Implementation of a Protocol for Cooperative Packet Recovery Over Hybrid Networks

Fraida Fund, Thanasis Korakis, Shivendra S. Panwar
Department of Electrical and Computer Engineering
Polytechnic Institute of New York University
Brooklyn, New York, USA
ffund@nyu.edu, korakis@poly.edu, panwar@catt.poly.edu

ABSTRACT

In this work, we consider the challenge of delivering high-quality multimedia content to users concentrated in a small physical location, e.g., a sports stadium in which fans may view extra video content on their mobile devices. Because each user experiences a different wireless channel condition, it is difficult to deliver this content efficiently to all users. A hybrid network architecture has been proposed that uses a peer to peer exchange of packets over an assistant network to supplement the primary infrastructure network in this scenario. We describe an implementation of this popular technique which may be used in a variety of network environments and applications. In trials on a wireless networking testbed, using WiMAX as the primary network and WiFi as an assistant network, we find that our implementation can recover up to 92% of packets lost over the primary network.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

Keywords

Wireless Testbed; Experimentation; Hybrid Networks

1. INTRODUCTION

Consumer demand for high-quality multimedia content has been a major driver for innovation in wireless networks. One particularly challenging task in this area is that of delivering live multimedia content to a large number of users concentrated in a densely packed group. This may arise, for example, in a stadium where sports fans can view personal video feeds on their mobile devices.

One instructive case study is that of the wireless network provisioning for the 2012 London Olympic and Paralympic Games. There, carriers tried to minimize congestion by using small cellular or WiFi cells in areas that were expected to see high traffic loads [3, 11]. However, this approach presents several challenges. Within a stadium "bowl," there are generally no natural physical boundaries that provide isolation between the small cells. Therefore, if frequencies must be reused between cells, it becomes difficult to keep inter-cell interference at an acceptable level. Unexpected sources of interference can degrade user experience; for example, during wireless network trials before the London Games, a crane carrying a wireless camera for transmitting video to the driver in the cab caused intermittent outages for WiFi users in a hockey stadium [11]. Also, macrocells serving the surrounding area can create external interference. Highly specialized (and expensive) radio equipment is needed to overcome these challenges. Even then, users without a direct line of sight (LOS) connection to a WiFi access point or cellular base station, such as those shadowed by overhanging tiers or in executive boxes, experience a poor signal quality.

A complementary approach to minimize congestion is to deliver popular multimedia content using broadcast or multicast services. Because of the density of users in this scenario and the likelihood that many of them are requesting the same content, it is possible to transmit a single copy of the content to all receivers using a multicast or broadcast service. However, since a single transmission rate must cover all users, the system-wide video quality is constrained by the few users with the weakest channel. To deliver acceptable service to these users, the transmitter will use a conservative wireless modulation and coding scheme. Under these conditions, either media quality or wireless bandwidth must be sacrificed.

In this scenario, we may exploit diversity to improve performance. Because packet losses due to the wireless channel are generally dependent on location, it is likely that wireless devices in the network will lose different packets. Furthermore, because many wireless devices have multiple radios, we can use a second radio to communicate with peers over a short-range network (e.g. WiFi, Bluetooth) to cooperatively recover lost packets from one another. Then, as long as one node in a "cooperation cluster" has received a packet correctly, all of the other nodes in the cluster can recover that packet. We can therefore use a higher order modulation and coding scheme to transmit a high quality media stream, since even though the clients with a poor signal will experience an unacceptable level of loss, they can rely on the cooperative network to recover missing packets.

The literature describes many variations on this theme [2, 4, 8, 10, 14–17] all of which are found to improve performance in simulations. However, we have not seen any practical implementation which has been tested in a real hybrid wireless network. The contribution of this work, therefore, is an implementation of a practical framework for cooperative packet recovery of multicast and broadcast streams using hybrid networks. We evaluate this framework in an operational WiMAX/WiFi hybrid network and show that devices with a poor signal quality effectively recover missing packets

WiNTECH'13, September 30, 2013, Miami, Florida, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2364-2/13/09 ...$15.00.
http://dx.doi.org/10.1145/2505469.2505479.

over the cooperative network. We also identify some issues that arise in a realistic radio environment that can limit the usefulness of this technique, which have not been identified in simulation and emulation studies of similar protocols.

