Network Coding based Transmission Schemes in DTNs with Group Meetings

Abdurrahman Arikan, Yuexin Mao, Xiaolan Zhang[†], Bing Wang, Shengli Zhou, Song Han
University of Connecticut, [†]Fordham University

***Abstract*—Most existing studies on Delay/Disruption Tolerant Networks (DTNs) consider pair-wise node encountering that assumes nodes only meet in pairs. In many mobile wireless networks, a group of nodes, instead of only a pair of nodes, may meet each other. In this paper, we study how to effectively transmit a set of packets from a source to a destination in such group meeting scenarios. The optimization goal is to minimize the delay for the packets to reach the destination while limiting the energy consumption. We first assume that node encountering is known beforehand, and develop an algorithm to obtain the minimum delay. We then develop two practical network coding based schemes. Both schemes use a token technique to limit the total number of transmissions, and only incur signaling at the beginning of a group meeting. One scheme requires nodes in a group to exchange their encoding matrices with each other, while the other only requires exchanging rank information. Simulation results demonstrate that both schemes achieve delays close to the minimum delay for moderate number of tokens. They present different tradeoffs in the number of transmissions and the signaling overhead.**

I. Introduction

Many mobile wireless networks are Delay/Disruption Tolerant Networks (DTNs) where there is no contemporaneous path from a source node to a destination node. Examples include wireless sensor networks for wildlife tracking [13], [7], underwater sensor networks [17], [19], networks for remote areas or for rural areas in developing countries [1], vehicular networks [4], [11] and Pocket-Switched Networks [10]. Due to lack of contemporaneous path, data packets in DTNs are transmitted in a "store-carry-forward" manner [22]: a node receiving a packet buffers and carries the packet as it moves, passing the packet on to new nodes that it encounters. The packet is eventually delivered to the destination when the destination meets a node carrying the packet.

In addition to lack of contemporaneous path, DTNs often have severe bandwidth limitation and power constraints. To address these challenges, a plethora of routing algorithms have been proposed for DTNs (e.g., [22], [7], [20], [21], [3]). Most studies assume pair-wise node encountering where nodes only meet in pairs. That is, when two nodes meet, no other nodes are in the neighborhood. While this assumption might be true for very sparse networks, in many DTNs, nodes can meet in groups where there can be more than two nodes and sometimes much more than two nodes. For instance, in wildlife tracking, a group of animals might meet together at a water hole; in underwater sensor networks, a group of nodes may be in the neighborhood of each other due to water currents; in Pocket-Switched Networks, a group of people may cluster at the same location, e.g., when attending a conference. In this paper, we study how to effectively transmit a set of packets from a source to a destination in group meeting scenarios. The main problem we address is how to schedule packet transmission among a group of nodes that meet each other and have only a limited transmission bandwidth, in order to minimize the end-to-end delivery delay of packets while limiting the total number of transmissions in the network.

Network coding [2] can facilitate *distributed* and *localized* routing strategies, where nodes make independent decisions relying on knowledge about the local neighborhood [6]. These strategies are particularly attractive for DTNs due to rapidly changing topology, intermittent connectivity and limited bandwidth in the network. Network coding has been used for DTNs with pair-wise node encountering patterns [26], [15], [28]. In this paper, we apply network coding in group meeting scenarios. Our main contributions are as follows.

- We propose an algorithm to calculate the minimum time to deliver a group of packets, given a prior knowledge of all future meetings. This provides a lower bound for us to quantify the effectiveness of heuristic schemes.
- We develop two network coding based heuristic schemes for group meetings: one scheme based on the coefficient matrices of the coded packets buffered at the nodes, the other simply based on the ranks of the coefficient matrices. Both schemes are distributed, localized, and easy to implement. Both schemes use a token based technique to limit the total number of transmissions.
- Simulation results demonstrate that the two heuristic schemes achieve delays close to the minimum latency for moderate number of tokens. The rank-based scheme requires slightly larger number of transmissions than the matrix-based scheme, while incurring much lower signaling and computation overhead. Therefore, the rank-based scheme may be a preferred choice especially for networks with limited bandwidth and computation capabilities.

The remainder of this paper is structured as follows. In Section II, we introduce the network model and performance metrics considered in this paper. Section III presents the algorithm that obtains the minimum time to deliver a group of packets. Section IV presents the two network coding based heuristic schemes. Section V describes performance evaluation. Section VI reviews related work. Finally, Section VII concludes this paper and presents future work.

Notation	Meaning
N	number of nodes in the network
$\mathcal{V}$	the set of nodes
$\mathcal{L}$	DTN meeting trace
K	generation size
b	#. of packets that can be exchanged during a meeting
$\mathcal{B}(u)$	#. of relay packets node u can store
$\mathbb{F}_q$	finite field, $q = p^n$ p is a prime, n is a positive integer.
D	group delivery delay
C	number of tokens

TABLE I. TABLE OF NOTATIONS

II. BACKGROUND

In this section, we present the network model, traffic setting and performance metrics studied in this paper. Table II summarizes the key notations for easy reference.

A. Network Model

We consider a network consisting of a set of N mobile nodes, denoted as $\mathcal{V}$, moving independently in a closed area. Each node is equipped with a wireless radio with a common transmission range so that when two or more nodes come within transmission range of each other (i.e., they *meet*), they can exchange packets. We refer to the list of meetings, sorted in temporal order, within a DTN during a certain time interval as *a DTN meeting trace*, denoted as $\mathcal{L} = l_1, l_2, l_3, ...$, where each meeting, l_i, is a tuple (t_i, G_i, b_i) with t_i denote the time of the meeting, $G_i \subseteq \mathcal{V}$ denote the set of nodes that come into contact with each other during this meeting, and b_i denote the total number of packets that can be transmitted during the meeting. Due to the broadcast nature of wireless medium, we assume the packet transmitted by any node $u \in G_i$ will be received by all other nodes in G_i. During each meeting, transmission scheduling decides the allocation of bandwidth to the nodes in the group G_i, and the transmission order, in order to optimize performance.

