis of DTN. Our own previous work [19, 18] has investigated voice messaging in mobile ad-hoc environments. All these system are constrained to asynchronous operation.

3. DELAY-TOLERANT ADAPTIVE MEDIA

Instead of requiring a user to choose between real-time interactive voice, two-way alternate voice, and messaging, our *delay-tolerant adaptive media (DAM)* design combines elements of all three to offer the most suitable means of communication achievable under the given circumstances. In a nutshell, starting out from synchronous voice call with RTP/UDP-based audio, the endpoints monitor path characteristics and switch to DTN-based voice messaging when required and back to RTP when deemed feasible. The switching is important as plain DTN bundle headers impose additional per-message overhead for small real-time voice packets.

Below, we briefly discuss call signaling to set up DAM voice calls between two endpoints using SIP and then focus our attention on the adaptive media transport. Our emphasis is on dealing with substantial path impairments that cannot be dealt with using repair mechanisms because the end-to-end path may not exist for some time. Therefore, while our mechanisms could be complemented by error repair and concealment techniques to deal with individual losses or short loss bursts, we do not consider such combinations yet. We similarly limit our DTN-based protocol to hop-by-hop reliability and do not add end-to-end error protection mechanisms on top. Both are improvements for further study.

3.1 Call Signaling

We assume that a voice call starts out as a regular SIP call with a three-way SIP INVITE handshake during which the two nodes carry out an SDP-based offer/answer exchange to determine which media to use as well as which codecs and transports are available. The initial exchange provides the basis for synchronous communication by communicating transport addresses to use for RTP, avoids the uncertainty how to encode voice messages in purely asynchronous scenarios as discussed in [18], and allows the peers to signal their DAM support.

The nodes use an additional media-level attribute to indicate DAM support along with the required parameters: the DTN endpoint identifier (EID) to send voice bundles to, the maximum voice message size per bundle, the supported payload types for bundle-based transport, and an optional convergence layer address. The following SDP fragment illustrates, using a strawman syntax, a case in which the sending node is prepared to receive RTP with PCM audio (static payload type 0) on port 54321 at mobile-1234.operator.example.com and RTCP on port 54322, using AVPF [30] (see below). Alternatively, the node can receive DTN-based RTP at EID dtn://mobile-id/rtp/audio-id containing a maximum of 10 s audio in PCM (again payload type 0).

```
m=audio 54321 RTP/AVPF 0
a=rtcp:54322
c=IN IP4 mobile-1234.operator.example.com
a=dam:dtn://mobile-id/rtp/audio-id mtime=10s pt=0
```

A node receiving an $a = dam$ attribute will respond with such if it supports DAM and wants to enable this option for the call; otherwise, it ignores it. DAM is enabled only if both offer and answer contain this attribute.

Call teardown as well as any updates to call or media state would occur via further regular synchronous SIP signaling. We do not consider the impact of temporary disconnections on SIP, nor any DTN-based SIP signaling in this paper, but focus on media transport and leave signaling-related discussions for future study.

3.2 Media Transport

For UDP-based real-time transmission, we simply use RTP and the codec-specific packetization formats. Since we want to use RTCP for frequent RTT and loss measurements, we cannot rely on the basic audio-visual profile (AVP) because it does not allow RTCP packets to be sent more frequently than every five seconds. Therefore, we use the AVP feedback profile (AVPF) [30] and configure the RTCP rate to support multiple packets per second.

For DTN-based media transport, we use the same RTP packets as for UDP, but carry them inside DTN messages (bundles) de-

[2] An IETF working group for video rate control—RTP Media Congestion Avoidance Techniques (rmcat)—is formed at the time of writing (http://www.ietf.org/iesg/evaluation/rmcat-charter.txt).

fined in the Bundle Protocol [41]. We provide a minimal framing to stack multiple of them in the same bundle payload: since a bundle payload is similar to a connection-oriented byte stream, we use a 16-bit prefix per RTP or RTCP packet to indicate the respective packet size as per [23]. Preserving complete RTP packets inside a bundle has the advantage that playout timing across media samples carried in the same message is preserved, even if pauses occur in the media stream (e.g., when using voice activity detection).

RTP and RTCP packets may be mixed inside a bundle, but we also send separate bundles carrying only RTCP for measuring RTT and reachability when media packets are sent less frequently.

