Abstract—Disruption Tolerant Networks (DTNs) are characterized by frequent network partitioning. This causes network connectivity to be opportunistic. In recent years, many collaborative applications have been envisioned for DTNs. These applications rely on the availability of reliable group communication protocols. To address this challenge in DTNs, we propose a scalable multicasting scheme that deterministically guarantees message delivery to all receivers in the group. Also, our scheme makes controlled use of non-multicast nodes to reduce latency of message delivery. Our work introduces a new measure called termination delay, which is the delay incurred in assuring that the receivers have obtained a copy of the message. Simulation results show that the proposed scheme has much smaller delivery latency and termination delay than other schemes in literature.

I. INTRODUCTION

Disruption Tolerant Networks (DTNs) are based on the premise that the network is often highly partitioned and there is no connected end-to-end path between most pairs of nodes at any given time instant. Such networks are increasingly being envisioned for many applications. For example, a vehicular battlefield network of sensing devices carried by troops or vehicles [1]. In this scenario, the needed information exchange between the devices must occur during the occasional contacts between them while they move along with the troops and vehicles.

Many of the collaborative applications envisioned for DTN require Multicasting. Although, multicasting can be implemented by sending separate unicast messages to each receiver in the DTN, this approach does not work well in many situations [2]. Also, recent research has shown that conventional network protocols do not work well in DTNs [3], [4]. Consequently, there is growing interest in developing new protocols tailored for DTNs [3], [5].

Zhao, Ammar, and Zegura were among the first to propose appropriate semantic models, knowledge models, and the associated routing algorithms for multicasting in DTNs [2]. These semantic and knowledge models reflect the fact that multicast group memberships are likely to change considerably in the time required to deliver a message to all the receivers. They also compare performance of several schemes, namely Static Tree-Based Routing, Dynamic Tree-Based Routing, Group-Based Routing, Broadcast-Based Routing, and Unicast-Based Routing under different knowledge models and traffic conditions. In [6] the authors propose a custodial bundle transfer that extends the custodial unicast to multicast. In [7], Ye et al. propose a tree based multicasting scheme. In this scheme, each node that receives a multicast message reconstructs the tree based on its most recent estimates of the connectivity between the nodes. A common link in these multicasting schemes is for nodes to have some kind of tree structure maintained in them. This tree structure introduces a fixed node meeting requirement, which may not be optimal in a store carry and forward network like DTN.

Another area of research has focused on using probabilistic forwarding schemes [8] for multicasting. In these schemes, the source node forwards every packet with a probability of 1 and every other node in the network forwards with a probability \( p < 1 \). A recent enhancement to probabilistic schemes included combining them with rateless coding techniques to optimize the energy efficiency of message delivery [9]. This algorithm provides a probabilistic guarantee on packet delivery. When reliability is of concern, such probabilistic schemes fail to have an optimal performance in terms of overhead.

Contribution: We focus on guaranteed message delivery in the context of DTNs. The key features of the proposed scheme are as follows. First, it does not rely on a fixed multicast tree to guarantee message delivery. Instead, it utilizes changes in connectivity as they occur to deliver the messages without any assumption on contact information. Second, it makes controlled use of non-multicast nodes to substantially improve the average latency of message delivery, without flooding the network. Third, in addition to delivery latency, the scheme also controls a measure called termination delay. We consider a message to have terminated when all custodial nodes (defined later) know that the message has been successfully delivered to all the receivers.

II. MODEL AND ASSUMPTIONS

In this section, we describe our DTNs model and define the problem of reliable multicasting. In [10] we introduced the mobility model and membership model for DTN. We briefly describe the model here and refer the interested reader to our earlier work.

A. DTN Model

1) Mobility Model: The network is comprised of \( N \) mobile nodes with communication range of nodes modeled in a way that there is limited connectivity between any two nodes. Initially, we assume that all \( N \) mobile nodes move in the same region \( R \). This is an interesting special case of a more popular model found in many DTN scenarios and traces collected from real World deployments [11].
2) Membership Model: Due to possibly large forwarding delays in these networks, multicast node membership may change during message delivery. As a result, it is necessary to make a distinction between multicast members and intended receivers of a message. In this paper, we assume the following semantics for the intended receivers of a given message. Suppose a message $m$ arrives/generated at time $t$ at the source node. A source node could be the originator of the message or a message forwarder. Let $M_m$ be the set of multicast members known to the source node at time $t$. Then, the intended receivers of message $m$ are $M_m$. We further assume that $M_m$ is encoded in the message header so that all nodes forwarding the message know the intended receivers. We refer to this as Generation Time Membership model. Note that, this model is an interesting special case of the Temporal Membership model proposed in [2], in which the membership time interval is comprised of only one time instant. We believe that this special case is of interest in many applications because it implies that a message needs to be delivered to all nodes that belong to the multicast group at the time of its generation.

