INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Benefits of Network Coding in Disruption Tolerant Networks

Xiaolan Zhang — Giovanni Neglia — Jim Kurose — Don Towsley

N° 7277

June 2010

Networks and Telecommunications

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--7277--FR+ENG

Benefits of Network Coding in Disruption Tolerant Networks

Xiaolan Zhang[*], Giovanni Neglia[†], Jim Kurose[‡§] , Don Towsley[‡¶]

Theme : Networks and Telecommunications
Networks, Systems and Services, Distributed Computing
Équipe-Projet Maestro

Rapport de recherche n° 7277 — June 2010 — 28 pages

Abstract: In this report, we investigate the benefits of applying a form of network coding known as Random Linear Coding (RLC) to unicast communications in mobile Disruption Tolerant Networks (DTNs). Under RLC, DTN nodes store and forward random linear combinations of packets as they encounter other DTN nodes. We first consider RLC applied to a single block of K packets where *(a)* all K packets have the same source and destination, *(b)* the K packets have different sources but a common destination and *(c)* the K packets each have a different source/destination pair; we also consider the case where blocks of K packets arrive according to a Poisson bulk arrival process. The performance metric of interest is the delay until the last packet in a block is delivered. We show that for the single block case, when bandwidth is constrained, applying RLC over packets destined to the same node achieves (with high probability) the minimum delay to deliver the block of data. We find through simulation that the benefit over non-network-coded packet forwarding increases further when buffer space within DTN nodes is limited. For the case of multiple blocks, our simulations show that RLC offers only slight improvement over the non-coded scenario when only bandwidth is constrained, but more significant benefits when both bandwidth and buffers are constrained. We remark that when the network is relatively loaded, the RLC scheme achieves improvements over non-coded schemes only if the spreading of the information is appropriately controlled.

Key-words: Delay Tolerant Networks, Disruption Tolerant Networks, opportunistic networks, wireless ad hoc networks, epidemic routing, performance modeling.

[*] Computer & Information Sciences Department, Fordham University, NY, U.S.A., zhang@cis.fordham.edu
[†] EPI Maestro, INRIA Sophia Antipolis Méditerranée, France, giovanni.neglia@inria.fr
[‡] Department of Computer Science, University of Massachusetts - Amherst, MA, U.S.A.
[§] kurose@cs.umass.edu
[¶] towsley@cs.umass.edu

Avantages du codage réseau dans les réseaux tolérant les perturbations

Résumé : Dans ce rapport, nous examinons les avantages de l'application d'une forme de codage réseau connu sous le nom de codage aléatoire linéaire (RLC) aux communications point-à-point dans les réseaux mobiles tolérant les perturbations (DTNs). Quand les nœuds utilisent le RLC, ils stockent et transmettent des combinaisons linéaires et aléatoires de paquets lors de rencontres avec d'autres nœuds du réseau DTN. Nous avons d'abord envisagé d'appliquer le RLC à un seul bloc de K paquets où *(a)* tous les K paquets ont la même source et la même destination, *(b)* les K paquets ont différentes sources, mais une destination commune et *(c)* les K paquets ont chacun une paire source-destination différente. Nous considérons également le cas où le processus d'arrivée des blocs de K paquets est un processus de Poisson. La métrique de performance qui nous intéresse est le délai jusqu'à la livraison du dernier paquet d'un bloc. Nous montrons que pour le cas d'un seul bloc, si la bande passante est limitée, l'application du RLC aux paquets destinés à un même nœud minimisera (avec une probabilité élevée) le délai pour livrer le bloc de données. Lorsque le tampon au niveau des nœuds DTN est de taille limitée, nous trouvons par simulation que l'usage du RLC est encore plus avantageux par rapport à une transmission de paquets sans codage réseau. Lorsque plusieurs blocs sont considérés, nos simulations montrent que si la bande passante est limitée l'usage du RLC n'améliorera que faiblement le délai par rapport au scénario sans codage. Toutefois, l'amélioration sera plus importante si les tampons sont en plus limités. Nous notons que lorsque le réseau est relativement chargé l'usage du RLC apportera des améliorations seulement si la diffusion de l'information est convenablement contrôlée.

Mots-clés : Réseaux tolérant les délais, réseaux tolérant les perturbations, réseaux opportunistes, réseaux ad hoc sans fil, routage épidémique, modélisation de performance.

1 Introduction

Network coding is a new field in information theory started by the seminal work by Ahlswede *et al.* [2] in 2000. Network coding refers to the new concept where the network nodes (e.g., routers, switchs) combine/mix previously received packets before forwarding them, rather than simply forwarding data received. [2] showed that network coding allows for higher network throughput for a single multicast flow case. Since then, network coding has found many intersting applications (see the short primer [9] which gave a nice review on previous works on network coding).

Random Linear Coding (RLC) is a form of network coding that was first proposed by Ho *et al.* [14]. Basically, under RLC, each network node forwards random linear combinations of the data it has received. Previous works have applied RLC to networking scenarios including P2P content distribution [10], multicast applications [5], gossip protocols [7] and distributed storage [6, 1].

In this paper, we investigate the benefit of applying Random Linear Coding (RLC) to unicast applications in DTNs with opportunistic contacts and resource constraints. To our knowledge, the only previous work applying network coding to a DTN setting is [31] by Widmer and Boudec. There, the authors consider *broadcast* data delivery using RLC; our focus here is on using RLC for unicast delivery.

For unicast applications, there are different possible ways to combine packets: each node can combine all the packets in its buffer, or only the packets destined to the same destination, or only the packets belonging to the same flow (i.e., same source-destination pair). We first consider these three possibilities in the simple case where a single block of K packets propagate in the network.

The performance metric of interest is the delay until the last packet in a block is delivered, but we will also comment on the average packet delay for a block. We show that for the single block case, when bandwidth is constrained, applying RLC over packets destined to the same node achieves the minimum delay with high probability. We find that this benefit increases further when buffer space within DTN nodes is limited. We also demonstrate that the "price" to be paid for the improved delay performance is a larger number of epidemically-spread copies of data in the network. However, when a token-based scheme is used to limit the number of transmissions made, the RLC based scheme yields a smaller average delivery delay under similar transmission overhead as non-coded schemes.

We then consider the scenario where there are multple source/destination pairs with blocks of K packets arriving according to a Poisson bulk arrival process at each source. We find that the RLC scheme achieves slightly smaller average block delay than non-coded schemes when only bandwidth is constrained, but shows more significant benefits when both bandwidth and buffers are constrained.

The remainder of this paper is structured as follows. We introduce the network model, the forwarding and recovery schemes, and the simulation setting in Section 2. Section 3 studies the benefit of the RLC scheme over non-coded schemes for the single generation case. Section 4 extends the study to multiple generation case. Section 5 reviews related work. Finally, Section 6 summarizes this paper and discusses future works.

notation	meaning
N	number of nodes in the network
β	pair-wise meeting rate between a node pair
K	block (generation) size
λ	block arrival rate to each flow
l	packet size in bits
b	number of packets can be exchanged in one direction during a meeting
B	number of relay packets a node can store
q	size of finite field GF_q, where $q = p^n$, p is a prime and n is a positive integer.
d	dimension of the packet in the finite field GF_q
A	encoding matrix
r	rank of the encoding matrix
D_{block}	the time to deliver the last packet of a block
L	per-packet token number

Table 1: Tables of notations

2 Network Model, Forwarding and Recovery Schemes

In this section, we first introduce the network model, and then describe the forwarding and recovery schemes we study in this work, and finally describe the simulation setting.

2.1 Network Model

We consider unicast communications (i.e. each message is destined for a single node) in a network consisting of N nodes moving according to a mobility model (discussed shortly) within a closed region. Each node has a limited transmission range, such that the network is sparse and disconnected.

