A New Approach to Coding in Content-Based MANETs

Joshua Joy*, Yu-Ting Yu*, Victor Perez, Dennis Lu, and Mario Gerla
University of California, Los Angeles, USA
Email: {jjoy, yutingyu,vperez,slu005,gerla}@cs.ucla.edu

Abstract —In Content-Based Mobile Ad-hoc Networks (CB-MANETs), random linear Network Coding (NC) can be used to efficiently and reliably disseminate large files under intermittent connectivity. Conventional NC involves unrestricted re-encoding at intermediate nodes, and consequently is vulnerable to pollution attacks. To avoid pollution attacks, the most computationally- and energy- efficient approach is to restrict the encoding at the source. However, source restricted encoding generally reduces the robustness of the code in the face of errors, losses and mobility induced intermittence. CB-MANETs introduce a new option. In CB-MANET, intermediate nodes can cache a file and exploit their processing power, storage space, and content awareness to forward the cached file with their own signature after the file is fully reassembled in their caches as if they were new sources for that file. Thus, NC packets can be encoded not only at the originator but also at the intermediate caches while still providing full protection from pollution. In fact, this approach, which is referred to as full cache coding, allows us to identify polluters by examining their signatures. The hypothesis we wish to test in this paper is whether in CB-MANETs with pervasive caching, coding restricted to full caches can perform as well as unrestricted coding. In the paper, we examine and compare unrestricted coding to full cache coding, source only coding, and no coding. Our results show that full cache coding performs almost as well as unrestricted coding while maintaining full protection against pollution attacks.

Index Terms—MANET, Content-Based, Network Coding

I. Introduction

Mobile Ad-Hoc Network (MANET) is an infrastructure-less network architecture constituted by mobile devices. The main advantage of MANET is that it can be formed at low cost in response to temporary needs, and thus is often used in battlefield and disaster-recovery networks. In such environments, one crucial application is sharing data among groups of nodes via dissemination, for example situation awareness. The major challenges of data dissemination in MANETs are due to mobility, intermittent connectivity, scarce bandwidth, and energy limitations of mobile devices. Nodes often move at different speeds in MANETs; the fast-changing topology and extreme packet loss cause high route construction and maintenance costs, and thus degrade the dissemination efficiency.

Manuscript received March 15, 2014; revised June 23, 2014.
*These authors contributed equally to this work
Corresponding author email: jjoy@cs.ucla.edu.
doi:10.12720/jcm.9.8.588-596

Content-Based Networking (CBN) is a natural fit to the MANET scenario due to its data identifiability, built-in security, and pervasive caching. In CBN, a data object is searched and retrieved based on the data object identity instead of the IP address of the node on which it resides. More importantly, all data objects must be signed at the time when they are published, providing integrity and improved security. While in-network caching is intuitively beneficial in MANETs, the interference and consequently high loss rates incurred by high communication overhead (i.e. content searching control message exchange, content advertising overhead, channel contention due to multiple caches) remain challenging issues. To this end, network coding as a technique to improve channel reliability, to reduce traffic, and to increase channel utilization has attracted much attentions recently in Content Based MANET (CB-MANET) research community [1][2]. In this paper, we investigate the intersection of network coding and security in CB-MANET.

Traditional Random Linear Network Coding (RLNC) in MANETs performs random network coding packet mixing at intermediate nodes. The benefit of RLNC is efficient and reliable dissemination of files despite mobility, random interference, and losses. However, the downside is that pollution attack becomes possible. A pollution attack occurs when a malicious or faulty node injects invalid linear combinations of blocks into the network. These polluted blocks then get re-encoded with valid linear combinations and go undetected by honest intermediate nodes. The attack is detected only when the receiver is unable to reconstruct the original file, e.g. the reconstructed file hash does not match the original file hash. At this point, the entire file must be retransmitted from the source, if the source is still available. To protect from pollution, homomorphic signature [3][4][5] which is preserved through linear combinations, can be used. This provides non-repudiation and the ability to track and find malicious nodes. The drawback of homomorphic signatures is the processing cost, which is two order of magnitude higher than the Conventional NC mixing cost - a prohibitive proposition in heterogeneous MANETs consisting of smart phones. While there exists less costly alternatives for preventing pollution attacks [6], these solutions place limitations on topologies, require loose clock synchronization on the order of 100ms, limit the hop count, require large field sizes, or demand that new public keys be generated per generation. Obviously, these requirements are not feasible in dynamic CB-MANETs.

This leaves us with two alternatives. One option is to perform source only coding, whereby only the publisher performs network coding and signs all the blocks. Since only the source encodes and signs, non-repudiation is provided whereby the integrity of the blocks and the linear combination used to generate the blocks can be verified. Thus, receivers can identify pollution attacks and blacklist the malicious source. Another approach is full cache coding. That is, to allow certified intermediate nodes that have fully reassembled the file to perform re-encoding. To provide non-repudiation, the certified intermediate node also signs the regenerated blocks in addition to the originator. In both cases, non-repudiation is provided and thus downstream nodes are protected from untraceable pollution attacks.