The rest of this paper is organized as follows. First, we give some background information and place this work in the context of the literature on this subject (Section 2). In Section 3 and Section 4 we describe the system architecture and the design of our implementation, including the *AFFIX* platform it is built on. Section 5 details the testbed and measurement infrastructure we use to verify the effectiveness of our implementation. An evaluation of its performance is in Section 6. Finally, Section 7 concludes with directions for future work.

2. RELATED WORK

Various related techniques have been proposed to address the issue of delivering multicast or broadcast content using a hybrid infrastructure/ad hoc network architecture. However, because these have only been evaluated in simulation and emulation environments, their performance in a realistic wireless environment is not fully understood.

The literature describes many variations on this theme [2, 4, 8, 10, 14–17]. These include a hierarchal version of the hybrid network architecture, a game-theoretic analysis of the costs and incentives associated with the technique, applications involving satellite/terrestrial hybrid networks, and other interesting contributions.

Yet we are not aware of a practical implementation of this or any related technique which has been tested in a real hybrid wireless network. Of the related work we have seen, only [15] describes an implementation and evaluation in a testbed setting, and it uses an emulated network as the primary network.

In this work, we describe an implementation of a simple cooperative protocol for recovery of a broadcast media stream in a hybrid network scenario. We verify its effectiveness in a hybrid network testbed using operational WiFi and WiMAX networks, and we identify characteristics of a live radio environment, such as burstiness of packet loss, that affect its performance.

3. ARCHITECTURE

In the hybrid architecture (Figure 1), every wireless device is equipped with multiple radios. One of these is a cellular radio, and another can be used for short-range communication (such as Bluetooth or WiFi).

Each wireless device connects to a principal network, which is an infrastructure-based cellular network, to receive multicast or broadcast data. Also, devices that are close to one another (with respect to latency over a secondary ad-hoc network between themselves) that are consuming the same content form a “cooperative cluster” and connect to a common ad-hoc network using short-range radio.

When packets sent over the primary network are lost or corrupted, physically co-located devices communicate over the short-range, low-latency assistant network to recover lost data cooperatively from peers. As long as any device in the cluster has successfully received a packet, the entire cluster can recover it.

This scheme assumes that the primary cellular network is “expensive,” because a single infrastructure network covers an area with densely packed users and bandwidth is quite scare. In contrast, the assistant network is assumed to be “inexpensive,” because cells can be made very small to reduce congestion. It is therefore designed to conserve the “expensive” primary network, at the cost of increased congestion in the assistant network.

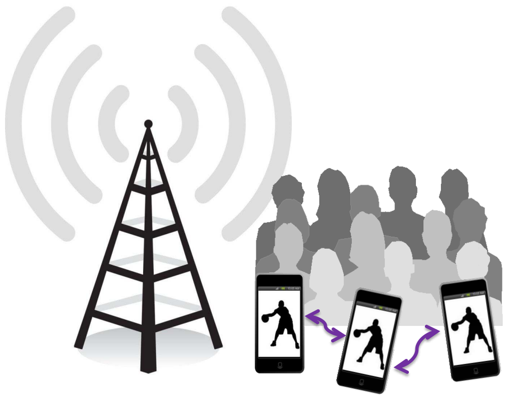

Figure 1: The principal network, which is an infrastructure-based cellular network, provides downlink multicast or broadcast services. The assistant network is an ad hoc cooperative network formed by physically co-located peers who are consuming the same media content, and is used to recover packets in a peer to peer fashion.

4. IMPLEMENTATION

In this section we describe the behavior of our implementation and some basic principles behind its design, including our use of the *AFFIX* framework for the implementation and our choice of cluster size.

4.1 Behavior

Our cooperative packet recovery implementation behaves as follows.

When the client application is launched, the user joins an ad-hoc 802.11 assistant network which is identified by a predetermined ESSID. This effectively groups clients that are in close proximity and are receiving video content. If no network with this ESSID is observed, the user starts a new one on a random channel; if multiple networks with this ESSID are visible, the user joins the one on which it sees the highest signal quality. Mobile users change from one assistant network to another if the signal quality observed on the new network becomes much greater than the quality of their current network, or start a new assistant network if none are visible to it. Low transmit powers are used by all clients on the assistant networks to keep the size of the network small and to minimize interference between neighboring networks. Once connected to the ad-hoc network, clients immediately begin listening for packet requests from peers on the network’s broadcast address.