We employ the *temporal network* model proposed by Kempe *et al.* in [17] to represent the dynamic network topology formed by the mobile nodes. Basically, a temporal network is an directed graph $G = (V, E)$ in which each edge e is annotated with a time label $\lambda(e)$ specifying the time at which its two endpoints "communicated". We extend this model such that each link also has a capacity attribute and assmume that one packet can be exchanged over each link. We construct the temporal network as follows (Figure 1): there are N vertices, each corresponding to a mobile node. For each contact between a pair of nodes that can exchange b packets in each direction, b directed edges are added in each direction between the corresponding vertices. Edges are labeled with the times that the contacts occur. A *time-respecting path* in the network is a path in the network where the successive edges have increasing timestamps. For example, there are three time-respecting paths from node 1 to node 4, i.e., two paths that goes through node 2 and one path that goes via node 3. There is no time-respecting path from node 4 to node 1. A set of paths are *independent* if they do not share edges. In this example, the two paths from node 1 to 4 going through node 2 are not independent, as they share the edge $(2, 4)$. Pathes $(1 \to 2 \to 4)$ and $(1 \to 3 \to 4)$ are independent.

Table 1 summarizes the notations used throughout this paper.

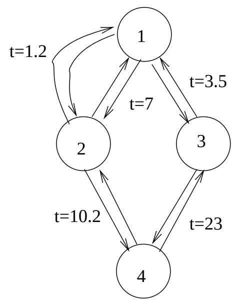


Figure 1: Random graph representing the contacts between nodes

As to the traffic model, we consider both single generation case and multiple generation case.

For the single generation case, we assume that K packets arrive at the same time in the network.We examine the following three scenarios:

- **SS_SD (Single Source/Single Destination):** the in K packets from a source are to be delivered to a single destination;
- **MS_SD (Multiple Source/Single Destination):** the K packets from different sources are to be delivered to the same destination;
- **MS_MD (Multiple Source/Multiple Destination):** the K packets (each from a different source) are to be delivered to different destinations.

We define *block delivery delay*, D_{block}, as the time from the arrival of the block in the network to the delivery of the whole block to the destination. We compare this time deliver a block of packets when the packets are forwarded without any coding or when RLC is applied to the block of packets. Depending on the specific application, other metrics could be more meaningful, like the average time to deliver a packet in the block, or the average time to deliver a packet in order. Note that D_{block} is the metric more favorable to RLC in the comparison. Other performance metrics of interest are the average number of packet copies or combinations made within the network, as this is a measure of resources consumed (bandwidth, transmission power, buffering) within the DTN.

2.2 Forwarding, Recovery Schemes

When two nodes come within transmission range of each other (i.e., they meet), they first each figure out if the other has some useful information and, if any, they try to exchange it. We detail this process with reference to the two mechanisms we are going to compare: non-coded packet-forwarding and RLC-forwarding.

Non-coded forwarding: When two nodes meet, each of them uniformly randomly selects one or more packets, depending on the available bandwidth, among the packets that the other node does not have, and forwards them to the other node. We refer to this scheme as the **random** scheme. We also consider a **RR_random** scheme in which the packet's source node chooses a packet to forward in round-robin manner, while intermediate nodes perform random selection. The round-robin scheduling at the source node gives each packet in the generation equal opportunity to be propagated, and thus speeds up the propagation of initial copies of each packet.

As each packet is duplicated by the nodes in the network, when it is first delivered to the destination, there are multiple copies of the packet in the network. A recovery scheme can be used to delete these obsolete copies from the network to free up storage space and avoid useless transmission [12]. In this paper, we focus on VACCINE recovery scheme, under which upon delivery of a packet, an *antipacket* is generated and propagated through the network (in the same way as data packets) to delete buffered copies of this packet. To simplify analysis and simulation, we assume that the storage and transmission of antipackets are not subject to bandwidth and buffer constraints.

Random Linear Coding based forwarding: RLC is applied to a finite set of K packets, $m_i, i = 1, 2, ..., K$, called a *generation*. Under the RLC based scheme, each packet is viewed as a d dimensional vector over a finite field [21], GF_q of size q. More specifically, a packet of l bit length is viewed as an $d = \lceil l/log_2(q) \rceil$ dimensional vector over GF_q, i.e., we have $m_i \in GF_q^d, i = 1, 2, ...K$.

A linear combination of the K packets is:

$$x = \sum_{i=1}^{K} \alpha_i m_i, \;\; \alpha_i \in GF_q,$$

where the coefficients $\alpha = (\alpha_1, ..., \alpha_K)$ are referred to as *encoding vector*, and addition and multiplication are over GF_q, and the generated linear combination, x, are referred to as the *encoded data*. Each original packet, m_i, can be viewed as a special combination with coefficients $\alpha_i = 1$, and $\alpha_j = 0, \forall j \neq i$. A set of r encoded data are called linearly independent if ...

Under the RLC scheme, linear commbinations of the packets, together with the coefficients, are stored and forwarded by network nodes. If a node carries r linearly independent encoded data, $X = (x_1, ..., x_r)$ (together with the corresponding encoding vectors), we say that the rank of the node is r, and refer to the $(K \times r)$ matrix made up by the encoding vectors as the node's *encoding matrix*, **A**. Essentially, a node with rank r has stored r linear equations with the K source packets as the unknown variables, i.e., $AM^t = X^t$, where M^t represents the original K packets. When a node (e.g., the destination) reaches rank K (i.e., full rank), it can decode the original K packets through matrix inversion, as $AM^t = X^t$ leads to $M^t = A^{-1}X^t$. We then can use the Gaussian elimination algorithm to solve for the original packets, M^t. [1].

Initially, the source node(s) carries the original packets, $M = (m_1, ..., m_K)$. When two nodes, say u and v meet, they first send their encoding matrices to each other, and then perform the following operations which we describe using node u as example. Node u, based on the matrix of node v, checks if it has useful information for node v[2]. If so, node u generates a random linear combination of the currently stored combinations, say $x_1, ..., x_r$, $x_{new} = \sum_{j=1}^{r} \beta_j x_j$, where the coefficients $\beta_1, ...\beta_r$ are chosen uniformly randomly from

[1] A packet can be decoded before the matrix reaches full rank, as long as the encoding matrix contains a simple encoding coefficient.

[2] In fact, if a node has at least one combination that cannot be linearly expressed by the combinations stored in another node, it has useful (i.e., *innovative*) information for the latter node.

the field GF_q. Obviously, x_{new} is also a linear combination of the original K packets. This new combination, along with the coefficients *with respect to the original packets*, is forwarded to node v. Given that node u has useful information for node v, this randomly generated combination is useful to node v (i.e., can increase the rank of node v) with probability greater or equaled to $1 - 1/q$, according to Lemma 2.1 in [7]. We further note that as node u has knowledge about the encoding matrix of node v, node u can iteratively generate random linear commbinations from its stored combinations, until a combination useful to node v is generated. Such processing pays computation overhead to attain savings in transmission bandwidth. We do not consider such processing in this thesis.

In addition to the above basic RLC scheme, we consider several implementation details that improve its performance in terms of average delivery delay using various optimizations. For example, the node (including destination) can decode packet before the matrix reaches full rank, and forward the decoded packets to destination directly (other than generate random linear combinations). The source node, instead of transmits random linear combinations of the block of packets to relay nodes, transmit the data packets in a round-robin manor. These approaches allow improvement in the average packet delivery delay for the RLC scheme.

Similar to non-coded scheme, when a generation is delivered to the destination, the destination generates an antipacket for the generation, which is subsequently propagated maximally in the network to delete buffered combinations of the generation (i.e., VACCINE recovery).

2.3 Simulation Setting

Throughout this paper, we mainly rely on simulation to quantify performance gains of the RLC scheme. We report simulation results based on a pair-wise Poisson meeting process between two nodes, rather than an actual mobility model such as random waypoint/direction mobility model. This simplification speeds up the simulation, and as [11] has shown, under the random waypoint/direction models, the inter-meeting time between a pair of nodes follows a Poisson process when node velocity is relatively high compared to the region size, and the transmission range is relatively small. We have also performed simulations using the actual mobility models and observe similar performance as the poission meeting simulation. For the results presented in this paper, we simulate a network of $N = 101$ nodes with a pair-wise meeting rate of $\beta = 0.0049$. We use a finite field of size $q = 701 = 701^1$, as field arithmetic operations of this field are simpler to implement than those of the commonly used field $GF(2^8)$ or $GF(2^{16})$.

3 Single Generation Case

Having described the network setting and the non-coded and RLC schemes, in this section, we investigate the benefit of the RLC scheme under simple setting where there is a single generation of packets in the network. In particular, we assume that K packets arrive at the same time in the network.We examine the following three scenarios: SS_SD, MS_SD, MS_MD.