The contribution of this paper is the following. First, we discuss the security concerns of unrestricted network coding in CB-MANETs and identify the alternatives of unrestricted coding: full cache coding and source only coding. Second, we perform a throughput comparison of unrestricted coding, full cache coding, and source only coding; we confirm that full cache coding remains competitive with unrestricted coding while maintaining full protection against pollution attacks in highly mobile, intermittent scenarios.

The remaining of this paper is organized as follows. In section II, we introduce the CB-MANET system considered in our evaluation. In section III, we discuss network coding, the pollution attacks to network coding and defines the alternatives. In section IV, we analyze the performance of these approaches first in a static model and carry out the hypotheses for the mobile scenario. We validate our hypotheses by the simulation results in section V. The related work is discussed in section VI. Finally, we conclude in section VII.

II. Background: Content Based MANET

We first introduce the CB-MANET system used to evaluate the different NC pollution protection strategies: Delay-Tolerant Information-Centric Ad hoc Network (DT-ICAN) [7]. DT-ICAN subsumes both the family of peer-to-peer content dissemination network (e.g. Haggle [8]) in which interests are propagated in an epidemic fashion as well as the family of Content-Based Networks in which data is cached as uniquely identifiable blocks (e.g. NDN [9]). It provides high data availability in disruptive MANETs and in the mean time preserves the possibility of content-aware caching/routing design. As in all CBNs, a data object is retrieved based on its content identity instead of the IP address of the node on which it resides. In DT-ICAN, files are segmented into blocks. The transmissions are performed in the unit of data blocks and all blocks are named as *filename/blockID*. With hierarchical naming [9], each block is associated with the data object (file) it belongs to. In the case of network coding, the blocks of a file are encoded as coded blocks and the block IDs are randomly generated. To leverage the wireless broadcast channel, all communications in this system are broadcast. In the following, we describe the major principles of DT-ICAN.

A. Bloomfilter-Based Content Searching

The major drawback of the current CBN architecture [9] in a frequently-disconnected MANET is its overhead of per-chunk interests. Unlike [9], DT-ICAN reduces the bandwidth consumption not only by per-content interest aggregation by the use of Pending Interest Table but also by per-node interest aggregation. This is done by separating the *node-interest* identifying the data a node wants, and the *request* specifying the data the node is currently asking for. In addition, each node periodically advertises its *cache summary* to assist efficient content requesting. For scalability, node-interest, request, and cache summary are all represented by Bloomfilters [10]. In the following, we introduce the usage of these three control messages in more details:

1) Node-interest

Node-Interest is a summary of data objects a node wants. Each node aggregates its interests from applications as a node-interest using bloom filter. The node-interest indicates the file object IDs instead of chunk IDs to speed up the retrieval in partitioned networks. Each node periodically broadcasts its own node-interest to one-hop neighbors, and the neighbors are not required to propagate the node-interests immediately. However, node-interests may be opportunistically disseminated over multiple hops when the relay has sufficient bandwidth. We will briefly discuss the dissemination of node-interest in II.D.

2) Request

As the encounter intervals in an ad hoc network can be very limited, the node-interests do not trigger data transmission immediately. We introduce the *request* message, which identifies a subset of data a node is willing to receive at the present time. Note that a request may consist of data IDs the node itself is interested in, or the data other nodes want.

Requests are broadcast only within one hop to retrieve contents from neighbors. When a new request comes in, nodes examine the request with the data they currently hold and initiate data transmissions for the matching data. The data transmission may be triggered in two conditions:

- When new contact is discovered: new contact may be discovered when receiving packets from neighbors. If a node detects a new contact, it broadcasts its current request.
- Periodically: The requests are also periodically broadcast to reflect the changes of the list of data of a node's interest.

Note that the node has the right to decide what contents to pull according to the volume of interests it receive, the network condition, and its local content prioritization policy. We assume nodes decide the amount of data to retrieve based on the available bandwidth, and

aggregate the desired retrievals in one request to prevent overwhelming data transmissions from multiple caches.

3) Cache summary

To assist the prioritization and ensure the compactness of requests, nodes periodically broadcast their \textit{cache summaries} to one-hop neighbors. The cache summaries may include the data object IDs, meaning the node has the complete data object, or the data block IDs if the node has only partial data object.

The cache summaries are leveraged by neighbors to decide which files/chunks to send or request. Without cache summaries, relay nodes may blindly pull redundant data based on an previously received node-interests from neighbors. The nodes also update neighbors' cache summaries based on the content names carried by the control messages and data they hear.

B. Handshake Protocol

When a data transmission is triggered by the request, a handshake procedure is needed to eliminate redundant transmission in the broadcast network. For each data chunk, the sending node first sends a Request-To-Send-Block (RTSB) carrying the chunk name to the target node. Upon receiving an RTSB, the target node sends a RTSB-Reply, which may accept or reject the block. If the block is rejected, a reject code is carried to indicate one of the three reasons: (1) The chunk is already received, (2) The complete object is already received, and (3) The chunk is being sent by other neighbors. The data is only transmitted if accepted. Once the target node receives the data, it acknowledges by an ACK. Note that all neighbors of the target node also update their cache summaries based on the RTSB-Reply and ACK.

