# A New Reliable Transport Scheme in Delay Tolerant Networks Based on Acknowledgments and Random Linear Coding

Arshad Ali[1], Eitan Altman[2], Tijani Chahed[1], Manoj Panda[2], Lucile Sassatelli[3]
[1]Institute Telecom, Telecom SudParis, UMR CNRS 5157, 9 rue C. Fourier - 91011 Evry Cedex - France
{arshad.ali, tijani.chahed}@it-sudparis.eu
[2]INRIA, 2004 Route des Lucioles, 06902 Sophia-Antipolis, France
{eitan.altman, manoj.panda}@sophia.inria.fr
[3]I3S, University Nice Sophia Antipolis-CNRS UMR 6070, 06903 Sophia Antipolis, France
sassatelli@i3s.unice.fr

***Abstract*—We propose a new reliable transport scheme for Delay Tolerant Networks (DTNs) based on the use of acknowledgments (ACKs) as well as coding. We, specifically, develop a fluid-limit model to derive expressions for the delay performance of the proposed reliable transport scheme and derive the optimal setting of the parameters which minimize the file transfer time. Our results yield optimal values for the number of outstanding random linear combinations to be sent before time-out as well as the optimal value of the time-out itself, which, in turn, minimize the file transfer time.**

## I. Introduction

Delay Tolerant Networks (DTNs) are Mobile Ad Hoc NETworks (MANETs) in which the number of mobile nodes per unit area is typically very low which makes the connectivity between the nodes intermittent. In a DTN environment, in the absence of connectivity, a source node has to rely on the mobility of other nodes which act as relays, and takes advantage of the transmission opportunities which occur when the mobile relays come into contact. Hence, the transfer of data in DTNs is very much dependent on the mobility of the nodes and the packet replication method. Such forwarding strategy is commonly known as *opportunistic routing* [1], [2].

In reliable data transfer, the data source wishes to ensure that all the information it sends arrives correctly at the destination. TCP is by far the most deployed protocol for reliable data transfer. In TCP, data packets are given unique sequence numbers so that the destination can detect which packet(s) is/are missing. To achieve reliability, the destination sends acknowledgements (ACKs) back to the source for each packet (or for each $k$-th packet, in case of delayed ACKs) received without errors. A packet $i$ is considered to be lost if either it is not acknowledged within some time-out period, $T_0$, or if the ACKs for three packets sent more recently than packet $i$ arrive at the source before the ACK for packet $i$ (so called triple duplicate ACKs). In addition to this, TCP implements a flow/congestion control mechanism wherein packets are sent within a so called congestion window, whose size increases with the receipt of ACKs and decreases after time-outs or in case of triple duplicate ACKs.

In general, TCP turns out to be very inefficient for reliable data transfer in MANETs because it misinterprets losses due to link failures as losses due to congestion [3]. This is even worse in the case of DTNs as connectivity intermittence is even more significant in this case [4]. Existing solutions in MANETs rely on cross-layer signaling between routing and transport so as to inform the latter about route failures. This strategy cannot be used in the DTN setting as only opportunistic routing can be performed. In this paper, we propose a new reliable transport scheme for DTNs based on ACKs and coding.

Protocols based on rateless codes are appealing alternatives to enhance reliability in a future Internet, in general, and in DTNs, in particular [5]. Suppose that a file consisting of $M$ "information" packets is to be transferred from a source to a destination. Protocols based on rateless codes allow the source and/or the relays to keep on sending newly generated "coded" packets as combinations of randomly selected packets, so-called *Random Linear Combinations* (RLCs), that they already have received. The loss of an RLC can be compensated by another RLC, whereas without coding, the loss of a packet has to be compensated by retransmitting the same packet. With coding, the only thing that matters for the destination is to meet a certain *Degree of Freedom* (DoF), corresponding to the number of linearly independent RLCs it has to receive in order to be able to decode the file, and not the receipt of specific packets.