3.3 Adaptation Algorithms

During a call, a sender needs to determine whether to use UDP or DTN for media transmission. Below, we present two initial algorithms to perform this adaptation: algorithm 1 for the sender and algorithm 2 for the receiver side. The algorithms are based on RTCP reports to switch to send RTP packets as DTN bundles if packets losses and delays are above the desirable levels and thus we assume RTCP operation as per [40, 30]. The algorithms are based on the assumption (backed by measurements we conducted) that DTN-based transmission is preferable when the delays and packets losses are high. Even though voice communication may be symmetric, our algorithms take their transmission choices independently per direction (but they could be coupled).

Algorithm 1 Sender

```
Send packets via: DTN OR UDP mode
  Initialize: Mode := UDP
  if RTCP received then
    if RTCP received via UDP then
      if p_RTPloss > p_ν then
        Mode := DTN
      end if
    end if
    if RTCP received via DTN then
      if ProbeFlag = Set then
        if T_avgDelay < T_κ then
          Mode := UDP
        end if
      end if
    end if
  else
    if N_RTCPloss > N_ζ then
      Mode := DTN
    end if
  end if
  if Mode = DTN then
    PacketizeRTPPacketsInBundle
    sendBundleOverDTN
    sendProbePacketsOverUDP
  else
    sendPacketsOverUDP
  end if
```

Algorithm 1 shows the sender operation: Initially the packet transfer uses RTP/RTCP over UDP. The sender regularly receives RTCP receiver reports from the receiver and observes packet loss and RTT. As long the loss thresholds is not exceeded, the sender remains in UDP mode (larger delays are compensated by more tolerance in playout delay adaptation). If the packet loss exceeds an acceptable threshold (p_ν), the mode is switched to DTN-based transmission. In any case, if N_ζ consecutive RTCP reports are missing (i.e., no RTCP was received for N_ζ times the RTCP transmission interval) then the transfer mode is set to DTN.

Algorithm 2 Receiver

```
Receive packets via: DTN OR UDP
  Initialize RTP/RTCP
  if RTPreceivedoverUDP then
    if RTPType = Probe then
      CheckProbeSequence
      Update ProbeFlag
    else
      UpdateRTCPReceiverReportBlock
      sendRTCPViaUDP
    end if
  else
    ExtractRTPpacketsFromBundle
    UpdateRTCPReceiverReportBlock
    UpdateRTCPProbeFlag
    sendRTCPViaDTN
  end if
```

In DTN mode, groups of RTP packets are sent as DTN bundles. In addition, the sender sends probe packets at regular time intervals; if the receiver receives N_ι consecutive probe packets, it sets a corresponding probe flag in subsequent RTCP reports to the sender.[3] The receiver reports about RTP/DTN and probe packets by sending its RTCP packets as DTN bundles. The sender checks the RTCP reception statistics and if the average delay is less than the maximum threshold (T_κ), the mode will be switched back to UDP, provided that the probe flag is set in the received RTCP report (which implicitly indicates a low loss rate for UDP). Using the probe packets and flag ensures the availability of a UDP-based end-to-end path, which cannot be taken for granted if DTN is used otherwise.

Algorithm 2 shows that the receiver mirrors the sender transport in the reporting behavior, sending RTCP reports over the same transport as the RTP packets are received. This ensures that the RTTs of the correct transport are measured. If probe packets are received (via UDP), the corresponding report block is updated and included in the next RTCP receiver report (over DTN) when N_ι consecutive probes were received to indicate that switching back to RTP/UDP may be possible.[4]

In both modes (not shown), the sender captures, encodes, timestamps, and encapsulates media samples and the receiver extracts the payloads from the RTP packets and inserts them into the playout buffer for rendering when due.

Based upon our measurements and simulations, we use the following parameters: $T_\kappa = 500\,\text{ms}$, $p_\nu = 0.1$, $N_\zeta = 3$, and $N_\iota = 5$. We aggregate 40 RTP packets to form a single bundle payload.