B. Problem Formulation

The problem of reliable multicasting in DTNs can be stated as follows. A multicast stream of messages arrives at a source node $s$. There are $p$ messages in the stream. Message $m$ arrives at node $i$ at time $t_m$ and it must be reliably delivered to the corresponding set of multicast receivers $M_m$. Message $m$ is delivered to receiver $j \in M_m$ at time $r_{ij}$. The end-to-end latency of delivering message $m$ is $\delta_m = (\max_{j \in M_m} r_{ij}) - t_m$ and the average latency in delivering the multicast stream is $\bar{\delta} = 1/p \sum_{m=1}^{p} \delta_m$.

However, the network is not done with a message when it is delivered to all the receivers. Reliable multicasting requires all nodes to “know” that their custodial responsibilities are complete to ensure that all receivers get the messages. Let $e_{ij}$ be the time at which node $j$ “decides” that its custodial responsibilities are complete. Then, the termination latency of message $m$ is $e_m = (\max_{j \in M_m} e_{ij}) - t_m$ and the average termination latency for the multicast stream is $\bar{e} = 1/p \sum_{m=1}^{p} e_m$.

The energy consumed in delivering message $m$ has two components. Let $C_m$ be the total energy consumed in the exchange of control messages required to deliver message $m$. Likewise, let $O_m$ be the total energy consumed in the transmission of message $m$ during the course of its delivery. Then, the total energy consumed in delivering message $m$ is $E_m = C_m + O_m$ and the average energy consumed in delivering the multicast stream is $E = 1/p \sum_{m=1}^{p} E_m$.

Clearly, the goal of any multicasting scheme is to reduce $\bar{\delta}$, $\bar{e}$, and $E$ to the maximum extent possible. However, it may not be possible to simultaneously reduce all three of these measures. Instead, there is usually a trade off between them.

III. REVIEW OF RELIABLE FORWARDING SCHEMES

A. Reliable Group Based Forwarding (RGBF)

This scheme was discussed and characterized in [12]. The scheme involves flooding of the multicast message among all the multicast receivers. There are two key drawbacks to this approach. First, the scheme does not utilize non-multicast nodes in the network to possibly reduce the latency of termination. Utilizing these nodes under RGBF results in flooding throughout the network, which in turn is very expensive in terms of energy and communication bandwidth. Second, the termination latency of RGBF may be large because each node must wait until it meets all other nodes in $M_m$ before relinquishing the custody.

A variant of RGBF is Source Assured Reliability (SAR), where only the source has custodial responsibility for all nodes in $M_m$. However, the delivery latency of SAR will be typically larger than RGBF. SAR also does not exploit other nodes in the network to improve the latency. SAR, however, uses much lesser communication overhead energy than RGBF.

B. Reliable Epidemic Algorithm (REA)

Epidemic algorithms was proposed by Vahdat and Becker [13]. There have been numerous variants of epidemic algorithms for datacasting [8]. Most of these protocols do not provide a reliable guarantee for message delivery. In application scenarios, where reliable multicasting is a requirement epidemic algorithms fail to meet those guarantees. Here, we extend the epidemic algorithm to provide reliable message delivery guarantees. We call this reliable epidemic algorithm as REA.

Packet Forwarding: The scheme involves flooding of the multicast message among all the multicast receivers and participating non-members. When a node $i \in M_m$ that has already received the message meets another node $j$, it forwards the message if and only if $j \notin M_m$ and $j$ has not received the message. In order to ensure all receivers get the message, each node in $M_m$ maintains custody of the message until it knows that all nodes in $M_m$ have received the message using the scheme below.

Packet Termination: Every multicast receiver maintains a list of nodes in $M_m$, who it knows have already received the message. When two nodes meet, they exchange their lists and update them as necessary. For instance, if node $j$ meets node $i$ and node $j$ knows that node $k$ has already received the message, then node $i$ will also update its list to indicate that node $k$ has received the message. A node terminates when it knows that nodes in $M_m$ have received the message.