C. Request Generation

Since node interest and request are both Bloomfilters, the most efficient way to generate a request is by merging a node's own node-interest and the ones received from its neighbors. This leads to a node-priority-based request generation policy. Namely, the data to include in the request is decided by a "node ranking" that is locally computed by the request generator. In our experiment, we assume a simple request generation algorithm that integrates the node's own interest and the *k* most recently received node-interests. However, note that the ranking algorithm used for request generation does not affect the performance of network coding.

D. Greedy Node Interest Propagation

While requests are only transmitted within one hop, the node-interests must be propagated so that the relays may request data from the data source. In our experiment, we assume the nodes propagate node-interest in a greedy way. That is, if there is still data to be requested, a node always requests the data first. Neighbors' interests are propagated when there is residual bandwidth after all data have been received. The order of node-interest broadcasts are decided using the same node ranking algorithm as that in request generation.

E. Breadcrumb Multi-hop Data Retrieval

As previously described, the requests are broadcast to one-hop neighbors and are not propagated. Therefore, the data transmission over multiple hops may be slowed down if the data is only transmitted upon requests when the network is connected. Therefore, we optimize the multi-hop data retrieval as follows.

When a data block is received, a relay propagates the data back to its original requester(s) by checking pending requests recently received from neighbors. If matches are found, the relay initiates a data transmission for the particular data. In this way, the data is delivered back to the original requestors via the trail of breadcrumbs. To eliminate redundant transmissions, if the data matches multiple interests, only one data transmission is initiated. This approach achieves the same benefit of per-content interest aggregation as in [9].

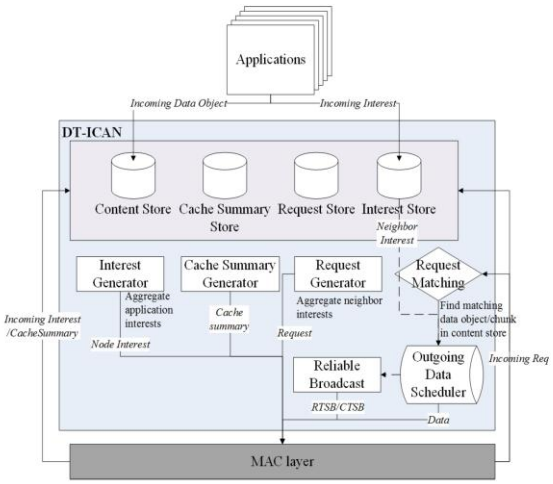


Fig. 1. DT-ICAN System Architecture

F. Reliable Broadcast

All transmissions in DT-ICAN are broadcast to better leverage the wireless broadcast channel. The consequence of this design is that there is no support from MAC layer retransmission and RTS/CTS mechanisms. Therefore, we implement a content-aware reliable broadcast layer to improve the robustness of the communication. The reliable broadcast is applied to short control messages destined to particular nodes such as RTSB and RTSB-Reply. It utilizes the data object IDs and node IDs carried in the messages to ensure delivery. An RTSB or RTSB-Reply packet is retransmitted up to two times if a relay node does not detect a progress is made, by checking the data chunk name in the incoming RTSB-Reply and Data, respectively.

The DT-ICAN system architecture is summarized in Fig. 1. The corresponding simulator, DT-ICANSIM, is available on Github (https://github.com/uclanrl/dt-icansim).

III. Network Codoing in CB-MANETs

A. Network Coding Background

We begin with a brief overview of the network coding encoding process [11], [12]. Suppose a source node wishes to disseminate a file F. The source node first transforms F into a set of m vectors $\mathbf{v_1},\ldots,\mathbf{v_m}$ in an n-dimensional vector space over a finite field $\boldsymbol{GF_p}$ where p is a prime number. These vectors are then linearly combined by drawing from the finite field $\boldsymbol{GF_p}$ n encoding coefficient $\mathbf{e_i}$ to linearly combine with the vector to create a coded block. The set of these coefficients then forms the encoding vector $\boldsymbol{e}$ which can be represented by $\mathbf{[e_1,...e_n]}$. The source generates m coded blocks $\mathbf{b_1, ..., b_m}$, where $m \geq n$.

To reconstruct the file, a node simply must recover enough (n) linearly independent coded blocks to be able to perform Matrix Inversion. First, we take the transpose of the received vectors such that $\mathbf{E^T= [e^T_1,..., e^T_n]}$, $\mathbf{B^T= [b^T_1,...,b^T_n]}$ and $\mathbf{V^T=[v^T_1,...,v^T_n]}$. Then we take $\mathbf{E^{-1}B}$ which will then reconstruct all the original blocks in the file.