3.1 Coding Benefit under Bandwidth Constraints

We first consider the bandwidth constrained case and assume when two nodes meet, they can send a maximum of b packets in each direction. Mobile nodes are assumed to have sufficient buffer space to store all packets in this section.

3.1.1 Analysis of RLC benefit

Recall that we use temporal network to represent the dynamic network formed by the nodes. It's easy to see that for the SS_SD case, where the source node at time $t = 0$ has K packets to send, the minimal time to deliver these K packets is the time when there are K independent paths from the source to the destination. Similarly, for the MS_SD case, the minimal time to deliver the K packets from the K source nodes to the single destination is the earliest time that there are K independent paths from the K source nodes to the destination.

Notice that in DTN routing schemes, the mobile nodes have no knowledge, or only delayed knowledge about packets transmission decision made by other nodes. As a result, a node along the K paths might choose to forward information that some other path is forwaring or has forwarded. Under the RLC scheme, rather than choosing from the K packets, nodes randomly and independently combine packets to generate "equally important" encode-packets. As the number of independent coded packets is much greater than K, the probability that some path forward data that is useless to the destination is much smaller than for a non-coded scheme. As [7] pointed out, such benefit of RLC scheme is well captured by the Coupon Collector Problem [8].

Consider the 4-node network as shown in Figure 1. Assume that at time $t = 0$, node 1 generates two packets m_1 and m_2 destined to node 4. By time $t = 23$, there are two edge-disjoint paths from node 0 to node 4, therefore the minimum delay to deliver the two packets is 23.

Under the RLC scheme, source node 1 forwards random linear combinations c_1 and c_2 to node 2, and c_3 to node 3 at the contacts at times $t = 1.2, 7, 3.5$ respectively. With proability $1 - 1/q$, c_1 and c_2 are independent. For the case where c_1 and c_2 are independent, node 2 stores both combinations. When node 2 meets node 4, it generates a random linear combination c_{12} of c_1 and c_2 and forwards it. If c_{12} and c_3 are independent, node 4 can decode the two original packets after node 3 delivers c_3 at time $t = 23$. Note that c_3 can be linearly expressed by c_1 and c_2, and with probability $1 - 1/q$, c_{12} is independent from c_3. For the case where c_1 and c_2 are linearly dependent, node 2 stores c_1, and forwards it to node 4 at $t = 10.2$. If c_3 and c_1 are independent (with probability $1 - 1/q$), node 4 reaches full rank at $t = 23$ and the two packets are delivered at the minimum delay. Summing up both cases, we conclude that for this particular contact scenario, RLC achieves minimum block delivery delay with probability $1 - 1/q$. In this particular example, the benefit of the RLC scheme over non-coded scheme is reflected in the forwarding decision made by node 2. Having no knowledge about the contents at other nodes (node 3), node 2 cannot choose to forward information that's not available at node 3. The RLC scheme decreases the probability that a useless infomation is forwarded through its increased randomness.

On the other hand, under non-coded scheme, node 1 forwards m_1 and m_2 to node 2 at time $t = 1.2, 7$ respectively, and one of the packets (say m_1) to node 3 at time $t = 3.5$. When nodes 2 and 4 meet at $t = 10.2$, node 2 randomly selects a packet and delivers to node 4 (as it has no global knowledge of past and future contacts for other nodes). With probability 0.5, packet m_2 is selected to forward to node 4, and thus when node 3 meets node 4 at $t = 23$, it has no useful information for node 4. Hence, the non-coded scheme achieves the minimum delay with probability 0.5.

Proposition 3.1 *If there is a single block of K packets in the network, for the SS_SD and MS_SD case, RLC achieves the minimum D_{block} with high probability. Let η be the total number of total edges along the first K independent paths from the sources to the destination, then:*

$$p_{achieve\_min\_delay} \geq (1 - 1/q)^{\eta}. \tag{1}$$

For the SS_SD case, the probability can be bounded as follows:

$$p_{achieve\_min\_delay} \leq (1 - 1/q^K)(1 - 1/q^{K-1})(1 - 1/q^{K-2})...(1 - 1/q). \tag{2}$$

Proof: The upper bound for the success probability is a direct application of Thorem 2. in **??**. For the SS_SD case, in order to achieve the minimal block delivery delay, the K combinations generated by the source node to send along the K paths must be linearly independent. Under the basic RLC scheme, the total number of ways of generating K combinations is $(q^K)^K$; among them, the number of ways of generating K linearly indepedent combinations of the K packets is $(q^K - 1)(q^K - q)(q^K - q^2)...(q^K - q^{K-1})$. Therefore the probability that K combinations generated by the source is linearly independent is given by $(1 - 1/q^K)(1 - 1/q^{K-1})(1 - 1/q^{K-2})...(1 - 1/q)$. ■

We note that to achieve the minimal block delivery delay, it's necessary that *each node* along the K paths chooses to forward a combination that is independent from combinations forwarded by nodes along the other paths. Quantifying this probability requires considerations for the exponentially large number of possible scenarios for the K paths. For example, if the K paths have no shared nodes, RLC scheme achieves minimal block delivery delay with probability 1. If the K paths ... For example, Figure 2 depicts two different meeting scenarios for a four-node network. We have, for the meeting scenario shown in Figure 2(a),

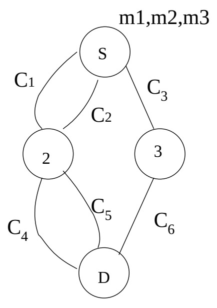
$$\begin{aligned}
p_{achieve\_min\_delay} &= prob\{c_1, c_2 \text{ and } c_3 \text{ are independent}\} \\
&\quad \times prob\{c_4 \text{ and } c_5 \text{ are independent}\} \\
&\quad \times prob\{c_6 \text{ is non trivial}\} \\
&= (1 - 1/q^3)(1 - 1/q^2)(1 - 1/q)(1 - 1/q^2)(1 - 1/q)(1 - 1/q) \\
&= (1 - 1/q^3)(1 - 1/q^2)^2(1 - 1/q)^3
\end{aligned}$$

Whereas, for the meeting scenario shown in Figure 2(b), we have

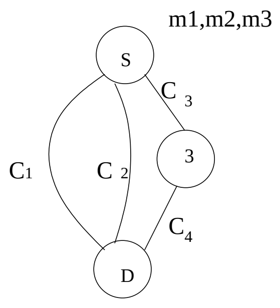
$$\begin{aligned}
p_{achieve\_min\_delay} &= prob\{c_1, c_2 \text{ and } c_3 \text{ are independent}\} \times prob\{c_4 \text{ is non trivial}\} \\
&= (1 - 1/q^3)(1 - 1/q^2)(1 - 1/q)(1 - 1/q) \\
&= (1 - 1/q^3)(1 - 1/q^2)(1 - 1/q)^2.
\end{aligned}$$

Table 2: Algorithm to find the minimal time to deliver K packets generated at *src* at t_0 destined to *dest*

```
$num\_path \leftarrow 0$
Initlialize event queue
while $num\_path < K$ do
  Generate a source pkt from $src$ destined to $dest$ at time $t_0$ {Simulate epidemic routing to find a minimal delay path for the pkt}
  while there are more events in the event queue do
    Get next event from queue, $l$
    Node $v_1(l)$ and $v_2(l)$ exchange packets
    Check if the packet is delivered to $dest$
    if the packet is delivered then
      $num\_path++$
      if $num\_path == K$ then
        break
      end if
      Update event queue: decrease bandwidth for contacts used by the packet, for the reversed direction, increase bandwidth by one
      break
    end if
  end while
end while
return $cur\_time$ as the minimal time to deliver $K$ packets
```

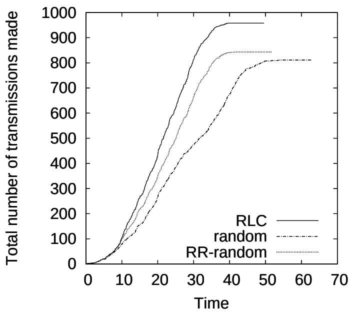


(a)

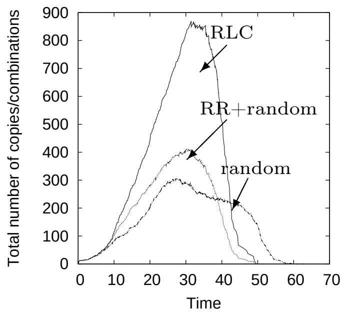


(b)

Figure 2: Two meeting scenarios for a 4-node network



(a) RLC scheme achieves faster propagation



(b) storage comsumption

Figure 3: RLC scheme versus non-coded schemes

3.1.2 Characteristics of the RLC Scheme

We now highlight several characteristics of the RLC scheme compared to the non-coded schemes using simulation.