4. EVALUATION

For our evaluation, we use voice traffic patterns obtained by capturing the RTP packets (on the sender side) of a voice call between two SIP endpoints located in Finland and France. The packets were generated by Linphone, an open source VoIP software client on MS Windows 7 and captured using Wireshark. The raw packets in the trace are all 214 bytes, yielding 160 byte audio payloads[5], equiv-

[3] The probe packets are sent in the same RTP session using a different SSRC identifier and a different payload type, so that they can be distinguished on the receiver side from regular audio packets.

[4] We assume symmetric reachability via IP for now.

[5] Excluding headers: 14 bytes Ethernet, 20 bytes IP, 8 bytes UDP, and 12 bytes RTP.

alent to 20ms of voice and were collected in the *rtpdump* format. We use ns-2 in our simulations. The sending node reads the rtpdump traces and generates the corresponding RTP stream using the RTP traffic generator, an application level extension to ns-2. For RTP/UDP mode, this results in an RTP data stream exactly as in the rtpdump. To implement the DAM mode, we use the ns2dtn package [22] which extends ns2 to support DTN, so that an application can choose to send packets via DTN or UDP on the sending side. When sending RTP over DTN, the source node aggregates several RTP packets inside a DTN bundle and sends the bundle to the destination node.

We measure the performance of the protocol by calculating the R-value for the two modes of operation. The Transmission Rating Factor, R value is the scalar quality rating of the E model calculations. R values are calculated based upon the simulation output files, considering packet delay, losses and jitter. We compare the R-values observed in our simulations against the reference value R_{ref} of the digital audio signal received over a perfect network. We use the difference $\Delta R = R_{ref} - R$ [26] as performance indicator for both RTP/UDP and DAM. $\Delta R = 0$ indicates the maximum achievable audio quality, $\Delta R = 93$ indicates the worst (equivalent to signal, i.e., packet, loss).

4.1 Helsinki Trace

Our first evaluation setup investigates voice communication via a 3G network between a fixed node and mobile node on a bus route between Helsinki and Espoo, Finland.

We first performed packet-based measurements by collecting traces form a UMTS data connection of a laptop user (HP Probook 4320s, MS Windows 7) on a bus to a passively observing server in our lab. The mobile node generated a steady stream of RTP packets using *rtpsend*, the packets matching the characteristics of the previously recorded RTP traces (160 bytes G.711 payload, encapsulated in RTP/UDP/IP, sent every 20ms). The fixed node recorded the incoming packets using *rtpspy*. The measurements were taken for some 12 minutes while the bus was moving and included stops. In order to measure accurate packet delays, both the nodes were synchronized before the experiment using NTP. The traces provide packet delays and losses for the RTP traffic.

The simulation consists of two well-connected fixed nodes in the ns2dtn setup. Both nodes modulate their packet reception to match the delay and loss patterns collected in the measurements for the bus routes. We run a unidirectional media flow from a sender to a receiver flow as per the RTP traces. The sending node generates 214 bytes of packets for every 20 ms for around 12 minutes. The destination node sends RTCP receiver reports regularly for every 2 seconds. For RTP/UDP, all packets are sent using UDP. For DAM, the sending node executes the adaptation algorithm describe above. The destination nodes acknowledges each bundle with a receiver report and also it sends RTCP receiver reports inside a DTN bundle at regular RTCP intervals. The DTN RTCP report consists of average delay of the packets in a RTCP interval. The bundle delays were calculated from receiving time stamp and sending timestamp embedded in the bundle meta data.

Results

Figure 1 shows the voice quality differences (ΔR) for two distinct runs over time: the crosses (+) indicate individual values for UDP-based transmission, the dots those for the DAM case. The peaks in the graph coincide with increases in packet losses and RTT, the latter of which causes packets to be discarded due to late arrival. Both graphs show that the DAM algorithms yields better audio quality if the network conditions worsen sufficiently long for the switching algorithm to kick in (e.g., at 200 s and around 550 s for the top and around 350 s and 620 s for the bottom plot). For short connectivity impairments, the quality is equal. Reviewing the logs in detail reveals that out of some 71,100 R value samples, the DAM mechanism improves quality in 4–8% of the time but yields a reduction in only about 0.5%. Due to the constantly good connectivity, the quality is equal for more than 90% of the time.