At the video source, an application-level header containing an incrementing ID number is appended to each outgoing packet, which is then broadcast over the primary network. The client application starts a FIFO buffer for each video feed it receives, with each buffer identified by a key composed of the video feed’s source address and port. Incoming packets are placed in their respective buffers as they are received. Initially, packets for a given feed are buffered until there is a range of packet IDs in the buffer greater than some value q. Thereafter, whenever the range of packet IDs in a buffer exceeds q, the first packet is removed from the buffer and passed to a video player. This strategy ensures that approximately q packets are kept in the buffer at all times.

When a packet with a non-contiguous ID is received for a particular feed, the application assumes that the intervening packet has been lost and broadcasts a request message, containing the ID of the missing packet and the key of the video feed, over the assistant network. If any peer receiving the request has the packet in its own buffer, it sends it over the assistant network to the requester as a unicast packet. At the receiver, if the ID of a recovered packet is higher than the lowest ID of any packet in the buffer, then the recovered packet is placed in the appropriate position in its buffer.

Unlike some other schemes described in the literature, there is no peer discovery phase in which clients identify neighbors with which they will cooperate. The protocol is essentially stateless, so we do not require any additional actions to be taken when mobile devices move between clusters or when clients join and leave clusters.

4.2 Using Software Defined Sockets

In order to make this implementation as broadly useful as possible, we developed it within the *AFFIX* software defined sockets framework [1].

Techniques for implementing new networking protocols, such as our cooperative recovery protocol, often involve modifying and rebuilding the Linux kernel or a network device driver. This introduces a steep learning curve and requires a prohibitive time investment. The implementations produced by this method are often not generalizable to heterogeneous platforms, and are difficult to deploy on real end-user devices.

Another popular method is to build functionality into an application or a software library. This complicates application logic and requires that the functionality be implemented in or ported to every application that is to benefit from it.

Software defined sockets (SDS) is a technique for implementing and deploying novel networking ideas without changing the kernel, or porting to every possible target environment and application. Much as software defined networking (SDN) allows programmability at the network routing layer and software defined radio (SDR) offers programmability at the radio layer, SDS allows researchers and software developers to program functionality at the boundary between the application layer and the OS networking stack, the socket API. *AFFIX*, the SDS framework used in this project, offers a convenient platform for implementing enhanced network functionality and making it available in a broad range of scenarios.

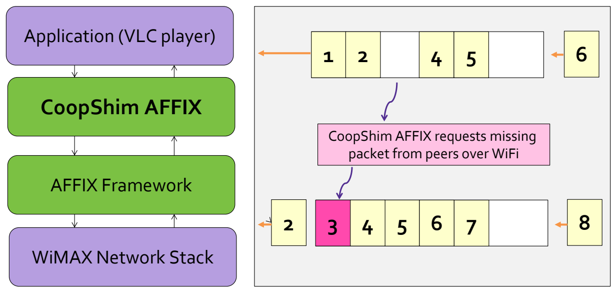

Figure 2: Operation of CoopShim implementation on the wireless receiver.

The structure of our implementation is shown in Figure 2. The AFFIX framework runs at the boundary between the operating system's network stack and the video player application and intercepts socket API calls made by the application. For example, when the video application on the client makes a `recv` call to read data from a video feed, this call is intercepted by the *CoopShim* AFFIX component, which has implemented its own `recv` behavior. The CoopShim `recv` behavior is composed of several parts. First, it calls the `recv` function of the legacy socket API, strips the packet ID from any received packets, and places them in their respective buffers. Next, it broadcasts packet requests if any missing packets are identified. Finally it returns to the application the lowest-numbered packet in the buffer if the range of packet IDs in the buffer is greater than some value q. Otherwise it raises an `EWOULDBLOCK` exception to the calling application, which signals that no data is available for the given socket.

This `recv` behavior is comletely hidden from the video application, which makes an ordinary `recv` call and receives either packet data or an `EWOULDBLOCK` exception in response. It is also hidden from the underlying network stack, which gets a `recv` call and responds with packet data or an `EWOULDBLOCK`.

Because this implementation operates entirely above the operating system's network stack and below the application, it is application- and network-agnostic. It will operate on any underlying primary and assistant network, and will work with any application that is configured to receive packets on the AFFIX-enabled port.

However, because this platform is not optimized for performance, it suffers a performance penalty. Also, this implementation can only access information that is available at the application layer, and cannot exploit information from lower layers (such as knowledge of the wireless channel quality) to improve performance.

