

Emergency Related Video Streaming in VANETs using Network Coding

Joon-Sang Park[†], Uichin Lee[†], Soon Young Oh[†], Mario Gerla[†], Desmond Lun^{*}

[†]UCLA CS

Los Angeles, CA 90095

{jspark,ucllee,soonoh,gerla}@cs.ucla.edu

^{*}UIUC CS

Urbana, IL 61801

dslun@uiuc.edu

Abstract—Vehicular communications are becoming a reality driven by various applications. Among those applications safe navigation support is of most significance. In designing such navigation safety applications, reliable dissemination of data, i.e., every affected vehicle receives data, is the key issue. Past research focused on the reliable dissemination problem of plain media type (e.g., text) safety message whereas we look at the problem of reliable and efficient dissemination of multimedia type (e.g., video, audio) safety information. Considering the potential volume of multimedia traffic and the unpredictability of vehicular networks (e.g., partitions, packet errors), reliable and efficient multimedia dissemination is non-trivial. By using a recently developed technique, *network coding*, we describe an efficient way to achieve reliable dissemination. Simulation results show that in a typical setting, with representative channel errors/losses our approach yields near 100% delivery ratio as compared to 92% delivery ratio by traditional multicasting. More importantly, the overhead is reduced by as much as 60%. If the column of vehicles on the road has gaps, network coding jointly with “data muling” on vehicles in the opposite direction can deliver the multimedia files to disconnected components faster than other known schemes.

I. INTRODUCTION

In the near future, car to car communications and networking will be driven mainly by navigation safety support applications. In the simplest example, vehicular communications can be used to exchange *virtual* tail lamp signals between vehicles more reliably than the conventional visual tail lamp system. Moreover, human reaction times can be reduced or nullified altogether. To implement such a safety warning system, a finite set of signals augmented with additional information such as vehicle location and speed can be represented as plain media type (e.g., text) data and packed into a very small number of short packets, resulting in rather modest data rates. Emerging vehicular networks will be equipped with advanced radios and wireless LAN grade bandwidth, thus allowing more aggressive data rates and applications. In the longer term, multimedia type data (e.g., audio, video), if feasible, will also be introduced to enhance navigation safety. In fact, multimedia data such as images and even short videos clips of an accident

or dangerous situation ahead (e.g., flood, fire, earthquake damage, terrorist attack damage, etc) will provide drivers with more precise information than simple text messages. It will allow them to make a more informed decision (whether to proceed or turn back) based on personal priorities and/or on vehicle capabilities. For instance, suppose that a critical traffic/safety situation occurs on a highway, say, a hurricane caused flood followed by major traffic congestion. In such cases, multimedia content transmission, say, image and video streaming, could be triggered on one or more lead cars and propagated to vehicles following several miles behind – to *visually* inform them of the problem and allow them to decide if they should turn around. All this will happen within minutes from the accident, much before helicopters and first responders, say, can come to the scene and broadcast warning signals and video feeds on the networks. As a vivid example, the reader may recall that a few years ago in the Mont Blanc tunnel between Italy and France a truck caught fire.¹ The vehicles approaching the truck tried to turn back, but were blocked by the oncoming traffic that, unaware of the danger, had filled both lanes. Many lives were lost. If oncoming drivers a few miles away had been warned of the danger, they would have turned back when it was still possible, or would have not entered the tunnel altogether. A generic text message warning should have been sufficient in this case, but a video clip of the developing fire situation would have been much more graphic and persuasive.

Besides private vehicles, also first responders and rescue operations can greatly benefit from more prompt and precise situation awareness delivered by multimedia car to car streams. Police and/or paramedics, for example can plan ahead and coordinate their intervention using the multimedia data collected on their way to the scene.

For the above warning systems to work, however, it is crucial that the broadcast be efficient and does not congest the network. Also, the safety related messages must be *reliably* delivered to all the *impacted* nodes in vehicular network so that they can cooperatively coordinate evasive actions. For instance, if only 75% of the vehicles in the tunnel can receive the visual

¹See http://en.wikipedia.org/wiki/Mont_Blanc_Tunnel

warning, it is possible that the remaining 25% who received a garbled signal will still congest the lanes. Thus reliable data dissemination is a key requirement for multimedia navigation safety applications. The application must be protected against unreliable wireless channel, e.g., random obstacles on the path and packet drops. It must also be robust to the unpredictable nature of the vehicular network, e.g., variable network density, frequent partitions and gaps in the column of cars driving in the same direction, etc.

There is a vast literature on reliable data dissemination in vehicular networks. The main focus however has been on short alert messages. The short message application can be supported by conventional solutions such as repetition of messages. The reliable data dissemination problem becomes much more complex with multimedia data. Multimedia files are inherently large. When disseminated without proper controls, they can cause severe congestion and shut down other safety traffic.

In data dissemination over vehicular networks, packets may be corrupted and lost because of many reasons. Fading, environment interference, and mobility can produce random like losses. Another cause of loss is packet collision. In particular, collisions among hidden terminals are quite frequent in broadcast where the RTS/CTS feature of the 802.11 like MAC is disabled. We assume in this study that the DSRC MAC protocol or the emerging 802.11p MAC protocol are used. Because of broadcast, packet losses tend to be random and locally diverse. That is, each node in a neighborhood has dissimilar, rapidly varying packet reception characteristic. Therefore, upon packet loss, a “local diversity” recovery strategy, i.e., neighbors helping each other, can be very efficient and effective at the same time. To fight locally correlated losses, e.g., losses experienced by multiple receivers in the same neighborhood due to common upstream congestion, broken link, etc, multipath diversity, i.e., the use of multiple, disjoint paths paves an effective way. Path diversity is abundant in urban vehicular networks when they are densely packed, during peak hour, say. The main question thus is how to utilize path diversity efficiently. Also, the overhead must be low, so that the scheme does not introduce extra inefficiency when the vehicle network is “sparse.” Our approach seeks to achieve reliable multimedia data dissemination using localized neighbor recovery approach and multipath diversity with very low overhead. The key ingredient of our approach is a *random network coding* scheme which transparently implements both localized neighbor recovery and path diversity with remarkably low overhead.