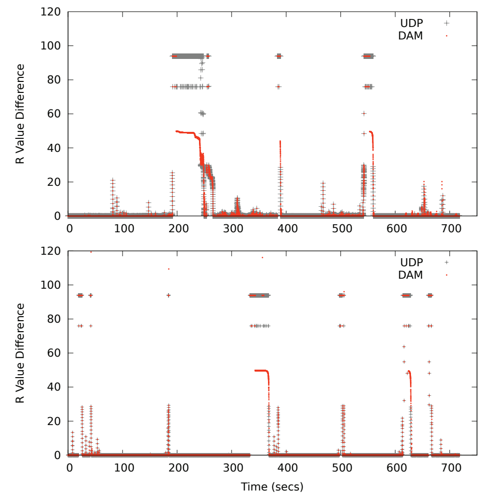


Figure 1: Helsinki Traces

4.2 Ad-hoc Communications

The above scenario has two limitations: 1) disconnections are rare and 2) there is no real need for DTN-based communications because we are concerned only with a single challenged link, which could also be bridged by TCP. We therefore complement the above scenario by one featuring ad-hoc communication between mobile nodes as a more challenging environment. We use 40 mobiles nodes using AODV or DTN with epidemic routing and choose one node pair for bidirectional real-time communication.

We use the Random Waypoint (RWP) mobility model for node mobility simulation. RWP has known deficiencies, including not reproducing human contact patterns appropriately and leading to infrequent contacts with often only short contact times. However, we consider the latter a feature because this yields a particularly challenging communication setup. We generate RWP movement files for 40 mobile nodes using *setdest* of the ns-2 (ns-allinone) package. Nodes move at random speeds of $[0; 20]$ m/s, with a pause time of 2 s. We use simulation area sizes ranging from 50×50 m to 2000×2000 m.

Finally, we use the dei80211mr library [27] to simulate the wireless channel as required by the ns2dtn [22] for ad-hoc network simulations, because the dei80211mr library provides a more realistic channel model compared to default model in ns-2. The library provides functionality for different transmission rates, modulation and coding schemes defined in the IEEE802.11b/g standards. Table 1 summarizes the 802.11g parameters used in our simulations.

Results

We explored the impact across area dimensions from 50×50m to 2000×2000m and calculated the mean ΔR for both RTP/UDP and the DAM mechanism. The gain achievable by using DAM is largest for mid-size areas of 250×250m through 750×750m, while it diminishes to zero for small areas (about 150×150m)—because

Table 1: IEEE 802.11g related simulation parameters.

Parameter	Value	Parameter	Value
noise_	9.75e-12 W	CWMin_	16
CSThresh_	1e-10 W	CWMax_	1024
Pt_	0.0178 W	RTSThreshold_	0 B
freq_	2.437e6 Hz	ShortRetryLimit_	8
L_	1.0	LongRetryLimit_	5
useShortPreamble_	true	SlotTime_	0.000009 s
gSyncInterval_	0.00001 s	SIFS_	0.000016 s

many packets are delivered either way in a perfectly connected dense network—and large area (2000 $\times$ 2000m)—because virtually no messages are delivered at all either way in a very sparse network. For very small networks, DAM performs slightly worse than RTP/UDP because of occasionally switching to RTP/DTN mode, whose flooding is detrimental to bundle delivery in dense networks. Across almost all densities, we find occasions where DAM quality is worse than RTP/UDP, which typically can be attributed to the latency when switching back to UDP and that RTP/DTN incurs higher network load.

Figure 2 shows the results for one node pair for the interesting node densities with a high gain. The connectedness of the network is reflected in longer periods with $\Delta R = 0$ (a), whereas sparse networks with many packet or bundle losses exhibit more periods with $\Delta R = 93$, which indicates total audio loss (c). We plot results for static DTN message size of 1 s (top) and 5 s (bottom).

Expectedly, RTP/UDP communication performs well as long as the network is sufficiently connected but otherwise quality immediately drops sharply to the worst values. In contrast, DAM is able to maintain some medium level speech quality for most of the time (figure 2a and b), even though short periods of lost bundles repeatedly occur. However, since those are mostly interspersed with received bundles, error resilience mechanisms could mitigate losses in the small and medium scenarios at least. When the network becomes sparser (c), at least occasional communication remains possible with DAM while RTP/UDP fails virtually all the time.