First, we observe that the RLC scheme allows faster propagation of the information in the network, but incurs more transmissions being made in the network. For example, for a particular run for the SS_SD case with $N = 101, K = 10$ case, Figure 3(a) and (b) respectively depict the cumulative number of transmissions made, and the total number of packet copies (for the non-coded schemes) or combinations (for the RLC scheme) in the entire network as a function of time. We observe that there are two factors causing more transmissions made under the RLC scheme. first, the RLC scheme allows faster propagation of information, as the random combination performed at each node allows two nodes that meet each other to have useful information to exchange more often. Secondly, under the RLC scheme, the recovery process starts only when the whole generation is delivered, much later than under non-coded approach, where the recovery process for individual packet starts immediately when the packet is delivered.

The second point to make concerns the performance metric. Throughout this paper, we mainly study the average block delivery delay as the performance metric; there are alternative metrics such as mean packet delay, in-order delay. For multiple simulation runs of the above setting, Figure 4 plots the empirical CDF for different delay metrics achieved by RLC and RR-random scheme. It shows that although RLC is able to decreases the block delivery delay, it sacrifices performance metrics such as mean packet delay and in-order packet delay.

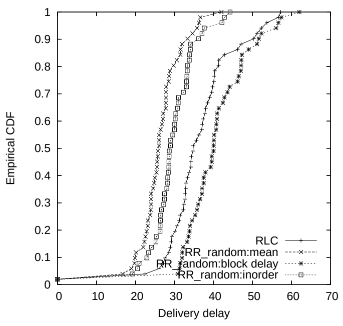
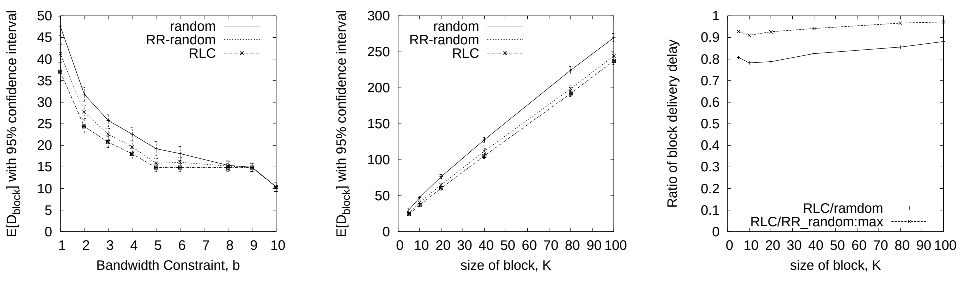


Figure 4: Comparison of different performance metrics under SS_SD with $N = 101, K = 10$

3.1.3 Performance Gain of the RLC scheme

We now quantify the performance gain of the RLC scheme through simulation. We note that due to the random nature of the contacts and the large size of the network in which we are interested, a quantitative analysis of delivery delay is difficult.

We first explore the relative benefit of the RLC scheme with respect to the non-coded schemes under varying bandwidth constraints. Figure 5(a) plots the $E[D_{block}]$ and its 95% confidence interval for the SS_SD case with $K = 10$ under varying bandwidth constraints. (The average block delivery delay, $E[D_{block}]$



(a) $E[D_{block}]$ under varying bandwidth (b) $E[D_{block}]$ under different block size (c) Benefit of RLC under different block size

Figure 5: Benefit of the RLC scheme under SS_SD

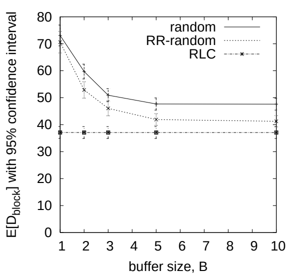
reported throughout Section 3 is the average value from 50 different simulation runs). The figure shows that that the RLC scheme achieves a lower $E[D_{block}]$ than both random and RR_random schemes. All schemes perform the same under $b = 10$ case where the $K = 10$ packets are propagated independently without competing for bandwdith; whereas as bandwidth decreases, the relative benefit of the RLC scheme increases.

We next study the sensitivity of the performance gain to the block size K. Figure 5(b) plots the average D_{block} for the SS_SD case with varying block size K and a bandwidth constraint of $b = 1$ (i.e., on every contact, only one packet can be sent in each direction. For the remainder of this paper, this is the default bandwidth constraint used in our simulation results), and Figure 5(c) plots the relative benefit of the RLC scheme over non-coded schemes. We observe that as the block size increases, the relative benefit of the RLC scheme over non-coded schemes decreases. This is because for non-coded schemes, with a larger block size, there are a larger number of packets to randomly choose from, and therefore the probability of two paths choosing to forward the same packets is smaller.

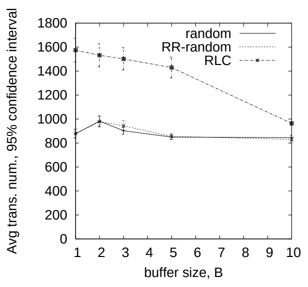
Our results for the MS_SD and MS_MD case are not shown here. We note that the benefit achieved by the RLC scheme for the MS_SD case is smaller than for the SS_SD case. Basically, under the MS_SD case, the K packets start to propagate from the K different source nodes, and the effect of relay nodes choosing the wrong packets to forward becomes less significant. For the MS_MD case, the RLC scheme performs worse than the non-coded scheme since the RLC scheme forces every destination node to receive K independent combinations to decode the one single packet destined to it.

3.2 Coding Benefit under Bandwidth and Buffer Constraints

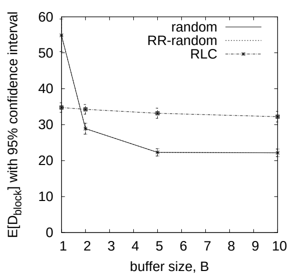
In the previous section, we have assumed that nodes have unlimited buffer capacity. In this section, we conisder the case where each node can store at most B $(B < K)$ *relay* packets or combinations, but has sufficient buffer space to store all source packets and packets destined to it. As the nodes do not have enough memory to hold the whole block of packets, an algorithm is needed to determine what to keep or drop when the buffer is full.



(a) $E[D_{block}]$ for SS_SD under varying buffer



(b) Number of transmissions made for SS_SD



(c) $E[D_{block}]$ for MS_MD under varying buffer size

Figure 6: Bandwidth and buffer constrained case

We consider the following buffer management schemes. For the RLC scheme, when a node receives a combination and its buffer is full, it randomly combines the new combination with an existing combination in the buffer and stores the result. For the non-coded scheme, a drophead scheme [35] is used. Under the drophead scheme, when a new relay packet arrives to a node, and the node's buffer is full, the node drops the relay packet that has resided in the buffer the longest.

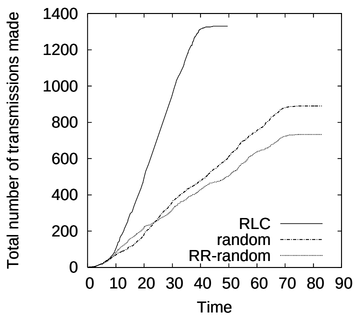


Figure 7: RLC scheme makes use of more transmission opportunities, B=1

Figure 6(a) plots the block delivery delay under the RLC scheme and the non-coded schemes for the case SS_SD (with $K = 10$) under varying nodal buffer sizes. We find that as nodal buffer sizes decrease, the performance of the RLC scheme degenerates only slightly; while the performance of the non-coded schemes degrade quickly. We examine the simulation trace closer to better understand the performance gain of the RLC scheme. For a particular run, Figure 7 plots the cumulative number of transmissions made as a function of time for different schemes. We see that the RLC scheme is able to make use of more transmission opportunities, and therefore propagates information much faster than the non-coded schemes. Further examination of the simulation traces reveals that under the RLC scheme, different information propagates evenly throughout the network. As different packets are mixed randomly by nodes, when a node drops a combination, an equal amount of information is lost for each packet. For the non-coded schemes, however, different packets in the block propagate at very uneven speeds: some packets spread quickly to a large number of nodes, while other packets spread much more slowly. The uneveness of propagation of the non-coded schemes can be explained by the adopted random selection scheme: the more copies a packet has in the network, the more likely the packet is copied to some other node with the effect of kicking out copies of other packets. As a result of such uneveness, it takes much longer to deliver the "slowest" packet using non-coded schemes than using the RLC scheme.