In comparison to these schemes, we will illustrate in next section how termination latency in our solution would be smaller since the custodial responsibilities are satisfied in a distributed fashion.

IV. PROPOSED RELIABLE MULTICASTING SOLUTION

As stated in Section II, in general, it is possible for each node to have its own territory. However, for ease of understanding, we first assume that all nodes move in the entire region and therefore there is a non-zero probability of any two nodes coming into communication range of each other.

A. STRAP: Single-Territory Reliability Assurance Protocol

We introduce a parameter $l$ that controls the number of nodes that a custodial node has to meet before it could terminate the multicast message $m$. 
To ensure reliable delivery of $m$ to all nodes in $M_m$, STRAP-$k$ employs a scheme in which nodes have certain “custodial” responsibilities for delivering the message. During the course of message delivery, a node may be assigned custodial responsibility for delivering the message to certain number of nodes. When a node is assigned custodial responsibilities for delivering the message to $d$ other nodes, it fulfills the responsibility in one of two ways: (i) by directly delivering the message to some $l \leq d$ nodes, and (ii) by transferring custodial responsibilities for the remaining $d-l$ nodes to other nodes. When a node completes its custodial responsibility for a message $m$, it can remove the message from its buffer and we say that the node has terminated with respect to the message.

In the proposed scheme, each node in the network periodically broadcasts a beacon. When node $i$ receives a beacon from node $j$, node $i$ first checks whether it exchanged information with node $j$ in the last $t_c$ time units, where $t_c$ is a design parameter. If yes, node $i$ ignores node $j$ and no specific action is taken. Otherwise, node $i$ sends the sequence number and the state (to be defined later) of each message in its queue to node $j$. It also receives the sequence number and the state of each message in node $j$’s queue. Further interactions for a particular message depends on its state at the two nodes.

Now consider a typical node $i$. In STRAP-$k$, node $i$ can be in one of the following four possible states with respect to message $m$.

- **Not Yet Received Message (NYRm):** A node in this state has not yet received message $m$.
- **Received Message No Custody (RmNC):** A node in this state has received the message but it has not yet been assigned any custodial responsibility for the message.
- **Received Message Have Custody (RmHC):** A node in this state has received the message and it has unfulfilled custodial responsibilities for the message.
- ** Terminated Message (Tm):** A node in this state has received the message and it has fulfilled all the custodial responsibilities for the message.

When node $i$ begins interaction with node $j$ there are 16 possible state combinations for each message. We describe STRAP-$k$ by specifying the actions in each of these 16 state combinations. Note that, in 10 out of these 16 cases, nodes $i$ and $j$ take no further action. Thus, we only explain the interesting cases here.

Case 11: $(i \in \text{RmNC}, j \in \text{NYRm})$: If $j \in M_m$, node $i$ forwards the message to node $j$ and node $j$ transitions to state RmNC. Otherwise, no further action is required.

Case 12: $(i \in \text{NYRm}, j \in \text{RmNC})$: Same as Case 11 except roles of $i$ and $j$ are reversed.

Cases 13, 14: $i \in \text{RmHC}$ and $j \in \{\text{NYRm}, \text{RmNC}\}$: This is one of the most interesting cases. The actions executed by node $i$ depends on whether or not node $j$ is a multicast node.

- **Subcase A:** $i \in M_m$ and $j \in M_m$. There are four key steps executed by node $i$. First, if node $j$ is in NYRm then node $i$ forwards the message to node $j$. Second, node $i$ transfers some of its custodial responsibilities to node $j$. More specifically, let $d_m$ be the number of nodes for which node $i$ was originally given the custodial responsibilities. Let, $c_m = \min \{d_m - 1, l\}$. Then, node $i$ transfers custodial responsibilities for approximately $(d_m - 1)/c_m$. It is approximate because $(d_m - 1)/c_m$ may not be an integer. There are simple ways of dealing with this situation. For instance, let $r = (d_m - 1)/c_m$. Then, for the first $r$ nodes, transfer custody for $\lfloor (d_m - 1)/c_m \rfloor + 1$. For the remaining $c_m - r$ nodes transfer custody of $\lfloor (d_m - 1)/c_m \rfloor$. This results in a net transfer of custody for $d_m - 1$ nodes.