B. How Network Coding Helps?

In dynamic, intermittent networks, bandwidth is scarce. The trend of increasingly cheap storage suggests to embrace the CB-MANET philosophy of compensating for intermittent connectivity with intermediate node caches. This implies that the requestors may download from multiple caches when the origin is unreachable. However, even with the help from caching, the epidemic CB-MANET still faces the following challenges in intermittent networks:

- **Last coupon problem**: Groups may form and split frequently, thus a file must be transmitted (and can be retrieved from caches) in a piecemeal fashion. Thus, pieces are often offered by caches randomly and out of order. This makes it difficult for the requestor to quickly and reconstruct the fragmented file.
- **Lack of end to end connectivity**: Hop by hop transmissions are required, with nodes acting as partial caches. Requestors must wait for the next contact opportunity to resume transmission.
- **Partial caches**: It is likely that various nodes only contain parts of a file. This means the likelihood of successful reassembling is lower when the file source is gone.
- **Busy caches**: A requestor may find out that a cache is busy serving other requestors, while the blocks being served to others are not useful for it. This in general causes the requestors must wait longer to complete transfer when many requestor nodes come and go for the same file.

Content network coding can help achieve efficient dissemination even when network partitions and severe disruptions occur and address the above challenges. The advantage of content coding can be summarized as follows.

- **Dispenses With Last Coupon Problem**: By using content coding, the last coupon problem is eliminated since with high probability each coded block received is innovative (i.e., helpful) and can be used to reconstruct the file. Thus, the throughput will be higher with content coding.
- **Overcoming Intermittent Connectivity**: Since transmissions are session-less and hop-by-hop, we cache blocks at intermediate nodes. A requestor can then ask nearby caches for network coded blocks. The neighbors pull coded blocks from their cache and either transmit as they are or mix them and transmit new coded blocks.
- **Leverage Partial Caches**: Intermediate nodes cache partial files as innovative blocks. Since each block is helpful, it is very likely the file can be reassembled as long as the total number of blocks in the network is sufficient.
- **Parallel Cache Download**: When a requestor finds a nearby cache busy serving other requestors, the coded blocks served by the cache are likely to be helpful for it too even if it has not yet been served. This speeds up the completion of file transfer and consequently increases the bandwidth utilization.

C. Protecting Network Coding From Pollution Attacks

Network coding across multiple caches and parallel downloading improves the throughput. However, as caches may often consist of only partial objects, *the partial contents cannot be signed since the signature implies that the intermediate node has received the full file, has verified the signature and has replaced in each block the originator signature with its own.* This leads us to the concern of two types of pollution attacks:

- A malicious node may mix and corrupt the coefficients such that downstream nodes are never able to successfully decode with unrestricted coding.
- The blocks may be polluted in such a way that downstream nodes are still able to decode. However, the reconstructed file's signature does not match the original file signature.

To protect network coding from pollution attack, it is required that all coded blocks must carry a valid signature. In traditional *unrestricted coding*, a relay may mix blocks from a partial file it holds but cannot sign the re-encoded blocks, and hence leaves the blocks in danger to be polluted. Two alternative options are *full-cache coding* and *source-only coding*.

In full-cache coding, we protect the coded blocks by only allowing cache re-encode blocks while generating valid signatures. That is, the cache must fully reassemble the file and verify integrity before it reissues newly re-encoded packets. We ensure the authentication, integrity, and non-repudiation as follows. The file is first signed by the source. The signature can be saved in the header of the payload. When an intermediate cache receives the full file, it verifies the source signature. Once the source signature has been validated, only then does the intermediate cache now assume responsibility for the integrity and non-repudiation. The re-encoded block is

signed by the cache owner (the intermediate node). Signing each block provides non-repudiation, as if a polluter is detected one can blacklist the polluter once it is identified. In practice, suppose a node receives from *N* caches and cannot decode, to recover from either form of pollution attacks, it requests blocks from one of the *N* caches at a time. The receiver must try to decode data from one cache at a time in order to isolate the faulty cache. The cache that provides an un-decodable stream or faulty signature is the polluter and must be investigated. Therefore, with source signatures when the file is published and with intermediate node signatures after full file reconstruction, the system is fully protected from pollution attacks.

Source-only coding refers to the approach in which only the data source (originator) is allowed to encode. In other words, relays are not allowed to re-encode even when they have received the full file, and may only forward the encoded blocks generated by the origin.

Comparing source-only coding and full-cache coding, there is a tradeoff between reassembly delay and improved orthogonality (i.e. linear independence) of the blocks. At this point, our question is whether a node should fully cache and decode/re-encode before forwarding and signing or should just forward the blocks as it receives them without generating new signatures. As we shall see, full-cache coding in some cases can improve performance considerably as compared to *source only coding*.

IV. Performance and Reliability Analysis

We hereby discuss and analyze the performance of no coding, source only coding, unrestricted coding and full-cache coding in terms of reliability under random packet losses in a static scenario as a start, and carry out our hypothesis for mobile scenarios at the end.

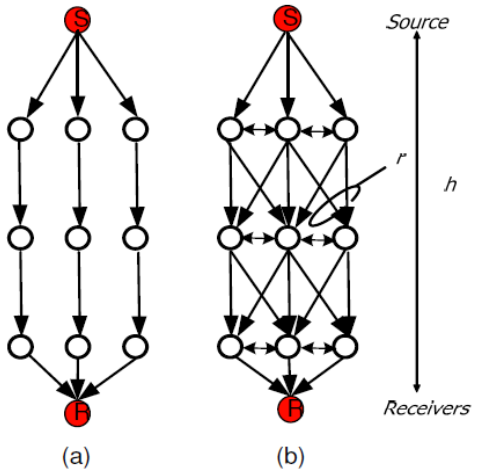


Fig. 2. 1-3-3-1 corridor model [13]. There is a single node which broadcasts at the top. A single receiver subscribes to all files.