Again, we note that the improvement in delay performance of the RLC scheme is achieved at the cost of more transmissions being made as shown in Figure 6(b). Notice that when there is no buffer constraint, at most K linear combinations of a generation (of size K) are sent to each node. This is however, not the case when there are buffer constraints. When a relay node cannot store all combinations of a generation, it can be repeatedly sent different combinations of a generation without increasing its rank.

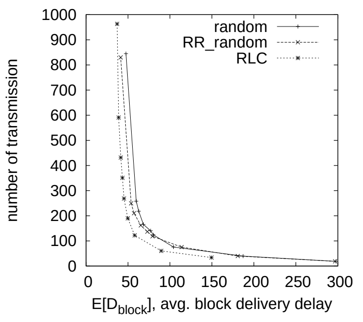
For the MS_SD case, we observe similar performance gains achieved by the RLC scheme (not shown here). For the MS_MD case with $K = 10$, where coding is applied to packets sent by different sources to different destinations, we observe that the RLC scheme out-performs the non-coded schemes when the buffer is very constrained ($K = 10, b = 1$ for this setting) as shown in Figure 6(c).

3.3 Controlling Transmission Power Consumption

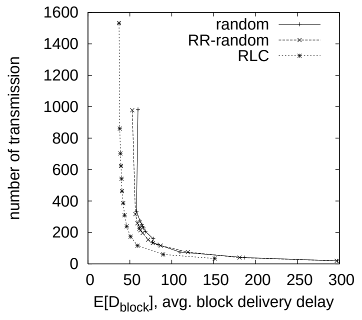
So far, we have seen that the RLC scheme delivers a block of data, or collects multiple packets from different sources faster than the non-coded schemes, at the "cost" of having more copies of packets present in the network, consuming more buffer space, transmission power and bandwidth (to send these copies). Can the RLC scheme achieve a smaller average block delivery delay than the non-coded schemes *(i)* under the same transmission power consumption, *(2)* under the same transmission power consumption and buffer constraint ? We answer these questions in this section.

To limit the number of copies made of a packet, we use a token-based scheme, extending the *binary spray and wait* scheme proposed in [29, 28]. We refer to the maximum number of copies made for a packet as *the number of per-packet tokens*. Each node carrying a copy of a packet is assigned a token number that denotes the number of copies the node can make for the packet.

The spray and wait protocol [29] with the number of per-packet tokens, L, consists of two phases: a spray phase to spread $L - 1$ copies of the packet, and a wait phase (if the destination node has not been reached) where each of L carriers (including the source) performs direct transmission to deliver the packet to the destination. There can be different ways to spread the initial $L - 1$ copies, one of them is binary spray and wait. Under binary spray and wait with number of per-packet tokens L, every new packet generated at the source is assigned $L - 1$ tokens. When the source node meets another node, the packet is copied to the other node and half of the tokens are assigned to the new copy, while the source node keeps the remaining half of the tokens. A relay node carrying a copy in turn does the same. When a packet copy has only a single token remaining, it can only be forwarded to the destination. [29] has shown that under an independently and identically distributed mobility model, binary spray and wait achieves minimum expected delay among all spray and wait routing schemes.



(a) Transmission power vs delay trade-off



(b) Transmission power vs delay trade-off under buffer constraint, B=2

Figure 8: Transmission number vs block delivery delay trade-off

We note that this scheme can be improved by allowing two nodes carrying copies of the same packet to average their token numbers when they meet, as the two nodes have equal opportunities to meet susceptible nodes or destination node (and to propagate and deliver the packet). Furthermore, to apply the notion of tokens to the RLC scheme, we associate a token number with each generation, which limits the total number of combinations that can be exchanged for the generation in the network. The *generation token number* equals the product of the number of packets in the generation and the per-packet token number. When a node sends a random combination to another node, its token number is decreased by one. After two nodes finish exchanging combinations of a generation, they average their token numbers: the sum of the two nodes' token number is reallocated to the two nodes in proportion to their ranks. Even if the two nodes meet each other have no information to exchange, they average their token numbers in the same way. The rationale behind averaging tokens in proportion to ranks is that the potential of a node to spread information is linearly proportional to the rank of the node, i.e., the "amount" of information the node carries for the generation.

We run simulations for the SS_SD case with $K = 10$ and the number of per-packet tokens varying between 5 and 90, and ∞. Figure 8 plots ((a) without buffer constraint, (b) with buffer constraint of $B = 2$), the number of transmissions versus delay tradeoff achieved under different per-packet token limits. The results show that even with similar transmission numbers, the RLC scheme is still able to outperform non-coded schemes. This is because the random mixing performed by the RLC scheme allows faster and more even propagation of independent information through the network. The results for limited relay buffer case further establish the usefulness of the RLC scheme in decreasing block delivery delay.

4 Multiple Generation Case

In the previous section, we examined the behavior of the RLC scheme in a single generation setting. We found that it provides faster delivery of a block of packets under bandwidth constraint, and that the delay performance degrades very slowly as nodal buffer becomes more and more constrained, at the cost of more transmissions. Although limiting token numbers leads to a larger average delay under both RLC and non-coded schemes, the RLC scheme achieves better transmission power versus delay trade-off than non-coded schemes.

The natural next question to ask is whether the benefit of RLC continues when one moves from a single generation case to the more realistic case where there are multiple continuous flows in the network. We address this question in this section by considering a scenario where there are multiple asynchronous continuous unicast flows in the network. In what follows, we first introduce the traffic process and scheduling schemes. We then present the results for the following two scenarios: when only bandwidth is constrained, and when both bandwidth and buffers are constrained. Finally, we discuss the feasible throughput of network under the non-coded schemes and the RLC scheme.

4.1 Traffic Process and Scheduling Schemes

We assume there are N flows in the network, with each node being the source of one flow and the destination of another flow. Each source independently generates a block of $K = 10$ packets according to a Poisson process with rate λ. Thus the total packet arrival rate to the network is $NK\lambda$. We only consider applying the RLC scheme to packets belonging to the same block, i.e. each block forms a generation, as this case has been shown to result in the largest benefit under the single generation setting.

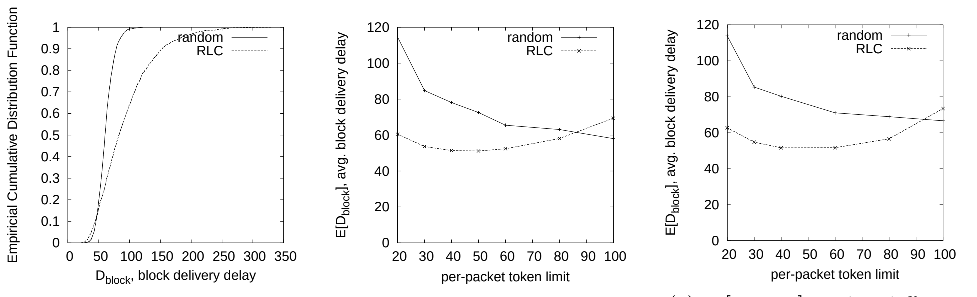
Our focus is on understanding the benefit of using RLC, not on designing an optimal scheme. Hence we adopt simple randomized scheduling. For non-coded schemes, when a node meets another node, it randomly selects a packet from the set of packets that it carries and the other node does not have, and forwards it. For the RLC scheme, the node first randomly chooses a generation from the set of generations that it carries which contain useful information for the other node, and then generates a random linear combination for this chosen generation to forward. For both cases, priorities are given to the packets/generations destined to the other node; furthermore, among such packets/generations, those originated from the sender itself are served first.

4.2 Coding Benefit under Bandwidth Constraint

We have seen that for one single generation, under bandwidth constraints, the RLC scheme achieves a smaller average delay than the non-coded schemes, because the RLC scheme can take advantage of more contact opportunities. We now examine the multiple generation case.