As for the buffer constraint, we assume each node can store an unlimited number of packets originated by itself or destined for itself, but can only carry a limited number of packets for other nodes. We represent the buffer constraint as a function, $\mathcal{B} : \mathcal{V} \to \mathbb{N}$ where $\mathcal{B}(u)$ is the number of relay packets that node u can carry.

B. Traffic Setting and Performance Metrics

We focus on unicast applications where each packet (generated by its source node) is destined to a single destination node. We suppose that each message generated by the application is segmented into a group of packets in order to take advantage of the short meetings. We denote the group of packets belonging to a message as $P_i, i = 1, 2, ..., K$, and the delivery delay of packet P_i as D_i for $i = 1, 2, ..., K$. The *group delivery delay*, D, is the time from the generation of the message, i.e., the group of packets, to the delivery of the entire group to the destination, and we have $D = \max_{1 \le i \le K} D_i$. In the following, we refer to group delivery delay simply as delivery delay. Another performance metric is the total number of transmissions in the network before the destination receives the message. We assume that once the destination receives the message, *recovery mechanisms* such as those in [7], [27] are used to remove the obsolete copies of the packets from the network to save resources. The third metric is the signaling overhead, i.e., the total amount of signaling data that a group of nodes use to exchange information so as to determine the transmission scheduling.

III. MINIMUM DELIVERY DELAY

In this section, we present an algorithm to calculate the minimum delivery delay under a meeting trace and buffer constraint.

We use the 4-tuple (s, d, t_0, K) to denote a group of K unicast packets generated by source node s at time t_0, all of which are destined for the same destination d. For (s, d, t_0, K) that can be delivered to the destination under the meeting trace $\mathcal{L}$ and buffer constraints $\mathcal{B}(\cdot)$, there is a minimum delivery time by which all the K packets can be delivered to the destination. This time is in general achievable only by a centralized oracle with knowledge of all future meetings. The minimum delivery time clearly lower bounds the delivery time achievable by any routing scheme, and therefore is an ideal benchmark to compare different routing schemes with.

We first consider the related problem of determining the maximum number of unicast packets (generated at s at t_0 to be delivered to destination node d) that can be delivered under a given meeting trace $\mathcal{L}$ and buffer constraint $\mathcal{B}(\cdot)$. Similar to related works [8], [28], we first build an *event-driven graph* $\mathcal{G}(\mathcal{L}, \mathcal{B}, (s, d, t_0, K))$ as follows, and then solve a maximum flow problem on $\mathcal{G}$. For ease of explanation, let $T = |\mathcal{L}|$, i.e., the number of meetings in the trace, and let $t_1, t_2, ..., t_T$ represent the times when meetings $l_1, l_2, ..., l_T$ occur.

1) For each node $u \in \mathcal{V}$, we introduce $T + 1$ nodes in $\mathcal{G}$, $u_0, u_1, ..., u_T$, to represent the snapshot of node u at $t_0, t_1, t_2, ..., t_T$ respectively.
2) We connect each snapshot of a node u_i to its next snapshot u_{i+1} with an **intra-node edge** (u_i, u_{i+1}), and set its capacity as follows $c(u_i, u_{i+1}) = \mathcal{B}(u)$, denoting that node u can buffer packets until a later time instance[1].
3) For each meeting $l_i = (t_i, G_i, b_i)$ in $\mathcal{L}$, where nodes in set $G_i \subseteq \mathcal{V}$ come into contact with each other at time t_i, and up to b_i packets can be exchanged in a broadcast fashion, we introduce **inter-node edges** to connect nodes in G_i so that every node (at time t_i) is connected to every other node (at time t_{i+1}). For example, if $G_i = \{u, v, w\}$, we add the following edges into $\mathcal{G}$ $(u_i, v_{i+1}), (u_i, w_{i+1})$, $(v_t, u_{i+1}), (v_t, w_{i+1})$, $(w_t, u_{i+1}), (w_t, v_{i+1})$, and assign capacity b_i to each of them.
4) For source node $s \in \mathcal{V}$, we add a super source node s to $\mathcal{G}$, and connect it to s_0 (source node at t_0) with an intra-node edge with capacity K, i.e., $c(s, s_0) = K$.
5) For destination $d \in \mathcal{V}$, we add a super sink node d to $\mathcal{G}$, and connect each node $d_0, d_1, ..., d_T$ to d with an intra-node edge of capacity K, i.e., $c(d_i, d) = K$, for $i = 0, 1, ..., T$.

[1]If nodes have unlimited buffer, we can set the capacity for all intra-node edges to K (the total number of packets to be delivered from s to d).

We use a network of 4 nodes in Fig. 1 to illustrate the construction of graph $\mathcal{G}$. We assume that the nodes are moving in an area divided into 2×2 grids, and all nodes in the same grid can communicate with each other. The lower figure represents the topology of the network at three consecutive time slots. The upper figure shows the constructed event-driven graph, when the source is a and the destination is node d. The intra-node edges are drawn in solid lines. The inter-node edges are drawn in dashed lines.