Consider the corridor model in Fig. 2 [13]. The origin of the file is the node S. Node R has issued an interest for the file. The interest has traced several paths (as shown in Fig. 2(a) and (b)) as relay's requests. The file is split into blocks which are broadcast on the breadcrumb paths (mesh or braid) created by the interests. Note that the broadcast mode precludes MAC layer ACKs and therefore there is no loss detection and retransmissions. The model depicts two possible multipath configurations in a MANET, with perfectly disjoint (Fig. 2(a)) and highly interfering paths (Fig. 2(b)), respectively. The reality will be "in the middle", so we will study both cases and discuss the average behavior.

A. No Coding

We first start from the no coding approach as a baseline. In this discussion, we assume the application requires perfect, loss-free transmissions in short delays, so either duplicate transmissions or network coding must be used to compensate the lossy channel. Without network coding, the only way to compensate loss is by duplicate transmissions. We consider both scenarios as an *h* hop network since a packet can be propagated by either of the three relays at each of the *h* stages and reach the node R in *h* hops.

Each packet is triplicated by broadcast. Thus, 3 copies travel along the braid (Fig. 2(a)). With random channel loss probability a packet has a chance to be lost in each single-hop transmission. Note that the probability of packet loss at each "hop" is lower in Fig. 2(b) than in Fig. 2(a), because of greater link redundancy in Fig. 2(b). However, the transmission in Fig. 2(b) is slower than in Fig. 2(a) due to interference. More precisely, in Figure 2(b) the transmission is at least three times slower as in Fig. 2(b) only one of the three nodes can transmit at a given time at each hop due to the interference, while the three paths in Fig. 2(a) can be traversed in parallel. Therefore, in principle, if we consider the delay requirement as the total transmission time of mesh scenario Fig. 2(b) without extra duplicate transmissions T_{mesh} as our benchmark, we could improve the reliability in Fig. 2(a) with redundancy, that is, by transmitting up to 3 duplicate blocks all together. However, to ensure the perfect delivery with lowest delay possible, suppose the link loss rate is *l*, in the case of the braid Fig. 2(a), each relay must transmit $\lceil \frac{1}{l} \rceil$ blocks at once, meaning the bandwidth consumption will always be $\lceil \frac{1}{l} \rceil$ times than the original and the delay at the receiver must be close to $\lceil \frac{1}{l} \rceil$ long as we must send the duplicates even if the packet is not lost. One may also consider retransmission upon requests, but this leads to even longer delay as the packet loss can only be detected by timeouts. Intuitively, network coding approaches may achieve the reliability with less bandwidth consumptions and lower delays in a lossy environment.

B. Source Only Coding

Now suppose the source encodes the blocks. By the principle of linear algebra, the node R is able to

reassemble the file as long as it receives enough number of encoded blocks. In this case, suppose the link loss rate can be estimated, the source S may calculate the number of blocks needed to ensure successful delivery. For example, given the link loss rate *l*, the probability of successfully deliver a packet to the node R over *h* hops on a path *i* is

$$P_i = (1-l)^h \quad (1)$$

Therefore, for the braid topology in Fig. 2(a), the probability of a packet loss on all *k* paths is

$$l_{all} = (1-l)^{hk} \quad (2)$$

The expected number of coded blocks to be transmitted from the source is

$$N_a = \frac{1}{(1-l)^{hk}} \quad (3)$$

Therefore, source coding has a performance gain as long as $N_a < \frac{1}{l}$, for example, when $l \leq 0.17$ for this particular braid scenario (*hk=9*).

As for the mesh scenario Fig. 2(b), since with source only coding the intermediate nodes are not allowed to re-encode, the packets from different paths have about the same effects, since the probability a packet is lost for all three nodes at hop *k* is negligible, as in no coding given all three nodes at each hop receives the same packets in the mesh scenario within time T_{mesh}.

C. Unrestricted Coding

We now discuss the reliability of unrestricted coding. Consider Fig. 2(a), if packets are transmitted at the time right when they are received at the relays, intermediate node re-encoding is not useful and the performance gain is equivalent to that of source encoding. In order to benefit from unrestricted coding we must accumulate multiple blocks at the relays and re-encode them to generate new encoded blocks to provide better block diversity. In this case, if a block is lost on a strand of the braid, the next coded block will allow recovery. The more blocks we accumulate at a node, the more losses we can recover by the intermediate node generated, new linearly independent blocks. In addition, when the blocks are reassembled at the end, the triple redundancy of the three strands also comes to help.

Next, consider the Fig. 2(b). In this case, because of the interconnection between the paths, two or three blocks are accumulated in each queue at each stage except for the first hop. These blocks can be re-encoded and will allow the recovery of the lost blocks. In summary, to exploit NC's reliability benefit, we must accumulate and re-encode at intermediate nodes, at the expense of a few block delays.