We perform simulations under varying block arrival rate with bandwidth constraint $b = 1$. We observe that the RLC scheme only shows a benefit when the traffic rate is low; and performs worse than the non-coded scheme when the traffic rate is high, as shown in Figure 9(a), which plots the empirical cumulative distribution function (CDF) of D_{block} under $\lambda = 0.00045$.

The reasons that the RLC scheme experiences worse performance than the non-coded scheme under relatively high traffic rates are two-fold. First, when the arrival rate λ is high, there is a large number of different packets in the network under the non-coded schemes; and it is more likely that two nodes have some useful information to exchange when they meet. As a result, the relative benefit of the RLC scheme through its increased randomization is smaller.



(a) Empirical CDF of D_{block} under $\lambda = 0.00045$ (b) $E[D_{block}]$ under different token limit (c) $E[D_{block}]$ under different token limit, $B = 5$

Figure 9: Block delivery delay under multiple generation case

Secondly, as we have shown in Figure 3(a), the RLC scheme generates more transmissions for each generation than the non-coded schemes; this means that when the block arrival rate is high and there are many simultaneous generations in the network, the contention for bandwidth is severer under the RLC scheme. An optimal scheduler should favor generations that have fewer combinations spread throughout the network, but the currently implemented random scheduling scheme does not consider this optimization.

The trade-off between the average number of transmissions and average block delivery delay shown in Figure 8 suggests a way to deal with this resource contention problem. Figure 8 shows that the RLC scheme can achieve similar block delivery delays as the non-coded schemes with a significantly smaller number of transmissions (left part of the curve), so we expect a significant benefit by appropriately limiting the number of copies made of a generation. Figure 9(b) confirms that this is the case. Figure 9(b) plots the $E[D_{block}]$ achieved for the RLC and random schemes for a block arrival rate $\lambda = 0.00045$, when the per-packet token limit is varied between 20 and 100. In particular there is an optimal token limit value for the RLC scheme between 40 and 50 tokens. If the token limit is too large, the system suffers severe contention that degrades performance; if too small, some useful meetings cannot be exploited because all the tokens have been consumed. For the non-coded scheme under this arrival rate, contention is not significant and the reduction in the token limit incurs larger delays. We do observe that under a higher block arrival rate, the non-coded scheme also benefits from limiting the number of copies made for a packet.

How to set the per packet token limit based on bandwidth constraint and block arrvial rate is an open problem. We can estimate an upper bound of the number of transmissions that can be made for each packet as the ratio between the total bandwidth available in the networks, $N(N-1)\beta$, and the total arrival rate, $NK\lambda$. For the specific setting considered here, this value is equal to 100.

4.3 Coding Benefit under Bandwidth and Buffer Constraints

We have seen in Section 3.2 that for a single generation case, the RLC scheme is especially useful when buffer is constrained. We now consider whether this is still the case when there are multiple generations in the network.

As for the single generation case, we assume that each node has limited buffer space for storing relay packets, but unlimited buffer space for storing its own source packets or packets destined to it. Since the source node always stores a packet until it is known to be delivered, there is no packet loss. Under the RLC scheme, when a node receives a combination and its buffer is full, it first selects one generation from its buffer to compress. This is done by randomly choosing one generation from the set of generations in its buffer that have the highest rank. If the newly received combination is for the chosen generation, the combination is combined with an existing combination within that generation; otherwise, the node compresses the matrix of the chosen generation by one [3] to make room for the new combination, and insert the new combination into the generation it belongs to. For the non-coded schemes, the drophead scheme is used.

When both bandwidth and buffer are constrained, limiting the number of transmissions made for each generation becomes even more important for the RLC scheme. As Figure 6(b) in Section 3.2 has shown, under a single generation case, the RLC scheme generates much more transmissions than the non-coded scheme. Therefore, when there are multiple generations in the network, resource contention is even greater than when buffer space is not constrained. We expect that a token scheme will allow bandwidth and buffer space to be allocated more evenly among different generations. We simulate the case of block arrival rate of $\lambda = 0.00045$, and every node only store $B = 5$ relay packets (combinations) under various token limits. As Figure 9(c) shows, the RLC scheme achieves a lower average block delivery delay than the non coding scheme, reducing the average block delivery delay by about 22.5%.

4.4 Feasible Throughput

In the previous two sections, we compared the average block delivery delay achieved by the RLC scheme and non-coded schemes under certain block arrival rates. An intersting question is whether network coding, can increase the throughput, i.e., the maximum per-flow block arrival rate that can be supported by the network.

When nodal buffer is not constrained, a DTN can be viewed as a traditional static network, where the link bandwidth represents the long term bandwidth available between the nodes (i.e., taking into account the meeting frequency). For the communication links in wireless networks, data transmission links along both directions share the same spectrum, therefore it's more natural to view the network as undirected network [20]. As conjectured by Li and Li [19], the benefit of network coding for multiple unicast sessions in direcional networks is likely to be non-existent.

The question of whether network coding scheme can increase throughput when nodal buffer is constrained remains to be answered.

5 Related Work

In this section, we first review briefly previous works that studied the benefit of network coding for wireless networks. Next, we compare in more details

[3] This is done by randomly combining two encoded packets into one.

our work with previous work studying network coding benefit for broadcast applications in DTNs and wireless ad hoc networks. Then we review previous works that performed analytic studies of the RLC scheme for similar settings. Finally, we compare network coding approach with the source-coding approach.

Several previous works have investigated the benefit of network coding for wireless network. For multicast applications, Lun *et al.* [26] and Wu *et al.* [33] studied the problem of minimum-energy multicast, and showed that allowing network coding greatly simplifies the problem (from an NP-complete problem to a linear optimization problem solvable in polynomial time). For broadcast applications, Widmer *et al.* [31, 32] proposed RLC based scheme for energy efficient broadcast in mobile and static networks. For unicast applications, Wu *et al.* [34] and Katti *et al.* [16] studied the benefit of network coding in taking advantage of the shared nature of the wireless medium. Such benefit is applicable to relatively dense network, but not applicable to the sparse mobile network we are considering.

We now compare in more detail our work with those by Widmer *et al.* [31, 32], in which a RLC-based scheme was proposed for broadcast applications in DTNs and wireless ad hoc network, and was shown to achieve higher packet delivery rates than non-coded schemes under the same forwarding overhead. Our work differs from these two works as we consider unicast applications rather than broadcast applications. Even though we consider epidemic style routing where a flooding protocol is used for unicast delivery, it's different from broadcast delivery in that we consider token scheme for limiting the total number of copies made and a recovery scheme for deleting obsolete copies once the first copy is delivered. Moreover, we are interested in the overhead in terms of the number of copies made for each packet. Although [31] considered generation management (i.e., how to decide which packets form a generation) and information aging (i.e., how to delete or compress information), they only reported simulation results for the single generation case. We have considered both single generation and multiple generation cases, and demonstrate that the coding based scheme is especially robust under relay buffer constraint.

Our findings that that under normal signaling, the relative benefit of RLC is much more significant than the intelligent beacon signaling. This result is also in line with findings in [7] showing with intelligent signaling, the benefit of RLC scheme over non-coded scheme to be less significant.

Our main focus in this work is to investigate the benefit of applying network coding to unicast applications in DTNs, therefore we rely on simulations so that we can quantify the benefits accurately. Recent work by Lin *et al.* [23] proposed an ODE model for analyzing delivery delay under an RLC-based scheme and replication (epidemic routing) scheme, for the case of a single block of K packets propagating in resource constrained network. The model is proposed based on the assumptions that *(i)* two nodes with ranks between 1 and $K-1$, i.e., carry some, but not all information about a generation, always have useful information for each other, *(ii)* for all such nodes, equal fractions of them are of rank $1, ..., B-1$, *(ii)* under replication based schemes, the K packets are equally likely to reside each nodes. We comment that the results therein confirm our findings in this paper, for example, the benefit of a RLC scheme under buffer constraints; and we have considered more complicated scenarios than them. A priority scheme is also proposed in the paper, which strictly transmits packets of different priorities in sequence. We observe that such a scheme is not optimal in making use of contact opportunity, and therefore can be improved by priority transmission scheduing at each nodes.

As pointed out in Section 3.1, the benefit of RLC observed in our setting is similar in spirit to that of rumor mongering as studied [7]. They studied the problem of simultaneously disseminating multiple message in a large network, under the "random phone call" communication model where in each time step, each node communicates with another node randomly uniformly chosen from all the nodes. Through rigorous stochastic analysis, asympotic bounds for the delay under coding and non-coded schemes were derived. As both the communication model and the schemes considered (no signaling) therein differ from ours, applying similar analysis to our setting is non-trival.