Let $f(\cdot)$ denote a flow from node s to node d in graph $\mathcal{G}$. The maximum number of packets (generated by node s and destined for node d) that can be delivered under meeting trace $\mathcal{L}$ is the same as the maximum value of flow, $|f| = f(s, s_0)$ (Theorem 4 in [8]). We want to maximize the value of flow $|f|$, subject to constraints described below:

$$\underset{f}{\text{maximize}} \ |f| = f(s, s_0) = \Sigma_u f(u, d)$$

subject to:

(1) $f(u_i, u_{i+1}) \le c(u_i, u_{i+1})$,
for each intra-node edge (u_i, u_{i+1})

(2) $f(u_i, v_{i+1}) \le c(u_i, v_{i+1})$,
for each inter-node edge (u_i, v_{i+1})

(3) $\Sigma_v f(u, v) = \Sigma_v f(v, u)$, for each node $u \ne s, u \ne d$

(4) $\Sigma_{u \in G_i} \Sigma_{v \in G_i, v \ne u} f(u_i, v_{i+1}) \le b_i$,
for each meeting $l_i = (t_i, G_i, b_i)$ in $\mathcal{L}$

Constraints (1) and (2) are the capacity constraints for intra-node edges and inter-node edges respectively. Constraints (3) specify the flow conservation property for all nodes other than the source and destination of the flow, i.e., s and d. Note that a flow of value $|f|$ in $\mathcal{G}$ corresponds to a set of end-to-end paths for delivering $|f|$ packets from s to d. As we are considering the minimum delay for a group of *unicast* packets, we only need to consider those end-to-end paths that deliver those packets first. Therefore, we do not need to take into consideration that when a node in a group transmits, all nodes in the group receive the packet. In other words, the flow conservation property stated in constraints (3) holds in our setting. Lastly, constraints (4) specify that for a group meeting $l_i = (t_i, G_i, b_i)$, the total flows from nodes in the group G_i at time t_i to other nodes in the group at time t_{i+1} are bounded by b_i, the total bandwidth of the meeting[2].

Again, we use the example in Fig. 1 to illustrate the correspondance between the maximum flow in the event-driven graph and a transmission schedule in the DTN. Suppose only one packet can be exchanged during each meeting (i.e., $b_i = 1, \forall i$). The maximum flow from the super source to super sink is 2, achieved by path $(source, a_0, a_1, a_2, d_3, sink)$ and path $(source, a_0, b_1, d_2, sink)$. The corresponding transmission schedule in DTN is: node a transmits the first packet in time 0 (nodes b, c receive the packet), node b then transmits the packet to destination d at time 1, and finally, node a transmits the second packet to destination d at time 2.

The above maximization problem is an integer linear programming problem, and can be solved using standard tools (e.g., CVX [18]). Since the constraints are integral, the solution is also integral.

[2]As $f(\cdot)$ is a non-negative mapping, capacity constraints on inter-node edges, i.e., constaints (2), are redundant as they are implied by constraints (4).

The above formulation allows us to calculate the maximum number of packets that can be delivered under a given meeting trace $\mathcal{L}$, which we denoted as $MaxPacketDelivered(\mathcal{L})$. Now, in order to calculate the minimum delivery time to deliver a group of K packets, we only need to find the shortest prefix of the meeting trace under which K packets can be delivered. A simple way to do this is to start with a short prefix $\mathcal{L}_1$ of the meeting trace, If $MaxPacketDelivered(\mathcal{L}_1)$ is greater than or eqaul to K, we know the the minimum delivery time lies within the range $(0, t(\mathcal{L}_1)]$ (here we use $t(\mathcal{L})$ to denote the time of the last meeting in the meeting trace $\mathcal{L}$). If $MaxPacketDelivered(\mathcal{L}_1)$ is smaller than K, we consider a longer prefix $\mathcal{L}_2$ of the meeting trace (e.g., by doubling the length of the prefix) to see whether K packets can be delivered, and repeat this procedure until we find a prefix $\mathcal{L}_n$ under which K packets can be delivered. In this case, we bound the minimum delivery time with range $(t(\mathcal{L}_{n-1}), t(\mathcal{L}_n)]$. To find the minimum delivery delay, we then perform a binary search in the above range to find the shortest prefix under which K packets can be delivered.

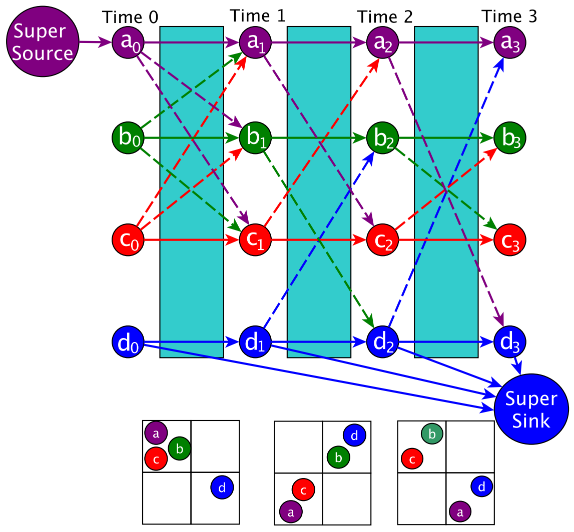


Fig. 1. Illustration of the formulation to obtain the minimum delivery delay.

IV. Network coding based heuristic schemes

In this section, we present two network coding based heuristic schemes for group meeting scenarios. The reason for using network coding is motivated by its significant benefits for DTN routing for pair-wise scenarios [26], [15], [28]. Specifically, our proposed schemes use Random Linear Coding (RLC) [9], a special form of network coding. In the following, we first describe the basic idea of using RLC for DTN routing, and then present our schemes.