While improving voice quality of individual short time periods is a good start, we also recognize an important issue of mode switching: the flow of the conversation is disrupted and the users have to adapt to the new situation, especially when switching from UDP to DTN. This would have an impact of user-perceived quality not captured by the E model. To provide an initial estimate on the impact on the user, we also plot which mode is chosen by the DAM algorithm (solid line at the top in figure 2) and thus the frequency of mode switches. We can see that mode switches do not occur very frequently for most of the time, but also that conditions with oscillations may occur. This frequency may partly be reduced by choosing larger messages—moving from 1 s to 5 s messages as shown—but this comes at the expense of higher latency and higher message load (since messages are flooded) and thus partly lower R values. Investigating algorithms to determine message sizes adaptively and hysteresis functions avoid oscillations is subject to future work.

5. CONCLUSION

In this paper, we have begun investigating expanding the operational range adaptive real-time communication towards mode delay tolerance in order to cope with challenging network scenarios. *Technically*, the basic idea to dynamically adapt to packet losses and delays by moving between (unreliable) packet-based RTP and more robust message-based RTP appears feasible for our initial evaluation scenarios. Various improvement options, e.g., adding error resilience mechanisms and adapting message sizes, require further study as do more scenarios, higher loads, and other codecs.

We have not yet touched upon *usability*, the probably more interesting issue to be addressed: How to convey significant adaptation steps to the humans on a call? Would the potential gain make up for the less predictable quality? How to manage user expectations? To assess these, user trials using a realistic implementation will be required; developing a prototype is subject of our ongoing work.

Acknowledgment

This work was partly funded by Teknologiateollisuus ry and the Academy of Finland in the RESMAN project (grant no. 134363).

6. REFERENCES

[1] Vlad Balan, Lars Eggert, Saverio Niccolini, and Marcus Brunner. An Experimental Evaluation of Voice Quality over the Datagram Congestion Control Protocol. In *Proceedings of IEEE INFOCOM*, 2007.

[2] N. Blum and T. Magedanz. PTT + IMS = PTM - towards community/presence-based IMS multimedia services. In *Proc. 7th IEEE International Symp. on Multimedia*, 2005.

[3] Jean-Chrysostome Bolot, Sacha Fosse-Parisis, and Don Towsley. Adaptive FEC-based error control for Internet telephony. In *Proceedings of IEEE INFOCOM*, volume 3, pages 1453–1460, 1999.

[4] Jean-Chrysostome Bolot and Andrés Vega-García. Control Mechanisms for Packet Audio in the Internet. In *Proceedings of IEEE INFOCOM*, volume 1, pages 232–239, 1996.

[5] Bob Briscoe. Flow Rate Fairness: Dismantling a Religion. *ACM Computer Communications Review*, 37(2):65–74, April 2007.

[6] Eli Brosh, Salman Abdul Baset, Dan Rubenstein, and Henning Schulzrinne. The Delay-Friendliness of TCP. In *Proc. of ACM SIGMETRICS*, 2008.

[7] L.-H. Chang, C.-H. Sung, H.-C. Chu, and J.-J. Liaw. Design and implementation of the push-to-talk service in ad hoc VoIP network. *IET Communications*, 3(5):740–751, 2009.

[8] Kuan-Ta Chen, Chun-Ying Huang, Polly Huang, and Chin-Laung Lei. Quantifying skype user satisfaction. In *Proceedings of ACM SIGCOMM*, 2006.

[9] Rui Santos Cruz, Mario Serafim Nunes, Guido Varatojo, and Luis Reis. Push-to-Talk in IMS Mobile Environment. In Jaime Lloret Mauri, Vicente Casares Giner, Rafael Tomas, Tomeu Serra, and Oana Dini, editors, *ICNS*, pages 389–395. IEEE Computer Society, 2009.

[10] Adam H. Li (ed.). RTP Payload Format for Generic Forward Error Correction. RFC 5109, December 2007.

[11] Ladan Garai. RTP with TCP Friendly Rate Control. Internet Draft draft-ietf-avt-tfrc-profile-09, Work in progress, 2007.

[12] A. Hafslund, Toan Tuan Hoang, and O. Kure. Push-to-talk applications in mobile ad hoc networks. In *Proceedings of the 61st IEEE Vehicular Technology Conference*, 2005.