4.3 Size of clusters

The implementation we describe here is specifically intended to be used with small clusters. This decision is informed by previous simulation-based studies [14], which found that between two or three helpers in a peer cluster, there is generally enough diversity to recover almost all missing packets, even under poor channel conditions. There is little benefit in increasing the number of peers in a cluster beyond four peers.

The small size of the peer clusters allows the recovery application to be kept very simple. Because the network is so small, we may broadcast recovery requests over the assistant network and make no effort to suppress redundant transmissions of recovery packets by other respondents. This very simple procotol has minimal computing requirements and therefore, introduces minimal latency and energy usage. While we create unnecessary traffic on the assistant network, contention levels for the 802.11 link remain low due to the small network size. The small size of the peer group also allows lower transmission powers to be used on the assistant network, further reducing energy usage on the wireless devices.

5. MEASUREMENT PLATFORM

This work was evaluated using a dedicated experimental WiMAX 802.16e testbed installed at the campus of the Polytechnic Institute of New York University (NYU-Poly) as part of the GENI project [5]. For highly controlled experimentation, this platform has a distinct advantage over commercial cellular networks because it allows us to isolate the effects of the wireless channel quality from other variables such as competing traffic, carrier routing and shaping policies, and radio configuration. This increases consistency and repeatability, while still being more true to life than a simulation or emulation environment.

In a previous study [7], we quantified the performance of this platform, particularly with regard to achievable data rates. We found that the data rates achieved on this platform are similar to typical data rates measured by others for users of commercial HSPA+ networks and users of commercial LTE networks [9]. This suggests that with regard to overall performance, this platform can be considered broadly representative of current wireless broadband networks.

The testbed includes a WiMAX base station (BS) and fifteen clients (nodes). The testbed nodes are managed by the cOntrol and Management Framework (OMF) [13], which configures network connectivity and orchestrates experiment scenarios on the nodes and the BS. Each node has a WiMAX network card, a WiFi network card, and an Ethernet interface for experiment control. The BS is a commercial WiMAX 802.16e product from NEC, operated via customized control software. It operates in licensed frequency centered at 2.595 GHz. Table 1 summarizes key BS configuration parameters for the experiments described here.

Access Mode	SOFDMA/TDD
Center Frequency	2.595 GHz
Channel Bandwidth	10 MHz
Transmit Power	38 dBm
TDD DL:UL ratio	35:12 (symbols)
Service Class	Best Effort (BE)

Table 1: Selected parameters of the WiMAX BS configuration.

The testbed platform also includes the suite of instrumentation and measurement utilities that make up the OML measurement framework [12]. We used OML, and the OML4Py Python module [6] in particular, to instrument the CoopShim software with measurement "hooks" that stream measurement data to a database at regular intervals while the application is running. We also use OML to collect measurements of WiMAX signal quality from the nodes during experiment runtime.

6. EVALUATION

We verified the effectiveness of CoopShim in a series of live over-the-air trials on the WiMAX testbed described in Section 5.

In our experiment scenario, the CoopShim sender is a host that is connected to the NYU-Poly campus network through an Ethernet interface. We use the *iperf* bandwidth testing tool to generate UDP broadcast traffic from this host.

Four testbed nodes, all physically located in the same room, serve as the cooperative receivers in this scenario. Each of these nodes have a WiMAX carrier to interference plus noise ratio (CINR) in the range of 19 to 30 dB, as shown in Table 2. The nodes are also connected to a dedicated 802.11g network in ad hoc mode.

Received	Average CINR (dB)
node11	26.7
node13	22.9
node5	19.5
node8	30.3

Table 2: WiMAX CINR of the receivers in the cooperative cluster ranges from approximately 19 to 30 dB.

Ordinarily, broadcast traffic from the WiMAX BS will be sent using QPSK modulation and a 1/2 code rate, to accommodate clients with a low CINR. For this experiment, we configured to WiMAX BS to broadcast using 64 QAM modulation and a 3/4 code rate for all clients, even those with low CINR. At this setting, UDP packet loss rates at the WiMAX receivers we have chosen will range from zero to 10%.

During each experiment trial, we broadcast UDP traffic at a rate of 300 kbps, with a packet size of 1000 B, for a duration of 60 seconds. Approximately 2250 packets are transmitted during each trial. We execute this experiment for seven different received buffer sizes, with four trials for each buffer size.