A. RLC based Routing Schemes

We assume that all packets have the same payload size equal to S bits. When RLC is used in packet data networks, the payload of each packet can be viewed as a vector of $\eta = \lceil S/\log_2(q) \rceil$ symbols from a finite field [14], $\mathbb{F}_q$ of size q.

A collection of packets that may be linearly coded together by network nodes is called a *generation*. For example, the K packets that make up an application message can constitute a generation. We denote by $\mathbf{m}_i \in \mathbb{F}_q^\eta$, the symbol vector corresponding to packet P_i. A linear combination of the K packets is:

$$\mathbf{x} = \sum_{i=1}^{K} \alpha_i \mathbf{m}_i,\ \alpha_i \in \mathbb{F}_q,$$

where addition and multiplication are over $\mathbb{F}_q$. The vector of coefficients, $\alpha = (\alpha_1, ..., \alpha_K)$ is called the *encoding vector*, and the resulting linear combination, $\mathbf{x}$, is called an *encoded packet*. We say that two or more encoded packets are linearly independent if their encoding vectors are linearly independent. Each original packet, $\mathbf{m}_i, i = 1, 2, ...K$, can be viewed as a special combination with coefficients $\alpha_i = 1$, and $\alpha_j = 0, \forall j \neq i$.

Under RLC schemes, network nodes store and forward encoded packets, together with their encoding vectors. If the set of encoded packets carried by a node contains at most r linearly independent encoded packets $\mathbf{x}_1, ..., \mathbf{x}_r$, we say that the rank of the node is r. We refer to the $r \times K$ matrix (denoted as $\mathbf{A}$) formed by the encoding vectors of $\mathbf{x}_1, ..., \mathbf{x}_r$ as the node's *encoding matrix*. Essentially, the node stores r independent linear equations with the K original packets as the unknown variables, i.e., $\mathbf{AM} = \mathbf{X}$, where $\mathbf{M} = (\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_K)^{\mathrm{T}}$ is a $K \times \eta$ matrix of the K original packets, and $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_r)^{\mathrm{T}}$ is an $r \times \eta$ matrix of the r encoded 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, solving $\mathbf{AM} = \mathbf{X}$ for $\mathbf{M} = \mathbf{A}^{-1}\mathbf{X}$ using standard Gaussian elimination algorithm.

We illustrate data forwarding under RLC schemes using the transmission from node u to node v as an example. Node u generates a random linear combination ($\mathbf{x}_{new}$) of the combinations stored in its buffer $\mathbf{x}_1, ..., \mathbf{x}_r$: $\mathbf{x}_{new} = \sum_{j=1}^{r} \beta_j \mathbf{x}_j$, where the coefficients $\beta_1, ...\beta_r$ are chosen uniformly at random from $\mathbb{F}_q$. Clearly, $\mathbf{x}_{new}$ is also a linear combination of the K original packets. This new combination, along with the coefficients *with respect to the original packets*, is forwarded to node v. If among $\mathbf{x}_1, ..., \mathbf{x}_r$, there is at least one combination that cannot be linearly expressed by the combinations stored in node v, node u has useful (i.e., *innovative*) information for node v, and $\mathbf{x}_{new}$ is useful to node v (i.e., increases the rank of node v) with probability greater than or equal to $1-1/q$ (*Lemma* 2.1 in [5].). When v receives $\mathbf{x}_{new}$, it stores $\mathbf{x}_{new}$ into its buffer if there is still space in its buffer; otherwise, one existing encoded packet in the buffer is replaced by its linear combination with $\mathbf{x}_{new}$.

B. RLC based Routing Schemes for Group Meeting Scenarios

When a group of $n \geq 2$ nodes meet and can only exchange b packets, the key decision is transmission scheduling, i.e., allocating the bandwidth to the nodes in the group, and determining the transmission order. Intuitively, nodes that have innovative packets for other nodes in the group should transmit first and use more bandwidth. In addition, to limit the energy consumption in delivering a generation of packets, the total number of transmissions that is allowed in the network should be limited.

Several previous studies in DTNs proposed schemes to limit the total number of transmissions. The binary spray-and-wait [20], [15] scheme assumes pair-wise meetings, and hence cannot be directly applied to group meeting scenarios. In our heuristics, we adopt the token-based RLC technique in [28] to limit the total number of transmissions in the network. Specifically, at the beginning, the source has C tokens; the rest of the nodes do not have any token. When a group of nodes meet, their tokens are aggregated together, used jointly and then distributed among the nodes at the end of the encountering. We will show that the total number of transmissions in the network is no more than $C + K$ with high probability.

Next, we present two transmission scheduling schemes for group meeting scenarios to be used with RLC based scheme as described in Section IV-A. In the matrix based scheme, nodes determine transmission scheduling based on their encoding matrices; in the rank based scheme, nodes determine transmission scheduling based on the ranks of their encoding matrices.

For ease of exposition, we assume that a group of n nodes, denoted as $v_1, \dots, v_n$ meet and can transmit up to b packets during the group meeting. For each node v_i, we denote its encoding matrix as $\mathbf{A}_i$, its rank as r_i, and its number of tokens as c_i. We define the number of innovative packets that v_i has *relative to* node v_j, denoted as r_{ij}, to be the rank of the matrix formed by $\mathbf{A}_i$ and $\mathbf{A}_j$ (i.e., the rank of v_j when it gets all the coefficient combinations from v_i) subtracted by the rank of v_j.

1) Matrix based Scheme: In the matrix based scheme, when the group of nodes meet, each node in the group, v_i, broadcasts its encoding matrix, $\mathbf{A}_i$, and the number of tokens, c_i, to the rest of the nodes in the group. After receiving the encoding matrices from other nodes, each node v_i calculates *i).* the rank of the encoding matrix of node v_j, denoted as r_j, $j = 1, \dots, n$, *ii).* the number of innovative packets it has relative to node v_j, i.e., r_{ij}, $j = 1, \dots, n$, and *iii).* the total number of tokens for the group, $c = \sum_{i=1}^{n} c_i$.