D. Full Cache Coding

Now consider the case of full cache coding. Suppose the blocks are coded at the source and broadcast on the Fig. 2(a). As unrestricted coding, full cache coding only improves the performance if the intermediate nodes can re-encode the blocks. Therefore, full cache coding nodes must assemble the file first before re-encoding to outperform source only coding. However, the nodes may also start forwarding source-encoded blocks before they assembled the files, and continue with innovative blocks when they receive the full file. Once the upstream nodes have assembled the files, they can re-encode the blocks and generate as many new coded blocks as necessary to compensate for the lost blocks. The reliability gain of full cache coding, once the cache has already obtained the full file, is as good as the unrestricted coding's. In other words, the only difference between the performance gain of full cache coding and unrestricted coding comes from the duration when the full file has not been received by the intermediate nodes, and all intermediate nodes are getting the same set of blocks from upstream. During this time, the reliability gain of full cache coding is the same as that of source only coding. In all, if we assume the delay requirement is T_{mesh}, the performance gain of full cache coding is only greater than source only coding if the file can be accumulated and reassembled at the first hop before the file transfer is complete. Intuitively, this assumption is valid when the link loss rate is high, while the performance gain of full cache coding depends also on the processing and transmission delay. To summarize, the performance gain of full cache coding is "in the middle" of that of source coding and unrestricted coding. In a static scenario where the processing and transmission delay are minimal and no waiting time or source diversity, the performance gain of full cache coding depends solely on the link loss rates. When the link loss rate is low, the performance of full cache coding is similar to that of source only coding; when the link loss rate is high, full cache coding has more time and higher chance to accumulate blocks and encode, and therefore will perform more closely to unrestricted coding.

E. Hypothesis

The performance gain of full cache coding is less predictable by analysis and hence is unclear under CB-MANETs due to the fact that in CB-MANETs the intermittency enlarges the waiting time between each blocks and also higher the chance of multiple full caches before the file transfer complete. By intuition, we have the following hypotheses for mobile and intermittent scenarios:

- **Full cache coding provides higher performance gain than source only coding.** The reason is similar to the case in static scenario. Suppose there is only one originator in the network for the particular file. Once the blocks have been reassembled by caches, a receiver may receive packets from multiple caches, and the probability of getting a full rank set is higher than if the neighbors provide two identical sets. On the other hands, if there are multiple originators for the same file, the block diversity provided by multiple

originators may shorten the performance gain difference between source-only coding and full cache coding.

- **Full cache coding provides comparable performance to unrestricted coding in an intermittent network.** The main advantage of unrestricted coding is that even partial caches can re-encode and thus creating more diversity. We argue that in a mobile and intermittent scenario, the fact that nodes are intermittently connected to the source has provided similar degree of block diversity from partial caches, even without partial cache re-encoding. In other words, as the network is intermittent, the blocks received by each partial cache are more likely to be different and thus linearly independent by nature. Meanwhile, consider the case that after a sufficient time of operations, when there are already multiple sources/full caches in the intermittent scenario, the encoded blocks provided by them can offer sufficient diversity even if the partial caches do not re-encode. In this case, the performance gain of full cache coding can be comparable to that of unrestricted coding. Note that considering full cache coding has the advantage of pollution attack protection, it is a better coding approach for CB-MANET if the performance gain is comparable to that of unrestricted coding.

We will validate our hypotheses by the simulation in the next section.

V. Simulation

We evaluated the throughput of unrestricted coding, full-cache coding, source-only coding, and no coding by simulation using DT-ICANSIM, which is implemented in Qualnet 6.1.

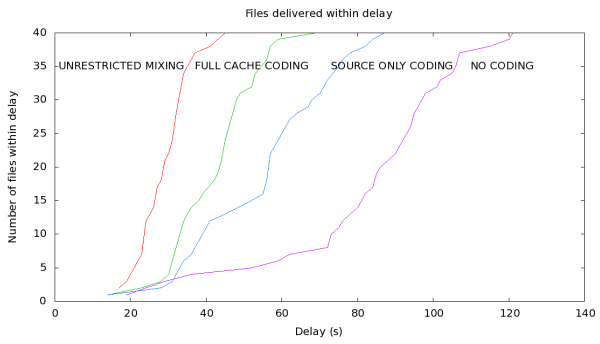


Fig. 3. Corridor model with 30% packet loss. Single publisher and single downstream receiver with partial and full intermediate caches.

A. Static Scenario

We first examine a static, lossy scenario using the corridor model in Fig. 3. We assume the channel is lossy due to interference and jamming. The loss rate is 30%. The MAC layer uses IEEE 802.11a and data rate 54Mbps. The transmission range is about 70 meters. In this scenario, we have one publisher (node S) and one subscriber (node R). For simplicity, we evaluate a single file transmission. Note that our results are generalizable to multiple files as our technique is not bound to number of files.

The results are shown in Fig. 3. Due to the lossy channel, many caches are partial in this scenario. However, a receiver can download from multiple caches in parallel. Note that for no coding we do not consider duplicate transmissions, the redundancy is only provided by multiple paths in the simulation. We observe that as expected, the performance gain of the three network coding approaches follows *Unrestricted coding>Full cache coding>Source only coding*. This matches our analysis as the unrestricted coding has the advantage of intermediate node re-encoding before the full file is received by the intermediate nodes. Full cache coding performs better than source only coding due to the ability to add diversity after the full file is obtained by the intermediate caches.