6.1 Buffer size and recovery rate

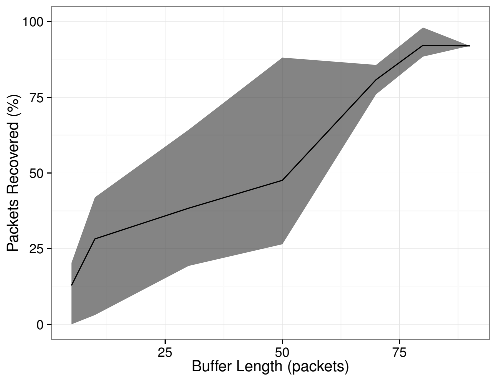

Figure 3: The solid line shows the average proportion of lost packets that are successfully recovered from peers over the assistant network, i.e. *requested packets/recovered packets*. The shaded region shows this proportion for the best- and worst-performing clients in the cluster. As buffer size grows, up to 92% of packets may be recovered on average.

Figure 3 shows the proportion of lost packets that are successfully recovered from peers over the assistant network under different buffering conditions.

Because each receiver experiences different loss and latency conditions on the WiMAX link, the range of packets in each receiver's buffer is slightly different. For larger buffer sizes, the packets held by two receivers are more likely to overlap, increasing the likelihood of peer recovery.

There is also a small possibility that latency in the assistant network will delay a recovery packet so that by the time it is received, a higher-numbered packet has already been delivered to the application. In this case, the recovery packet will be discarded. With large buffer sizes, each packet spends more time in the buffer before it is delivered to the application, which can compensate for latency in the assistant network.

Figure 3 supports this intuition, with significantly increased performance for large buffer sizes. Even for a buffer size of 10 packets, an average of 28% of requested packets are recovered from peers. As buffer size grows, as many as 92% of missing packets are recovered on average.

However, larger buffers come at the cost of increased delay. The delay associated with buffering is $\frac{lq}{R}$ seconds, where R is the packet rate in bits per second, l is the packet length in bits, and q is the length of the buffer in packets. The particular delay that is considered tolerable depends on the application. For live streaming video, such as instant replays of a sporting event, delays of several seconds may be tolerated, while for real time communications, user experience degrades significantly after several hundred milliseconds.

6.2 Packet recovery time

In Figure 4, we show the cumulative distribution of the time elapsed between the transmission of a packet request over the assistant network, and the receipt of the recovery packet from a peer. This recovery time is the sum of the round trip latency in the assistant network and the computational time associated with finding and retrieving a recovery packet from the buffer. Therefore, it may be expected to increase with the size of the cooperative cluster and with buffer size, because with larger clusters or larger buffers, more traffic is generated leading to more congestion and latency.

In our scenario, recovery time was generally low. The average recovery time was 45.8 ms, and the minimum and maximum times

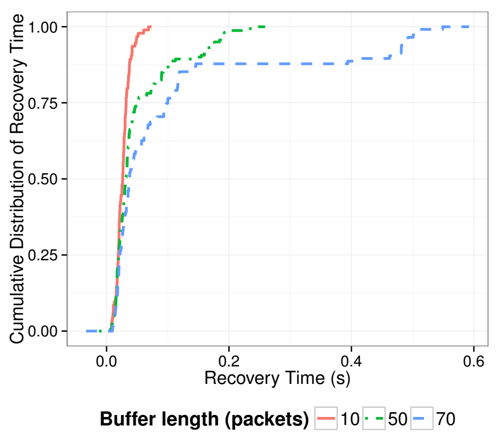

Figure 4: The cumulative distribution of recovery time for selected buffer lengths shows that recovery time increases with buffer length.

were 4.9 ms and 549.1 ms, respectively. In a more congested radio environment, however, latency in the assistant network is likely to increase, and buffer size will need to be calibrated to compensate.

6.3 Fairness and motivation to participate

We are also interested in balancing the costs and incentives associated with cooperation. For wireless devices to willingly participate in the cooperative recovery protocol, the benefit of reduced packet loss must outweigh the costs in battery life and processing power associated with sharing packets over the assistant network. This is especially important for devices with a good channel, which already have acceptable quality without the cooperative protocol, and which are likely to bear the brunt of the burden in sharing packets with peers.

Figure 5 shows the number of packets shared over the assistant network for each pair of nodes. As expected, the node with the highest WiMAX CINR (as per Table 2) contributes the most packets to the cluster and receives the fewest. The node with the weakest WiMAX CINR contributes the fewest packets, some three times less than the node with the strongest CINR. Depending on the cost associated with using the assistant networks, devices with a strong signal on the primary network may choose to opt out of the cooperative recovery protocol based on these results.