When the bandwidth is b and the total number of tokens in the group is c, there can be at most $\min(b, c)$ rounds of packet transmissions (in a round, one node generates and transmits a new packet which is a linear combination of the packets in its buffer as described in Section IV-A). Let b_g and c_g denote respectively the remaining bandwidth and the remaining number of group tokens. Initially $b_g = b$ and $c_g = c$. Each node maintains a copy of r_{ij}, r_i, $i, j = 1, \dots, n$, b_g and c_g.

We first consider the scenario where the destination node is not in the group, i.e., $v_i \neq d$, $i = 1, \dots, n$. In each round, node v_i can transmit only when all the following four conditions hold: $b_g > 0$, $c_g > 0$, v_i has at least one innovative packet for other nodes, i.e., $\sum_j r_{ij} > 0$, and v_i has the largest rank among all the nodes that have at least one innovative packet for other nodes.

After node v_i's transmission, each node reduces its copy of b_g (the remaining bandwidth) and c_g (remaining token) by one. In addition, each receiving node v_j updates its copy of r_{ij} to $\max(0, r_{ij}-1)$, $j = 1, \dots, n, j \neq i$. In other words, we assume that the number of innovative packets that v_i has relative to other nodes is reduced by one. We make this assumption since it happens with high probability (more specifically, the probability is greater than or equal to $1 - 1/q$ (*Lemma* 2.1 in [5]). There is no verification on whether this is indeed the case, because the verification requires additional signaling overhead and hence additional energy consumption. Last, each node v_j updates its copy of r_j to $\min(K, r_j + 1)$, for $j = 1, ..., n$, and $j \neq i$, again because this happens with high probability.

The group of nodes repeat the above transmission until $b_g = 0$, or $c_g = 0$, or none of the nodes in the group has any innovative packet to send. When transmission ends, the remaining group tokens are distributed among the group of nodes proportional to the current ranks of the nodes.

For the case when the destination node is among the group of nodes that meet, we remove the restriction of the group tokens, and allow each non-destination node v_i to transmit combinations as long as there is bandwidth and it has innovative packets relative to destination d, i.e., $r_{id} > 0$. Since the number of independent packets is K, the number of transmissions in this case is bounded by K with high probability.

In summary, for the matrix based scheme, the total number of transmissions in the network is no more than $C + K$ with high probability since the total number of transmissions made to non-destination nodes is bounded by C, and the number of transmissions to destination is bounded by K with high probability. The actual number of transmissions is smaller when a recovery scheme is used.

2) Rank based Scheme: The rank based scheme differs from the matrix based scheme in that a node broadcasts the rank of its encoding matrix, instead of the encoding matrix itself, at the begining of each meeting. More specifically, when the group of nodes meets, each node v_i broadcasts its rank, r_i, and the number of tokens, c_i, to all other nodes in the group. Each node v_i stores and maintains a copy of r_j, $j = 1, \ldots, n$. In each transmission round, node v_i can transmit only when all the following four conditions hold: $b_g > 0$, $c_g > 0$, v_i has the highest rank, i.e., $r_i > 0$ and $r_i \geq r_j$ (when multiple nodes have the highest rank, we break the tie randomly), and there still exists at least a node with rank below K.

After v_i's transmission, each node reduces its copies of b_g and c_g by one, and updates $r_j = \min(K, r_j + 1)$, for $j = 1, \ldots, n, j \neq i$. For similar reason as described for the matrix based scheme, the total number of transmissions in the network for the rank based scheme is no more than $C + K$ with high probability.

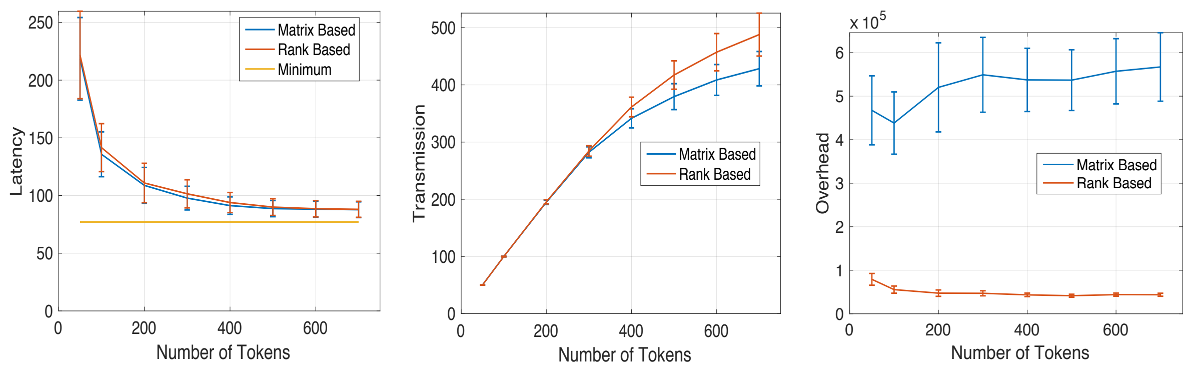
The rank based scheme incurs less signaling overhead than the matrix based scheme. Specifically, for the matrix based scheme, the signaling overhead for a group of n nodes is $\sum_{i=1}^{n} \text{size}(\mathbf{A}_i) \log_2 q + n \log_2 C$ bits. The first term is the signaling overhead for transmitting the encoding matrices, where $\text{size}(\mathbf{A}_i)$ represents the number of elements in $\mathbf{A}_i$ and each element has $\log_2 q$ bits. The second term is the signaling overhead for transmitting the number of tokens (since C is the maximum number of tokens at a node). For the rank based scheme, the signaling overhead is $n\lceil \log_2 K \rceil + n \log_2 C$ since the signaling overhead for transmitting the ranks is $n\lceil \log_2 K \rceil$ (a rank is no more than K). In our simulation setting, since $K = 10$, $q = 2^8$, and $\sum_{i=1}^{n} \text{size}(\mathbf{A}_i)$ can be significantly larger than n, the signaling overhead for each meeting under the rank based scheme can be much lower than that under the matrix based scheme. On the other hand, in the matrix based scheme, each node has an accurate estimate of the number of innovative packets that it has relative to other nodes, which may lead to better decisions, and hence shorter delivery delay and less transmissions. We compare the performance of these two schemes in Section V.