For low-complexity rateless codes, such as Fountain and Raptor codes [6] [7], if the number $M'$ of coded packets received at the destination is a little more than the initial number of information packets $M$, then with high probability, the destination can reconstruct the whole file. This success probability can be set as high as desired by properly choosing $M'$. For Random coding, or random linear network coding [8], $M'$ can be equal to $M$ to get high success probability, but the encoding and decoding complexities are, in this case, maximum. In this paper, we focus on this latter coding scheme.

In the absence of feedback from the destination, the source

978-0-9836283-0-9 



This paper was peer reviewed by subject matter experts for publication in the Proceedings of ITC 2011

cannot know for sure how many packets, be they coded or not, made it successfully to the destination, and hence only a *probabilistic* form of reliability is obtained: the source can only ensure a certain probability of successful delivery of packets to the destination. Coding is used to enhance the delay and/or the throughput performance by increasing the success probability of received RLCs within a given time. Our aim in this work, however, is to ensure a *deterministic* form of reliability, i.e., we want to be sure that the RLCs have indeed reached the destination. And so, we also make use of ACKs, in addition to coding, to cater to the losses in the network.

We propose a scheme in which transmission is organized within cycles. During one cycle, the source sends a given number of RLCs (which is a function of the number of missing DoFs) in a back to back manner without waiting for any feedback. The source then allows the RLCs to propagate to the destination and then waits for the ACKs to come back until a so called *cycle time-out* period. Each RLC received at the destination triggers the sending back of an ACK, indicating to the source the new number of DoFs still missing at the destination. In order that no flow or subset of flows monopolizes the (finite size) buffers at the relays, we allow for a so called *buffer expiry time-out* mechanism for each individual RLC in the relay buffer, at the end of which the given RLC is dropped. The cycle ends at the end of the cycle time-out, at which time, the source sends new RLCs, the number of which corresponds to the missing number of DoFs as indicated by the most up-to-date ACK the source has received so far. The process is repeated until the completion of the transmission of the whole file.

**Literature Survey:** The literature on DTNs is mainly concerned with routing and only a few works deal with transport. The Bundle Protocol [9] specifies a framework rather than a concrete protocol implementation. The Saratoga protocol [10] provides an IP-based convergence layer in DTNs supporting store-and-forward of bundles [10]. It performs UDP-based transfer of IP packets with Selective Negative Acknowledgements (SNACKs). The Licklider Transmission Protocol (LTP) [11] is designed to serve as a DTN convergence layer protocol. It provides retransmission-based reliable transfers over single-hop connections and also supports unreliable transmission.

The literature on reliable transport in DTNs is mostly concerned with deep-space communication. The CFDP protocol [12] provides file copy services over a single link and requires all parts of a file to follow the same path to the destination. The DS-TP protocol [13] is based on Double Automatic Retransmission to provide proactive protection against link errors. The TP-Planet protocol [14] employs Additive Increase Multiplicative Decrease control mechanism and uses time-delayed SACKs to deal with asymmetric bandwidth.

**Our contributions:** We make the following contributions:

- We propose a new reliable transport scheme for DTNs based on ACKs and coding at the source.
- We develop a fluid-limit model of the evolution of the network under our proposed scheme and demonstrate its accuracy by comparing with simulations.
- Based on our fluid-limit model, we carry out a joint optimization of the number of outstanding RLCs to be sent in one cycle, as well as of the value of the time-out, so as to minimize the transfer time of a file.

The remainder of this paper is organized as follows. We describe our network setting in Section II. In Section III, we provide a detailed account of our proposed reliable transport scheme. In Section IV, we develop a fluid-limit model of our proposed reliable transport scheme, and also describe a joint optimization procedure to minimize the file transfer time. In Section V, we demonstrate the accuracy of our fluid-limit model and the joint optimization procedure. Section VI concludes the paper providing future directions of research.

## II. Network Setting

To better explain our reliable transport scheme and to be able to develop a simple analytical model, we focus on the transfer of packets from a single source to a single destination corresponding to a single *flow*. The network consists of $N+2$ mobile nodes. There is one source node, one destination node, and the remaining $N$ nodes act as relays for the considered flow. The source sends packets to the destination and the destination sends back ACKs indicating the number of DoFs still missing at the destination.