6.4 Desynchronization with bursty packet loss

Packet losses on a wireless medium often occur in bursts. In our protocol, bursty packet loss that cannot be recovered may cause desynchronization between different receivers, i.e., the state of having no overlap at all between the range of packets held in their buffers.

Because the buffer acts as a “sliding window,” when a receiver experiences bursty packet loss, the highest-numbered packet in its buffer may actually be less than the lowest-numbered packet held by any other receiver. (This occurs when packet loss occurs in bursts greater than the buffer length, q.) In this state, the affected receiver cannot recover any missing packets.

This effect is illustrated in Figure 6 and Figure 7, for four receivers in a cooperative network. These are not the same as the receivers identified in Table 2; the WiMAX CINR values for receivers 1, 2, 3, and 4 in Figure 6 and Figure 7 are 29, 26, 23, and 21 dB, respectively.

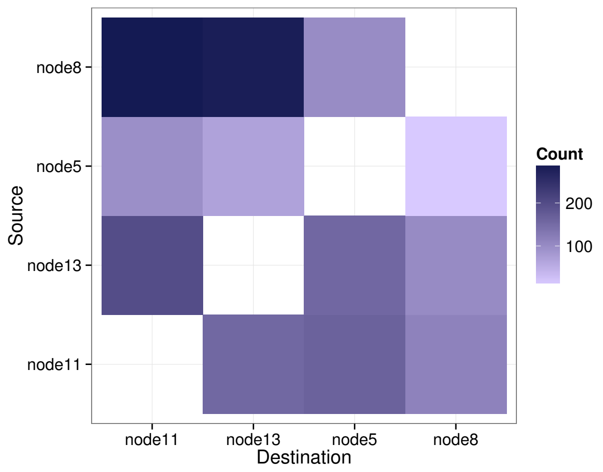

Figure 5: For each node pair in the cooperative cluster, we show the number of packets transmitted from one node to the other over the assistant network.

The color gradient in Figure 6 shows the timestamp at which a given packet is delivered to the application (or equivalently, the lowest-numbered packet in the buffer at a given point in time).

The receiver identified with ID 4 has a poor signal quality over the WiMAX link, and experiences bursty packet losses from which it is unable to recover, identified by blank gaps in the gradient. Each time there is a gap in the gradient for Receiver 4, the difference between its highest-numbered buffered packet and the lowest-numbered packet held by other receivers increases. This is illustrated by the shift in the gradient between Receiver 4 and the other receivers in Figure 6.

Figure 7 shows the source of each packet received by these four receivers. We observe that initially, Receiver 4 receives packets from both its peers and the WiMAX link. After Packet 2086, there is no longer any overlap between its buffer and its peers’ buffers. At this point cooperative recovery is not possible and it only receives packets from the WiMAX link, which suffers from severe packet loss.

Various strategies could be implemented to avoid this state, such as detecting desynchronization at a receiver and skipping ahead some number of packets, or using a timestamp-based window for buffering instead of a window based on packet ID. However, these would increase the complexity of the receiver and might introduce new performance tradeoffs.

7. CONCLUSION

In this work, we have demonstrated that we can achieve practical gains on live networks with a simple scheme for cooperative packet recovery using a hybrid network architecture. Our implementation of this scheme is built on the AFFIX platform, and so it is application- and network-agnostic. In our evaluation of this scheme, we demonstrated the tradeoff between buffer-induced delay and packet recovery. We explored two factors that inhibit packet recovery, latency over the assistant network and desynchronization of receiver buffers. We also measured the level of asymmetry between peers.

In future work, we might consider several enhancements to our scheme. In our simple implementation, all of the wireless clients are receiving the same video feed and belong to the same cooperation cluster. We would like to extend this experiment to a multi-