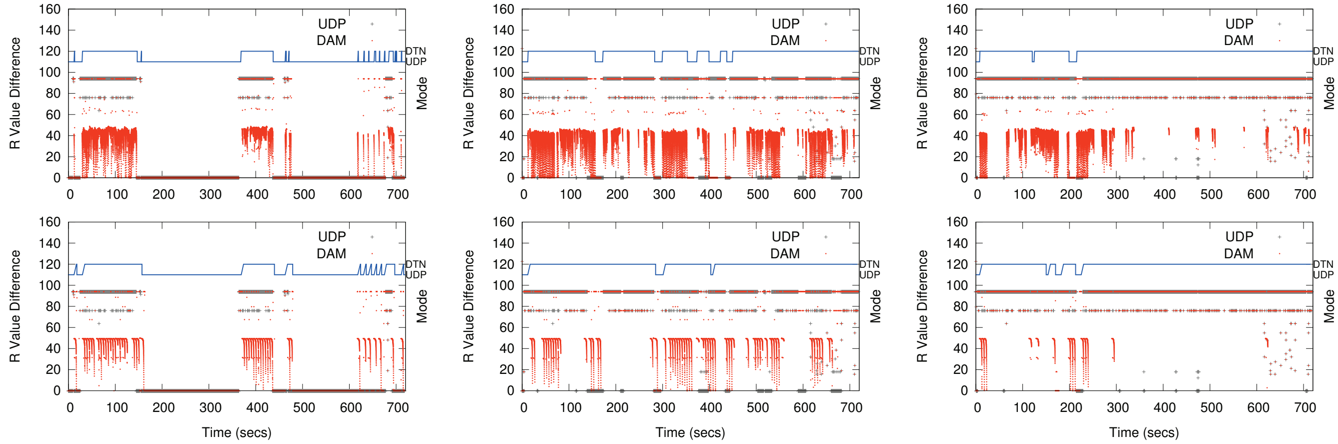
[13] Mark Handley and Colin Perkins. *Guidelines for Writers of RTP Payload Format Specifications*, 1999. RFC 2736.

[14] Kurtis Heimerl, RJ Honicky, Eric Brewer, and Tapan Parikh. Message phone: A user study and analysis of asynchronous messaging in rural uganda. In *Proc. ACM Workshop on Networked Systems for Developing Regions*, 2009.

[15] R.J. Honicky, Omar Bakr, Michael Demmer, and Eric Brewer. A message oriented phone system for low cost connectivity. In *Proc. of ACM HotNets VI*, 2007.

[16] Jenq-Muh Hsu, Wei-Bin Lain, and Jui-Chih Liang. A Context-Aware Push-to-Talk Service. In *Proc. 2nd International Conference on Multimedia and Ubiquitous Engineering*, 2008.

[17] International Telecommunication Union, Telecommunication Sector (ITU-T). Transmission Systems and Media, General Recommendation on the Transmission Quality for an Entire



(d) Area Size 250 × 250m (small)

(e) Area Size 500 × 500m (medium)

(f) Area Size 750 × 750m (large)

Figure 2: Ad-hoc communications: Random waypoint with 40 nodes, 1s messages (top) and 5s message (bottom)

International Telephone Connection; One-Way Transmission Time. Recommendation G.114, March 1993.

[18] Md. Tarikul Islam, Anssi Turkulainen, , and Jörg Ott. Voice Messaging for Mobile Delay-Tolerant Networks. In *Proceedings of ACM COMSNETS*, 2011.

[19] Md. Tarikul Islam, Anssi Turkulainen, Teemu Kärkkäinen, Mikko Pitkänen, and Jörg Ott. Practical Voice Communications in Challenged Networks. In *Proceedings of the ExtremeCom workshop*, 2009.

[20] P. Kim, A. Balazs, E. van den Brock, G. Kieselinann, and W. Bohm. IMS-based push-to-talk over GPRS/UMTS. In *Proc. of IEEE WCNC*, 2005.

[21] Michael Kropfberger and Hermann Hellwagner. Evaluation of RTP Immediate Feedback and Retransmission Extensions. In *Proc. of IEEE ICME*, 2004.

[22] Jani Lakkakorpi, Mikko Pitkänen, and Jörg Ott. Adaptive Routing in Mobile Opportunistic Networks. In *Proc. of ACM MSWiM*, October 2010.