V. Performance Evaluation

We evaluate the performance of the matrix and rank based schemes using simulation. Specifically, we simulate a grid network of 15×15 grids. There are 101 nodes in the network. Among them, 100 nodes are mobile, and one node is static. The mobile nodes are initially uniformly distributed in the network. They move in time slots. In each time slot, a node moves in one of the four directions, left, right, up or down, into the adjacent grid. If following the direction does not lead to a valid adjacent grid (i.e., if a node is in the top left grid, then moving left or up does not lead to a valid adjacent grid), then the node stays in the current grid. One of the mobile nodes is randomly chosen as the source. The source generates a generation of $K = 10$ packets and encodes them using RLC at the beginning of a simulation run. The single static node, located at the center of the network, acts as the destination.

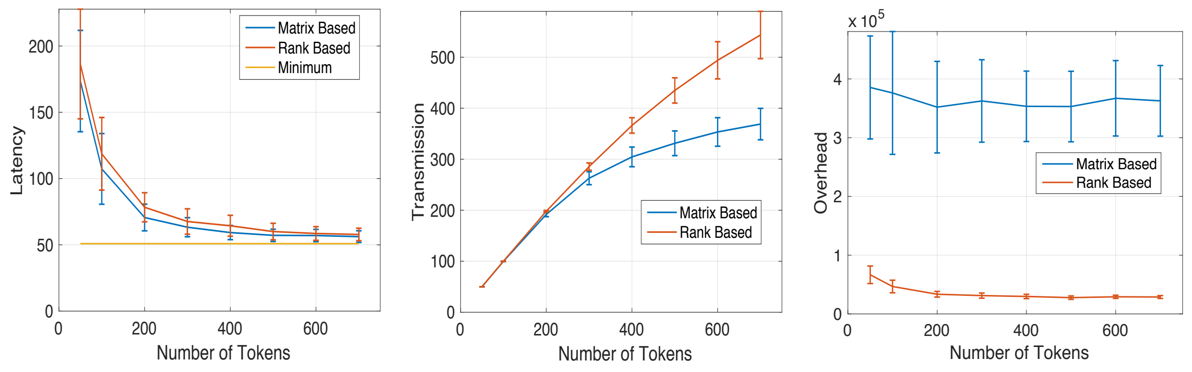
We assume nodes in the same grid can transmit to each other. In addition, due to the broadcast nature of wireless transmission, when one node transmits, the rest of the nodes in the grid can receive the packet. We assume each node has sufficient amount of buffer space (specifically the buffer can hold 200 packets). For a group of nodes in the same grid, the transmission bandwidth b is set to allow 1, 3, or 9 packet transmissions during an encountering. The number of tokens allowed to transmit a generation of packets, C, is set to $50, 100, 200, 300, 400, \ldots, 700$. For each simulation setting, we use the algorithm in Section III to obtain the minimum delivery delay. For the matrix and rank based schemes, we obtain the delivery delay, the total number of transmissions that is needed before the destination recovers the original packets, and the total amount of signaling overhead. For each setting, we generate 30 meeting graphs using random seeds, and obtain the average results and the standard deviation.

We first present the results when the transmission bandwidth, $b = 1$. Fig. 2 plots the performance of the matrix and rank based schemes. Specifically, Fig. 2(a) plots the delivery delay versus the number of tokens; the minimum delivery delay is also plotted in the figure (which is independent of the number of tokens and hence is a horizontal line). We observe that the performance of the matrix and rank based schemes is similar. This is because, when $b = 1$, only a single node can transmit a single packet when a group of nodes meet; the node with the most innovative packets (in the matrix based scheme) is likely to coincide with the node with the highest rank (in the rank based scheme). For both the matrix and rank based schemes, the delivery delay decreases when the number of tokens, C, increases. This is expected since a larger number of tokens allows more transmissions in the network and more opportunities for nodes to exchange information. In addition, there is a diminishing gain in increasing the number of tokens: the decrease in delivery delay is significant at the beginning and then becomes less significant afterwards. For instance, under the matrix based scheme, the delivery delay