In addition to delivery ratio and traffic overhead, a critical measure for multimedia video delivery is end to end delay. We must distinguish between real time multimedia (e.g., video conference) and non real time multimedia (e.g., one way video streaming). For real time, the delay average and jitter are subject to tight constraints. For non real time, the average delay and

jitter are subject to much more relaxed constraints (in the order of seconds). Basically, the receiver reassembles packets and orderly streams them to the application using a reconstruction buffer. The attention then shifts to buffer requirements and user buffer limitations. In this study, we address only non interactive real time streaming. Note that the delay requirements are more strict than in conventional video streaming (e.g., video on demand). Drivers in the proximity of each other must take coordinated actions based on what they are shown. So, discrepancies in the order of several seconds are not tolerated. Regarding buffering requirements, this constraint will be ignored since the vehicle (as opposed to handheld or sensor based platforms) has practically unlimited buffer capacity.

We mentioned earlier that cars may become separated on the highway, forming platoons. If an accident occurs, a platoon that follows by 30 seconds, say, will automatically incur a 30 second delay. There are two approaches to this problem. First, the applications discussed in this paper are intrinsically delay-tolerant, in the sense that reactive action to the accident alert (e.g., stopping and turning around) is generally required only when the second platoon establishes radio contact with the first one. Thus, it is important that the vehicles in the second platoon learn of the accident all at the same time (albeit with a 30 second latency).

The second approach exploits vehicles coming in the opposite direction (assuming that the highway has multiple lanes in both directions). The opposing traffic may actually consist of vehicles that have already turned around after the accident. This second approach uses the vehicles in the opposing directions as “data mules.” It can be used in conjunction with the first approach and can speed up the delivery of alarms with obvious navigation safety benefits.

In our study we address both the reliable delivery within a contiguous platoon and the delayed delivery to multiple platoons separated by random gaps. The goal of our reliable dissemination service design within a platoon is to use loss recovery efficiently, while at the same time keeping low overhead in check. Packet losses cause quality degradation in the multimedia data but 100% recovery is not the right answer since full recovery may incur too much overhead and most of multimedia data streams tolerate a certain level of packet losses 1-2%. Delay must also be kept in check when considering recovery. A few seconds discrepancy between neighbor vehicles is tolerated. However, an end to end 100% recovery scheme (for example, end to end recovery of individual packets using TCP) that leads to delays in the order of several seconds would be unacceptable in vehicular safety video streaming. We will show that network coding can be a very effective solution for reliable, low overhead, limited delay delivery within a contiguous platoon.

In the delayed delivery study we address the opposing

traffic “data muling” of a video streams to disconnected platoons. Again we show that network coding provides an efficient solution, leading to lower completion delays or conversely better accuracy than other techniques.

The rest of paper is organized as follows. Section II discusses related work, Section III illustrates the network coding based data dissemination protocol, Section IV evaluates the protocol via simulations, and an analysis of the protocol is presented in Section V. Finally, Section VI concludes this paper.

II. RELATED WORK

Urban Multi-hop Broadcast (UMB) [10] supports directional broadcast in VANET. UMB tries to improve reliability of broadcast by alleviating a hidden terminal problem through an RTS/CTS-style handshake, and broadcast storms through black-burst signals to select a forwarding node that is farthest from the sender using location information. Unlike UMB, Broadcast Medium Window (BMW) [20] and Batch Mode Multicast MAC (BMMM) [19] requires all the receiving nodes to send back ACK to the sender in order to achieve reliability. BMMM has also adapted to directional MAC in VANET [23]. However, all the previous work basically requires considerable amount of contention resolution time for each transmission, and thus, it is not suitable for real-time streaming which could potentially generate a large number of packets for a relatively short period of time. Our scheme can not only handle this, but also efficiently deal with broadcast storm and reliability issues using random linear network coding. Moreover, our scheme can handle intermittent connectivity using the carry-and-forward “data muling” method.

At the applications level, several cooperative peer to peer type schemes have been proposed for vehicular environments. TrafficView [15] disseminates, or pushes (through flooding), information about the vehicles on the road, thus providing real-time road traffic information to drivers such as speed of vehicles. To alleviate broadcast storms, this work focuses on data aggregation/fusion based on distance from the source. Vehicular Information Transfer Protocol (VITP) [4] provides on-demand, location-based, traffic-oriented services to drivers using information retrieved from vehicular sensors. A user can pull information from virtual ad hoc servers (i.e., collection of private vehicles) at the target location by sending a location-aware query. Similarly, V3 [5] supports a video request query (i.e., video trigger message) to the target location. Multiple vehicles at the destination could forward the video data to the query originator, and intermittent connectivity is handled by the carry-and-forward method. Our work is different from V3 in the following aspects. First, our protocol “pushes” urgent video streams regarding emergency situations such as natural disaster, traffic accidents, terrorist attacks etc which cannot be “pulled” by remote customers a prior.

These emergency streams must be disseminated to other surrounding vehicles in order to help drivers effectively avert the danger. Second, our protocol exploits random linear network coding to provide reliable streaming. Finally, we address the carry-and-forward approach to deliver delayed streams to disconnected platoons. In the vehicle environment, the limited bandwidth allows only piecemeal transfer between vehicles traveling at high speeds in opposite direction, thus creating the video data packets collection problem (i.e., the “coupon” collection problem). However, random linear network coding elegantly solves this problem.