Two previous works have investigated the application of erasure coding to DTNs, where the source node uses an erasure coding algorithm such as Reed-Solomon codes [27] and Low-Density Parity-Check (LDPC) based coding (e.g., Gallager codes, Tornado codes [25], or LT codes [24]) to encode a message into a large number of code blocks, such that if a fraction of $1/r$ or more of the code blocks is received, the message can be decoded. For DTNs where there are prior knowledge about paths and their loss behavior, Jain *et al.* [15] addressed the problem of allocating the source-erasure-coded blocks to the multiple paths between source and destination each with different loss behavior, in order to maximize the message delivery probability. Wang *et al.* [30] considered DTNs with unpredictable node mobility, and proposed to source-erasure-code message with a fixed overhead, and then send the large number of coded blocks over a large number of relays that then try to deliver them to the destination. Such erasure coding scheme allows for the usage of a large number of relays, in order to decrease the variance of the delivery delay, while maintaining a small fixed redundancy. Chen *et al.* [4] later proposed a hybrid scheme that combines the erasure-coding based scheme [30] with a scheme that aggregressively forwards coded blocks, to achieve both good worst-case performance and small delay performance. Compared to the erasure coding scheme, RLC based schemes have different benefits and characteristics. An intersting open problem to consider is how to combine these two type of codings to attain the different benefits simultaneously.

6 Summary and Future Work

We study the benefits of applying RLC to unicast applications in mobile DTNs in this paper. When there is a single generation in the network, we found that RLC achieves minimum block delay with high probability for a block of data destined to the same destination. Larger gains are achieved by the RLC scheme when buffer space is also constrained. Although the RLC scheme generate more transmissions, by using a token limit scheme, it can achieve better transmission power/delay tradeoff than non-coded schemes. When there are multiple generations in the network, under appropriately chosen token limit, the RLC scheme achieve a slight gain over non-coded schemes under only bandwidth constraint, and significant gains when nodal buffer is also constrained. Essentially, for epidemic style routing (i.e., replication based scheme) to work effectively in resource constrained DTNs, the most challenging problem is how to schedule packet transmissions and manage node buffers. RLC based schemes, where each node randomly combine multiple packets together to transmit to downstream node, and randomly evict a combination on buffer full, has higher degree of randomness compared to a randomized scheme. As a result, under RLC schemes, the probability that a node forwards/keeps a piece of information useful for the eventual delivery is greater than the case where random selection is done on per-packet base.

In the future, we plan to study several practical issues in applying RLC. First of all, we will analyze the computational complexity, and storage and transmission (signaling and data transmission) overhead of RLC scheme. As the finite field size, q, affects the probability of achieving minimum delay, the complexity of encoding/decoding, and the storage/transmission overhead, choosing q is an important practical issue. Another practical issue concerns generation management. We have assumed that packets arrives to the source node in batches, which could arise from applications that generate large messages that are then fragmented to smaller packets to take advantage of more transmission opportunities. For applications that generate small messages, it is not reasonable to fragmented the message to even smaller packets, because the relative per (coded) packet overhead will be too large. For such scenarios, RLC can be applied to a group of packets whose generation times are close to each other. We expect the benefit of RLC schemes will be smaller.

We are also interested in performing analytic studies of the performance of RLC-based schemes and non-coded schemes to obtain closed-form (asymptotic) results. We expect an analysis for an ODE model such as done in [3] might be promising.

A Minimum delivery time of k messages

We assume that the input is a time-ordered list L of all the data transfer opportunities among the nodes. Data transfer opportunities among different pairs of nodes occurring at the same time instant can be arbitrarily ordered.

Starting from this list we can follow the algorithm in [13] in order to build the time-independent *event-driven graph*, where each transfer opportunity is represented as a node. We denote by V the time-ordered set of such nodes (each node has a time label corresponding to the time the contact occurs). Two nodes are connected if either they are the sending and receiving event of the same contact or if both events occur on the same DTN node and no event in that node occurs between them. We denote respectively *inter-edges* and *intra-edges* such links and E the time-ordered set of links. Links with the same time label can be arbitrarily ordered. Inter-edges links have a capacity value corresponding to the maximum number of messages that can be transfered in that direction. Intra-edges links have a capacity corresponding to the buffer size of the corresponding mobile, we consider it to be infinity.

We consider first the case where in each meeting only one message can be transfered by each node to the other and there is a single source (s) and a single destination (t). We will discuss later how to extend the algorithm.

Being that the event-driven graph is a static graph, we can apply Ford-Fulkerson algorithm for the maximum-flow problem (or to find the maximum set of edge disjoint paths) [18]. The basic step of the Ford-Fulkerson algorithm consists in considering a new path in the residual graph, augmenting the flow and updating the residual graph. The algorithms terminates when there are no more s-t paths in the residual graph. When this condition is met the flow found is guaranteed to be the max flow in the graph. We observe that the choice of a new path can be arbitrarily as regards the final result, it only affects in general the convergence speed of the algorithm.

Procedures residual-graph(V,$E1$,f) and augment(G_f, f, P) implement exactly two of the basic steps of Ford Fulkerson algorithm. Procedure augment(G_f, f, P) takes into account that in our case every new path increases by one the value of the flow.

The peculiarity of our algorithm is that it works on progressively larger subgraphs (V,$E1$) -starting from (V,$\emptyset$)- and enlarging it progressively by adding orderly edges from E until we do not find a new s-t path. Every time the procedure enlarge-until-path is executed we find one new path (if any) and a new flow evaluated by the procedure augment whose value is increased by one. If the graph (E,V) has k disjoint paths, then the algorithm terminates after k iterations and provides a flow of value k or equivalently k edge-disjoint paths. Observe that $E^{(k)}$ at the end of the algorithm is a left subsequence of the sequence E and it contains the events that allow to build a flow of value k. In particular the last event in $E^{(k)}$ corresponds to the time -say it $t^{(k)}$- by which there is for sure a flow of value k in the DTN graph. Otherwise, if the graph (E, V) has not k disjoint paths, the algorithm would end with $E^{(k)} = E$ and f a maximum flow in the graph of value less than k.

Let us consider that the graph has k disjoint paths. We want to prove that it is not possible to get a flow of value k *earlier*, i.e. that for each E' left subsequence of $E^{(k)}$ (different from $E^{(k)}$) it is not possible that there is a flow of value k in (V, E'). Let us assume that this is not the case and such E' exists. Imagine to apply our algorithm to the graph (V, E'). The algorithm would terminate when $E_1 = E'$ without having found a flow of value k, because otherwise it would have not processed further to $E^{(k)}$ when applied to (V, E). Now the algorithm is a particular implementation of Ford-Fulkerson on the graph (V, E'), then the value of the maximum flow is lower than k and a contradiction follows.

Finally let us evaluate the computational complexity of our algorithm. The construction of the event-driven graph takes $O(|L|)$ time [13]. For the rest the cost of our algorithm is due to find k augmenting path hence the total time is $O(k|L|)$ or $O(k|E|)$. We observe that for our purpose we may not need to operate on the complete event-driven graph and we can progressively apply the transformation and add nodes to E and links to V as it is required, hence the actual time required from the algorithm is $O(k|E^{(k)}|)$ that can be much smaller.

References

[1] S. Acedanski, S. Deb, M. Medard, and R. Koetter. How Good is Random Linear Coding based Distributed Networked Storage. In *IEEE Workshop on Network Coding, Theory, and Applications (NETCOD)*, 2005.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. *IEEE Trans. on Information Theory*, 46:1204–1216.