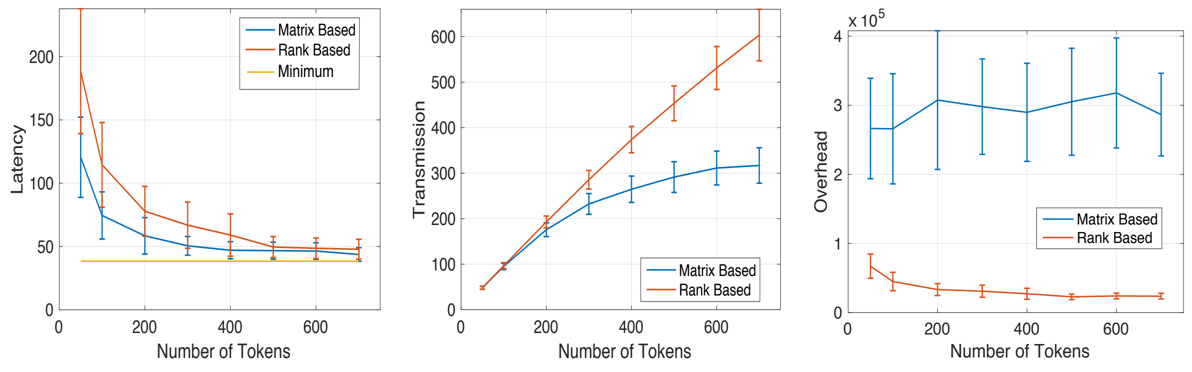
(a) Delivery delay. (b) Number of transmissions. (c) Signaling overhead.

Fig. 2. Performance of the matrix and rank based schemes when $b = 1$.



(a) Delivery delay. (b) Number of transmissions. (c) Signaling overhead.

Fig. 3. Performance of the matrix and rank based schemes when $b = 3$.



(a) Delivery delay. (b) Number of transmissions. (c) Signaling overhead.

Fig. 4. Performance of the matrix and rank based schemes when $b = 9$.

for $C = 400$ is similar to that when $C = 700$, respectively 18.5% and 13.9% larger than the minimum latency.

Fig. 2(b) plots the number of transmissions when $b = 1$. The results for both the matrix and rank based schemes are plotted in the figure. These two schemes achieve similar performance for relatively small C. For relatively large C, the number of transmissions under the matrix based scheme is noticeably lower. This might be because under the matrix based scheme, a node knows accurately whether it has innovative packets for other nodes or not, and may not transmit packets even if there are tokens allowing it to transmit. For the rank based scheme, as long as there are tokens and available bandwidth, a node will transmit unless its rank is K and the ranks of all the other nodes are K. Fig. 2(c) plots the signaling overhead when $b = 1$. As explained in Section IV-B, the matrix based scheme leads to significantly higher overhead.

We next compare the total communication overhead, i.e., the sum of the overhead for transmitting data packets and the signaling overhead, of the two schemes. Assume each data packet is 1500 bytes (i.e., the Maximum Transfer Unit in a typical network). When $C = 400$, from Fig. 2(b), the average numbers of transmissions under the rank and matrix based schemes are 361.6 and 341.3 respectively, and hence the overhead for transmitting data packets under the rank based scheme is $1500 \times 8 \times (361.6 - 341.3) = 2.4 \times 10^5$ bits more than that of the matrix based scheme. On the other hand, from Fig. 2(c), the signal overheads of the rank and matrix based schemes are 4.3×10^4 bits and 5.4×10^5 bits respectively, and hence the signaling overhead of the rank based scheme is $(5.4-0.43)\times 10^5 = 5.0\times 10^5$ bits lower than that of the matrix based scheme. Overall, the total communication overhead of the rank based scheme is $(5.0 - 2.4) \times 10^5 = 2.6 \times 10^5$ bits lower than that of the matrix based scheme. On the other hand, the delivery delay under the rank based scheme is 2.8% larger than that of the matrix based scheme (93.9 versus 91.3, see Fig. 2(a)).

We now present the results when the transmission bandwidth, b, is larger. Fig. 3 plots the results when $b = 3$. We observe similar trends as those when $b = 1$. The delivery delay under these two schemes is still similar for the various values of C, although the difference between the two schemes is more noticeable than that when $b = 1$. For the number of transmissions, the performance under these two schemes is similar for small C, while for large C, the difference is more significant than that when $b = 1$. For the signaling overhead, the rank based scheme is still significantly lower than that of the matrix based scheme. In general, we observe similar tradeoff as that when $b = 1$. For instance, when $C = 300$, the delivery delay under the rank based scheme is 6.8% larger than that of the matrix based scheme (67.6 versus 63.3), while the total communication overhead (i.e., considering both data transmission and signaling; assuming each data packet is 1500 bytes) is 8.0×10^3 bits lower.

Fig. 4 plots the results when $b = 9$. While the overall trends are similar to those when $b = 1$ and $b = 3$, the delivery delay under the matrix based scheme is significantly lower than that of the rank based scheme when C is below 400, indicating that when b is large, using encoding matrices to determine the transmission scheduling achieves more benefits than simply using rank information. In addition, Fig. 4(b) shows that the number of transmissions under the matrix based scheme increases much more slowly than that under the rank based scheme. As a result, the rank based scheme can lead to more total communication overhead than the matrix based scheme. For instance, when $C = 300$, again assuming each data packet is 1500 bytes, the overhead of data transmission under the rank based scheme is 6.3×10^5 bits more than that of the matrix based scheme, outweighing its savings in signaling overhead (which is 2.7×10^5 bits less than that of the matrix based scheme).

Summarizing the above, we observe that both the matrix and rank based schemes achieve delivery delay similar to the minimum delivery delay for moderate number of tokens (when $C = 300$ or 400). These two schemes present different tradeoffs in the delivery latency and communication overhead. In general, when b is small, the rank based scheme seems to be more preferable; while when b is very large, the matrix based scheme seems to be more preferable. Both schemes are easy to implement and only incur signaling at the beginning of the group encountering. For large b, the number of transmissions under the rank based scheme can be reduced by adding feedback, i.e., after a node transmits a packet, the rest of the nodes provide feedback on whether their ranks are indeed increased. The additional feedback, however, adds more complexity to the scheme.