The effect of node mobility on vehicle communications was addressed in [24]. This work confirms the idea that as mobility increases, the number of encounters increases whereas the duration of an encounter decreases. In this respect, [22] has proposed an analytical model for information propagation delay of a single packet when the carry-and-forward approach is used. As shown in [24], it is better to employ multiple vehicles in parallel (when available) to deliver a relatively large video file using the carry-and-forward approach. In this paper, we have followed [24] suggestion and have developed a simple analytical model to estimate the file delivery delay using multiple, rather than a single, packet carrier. We show that network coding minimizes the delay.

By network coding, we refer to the notion of performing coding operations on the contents of packets throughout a network. This notion is generally attributed to Ahlswede et al. [1], who showed the utility of the network coding for multicast. The work of Ahlswede et al. was followed by other work by Koetter and Médard [9] that showed that codes with a simple, *linear structure* were sufficient to achieve the capacity of multicast connections in lossless, wireline networks. This result was augmented by Ho et al. [6], who showed that, in fact, a *random construction* of the linear codes was sufficient.

The utility of such *random linear codes* for reliable communication over lossy packet networks—such as VANET—was soon realized [14]. In [13], a prescription for the efficient operation of VANET is given, which proposes using the random linear coding scheme of [14] coupled with optimization methods for selecting the times and locations for injecting coded packets into the network. This problem of selecting the times and locations for injecting packets is called *subgraph selection*. The prescription given in [13] allows potentially to find the optimal way of setting up a single connection. The optimal solution however may be complex, especially under the unpredictable characteristics and constraints imposed by VANET. Previous works related to network coding’s application to wireless networks include [7], [8] both of which focus on protocols solely for unicast and [21] which focuses on energy efficiency under a collision-free ideal MAC. Also in [18], authors proposed a network coding based multicast protocol for ad hoc networks and show how it can improve efficiency and

reliability in such networks. The coding schemes and/or packet header formats used in [21], [18] are similar to ones we use, which are variations of the standard techniques originally proposed in [2]. We show here how the techniques can be used in vehicular ad hoc networks for reliable multimedia type data dissemination and how it can be seamlessly extended to delay tolerant operations.

III. RELIABLE MULTIMEDIA DELIVERY

In this section, we describe a network coding based reliable multimedia delivery service.

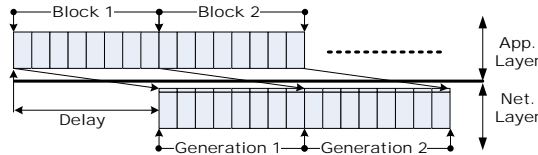


Fig. 1. Multimedia streaming with $blocksize = 8$

Suppose a multimedia data source generates a stream of frames $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \dots$ where subscripts denote unique and consecutive sequence numbers (see Figure 1). We assume that streams can be uniquely distinguished by the source address and port number pair or a globally unique identification number assigned to each stream. We use a tuple $(blockid, blocksize)$ where $blocksize > 0$ to indicate a block of frames with sequence numbers greater than or equal to $blockid$ and smaller than $(blockid + blocksize)$ (i.e., $\mathbf{p}_{blockid}, \dots, \mathbf{p}_{blockid+blocksize-1}$) belong to the block. A *coded packet* $\mathbf{c}_{(blockid, blocksize)}$ is a linear combination of frames in $(blockid, blocksize)$. That is,

$$\mathbf{c}_{(blockid, blocksize)} = \sum_{k=1}^{blocksize} e_k \mathbf{p}_{(k-1+blockid)}$$

where e_k is an element in a finite field \mathbb{F} over which every arithmetic operation is. Data frames \mathbf{p} 's and coded packets \mathbf{c} 's are also regarded as vectors over the field. In the header of a coded packet, the coefficient vector $\mathbf{e} = [e_1 \dots e_{blocksize}]$ is stored along with $blockid$ and $blocksize$ for the purpose of *decoding* packets on receivers. When generating a \mathbf{c} , each e_k is chosen randomly from \mathbb{F} , which is in general referred to as the random linear coding. \mathbf{c} 's with the same label $(blockid, blocksize)$ are said to be in the same *generation* (Figure 1). Throughout this paper we abuse lowercase boldface letters to denote vectors, encoding vector, frames, or packets, uppercase letters to denote matrices or constant numbers, and italics to denote variables or fields in packet headers.

The reliable delivery service agent (or layer) residing on the video source generates and transmit code packets to the receivers. Since a block of frames is required to generate a coded packet, the agent residing on the video source collects frames generated by application

and buffer them. For simplicity, we assume that data storage on a vehicle is large enough to store all the data for a limited amount of time. Under the general frame work of random linear coding, many variant strategies regarding encoding and forwarding can exist. Here we present two encoding schemes: *progressive coding* scheme and *block coding* scheme. In the progressive coding scheme, whenever a new frame \mathbf{p}_k becomes available a coded packet $\mathbf{c}_{(k-blocksize+1, blocksize)}$ is generated by combining the new frame with the preceding $blocksize - 1$ frames and is transmitted instantly. In the block coding scheme, the agent implements a delayed-transmission strategy: instead of transmitting a packet instantly when a frame becomes ready, a series of coded packets, $\mathbf{c}_{(blockid, blocksize)}$'s, are transmitted when all the frames in $(blockid, blocksize)$ are collected and blocks are organized such that no original data frame appears in two different blocks. In fact, $blocksize$ needs not be a fixed value. If the agent receives application frames of known format is known a priori, $blocksize$ can be determined on-the-fly according to the delay constraints of the packets. The objective is to make the size of each block big enough to gain efficiency while minimizing the possibility of delivering packets with delay constraint violations. (In general, the bigger the block size the greater the efficiency gain is, and also the delay) Otherwise, the agent uses a predefined number to limit maximum wait time in the buffer. Once the whole block is amassed, the agent generates $(blocksize)$ coded packets and broadcast them to the neighborhood. Due to the page limit and for the ease of explanation, we hereafter assume that we only use the block coding scheme.