[23] John Lazzaro. Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packtes over Connection-Oriented Transport. RFC 4571, July 2008.

[24] Jiun-Ren Lin, Ai-Chun Pang, and Yung-Chi Wang. iPTT: peer-to-peer push-to-talk for VoIP. *Wireless Communications and Mobile Computing*, 8(10):1331–1343, 2008.

[25] Johnny Matta, Christine Pépin, Khosrow Lashkari, and Ravi Jain. A Source and Channel Rate Adaptation Algorithm for AMR in VoIP Using the Emodel. In *Proceedings of ACM NOSSDAV*, pages 92–99, 2003.

[26] Wojciech Mazurczyk and Zbigniew Kotulsk. Adaptive VoIP with Audio Watermarking for Improved Call Quality and Security. *Information Assurance and Security*, 2(3):226–234, 2007.

[27] University of Padova. dei80211mr: a new 802.11 implementation for NS-2. http://www.dei.unipd.it/wdyn/?IDsezione=5090.

[28] Eoin O'Regan and Dirk Pesch. Performance Estimation of a SIP based Push-to-Talk Service for 3G Networks. In *Proceedings of the 5th European Wireless Conference*, 2004.

[29] Jörg Ott. Towards more adaptive voice applications. In *Proc. of the International Symposium On Leveraging Applications of Formal Methods, Verification and Validation*, 2010.

[30] Jörg Ott, Stephan Wenger, Noriyuki Sato, Carsten Burmeister, and Jose Rey. Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-based Feedback (RTP/AVPF). RFC 4585, July 2006.

[31] Jörg Ott and Lu Xiaojun. Disconnection Tolerance for SIP-based Real-time Media Sessions. In *Proc. of ACM MUM*, December 2007.

[32] A. Parthasarathy. Push to talk over cellular (PoC) server. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, 2005.

[33] C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J.C. Bolot, A. Vega-Garcia, and S. Fosse-Parisis. RTP Payload for Redundant Audio Data. RFC 2198, September 1997.

[34] Colin Perkins. RTP and the Datagram Congestion Control Protocol. Internet Draft draft-ietf-dccp-rtp-07, Work in progress, 2007.

[35] Colin Perkins, Orion Hodson, and Vicky Hardman. A survey of packet loss recovery techniques for streaming audio. *IEEE Network*, pages 40–48, Sep/Oct 1998.

[36] S.K. Raktale. 3PoC: an architecture for enabling push to talk services in 3GPP networks. In *Proc. IEEE International Conference on Personal Wireless Communications*, 2005.

[37] J. Rey, D. Leon, A. Miyazaki, V. Varsa, and R. Hakenberg. RTP Retransmission Payload Format. RFC 4588, July 2006.

[38] V. Ronnholm. Push-to-Talk over Bluetooth. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, 2006.

[39] Jeremiah Scholl, Lambros Lambrinos, and Anders Lindgren. Rural telemedicine networks using store-and-forward Voice-over-IP. *Stud Health Technol. Inform.*, 150:448–452, 2009.

[40] Henning Schulzrine, Stephen Casner, Ron Frederick, and Van Jacobson. *RTP: A Transport Protocol for Real-Time Applications*, July 2003. RFC 3550.

[41] Keith Scott and Scott Burleigh. Bundle Protocol Specification. RFC 5050, November 2007.

[42] J.W. Seo, S.J. Woo, and K.S. Bae. A study on the application of an AMR speech codec to VoIP. In *Proc. of IEEE ICASSP*, 2001.

[43] S. Shalunov, G. Hazel, J. Iyengar, and M. Kuehlewind. Low Extra Delay Background Transport. Internet Draft draft-ietf-ledbat-congestion-09, Work in Progress, 2011.

[44] Lin-Yi Wu, Meng-Hsun Tsai, Yi-Bing Lin, and Jen-Shun Yang. A client-side design and implementation for push to talk over cellular service. *Wireless Communications and Mobile Computing*, 7(5):539–552, 2007.

[45] Árpád Huszák and Sándor Imre. DCCP-based Multiple Retransmission Technique for Multimedia Streaming. In *Proc. of ACM MoMM*, 2008.