VI. Related Work

Previous works have studied the application of network coding to broadcast and unicast applications in DTNs. For broadcast applications, Widmer *et al.* [24], [25] showed that a RLC routing scheme achieves higher packet delivery rates than the non-coding scheme with the same forwarding overhead. For unicast applications, Zhang *et al.* [26], [28] investigated the benefits of RLC through analysis and simulation, and proposed a token based scheme to limit the number of transmissions. Lin *et al.* proposed and analyzed a different replication control scheme [15], and proposed Ordinary Differential Equation (ODE) models for estimating delivery delay and number of transmissions for RLC schemes and non-coding schemes ([16]). The above studies assumed pair-wise contacts. In this paper, we also study RLC based schemes for unicast applications in DTNs. However, our work differs from the above works in that we consider group meeting scenarios, and focus on the resulting transmission scheduling problem under such scenarios.

Several studies [12], [23] are on the application of erasure coding to DTNs, where the source encodes a message into a large number of blocks, such that as long as a certain fraction or more of the coded blocks are received, the message can be decoded. For DTNs where there is prior knowledge about paths and their loss behavior, Jain *et al.* [12] studied how to allocate the coded blocks to the multiple lossy paths in order to maximize the message delivery probability. To reduce the variance of delivery delay in DTNs with unpredictable mobility, Wang *et al.* [23] proposed to encode each message into a large number of coded blocks which are then transmitted to a large number of relays helping to deliver the coded blocks to the destination. Our study differs from them in that we focus on network coding, and specifically RLC.

VII. Conclusion and Future Work

In this paper, we studied how to effectively transmit a set of packets from a source to a destination in a DTN with group-based meetings. We first assumed that node encountering is known beforehand, and developed a max-flow based algorithm to obtain the minimum latency. After that, we developed two practical network coding based schemes, i.e., the matrix and the rank based schemes. Both schemes use a token based technique to limit the total number of transmissions, and only incur signaling at the beginning of a group meeting. Simulation results demonstrate that both schemes achieve delays close to the minimum latency for moderate number of tokens. They present different tradeoffs in the number of transmissions and the signaling overhead.

As future work, we will pursue the following directions. First, we will explore how to determine the optimal number of tokens under these two network coding based schemes. Secondly, we will explore scenarios where there are data transmissions between multiple source and destination pairs (this study focuses on transmitting data from a single source to a single destination).

Acknowledgements

The work was partially supported by National Science Foundation (NSF) under award 1310406. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF. The first author was supported by Ministry of National Education, Republic of Turkey.

References

[1] First Mile Solutions. http://firstmilesolutions.com/.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. *IEEE Trans. on Information Theory*, 46:1204–1216, July 2000.

[3] A. Balasubramanian, B. N. Levine, and A. Venkataramani. DTN Routing as a Resource Allocation Problem. In *ACM Conference on Applications, Technologies, and Protocols for Computer Communication (SIGCOMM)*, pages 373–384, 2007.

[4] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2006.

[5] 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.

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

[7] Z. J. Haas and T. Small. A New Networking Model for Biological Applications of Ad Hoc Sensor Networks. *IEEE/ACM Transactions on Networking*, 14:27–40, February 2006.

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

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

[10] P. Hui, A. Chaintreau, R. Gass, J. Scott, J. Crowcroft, and C. Diot. Pocket Switched Networking: Challenges, Feasibility, and Implementation Issues. In *IFIP TC6 International Workshop on Autonomic Communication (WAC)*, 2005.

[11] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. K. Miu, E. Shih, H. Balakrishnan, and S. Madden. CarTel: A Distributed Mobile Sensor Computing System. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2006.

[12] 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.

[13] P. Juang, H. Oki, Y. Wang, M. Martonosi, L.-S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2002.

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

[15] Y. Lin, B. Li, and B. Liang. Efficient network coded data transmissions in disruption tolerant networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2008.

[16] Y. Lin, B. Li, and B. Liang. Stochastic analysis of network coding in epidemic routing. *IEEE Journal on Selected Areas in Communications*, pages 794–808, 2008.

[17] A. Maffei, K. Fall, and D. Chayes. Ocean Instrument Internet. In *Proc. ASU Ocean Science Conference*, 2006.

[18] Michael C. Grant, Stephen P. Boyd. CVX: Matlab software for disciplined convex programming. http://www.cvxr.com/.

[19] J. Partan, J. Kurose, and B. N. Levine. A Survey of Practical Issues in Underwater Networks. In *ACM International Workshop on UnderWater Networks (WUWNet)*, 2006.

[20] T. Spyropoulos, K. Psounis, and C. Raghavendra. Efficient Routing in Intermittently Connected Mobile Networks: The Multi-copy Case. In *ACM/IEEE Transactions on Networking*, volume 16, pages 130–143, 2007.

[21] T. Spyropoulos, K. Psounis, and C. Raghavendra. Efficient Routing in Intermittently Connected Mobile Networks: The Single-copy Case. In *ACM/IEEE Transactions on Networking*, volume 16, pages 63–76, 2007.

[22] A. Vahdat and D. Becker. Epidemic Routing for Partially Connected Ad Hoc Networks. Technical Report CS-200006, Duke University, April 2000.

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

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

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

[26] X. Zhang, G. Neglia, J. Kurose, and D. Towsley. On the Benefits of Random Linear Coding for Unicast Applications in Disruption Tolerant Networks. In *IEEE Workshop on Network Coding, Theory, and Applications (NETCOD)*, 2006.

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

[28] X. Zhang, G. Neglia, J. Kurose, D. Towsley, and H. Wang. Benefits of network coding for unicast application in disruption tolerant networks. *IEEE/ACM Trans. on Networking*, 21(5):1407–1420, October 2013.