On reception of a coded packet $\mathbf{c}_{(blockid, blocksize)}$, every node stores the packet in its local memory for later decoding and forwarding. To recover $blocksize$ original frames belonging to $(blockid, blocksize)$, a node should collect a $blocksize$ number of coded packets tagged with $(blockid, blocksize)$ and encoding vectors that are linearly independent of each other. Once collected, the reliable delivery service agent recovers the $blocksize$ original data frames and deliver them to the upper layer. Let \mathbf{c}_k be a coded packet labeled $(blockid, blocksize)$ in a node's local memory, \mathbf{e}_k be the encoding vector prefixed to \mathbf{c}_k , and $\mathbf{p}_{blockid+k-1}$ be an original data frame to be recovered where $k = 1, \dots, blocksize$. Further, let $\mathbf{E}^T = [\mathbf{e}_1^T \dots \mathbf{e}_{blocksize}^T]$, $\mathbf{C}^T = [\mathbf{c}_1^T \dots \mathbf{c}_{blocksize}^T]$, and $\mathbf{P}^T = [\mathbf{p}_{blockid}^T \dots \mathbf{p}_{blockid+blocksize-1}^T]$, then conceptually $\mathbf{P} = \mathbf{E}^{-1}\mathbf{C}$, which corresponds to the original data frames where superscripts T to denote the transpose operation. Note that all \mathbf{e}_k 's must be linearly independent to be able to invert \mathbf{E} .

When a node receives, a coded packet with a new tuple $(blockid, blocksize)$, it sets up a timer for the tuple $(blockid, blocksize)$ expiring in $blocktimeout$ seconds. When the timer expires it broadcasts one coded packet $\hat{\mathbf{c}}_{(blockid, blocksize)}$ after local re-encoding to its neighbors.

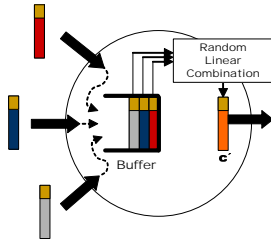


Fig. 2. Re-encoding at an intermediate node

The local re-encoding is through the same process that the data source has undergone to generate a coded packet, i.e., a random linear combination of packets with the same $(blockid, blocksize)$ available in local memory (see Figure 2). Note that though the packets in memory are coded ones thus the re-encoded packet $\hat{\mathbf{c}}^{(blockid, blocksize)} = \sum_{k=1}^{blocksize} \hat{e}_k \mathbf{c}_k$ is tagged with the encoding vector $\hat{\mathbf{e}} = \sum_{k=1}^{blocksize} \hat{e}_k \mathbf{e}_k$ where each \hat{e}_k is drawn uniformly from \mathbb{F} and \mathbf{c}_k and \mathbf{e}_k are again a coded packet labeled $(blockid, blocksize)$ in memory and the encoding vector prefixed to \mathbf{c}_k respectively (see Figure 2). The timer for $(blockid, blocksize)$ is reset on expiration unless decodable set of packets is collected for the tuple $(blockid, blocksize)$.

On the expiration of the timer for $(blockid, blocksize)$, even though there are less number than $blocksize$ of packets of $(blockid, blocksize)$ in the local memory, a node has to generate and transmit a coded packet using packets available in memory. The number of frames/packets that are combined to yield a coded packet is recorded in the field *rank* in the header of the coded packet. Since a coded packet $\mathbf{c}^{(blockid, blocksize)}$ with *rank* smaller than $blocksize$ indicates that the sender of \mathbf{c} is in need of more coded packets tagged with $(blockid, blocksize)$ to recover original frames, on reception of such packets, a node transmitting another coded packet to help the sender of \mathbf{c} collecting more coded packets. Owing to this recovery process in combination of buffering of packets, our protocol can deliver packets efficiently and reliably across partitions. Suppose that a vehicle encounters a platoon of other vehicles carrying data that the vehicle do not have. The vehicle runs the recovery process and it collects data from the platoon. By recovering process we mean a vehicle sending out a coded data packet tagged with $blockid$, $blocksize$, and *rank* where $rank < blocksize$ and in response to the *helprequest* packet, neighbors of the vehicles sending appropriate coded data packets. If a vehicle in seek of help has no data, then it just sends out header only packets with $rank = 0$. In fact, the help request and responses handshaking is not necessarily to be done block by block (or generation by generation) if a vehicle wants to collect a consecutive blocks of frames. There also can be a case where $blocksize$ is unknown. Suppose a vehicle wants to collect frames with consecutive sequence numbers from M to N where $N - M$ is larger than $blocksize$

or $blocksize$ is unknown. Then the vehicle broadcast a header only packets with $blockid = M$, $blocksize = (N - M)$ to its neighborhood and neighbors respond with coded packets $\mathbf{c}^{(blockid, blocksize)}$'s where $blockid \geq M$ and $(blockid + blocksize) \leq N$.

Since every transmission is MAC/link layer broadcasting, a small random amount of wait time called broadcast jitter is applied to every transmission. Without broadcast jitter, MAC/link layer broadcasting suffers severely from the hidden terminal problem.

IV. SIMULATION RESULTS: DELIVERY WITHIN A SIMPLE PLATOON

In this section, we study via simulation the performance of our Network Coding based Data Dissemination (NCDD) scheme when applied to a single connected vehicular networks—say, a platoon of cars—and compare it with conventional multicast. To this end, we implemented NCDD in QualNet [17], a packet-level network simulator, and conducted a set of simulations using following settings: 802.11 DCF MAC; Two-ray ground path-loss propagation model; 376 *m* of transmission range and 3 *Mbits/sec* of bandwidth (which is the minimum data rate of DSRC); one data source generates a constant bit-rate 10 *Kbytes/sec* traffic using fixed 512 bytes packet size and every node receives the traffic. Nodes move according to the Real-Track (RT) mobility model [16]. RT permits to model vehicle mobility in an urban environment more realistically than other simpler and widely used mobility models, such as Random Way Point (RWP) by restricting the areas where nodes can appear (e.g., roads). Also, in the RT model vehicles tend to aggregate and move in groups because of traffic signals and because directions change only at road intersections. The minimum node velocity is fixed to zero and we vary the maximum node velocity. Results are averaged over 10 runs with various random seeds. For NCDD, we set 40 *ms* for *blocktimeout* and $blocksize = 8$.