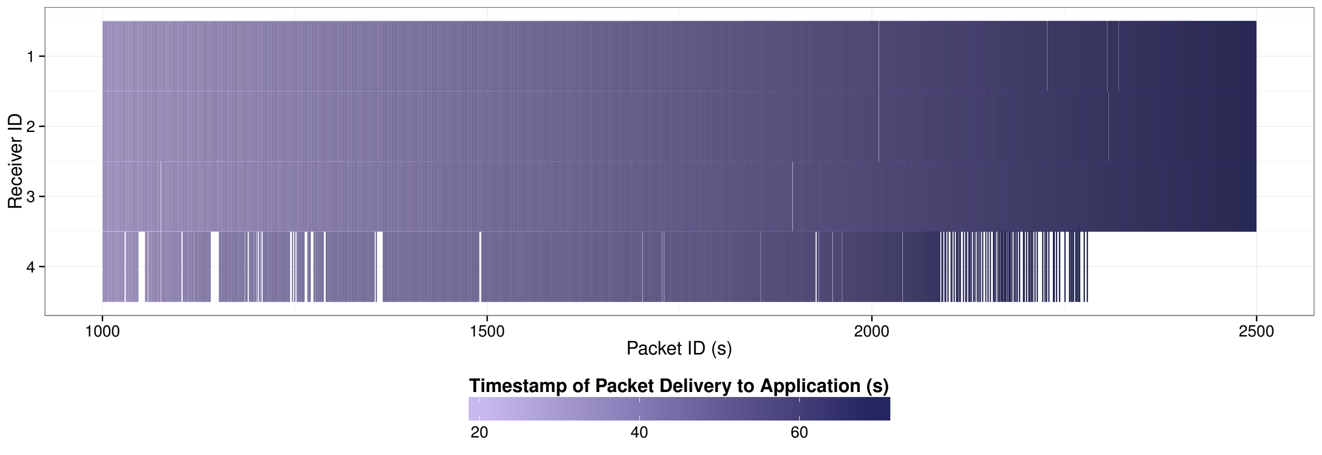

Figure 6: The color gradient shows the timestamp at which a given packet is delivered to the application (or equivalently, the lowest-numbered packet in the buffer at a given point in time). Gaps indicate packet losses which are not recovered over the assistant network. The buffer desynchronization issue is illustrated by the shift in the gradient between Receiver 4 and the other receivers.

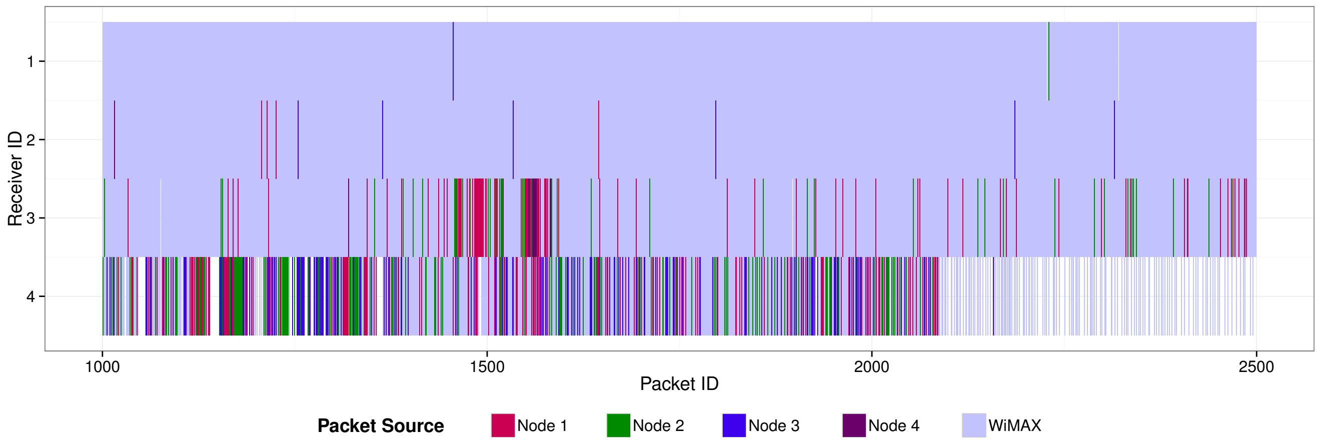

Figure 7: Source of each packet received for every node in a cooperative cluster of four nodes. Packets may be received from the broadcast service over the WiMAX network or from one of the other nodes in the cluster. The WiMAX CINR values for receivers 1, 2, 3, and 4 are 29, 26, 23, and 21 dB, respectively; we observe that the number of packets received from peers increases as signal quality is degraded.

cluster scenario to explore the impact of interference between neighboring clusters, and how this could be mitigated by designing an algorithm according to which clients would tune their transmission power on the assistant network.