Table 3: Algorithm to evaluate the residual graph for a graph (V,$E1$) with a flow f

```
residual-graph(V,E1,f)
$E_f \leftarrow \emptyset$
**for all** $e = (u, v) \in E1$ **do**
  **if** $f(e) < c_e$ **then**
    $e' \leftarrow e$
    $e'$ is a forward edge
    $c'_e \leftarrow c_e - f(e)$
    $E_f \leftarrow E_f \bigcup e'$
  **else**
    $e' \leftarrow (v, u)$
    $e'$ is a backward edge
    $c'_e \leftarrow f(e)$
    $E_f \leftarrow E_f \bigcup e'$
  **end if**
**end for**
**return** $(V, E_f)$
```

Table 4: Algorithm to augment a path P in a residual graph G_f

```
augment($G_f, f, P$)
**for all** edge $(u, v)$ in $P$ **do**
  **if** $e \leftarrow (u, v)$ is a forward edge in the residual graph $G_f$ **then**
    $f(e) \leftarrow f(e) + 1$
  **else**
    $e \leftarrow (v, u)$
    $f(e) \leftarrow f(e) - 1$
  **end if**
**end for**
**return** $f$
```

Table 5: Algorithm to extend the graph until a new path is found.

```
enlarge-until-path($G_f, E_1, E_2, s, t$)
$S \leftarrow \{s\}, E'_1 \leftarrow E_1, E'_2 \leftarrow E_2$
**for all** links $(u, v) \in G_f$ **do**
  **if** $u \in S$ **then**
    $S \leftarrow S \bigcup v$
  **end if**
**end for**
**while** $(t \notin S)$ or $(E'_2 = \emptyset)$ **do** {(there is no $s - t$ path in the residual graph $G_f$)}
  take the first edge $e = (u, v)$ from $E_2$
  $E'_2 \leftarrow E'_2 - e$
  $E'_1 \leftarrow E'_1 \bigcup e$
  **if** $u \in S$ **then**
    $S \leftarrow S \bigcup v$
  **end if**
**end while**
Let $P$ be the path $s - t$ found
**return** $(P, E'_1, E'_2)$
```

Table 6: Find the first k disjoint paths in a DTN graph

```
Input: V, E, k
$G = \emptyset$
**for all** $e \in E$ **do**
  $f(e) = 0$
**end for**
$F = 0$
$E_1 = \emptyset$
$E_2 = E$
**while** $(E_2 \neq \emptyset)$ and $(F < k)$ **do**
  $G_f$ =residual-graph($V, E_1, f$)
  $(P, E'_1, E'_2)$ =enlarge-until-path($G_f, E_1, E_2, s, t$)
  f'=augment($G_f, f, P$)
  $f \leftarrow f', E_1 \leftarrow E'_1, E_2 \leftarrow E'_2$
  $F \leftarrow F + 1$
**end while**
$E^(k) = E_1$
**return** $(f, E^(k))$
```

[3] A. Broder and M. Mitzenmacher. Using Multiple Hash Functions to Improve IP Lookups. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2001.

[4] L-J Chen, C-H Yu, T Sun, Y-C Chen, and H-H Chu. A Hybrid Routing Approach for Opportunistic Networks. In *ACM SIGCOMM Workshop on Challenged Networks (CHANTS)*, 2006.

[5] P. A. Chou, Y. Wu, and K. Jain. Practical network coding. *Allerton Conference on Communication, Control, and Computing*, 2003.

[6] S. Deb, C. Choute, M. Medard, and R. Koetter. Data Harvesting: A Random Coding Approach to Rapid Dissemination and Efficient Storage of Data. Technical report, M.I.T. LIDS Technical Report, 2004.

[7] S. Deb, M. Medard, and C. Choute. Algebraic Gossip: A Network Coding Approach to Optimal Multiple Rumor Mongering. *IEEE/ACM Transactions on Networking, special issue on networking and information theory*, pages 2486–2507, 2006.

[8] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1,2. J. Wiley and Sons, New York, 1964.

[9] C. Fragouli, J.-Y. Le Boudec, and J. Widmer. Network Coding: An Instant Primer. *ACM SIGCOMM Computer Communication Review*, 36, 2006.

[10] C. Gkantsidis and P. Rodriguez. Network Coding for large scale Content Distribution. *IEEE International Conference on Computer Communications (INFOCOM)*, 2005.

[11] R. Groenevelt, P. Nain, and G. Koole. The Message Delay in Mobile Ad Hoc Networks. In *Performance*, October 2005.

[12] Z. J. Haas and T. Small. A New Networking Model for Biological Applications of Ad Hoc Sensor Networks. *IEEE/ACM Transactions on Networking*, February 2006.

[13] D. Hay and P. Giaccone. Optimal routing and scheduling for deterministic delay tolerant networks. In *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, pages 27–34, 2009.

[14] T. Ho, R. Koetter, M. Medard, D.R. Karger, and M. Effros. The Benefits of Coding Over Routing in a Randomized Setting. In *IEEE International Symposium on Information Theory (ISIT)*, 2003.

[15] S. Jain, M. Demmer, R. Patra, and K. Fall. Using Redundancy to Cope with Failures in a Delay Tolerant Network. In *ACM SIGCOMM (Conference on Applications, Technologies, and Protocols for Computer Communication)*, 2005.

[16] S. Katti, D. Katabi, W. Hu, and R. Hariharan. The Importance of Being Opportunistic: Practical Network Coding For Wireless Environments. In *Allerton Conference on Communication, Control, and Computing*, Sep 2005.

[17] D. Kempe, J. Kleinberg, and A. Kumar. Connectivity and Inference Problems for Temporal Networks. In *Journal of Computer and System Sciences, Volume 64,Special issue on STOC 2000*, 2002.

[18] Jon Kleinberg and Eva Tardos. *Algorithm Design*. Addison Wesley, united states ed edition, March 2005.

[19] Z. Li and B. Li. Network Coding: The Case of Multiple Unicast Session. In *Allerton Conference on Communication, Control, and Computing*, 2004.

[20] Z. Li, B. Li, D. Jiang, and L-C Lau. On Achieving Optimal Throughput with Network Coding. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2005.

[21] R Lidl and H. Niederreiter. *Finite Fields, 2nd edition*. Cambridge, England: Cambridge University Press, 1997.

[22] Y. Lin, B. Li, and B. Liang. Efficient network coded data transmissions in disruption tolerant networks. In *Proc. of INFOCOM*, 2008.

[23] Y. Lin, B. Liang, and B. Li. Performance Modeling of Network Coding in Epidemic Routing. In *Proc. MobiOpp*, 2007.

[24] M. Luby. LT codes. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 271–282, 2002.

[25] M. Luby, M. Mitzenmacher, A. Shokrollabi, and D. Spielman. Efficient Erasure Correcting Codes. *IEEE Transactions on Information Theory*, February 2001.

[26] D. S. Lun, M. Medard, T. Ho, and R. Koetter. Network Coding with a Cost Criterion. In *International Symposium on Information Theory and its Applications (ISITA)*, 2004.

[27] I.S. Reed and G. Solomon. Polynomial Codes over Certain Finite Fields. *Journal of the Society for Industrial and Applied Mathematics*, 8:300–304, 1960.

[28] T. Small and Z. J. Haas. Resource and performance tradeoffs in delay-tolerant wireless networks. In *SIGCOMM Workshop on Delay Tolerant Networking (WDTN)*, 2005.

[29] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *SIGCOMM Workshop on Delay Tolerant Networking (WDTN)*, 2005.

[30] Y. Wang, S. Jain, M. Martonosi, and K. Fall. Erasure-coding based routing for opportunistic networks. *SIGCOMM Workshop on Delay Tolerant Networking (WDTN)*, 2005.

[31] J. Widmer and J.-Y. Le Boudec. Network Coding for Efficient Communication in Extreme Networks. *SIGCOMM Workshop on Delay Tolerant Networking (WDTN)*, 2005.

[32] J. Widmer, C. Fragouli, and J.-Y. Le Boudec. Energy-efficient broadcasting in wireless ad-hoc networks. In *NETCOD (IEEE Workshop on Network Coding, Theory, and Applications)*, 2005.

[33] Y. Wu, P. A. Chou, and S.-Y. Kung. Minimum-Energy Multicast in Mobile Ad hoc Networks using Network Coding. In *IEEE Information Theory Workshop*, 2004.

[34] S.-Y. Kung Y. Wu, P. A. Chou. Information Exchange in Wireless Networks with Network Coding and Physical-Layer Broadcast. Technical report, Microsoft Technical Report, MSR-TR-2004-78, August 2004.

[35] X. Zhang, G. Neglia, J. Kurose, and D. Towsley. Performance Modeling of Epidemic Routing. *Elsevier Computer Networks journal*, 51/10:2859–2891, 2007.

Contents

Centre de recherche INRIA Sophia Antipolis – Méditerranée
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399