We contrast NCDD to the plain UDP service running on top of a conventional multicast protocol in mobile settings. In the comparison, UDP assumes underlying On Demand Multicast Routing Protocol (ODMRP) [11] routing protocol. We restrict our attention to ODMRP since as shown in [11] it is one of the best performing multicast protocols for the one-to-many communications especially in mobile and lossy channel settings in which we are specially interested. The challenge of vehicular networks is, in fact, maintaining network operations in face of nodes' mobility and lossy wireless channel. To simulate a lossy channel, nodes are forced to drop successfully received packets randomly with some probability. In figures, NCDD-dp β denotes NCDD with packet drop probability $\beta\%$ and similarly UDP-dp β denotes UDP for the packet drop probability $\beta\%$ case.

First, we look at a *street scenario*. Two hundred vehicles are moving along the streets in a confined

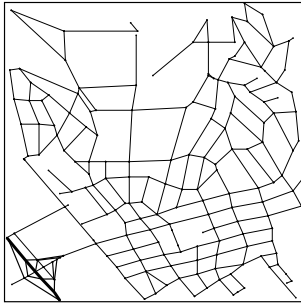


Fig. 3. Street map in the vicinity of UCLA

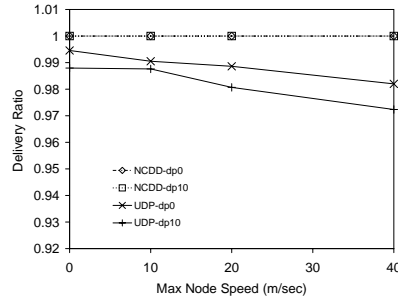


Fig. 4. Packet delivery ratio (street)

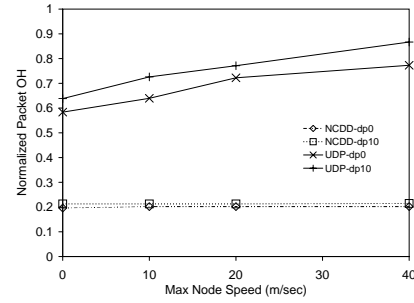
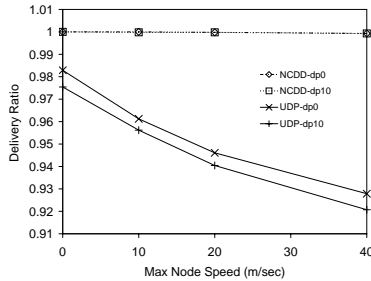
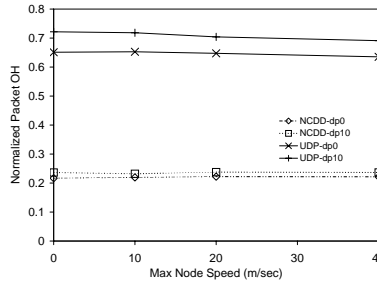


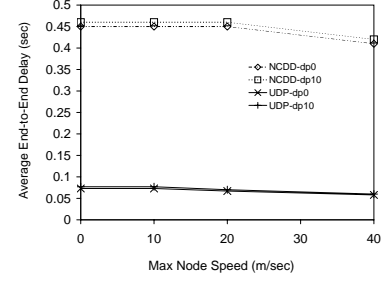
Fig. 5. Normalized packet overhead (street)



(a) Packet delivery ratio



(b) Normalized packet overhead



(c) End-to-end delay

Fig. 6. Performance of NCDD and ODMRP in the highway scenario

2000m × 2000m area as shown in Figure 3. In Figure 4 NCDD demonstrates near 100% data delivery regardless of mobility types and packet drop probability. To vary mobility types, we used four different maximum node velocities represented by the x-axis in the figure. On the other hand, the packet delivery ratio of the conventional multicasting represented by ODMRP degrades from 99% to 97% as mobility and packet drop probability increase. The packet delivery ratio is defined as the ratio of data packets received by all receivers over total data packets sent. More importantly, as shown in Figure 5, NCDD incurs less overhead than ODMRP. When the maximum node speed is 40 m/sec the reduction in overhead is as much as 70%. To measure protocol overhead, we use a common metric, the normalized packet overhead defined as the total number of packets transmitted to the wireless channel by any node in the network divided by the total number of data packets delivered to any receiver. As maximum node velocity increases, ODMRP’s overhead also increases. The reason is that ODMRP is designed to use more and more nodes as forwarding nodes to effectively counter frequent route breakages due to the increased mobility, which is equivalent to trading overhead off for high packet delivery ratio to cope with mobility.

Second, we look at the *highway scenario*. Two hundred vehicles are moving either one direction or the opposite with different speeds along the 10 km long 50 m wide track. Similar to the previous street scenario, Figure 6 shows that NCDD demonstrates near 100% data delivery regardless of mobility types and packet

drop probability whereas ODMRP’s packet delivery ratio degrades from 99% to 92% as mobility and packet drop probability increase. The overhead of ODMRP rather remains static, meaning that the number of forwarders remains constant. In a narrow highway, ODMRP fails to increase the forwarding group size and thus becomes prone to route breakage which in turn translates into low packet delivery ratio. Figure 6(c) shows the major drawback of NCDD, namely, end-to-end delay. The end-to-end delay is the difference between packet generation time at a source and packet delivery to the application at the receiver. In NCDD, a certain level of increase in end-to-end delay is inevitable since at the source it takes time to collect a block of packets such that coding over the block is possible. In our simulations, the application generates packets at a rate of 20 packets/sec so if the block size is 8 packets each packet spends on average around 0.2 seconds waiting in the buffer at the source. Note that the progressive coding scheme does not necessarily incur this delay, which is well-suited to real time applications. Propagation delay (end-to-end path can be as long as 10 km in our example) as well as packet recovery delay (which does not exist in the ODMRP case) also contribute to the NCDD’s high end-to-end delay.