Third, the nodes also pass along information to bound the number of non-multicast nodes that can be used. In particular, when node $i$ was given custody (i.e., when it transitioned to RmHC), it was also given another parameter $s_m$ which specifies the maximum number of non-multicast nodes it can use in fulfilling its custodial responsibilities. For the source node, $s_m = k$. When node $i$ transfers custody to node $j$, it also allocates approximately $s_m/c_m$ non-multicast nodes to node $j$. Note that $s_m$ is always less than $k$ the maximum permissible non-multicast node usage in the network.

Finally, if $j$ is the first node to which it transferred custody, then node $i$ terminates for $m$, i.e., it transitions to Tm. Node $j$ transitions to RmHC.

- **Subcase B:** $i \in M_m$ and $j \not\in M_m$. When node $i$ is given custody, it was also given another parameter $s_m$ which specifies the maximum number of non-multicast nodes it can use in fulfilling its custodial responsibilities. If $s_m > 0$, then the steps taken are as described in Subcase A. On the other hand, if $s_m = 0$, then node $j$ is ignored and no further action is taken.

- **Subcase C:** $i \not\in M_m$ and $j \in M_m$: The actions taken are the same as in Subcase A except for the following two differences. First, $c_m$ is defined as $\min \{d_m, l\}$. Second, node $i$ transfers custody for approximately $d_m/c_m$ nodes.

- **Subcase D:** $i \not\in M_m$ and $j \not\in M_m$: If $s_m > 0$ for node $i$, then the steps taken are as described in Subcase C. On the other hand, if $s_m = 0$, then node $j$ is ignored and no further action is taken.

Cases 15, 16: $(i \in \{\text{NYRm, RmNC}\}, j \in \text{RmHC})$: Same as Cases 13 and 14 except roles of $i$ and $j$ are reversed.

STRAP-$k$ algorithm assures message delivery to all multicast receivers as described above. Every node $i \in M_m$ will start in NYRm and transition to Tm state.

### B. Performance Evaluation of Single-Territory Scenario

To characterize the effectiveness of STRAP we present OpnetTM simulation results comparing their performance to that of SAR and REA. The DTN has 50 nodes. The number of participating non-multicast nodes is varied to characterize the effect of $k$ on the performance. The communication range of a node was chosen to model a DTNs by ensuring the average strongest connected component has fewer than 50% of the nodes. For REA, participating non-member nodes are fixed among the 50 nodes, which prevents opportunistic connectivity. On the other hand, in STRAP-$k$, the $k$ is chosen randomly among the available non-member nodes.

The simulation parameters used are given in Table I. All nodes follow a random waypoint movement pattern. We ensured that there is a non-zero probability of commu-
communication between any two pairs of nodes in this model. The performance is characterized using three measures: (i) average end-to-end delivery latency \((\bar{\delta})\), (ii) average termination latency \((\bar{\epsilon})\), and (iii) average energy consumed in the data and control transmissions \((E)\). The simulated multicast stream has 10,000 messages over which the averages are computed.

Figure 1 illustrates the packet delivery latency for 10 multicast receivers. In this plot, we compare only SAR and STRAP-\(k\) schemes. Although, we haven’t shown the delay for REA the packet delivery procedure and hence the delay is similar to SAR and STRAP schemes. The plot represents average packet delay (normalized to STRAP-0) for SAR and STRAP-\(k\) schemes against area. For the participating non-members, the \(k\) in each of the STRAP schemes were chosen randomly. The comparison with SAR shows that packet delay in the proposed scheme is comparable to that of epidemic algorithms. We observe that with increase in non-member nodes \(k\), the packet delivery latency decreases. This reduction in packet delay is due to the number of participating nodes that forward packets in comparison to STRAP-0. Finally, as the connectivity varies from sparsely connected \((15 \times 15 \text{Km}^2)\) to almost connected \((1 \times 1 \text{Km}^2)\) we observe the packet delay reducing rapidly and to a point where \(k\) does not have significant impact on packet delivery latency, since the member nodes itself have more opportunities to meet each other more often. Figures 1(b) characterize the performance of STRAP-\(k\) with respect to packet termination latency. Both plots, represent the average termination latency (normalized to STRAP-0) for 10 and 25 multicast receivers against varying area. Here we compare STRAP-\(k\) against REA-\(k\) and SAR schemes. In Figure 1(b), we have termination latency \((\bar{\epsilon})\) for 10 multicast receivers. The first observation we make is that STRAP and REA are consistently better than SAR for all values of area. We notice that as the connectivity is sparse in \(15K \times 15K\), STRAP has better performance than reliable epidemic algorithm. Progressively as the area reduces, connectivity is more dense and epidemic algorithms benefit and the termination latency for REA at lower areas are comparable if not better than STRAP schemes. With increase in \(k\) and increase in connectivity (smaller areas), STRAP-10 does not improve much in comparison to STRAP. This is due to the unnecessary custodial transfers that might occur in STRAP-10 and for smaller areas there are more opportunistic node meetings between all pairs of nodes, as a result STRAP-0 terminates faster. Figure 2 shows the distribution of average termination latency for 10 multicast receiver scenario. The plot shows data points for REA and STRAP schemes. Each data point represents the termination latency for node 1, node 2, ..., node 10 to terminate; for convenience the node ids are permuted so that the node which terminates first is called node 1, then node 2 and so on until the last node to terminate is called node 10. As we observe in figure 2(a), STRAP allows nodes to terminate as soon as they complete their custodial responsibility, as a result more than 50% of nodes were able to terminate before the first node terminated through REA.