B. Mobile Scenario

We next study a mobile scenario using random waypoint model. This scenario consists of 10 nodes, including three publishers and seven receivers. The territory size is 1000 by 1000 meters. The parameters of the random waypoint model are minimum speed of 1 m/s, maximum speed of 3 m/s, pause time of 1 second, and a total duration of 10 minutes.

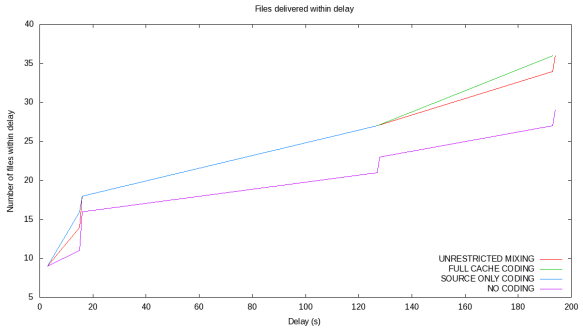


Fig. 4. 10 node mobility with 3 publishers and 7 receivers.

The results for our mobile scenario is presented in Fig. 4. We observe that full cache coding performs as well as unrestricted coding. As argued in our hypothesis, unrestricted coding gains its power when it is able to offer more diverse, innovative blocks to the receivers from intermediate node re-encoding. With mobility and the presence of multiple sources, the degree of diversity provided by partial cache re-encoding is similar to the diversity naturally obtained from the environment by full cache coding. This can also be seen from the similar performance of source only coding. As argued earlier, source only coding can have better performance as there are more originators. In this case, three originators are sufficient. Among the three network coding approaches, full cache coding does the best. The reason is that as unrestricted coding requires accumulating innovative blocks before starting forwarding, in the intermittent scenario, the delay can become longer due to the time for accumulation, and full cache coding outperforms by

simply forwarding single blocks at some relays. Meanwhile, full cache coding still has the slight advantage over source only coding even in a multiple source scenario as the number of encoders keep increasing. In all, the results show that the intermittence and multi-source have offered enough diversity for full cache coding and source only coding, and thus the performance gain is comparable to that of unrestricted coding. Given the fact that full cache coding naturally creates more sources and its resistance to pollution attacks, full cache coding is an ideal approach for coding in CB-MANET.

VI. Related Work

Network coding in MANETs has been studied in CodeTorrent and CodeCast whereby coded blocks are broadcasted and mixed at intermediate nodes [14], [15]. Lee et al. showed that by exploiting partial caches, unrestricted coding is able to greatly decrease the delay required to deliver files. However, [14], [15] does not look into the effect of pollution attacks and its remedies.

For protecting network coding from pollution attacks, Oh and Gerla showed that it is sufficient in a MANET for a small fraction of nodes to use homomorphic signatures with unrestricted network coding, while the other nodes simply forward [16]. This is useful in heterogeneous radio scenarios with powerful laptops and light smart phones internetworked in the battlefield. Untrusted nodes are only able to forward blocks; thus, signatures are preserved and pollution attacks are prevented. Only trusted nodes are able to code and append a secure "digest" so that downstream nodes can verify the digest and discard polluted blocks. Our contribution in this paper is different as full cache coding is suitable for general network devices and does not require the overhead of homomorphic signatures. Homomorphic cryptography is computationally expensive and on the order of 2 times more expensive than unrestricted coding [3]-[5]. This makes homomorphic cryptography infeasible for mobile devices such as smartphones.

More practical approaches for wireless networks have been proposed which utilize checksums [6]. However, these approaches require the receiver to establish loose time synchronization with the sender. Additionally, attacker identification requires joint cooperation between the receiver and source. Both of these constraints are difficult if not impossible to achieve in CB-MANETs and DTN type environments.

VII. Conclusions

In this paper we have considered intermittent MANET scenarios and have studied the impact of content routing and caching on network coding. The main focus has been the effect of caching on network coding performance as well as protection from pollution attacks. In a CB-MANET, we identify two alternative approaches to protect network coding from pollution attacks, namely: full cache coding and source only coding. To evaluate these strategies, we have conducted simulation experiments. The results show that full cache coding does as well as unrestricted coding in intermittent, mobile scenario. In fact, unrestricted coding is dominated by intermittent connectivity, limiting its ability to accumulate and re-encode. Moreover, the full cache coding strategy, by simply forwarding blocks and waiting for a full cache to re-encode, enables intermediate nodes to have throughput comparable to unrestricted coding. Based on these results, we argue that although the performance gain of unrestricted coding in static ad-hoc network is higher than that of full cache coding, this advantage is compensated in CB-MANET by the existence of multiple caches, and it is often eliminated by topology intermittence. Given the extra bonus to resist pollution attacks, full cache coding emerges as the preferred network coding approach for CB-MANETs.

References

[1] S. Wood, J. Mathewson, J. Joy, M. O. Stehr, M. Kim, *et al.*, "Iceman: A practical architecture for situational awareness at the network edge," 2013.