V. DELAYED DELIVERY ACROSS PLATOONS

In this section we study the model with opposite direction vehicles that act as “data mules.” We are interested in evaluating different schemes and their delayed delivery performance. Figure 7 shows the overall

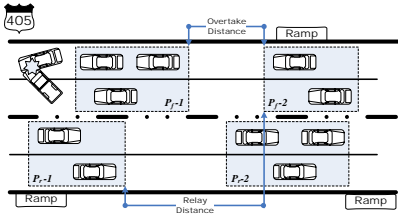


Fig. 7. Freeway relay model scenario

diagram of short segments of the freeway. Let P_{f-k} and P_{r-k} denote the k -th platoon in a forward and reverse direction respectively. In our specific scenario, the vehicles exchange information about an accident. Namely, vehicles located just in front of the accident site multicast video streams to the platoon P_{f-1} . Obviously the data cannot propagate immediately to other disconnected platoons such as P_{f-2} . Our freeway relay model uses platoons in the opposite direction to pick up, carry, and forward the data to the other platoons on the forward direction, e.g., P_{f-2} .

In this paper we are interested in calculating the delay of delivering the complete data file to the other platoons. This can be simply expressed as follows:

$$\text{Delay} = \min(\text{overtake delay}, \text{relay delay})$$

Overtake delay is defined as the time for a random platoon to catch up with (reach within its communication range) the source platoon (or other platoons that have merged into the source platoon) that is driving in the *same* direction. Relay delay is defined as the time for a target platoon to receive the whole data from data mule platoons driving in the opposite direction. For example, let us say that we have a target platoon P_{r-2} as shown Figure 7. Overtaking delay is the time for P_{f-2} to catch up with the source platoon P_{f-1} and relay delay is the time for P_{r-1} to encounter P_{r-2} . Given the limited data transfer rate between two crossing platoons, the target platoon may have to receive different parts of the data file from different platoons, i.e., data mules. This is the main focus of our analysis.

When relaying data, we could image the following strategies:

- *Relay without coding (R-WC)* - A platoon passing by the accident site randomly picks up a number of packets and “data mules” them to the disconnected target platoon.
- *Relay with erasure coding (R-EC)* - A source encodes the data using erasure coding. Erasure coding protects from packet loss caused by relaying platoons that may exit the freeway.
- *Relay with network coding (R-NC)* - The data is distributed using random linear network coding.

Vehicles arrive at independently distributed random intervals. In the traffic theory [12], this type of random arrival is often modeled using an exponential distribution with parameter λ where λ is the flow in vehicles/second.

Without loss of generality this can be extended also to a platoon arriving at the scene of the accident on the freeway. The size of a random platoon could be represented as a geometric distribution with mean s_p . Then, the overall arriving process is simply modeled as a compound Poisson process. Since our focus is calculating the relay delay, we simply assume that the speeds of platoons are constant with v_0 . The distance between two platoons are purely determined by the underlying Poisson arrival process.

Vehicles within a platoon may leave the freeway. This depends on the density of ramps along the freeway as well as the probability of defecting from a given platoon. Let d_r denote the density of ramps and \bar{p}_l denote the average defecting probability from a given platoon.² The ineffectiveness of a random platoon in terms of delivering packets can be expressed $p_{ie} \propto d_r \cdot \bar{p}_l$, thus $p_{ie} = \alpha \cdot d_r \cdot \bar{p}_l$ where α is a constant. Let p_e denote the effectiveness of a random platoon, which is given as $p_e = 1 - p_{ie}$. Let $N_p(t)$ denote the number of packets that a random platoon can pick up at time t ; for example, in the figure how many packets P_{r-1} can pick up from P_{f-1} .³ For ease of analysis, we assume that a platoon can pick up on average \bar{N}_p packets. The effective number of packets delivered to the target platoon is simply given as $p_e \bar{N}_p$.

Let N_d denote the total number of packets for a given data file that must be delivered. We assume that $N_d > p_e \bar{N}_p$, or we need multiple number of platoons to get the whole data. Since a random platoon arrives at the highway with a Poisson process, the average delay between two platoons is simply given as $\bar{T}_h = 1/\lambda$. From this, we see that the average distance is simply $\bar{D}_h = \bar{T}_h \times v_0$. We model a low traffic flow scenario such that the average distance is larger than the communication range, i.e., two consecutive platoons cannot directly communicate. Now we are ready to analyze the proposed schemes.

Relay without coding – Each “mule” platoon randomly picks up \bar{N}_p packets and delivers on average $p_e \bar{N}_p$ packets to the target platoon. This problem is analogous with the *coupon collection problem*. The number of coupons, i.e., the number of data packets that we need to deliver, is given as N_d . At each step, we draw a coupon whose type is uniformly distributed among all N_d types. Thus, the average number of trials required at each step to get a new coupon increases as more coupons have been collected. Let T_{N_d} be the time by which we have collected coupons belonging to all N_d distinct types. Then, the expected number of trials that we need to make

²The defecting probability mainly depends on the size of a platoon.