Improvements in packet delay and the termination latency occur at the expense of additional energy consumption (see Figure 3). The plot shows a ratio of 1 : 10, that is 1 unit of control overhead is equivalent to 10 units of data overhead for a \(15K \times 15K\) area. Both the data energy and control energy increases with \(k\). The control overhead is much higher than the data overhead, since the data packets are exchanged only if nodes do not have the message, as a result the number of data packets is proportional to the number of participating nodes in the network. On the other hand, control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>1 - 255 Km²</td>
</tr>
<tr>
<td>Speed</td>
<td>Uniform (10, 20) m/s</td>
</tr>
<tr>
<td>Number of Members</td>
<td>10 - 25</td>
</tr>
<tr>
<td>Number of Non-members</td>
<td>5 - 25</td>
</tr>
<tr>
<td>MAC</td>
<td>802.11b</td>
</tr>
<tr>
<td>Traffic</td>
<td>Source generates a message every 1000 secs</td>
</tr>
</tbody>
</table>

**TABLE I**

**SIMULATION PARAMETERS - STRAP**
packets are used to exchange beacons and other related state information. Since, these are exchanged often, control overhead is higher than data overhead. In particular, for our scenario beacon is the main factor attributing to the control overhead. We can control the beacon interval based on the connectivity. There is a tradeoff between energy consumed and the latencies achieved. However, comparing the results for REA and STRAP it shows that epidemic algorithms have higher control overhead than STRAP. The distribution of multicast receiver termination latency as shown in Figure 2 plays a role in lower overhead for STRAP. Since, nodes have terminated, they do not exchange many of the control packets, which otherwise would have been exchanged in the epidemic case.

C. A combined Scheme: STEP

REA is better than STRAP when the network is well connected and STRAP is better than REA when the network is sparsely connected. We want to make use of the fact that nodes benefit from both epidemic algorithm and STRAP based on the DTN application connectivity. The combined scheme (STEP) tries to achieve the same by combining REA and STRAP. In the combined scheme, packet forwarding is similar to STRAP.

Packet Termination: Each node maintains the data structure for both STRAP and REA. A node terminates if it can complete its custodial responsibility through STRAP or it knows that all other members in $M_{on}$ have received the message through REA. The termination latency in STEP is the min of termination latency obtained either through STRAP or REA. The intuition behind this reasoning is that some nodes will be able to terminate through STRAP, while others terminate through REA. As a result, the combined scheme is advantageous in comparison to both STRAP and REA.

Figure 4 shows the average packet termination latency for 10 receivers. The key observation is that the combined scheme is able to terminate faster than STRAP and REA for larger areas. The reason is that STRAP is better than REA for these scenarios, and as a result termination improves than a pure STRAP scheme. On the other hand, when REA has better termination than STRAP, the combined scheme reaches only the performance of REA, this is due to the fact that once nodes start terminating through REA, nodes cannot be terminated through STRAP, thus the termination latency for lower area (almost connected network) saturates at the termination latency of REA.

V. CONCLUSION

This paper proposed an algorithm designed to overcome the shortcomings of frequent network partitioning that characterizes DTNs. The performance of the algorithm was evaluated through simulations. Comparative studies of STRAP-k performed against RGBF and SAR showed that our algorithm outperforms them in terms of the termination delay and message latency.

REFERENCES