[2] J. Joy, Y. T. Yu, M. Gerla, S. Wood, J. Mathewson, and M. O. Stehr, "Network coding for content-based intermittently connected emergency networks," in *Proc. 19th annual Internationaz conference on Mobile Computing & Networking. ACM*, 2013, pp. 123–126.

[3] S. H. Lee, M. Gerla, H. Krawczyk, K. W. Lee, and E. Quaglia, "Performance evaluation of secure network coding using homomorphic signature," in *Proc. International Symposium on Network Coding*, 2011, pp. 1–6.

[4] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin, "Secure network coding over the integers," in *Proc. 13th International Conference on Practice and Theory in Public Key Cryptography*, Berlin, Heidelberg: Springer-Verlag, 2010, pp. 142–160.

[5] D. Boneh, D. Freeman, J. Katz, and B. Waters, "Signing a linear subspace: Signature schemes for network coding," in Proc. *12th International Conference on Practice and Theory in Public Key Cryptography: PKC '09*, Berlin, Heidelberg: Springer-Verlag, 2009, pp. 68–87.

[6] J. Dong, R. Curtmola, and C. Nita-Rotaru, "Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks," in *Proc. Second ACM Conference on Wireless Network Security*, 2009, pp. 111–122.

[7] Y. T. Yu, J. Joy, M. Gerla, and M. Y. Sanadidi, "Dt-ican: A delay-tolerant information-centric ad-hoc network," Department of Computer Science, University of California, Los Angeles, Tech. Rep. TR130017, 2013.

[8] J. Scott, J. Crowcroft, P. Hui, C. Diot, *et al.*, "Haggle: A networking architecture designed around mobile users," in *Proc. Third Annual Conference on Wireless On-demand Network Systems and Services*, 2006, pp. 78–86.

[9] V. Jacobson, M. Mosko, D. Smetters, and J. Garcia-Luna-Aceves,"Content-centric networking," *Whitepaper, Palo Alto Research Center*, pp. 2–4, 2007.

[10] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[11] T. Ho, M. M ́edard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to

multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[12] C. Fragouli, J. Y. Le Boudec, and J. Widmer, "Network coding: an instant primer," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 63–68, 2006.

[13] S. Y. Oh, M. Gerla, and A. Tiwari, "Robust manet routing using adaptive path redundancy and coding," in *Proc. First International Conference on Communication Systems and Networks*, Piscataway, NJ, USA: IEEE Press, 2009, pp. 224–233.

[14] U. Lee, J. S. Park, J. Yeh, G. Pau, and M. Gerla, "Code torrent: Content distribution using network coding in vanet," in *Proc. 1st International Workshop on Decentralized Resource Sharing in Mobile Computing and Networking*, New York, NY, USA: ACM, 2006, pp. 1–5.

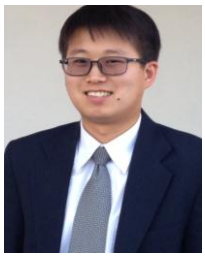
[15] J. S. Park, M. Gerla, D. Lun, Y. Yi, and M. Medard, "Codecast: A network-coding-based ad hoc multicast protocol," *Wireless Communications, IEEE*, vol. 13, no. 5, pp. 76–81, 2006.

[16] S. Oh and M. Gerla, "Protecting network coded packets in coalition networks," in *Proc. Seventh International Conference on Wireless On-demand Network Systems and Services*,2010, pp. 168–175.

Joshua Joy is currently pursuing his Ph.D. degree with the Department of Computer Science at UCLA under the guidance of Professor Mario Gerla. His research interests include mobile information centric networks, the mobile cloud, and privacy in future edge networking. He received his M.S. degree also from UCLA in Computer Science.

Yu-Ting Yu was born in Taipei, Taiwan. She received her B.S. degree from the National Chiao Tung University (NCTU), Hsinchu, in 2005 and her M.S. degree from the University of California, Los Angeles, in 2012, both in computer science. She is currently pursuing the Ph.D. degree with the Computer Science Department, University of California, Los Angeles. Her research interests include mobile and wireless networking, content-based networking, and future Internet.

Victor Perez was born in Los Angeles, California. He received his B.S. degree in Electrical Engineering and Computer Science from the University of California, Berkeley in 2006. He obtained his M.S. degree in Computer Science from the University of California, Los Angeles in 2013. He currently works as a software engineer. His research interests include operating systems, programming languages, mobile ad hoc networks, and content-based networking.



Dennis Lu received his B.S. degree in Computer Science from the University of California, Riverside. He obtained his M.S. degree in Computer Science from the University of California, Los Angeles. His research interests include information centric networks and content-based mobile ad hoc networks.

Mario Gerla is a Professor in the Computer Science Dept at UCLA. At UCLA, as a student in 1969 he was part of the team that developed the early ARPANET protocols under the guidance of Prof. Leonard Kleinrock. He joined the UCLA Faculty in 1976. His research involves mobile communications and applications. His team is developing a Vehicular Testbed for safe navigation, content distribution, urban sensing and intelligent transport.