³ $N_p(t)$ is directly related to the length of a platoon as well as the speed of a platoon.

is simply given as:

$$\mathbb{E}[T_{N_d}^{R-WC}] = 1 + \frac{N_d}{N_d - 1} + \dots + \frac{N_d}{1} \quad (1)$$

$$= N_d \sum_{k=1}^{N_d} \frac{1}{k} \quad (2)$$

$$\simeq N_d \ln N_d \quad (3)$$

Recall now that each platoon can carry $p_e \bar{N}_p$ number of packets, and thus, the expected number of platoons that a target platoon must see (i.e., passes the data mule can make in order to transfer the file) is given as $\lceil \mathbb{E}[T_{N_d}^{R-WC}] / p_e \bar{N}_p \rceil$. It is interesting to note that since the target platoon travels in an opposite direction, the expected delay for the target platoon to meet a random platoon is simply given as $\bar{T}_h/2$. Therefore, the overall delay D_{R-WC} is given as

$$D_{R-WC} = \left\lceil \frac{\mathbb{E}[T_{N_d}^{R-WC}]}{p_e \bar{N}_p} \right\rceil \times \frac{\bar{T}_h}{2} \quad (4)$$

$$= \left\lceil \frac{N_d \ln N_d}{p_e \bar{N}_p} \right\rceil \times \frac{\bar{T}_h}{2} \quad (5)$$

Thus, the time grows with $N_d \ln N_d$, where N_d is the data file size.

Relay with erasure coding – Given that redundancy factor is r (>0), erasure coding produces $N_d(1+r)$ coded packets where N_d is the number of the original data packets. The key property of erasure coding is that the original data can be reconstructed from any of N_d packets. R-EC is quite similar to R-WC but we now only collect any subset with size N_d of $N_d(1+r)$ coded packets; then, we may want to ask what the relative advantage of using erasure coding in terms of the average delay. Similar to R-WC, the expected number of trials for R-EC is given as:

$$\mathbb{E}[T_{N_d}^{R-EC}] = 1 + \frac{N_d(1+r)}{N_d(1+r) - 1} + \dots + \frac{N_d(1+r)}{N_d r} \quad (6)$$

$$= N_d(1+r) \sum_{k=N_d r}^{N_d(1+r)} \frac{1}{k} \quad (7)$$

$$\simeq N_d(1+r) \ln \left(1 + \frac{1}{r} \right) \quad (8)$$

The overall delay D_{R-EC} is given as

$$D_{R-EC} = \left\lceil \frac{N_d(1+r) \ln(1 + 1/r)}{p_e \bar{N}_p} \right\rceil \times \frac{\bar{T}_h}{2} \quad (9)$$

Note that the constant logarithm factor $\ln(1 + 1/r)$ plays a key role determining $\mathbb{E}[T_{N_d}^{R-EC}]$. If $r \geq 0.5$, we have $\mathbb{E}[T_{N_d}^{R-EC}] \simeq N_d(1+r)$. It is important to note that the delay improvement of R-EC comes at the cost of increased redundancy factor r . The video source has to generate and broadcast redundant packets proportional to the total number of packets ($r \times T_d$); thus, we are utilizing the channel less efficiently (also potentially

causing more collisions).

Relay with network coding – Both R-WC and R-EC are analogous to coupon collection. The intrinsic problem of coupon collection is that once we have collected half of the coupons, it takes a progressively longer and longer time to collect the rest of coupons. On the other hand, “algebraic mixing” of the original data in random network coding help us to attain near optimal bound of data dissemination. The key idea of random network coding is that no matter how many coded packets you have collected so far, any new random coded packet is “helpful” with high probability. This claim has proven in [3], and for the sake of completeness, we include the result in Lemma 1. Suppose that we use random linear coding with a finite field of size q . As mentioned earlier, a coded packet includes a code-vector as well as coded data. In order to decode packets, one must collect N_d independent code vectors (i.e., rank N_d).

Lemma 1: Suppose node u transmits a coded packet to node v . Let S_u^- and S_v^- denote the subspaces spanned by the code-vectors with u and v respectively at the beginning. Let S_u^+ denote the subspaces spanned by the code-vectors by u after receiving a coded packet from v . Then,

$$Pr(\dim(S_u^+) > \dim(S_u^-) | S_v^- \not\subseteq S_u^-) \geq 1 - \frac{1}{q},$$

where q is the size of the field. ■

Let $\tilde{T}_{N_d}^{R-NC}$ denote the number of IID Bernoulli trials with success probability $p = 1 - 1/q$. Trials with probability p are being continued until the N_d -th success. The distribution gives the probability that z experiments are needed to reach N_d successes, which is known as a negative binomial distribution as shown below.

$$Pr(\tilde{T}_{N_d}^{R-NC} = z) = \binom{z-1}{N_d-1} p^{N_d} (1-p)^{z-N_d}, \quad (10)$$

where $z = N_d, N_d + 1, \dots$. Thus, the expected number of trials is given as

$$\mathbb{E}[\tilde{T}_{N_d}^{R-NC}] = \frac{N_d}{p} = N_d \frac{q}{q-1} \quad (11)$$

From Lemma 1, the actual success probability is greater than p , and thus, $\mathbb{E}[\tilde{T}_{N_d}^{R-NC}]$ will be the upper bound of the mean number of trials:

$$N_d < \mathbb{E}[T_{N_d}^{R-NC}] \leq \mathbb{E}[\tilde{T}_{N_d}^{R-NC}] \quad (12)$$

As the field size increases, Equation 11 shows that the upper bound approaches to the lower bound. For a large field size, we conclude that

$$\mathbb{E}[N_d^{R-NC}] \simeq N_d \quad (13)$$

Therefore, the overall delay D_{R-NC} is given as

$$D_{R-NC} = \left\lceil \frac{N_d}{p_e \bar{N}_p} \right\rceil \times \frac{\bar{T}_h}{2} \quad (14)$$

Note that compared to R-WC where the delivery time is proportional to $N_d \ln N_d$, that of network coding is proportional to N_d . R-EC can achieve the same delay, but again with great cost. In conclusion, network coding provides a considerable improvement in the time required to propagate reliably via “data muling” critical video streams to disconnected platoons.