Also, the primary application we have considered, of a broadcast multimedia stream, is relatively uncomplicated from a security perspective. Under other circumstances, however, additional considerations may arise. For example, a similar architecture may be used to deliver over the air (OTA) firmware updates to devices using a multicast service and a device-to-device assistant network. In this scenario, we would require security enhancements to make sure that a malicious actor could not induce its peers to install an inauthentic firmware.

8. ACKNOWLEDGEMENTS

This work is supported by the New York State Center for Advanced Technology in Telecommunications (CATT) at the Polytechnic Institute of New York University, by NYU WIRELESS, and by the National Science Foundation under the Graduate Research Fellowship Program Grant No. 1104522.

9. REFERENCES

[1] AFFIX. `https://affix.poly.edu/`, 2012.

[2] K. Akkarajitsakul, E. Hossain, and D. Niyato. Cooperative packet delivery in hybrid wireless mobile networks: A coalitional game approach. *IEEE Transactions on Mobile Computing*, 12(5):840–854, 2013.

[3] D. Fraley, D. Sanderson, and S. Mitchell. Delivering London 2012: meeting the mobile data demand challenge. In *Delivering London 2012: ICT enabling the Games*, IET special interest publication, pages 33–44. The Institution of Engineering and Technology (IET), 2012. `http://www.theiet.org/sectors/information-communications/ict-2012.cfm`.

[4] N. M. Do, C.-H. Hsu, J. Singh, and N. Venkatasubramanian. Massive live video distribution using hybrid cellular and ad hoc networks. In *World of Wireless, Mobile and Multimedia*

Networks (WoWMoM), 2011 IEEE International Symposium on a, pages 1–9, 2011.

[5] C. Elliott and A. Falk. An update on the geni project. *SIGCOMM Comput. Commun. Rev.*, 39(3):28–34, June 2009.

[6] F. Fund. oml4py: An OML client module for Python. `https://pypi.python.org/pypi/oml4py`, 2012.

[7] F. Fund, C. Wang, T. Korakis, M. Zink, and S. Panwar. GENI WiMAX performance: Evaluation and comparison of two campus testbeds. In *Proceedings of the 2nd GENI Research and Educational Experiment Workshop*, GREE '13, 2013.

[8] S. Hua, Y. Guo, Y. Liu, H. Liu, and S. Panwar. Scalable video multicast in hybrid 3G/ad-hoc networks. *Multimedia, IEEE Transactions on*, 13(2):402–413, 2011.

[9] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4G LTE networks. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 225–238, 2012.

[10] A. Laya, K. Wang, L. Alonso, and J. Alonso-Zarate. Multi-radio cooperative retransmission scheme for reliable machine-to-machine multicast services. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, pages 1–6, 2012.

[11] P. Leonhardt. Wireless at the "connected games": How the London 2012 Olympic and Paralympic games utilized the latest Wi-Fi technology. *Journal of Telecommunications and Information Technology*, 01/2013, Jan. 2013. `http://www.itl.waw.pl/czasopisma/JTIT/2013/1/5.pdf`.

[12] O. Mehani, G. Jourjon, T. Rakotoarivelo, and M. Ott. An instrumentation framework for the critical task of measurement collection in the future internet. Technical report, Nicta, Eveleigh, Sydney, NSW, Australia, October 2012.

[13] T. Rakotoarivelo, M. Ott, G. Jourjon, and I. Seskar. OMF: a control and management framework for networking testbeds. *SIGOPS Oper. Syst. Rev.*, 43(4):54–59, Jan. 2010.

[14] K. Sinkar, A. Jagirdar, T. Korakis, H. Liu, S. Mathur, and S. Panwar. Cooperative recovery in heterogeneous mobile networks. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON '08. 5th Annual IEEE Communications Society Conference on*, pages 395–403, 2008.

[15] P. Vingelmann, M. Pedersen, F. Fitzek, and J. Heide. On-the-fly packet error recovery in a cooperative cluster of mobile devices. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–6, 2011.

[16] E. Weigle, M. Hiltunen, R. Schlichting, V. Vaishampayan, and A. Chien. Peer-to-peer error recovery for hybrid satellite-terrestrial networks. In *Sixth IEEE International Conference on Peer-to-Peer Computing*, P2P 2006, pages 153–160, 2006.

[17] Q. Zhang and F. Fitzek. Cooperative retransmission for reliable wireless multicast services. In F. H. Fitzek and M. D. Katz, editors, *Cognitive Wireless Networks*, pages 485–498. Springer Netherlands, 2007.