VI. CONCLUSIONS

In this paper we have considered the problem of disseminating emergency video streams to oncoming vehicles after an accident or a major disaster. To this end, we have proposed Network Coding and have evaluated its efficacy and suitability for this task. We have found via simulation that the video dissemination to vehicles connected to the source (i.e., single platoon) can benefit from Network Coding especially in fast mobility and when the radio channel is degraded by errors and interference. For example, for average speeds in the order of 40 m/s and channel drop rates of 10%, Network Coding offers 100% delivery ratio while one of the most robust multicast protocols, ODMRP, yields only 92% delivery ratio. The overhead also tends to be lower in Network Coding.

We also explored the problem of “delayed” delivery of the video file to disconnected platoons using the vehicles traveling in the opposite direction as “data mules.” Considering the fact that the opposite vehicles can randomly pick up (by “osmosis”) only a limited number of the packets at each pass, this problem was reformulated as a “coupon collector” problem. Using simple analytic models, we have shown that Network Coding outperforms two previously proposed schemes by featuring a delivery time linear with file size N , as opposed to $N \log N$ as random picking without coding, and random picking with erasure coding. This result is important as it will enable much prompter situation awareness in both urban grid and intercity highways.

Future work in this area will be focused to the development of more accurate simulation models for single platoon as well as multiple disconnected platoon dissemination. In the latter case, we will seek validation of our analytic approximations. We will also investigate the usefulness of network coding for a broader range of vehicle applications and scenarios beyond safe navigation.

REFERENCES

- [1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. *46(4)*:1204–1216, July 2000.
- [2] P. Chou, Y. Wu, and K. Jain. Practical network coding. 2003.
- [3] S. Deb, M. Médard, and C. Chout. Algebraic Gossip: A Network Coding Approach to Optimal Multiple Rumor Mongering. In *Allerton'04*, Allerton, IL, Sep. 2004.
- [4] M. D. Dikaiakos, S. Iqbal, T. Nadeem, and L. Iftode. Vtip: An information transfer protocol for vehicular computing. In *VANET'05*, Sept. 2005.
- [5] M. Guo, M. H. Ammar, and E. W. Zegura. V3: A vehicle-to-vehicle live video streaming architecture. In *PerCom'05*, Mar. 2005.
- [6] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. Submitted to *IEEE Trans. Inform. Theory*.
- [7] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Médard. The importance of being opportunistic: Practical network coding for wireless environments. In *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*, Sept. 2005.
- [8] R. Khalili and K. Salamatian. A new relaying scheme for cheap wireless relay nodes. In *Proc. Third International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WIOPT 2005)*, pages 197–206, Apr. 2005.
- [9] R. Koetter and M. Médard. An algebraic approach to network coding. *11(5)*:782–795, Oct. 2003.
- [10] G. Korkmaz, E. Ekici, F. Özgüner, and U. Özgüner. Urban Multi-Hop Broadcast Protocols for Inter-Vehicle Communication Systems. In *VANET'04*, Philadelphia, PA, USA, Oct. 2004.
- [11] S.-J. Lee, W. Su, and M. Gerla. On-demand multicast routing protocol in multihop wireless mobile networks. *To appear in ACM/Kluwer Mobile Networks and Applications, special issue on Multipoint Communications in Wireless Mobile Networks*, 2002.
- [12] H. Lieu. *Revised Monograph on Traffic Flow Theory*. U.S. Department of Transportation Federal Highway Administration, 2003.
- [13] D. S. Lun, M. Médard, and R. Koetter. Efficient operation of wireless packet networks using network coding. In *Proc. International Workshop on Convergent Technologies (IWCT) 2005*, June 2005. Invited paper.
- [14] D. S. Lun, M. Médard, R. Koetter, and M. Effros. On coding for reliable communication over packet networks. Submitted to *IEEE Trans. Inform. Theory*.
- [15] T. Nadeem, S. Dashtinezhad, C. Liao, and L. Iftode. TrafficView: Traffic Data Dissemination using Car-to-Car Communication. *ACM Mobile Computing and Communications Review (MC2R)*, 8(3):6–19, July 2003.
- [16] A. Nandan, S. Tewari, S. Das, M. Gerla, and L. Kleinrock. Adtorrent: Delivering location cognizant advertisements to car networks. In *Proc. Third Annual Conference on Wireless On demand Network Systems and Services WONS'06*, 2006.
- [17] S. Networks. <http://www.scalable-networks.com>.
- [18] J.-S. Park, D. S. Lun, Y. Yi, M. Gerla, and M. Médard. Code-Cast: A Network Coding Based Ad Hoc Multicast Protocol. *submitted for publication*, 2006.
- [19] M. T. Sun, L. Huang, A. Arora, and T. H. Lai. Reliable MAC Layer Multicast in IEEE Wireless Networks. In *ICCP'02*, Vancouver, Aug. 2002.
- [20] K. Tang and M. Gerla. MAC Reliable Broadcast in Ad Hoc Networks. In *IEEE MILCOM'01*, Washington, D.C., Oct. 2001.
- [21] J. Widmer, C. Fragouli, and J.-Y. Le Boudec. Low-complexity energy-efficient broadcasting in wireless ad-hoc networks using network coding. In *Proc. WINMEE, RAWNET and NETCOD 2005 Workshops*, Apr. 2005.
- [22] H. Wu, R. Fujimoto, and G. Riley. Analytical models for data dissemination in vehicle-to-vehicle networks. In *VTC'04*, Sept. 2004.
- [23] R. M. Yadumurthy, A. C. H., M. Sadashivaiah, and R. Mankaboyina. Reliable mac broadcast protocol in directional and omni-directional transmissions for vehicular ad hoc networks. In *VANET'05*, Sept. 2005.
- [24] W. Yuen, R. Yates, and C. Sung. Effect of node mobility on highway mobile infostation networks. In *MSWiM'03*, Sept. 2003.