We consider *epidemic routing* in which each mobile keeps forwarding a copy of its packet (or ACK) to the other mobiles it encounters, which also spread the packet (or the ACK) in that way. We say that two nodes "meet" when they come within the communication range of each other. We assume that the successive inter-meeting times between any two nodes are exponential random variables with mean $1/\beta$. Our assumption of i.i.d. exponential inter-meeting times is motivated by the works [15], [16], [17], wherein it was shown via simulations that, for "random waypoint" and "random direction" mobility models, the assumption of i.i.d. exponential inter-meeting times provides extremely accurate approximations for actual inter-meeting times provided that the communication range $r \ll L$, where $L \times L$ denotes the area of the network.

We assume that the relays have buffer capacity to store at most one packet or one ACK at any point in time. In reality, a relay could store multiple packets and ACKs. However, in real DTNs, there would be several other competing flows sharing the buffer space at the relays. Thus, our assumption of buffer capacity of one packet or ACK can be viewed as saying that the buffer capacity of a relay is limited to one packet or ACK *per flow*. Furthermore, to make room for packets and ACKs of other flows, a packet (of the flow under consideration) is retained in a relay buffer only for a duration $\tau_e$, called the *buffer expiry time-out*, and then dropped. However, we assume that ACKs are never dropped to make room for other packets or ACKs, since ACKs are much smaller than packets and they contain valuable information. The duration $\tau_e$ depends on several factors such as the number of simultaneous flows, the buffer capacity at the relays etc. We view the buffer expiry time-out $\tau_e$ as a constraint imposed by such external factors, and *we assume that $\tau_e$ is a given network parameter*.

We consider coding only at the source. The source generates RLCs of the information packets it wishes to send to the destination. The RLCs are generated over $\mathbb{F}_q$, where $\mathbb{F}_q$ denotes the Galois field of size $q$. The random coefficients used to generate an RLC, called the *encoding vector*, is included in the header of the coded packet. The case of coding at the relays is beyond the scope of the present work and will be considered in the future.

## III. Our Proposed Reliable Transport Scheme

The objective is to transfer a file consisting of $M$ information packets from the source to the destination in a reliable manner. Upon meeting relays, the source generates RLCs of the $M$ information packets. The relays carry and replicate the RLCs as detailed in Section III-B. To recover the $M$ information packets, the destination needs to receive $M$ DoFs, i.e., the rank of the matrix, formed by accumulating the encoding vectors of the received RLCs, must be equal to $M$.

To transfer the $M$ information packets, we propose a variant of the packet transfer mechanism introduced in [18]. As in [18], the ACKs in our scheme carry the number of DoFs still missing at the destination to recover the $M$ information packets. To ensure reliability in presence of packet drops due to buffer expiry time-out, our scheme: (1) evaluates the progress of the transfer at appropriate intervals using the feedback information provided by the ACKs, and then (2) takes corrective actions accounting for the feedback information provided by the ACKs. Thus, the $M$ information packets are transferred from the source to the destination in a reliable manner over multiple *cycles*.

### A. Algorithm

Our scheme is detailed as follows:

- **Initialization:** $i \longleftarrow M$.
- **While** $i > 0$,
  - A new cycle begins. The source sends $M_i$ RLCs back to back, where $M_i$ is a function of $i$ and $i$ denotes the missing DoFs in the beginning of the cycle (as viewed by the source). Each time an empty relay meets the source, the source gives a new RLC to the relay until $M_i$ RLCs have been sent.
  - Each RLC is spread for a duration $\tau_{i,S}$, called the *spreading time*, according to the replication scheme described in Section III-B.
  - Each time a relay meets the destination, the destination sends an ACK informing the source how many DoFs are still needed to recover the $M$ information packets. (The ACK generation and replication scheme is detailed in Section III-B.)
  - After emitting the $M_i$th RLC, the source waits for a duration $\tau_{i,S}$ to let the $M_i$th RLC spread in the network, and then waits further for a duration $\tau_{i,W}$, called the *waiting time*. The purpose of the waiting time is to allow the ACKs to reach the source.
  - Replication of the RLCs stops during the ACK-wait phase. However, replication of the ACKs continues throughout the cycle. Furthermore, a copy of an RLC is retained in a relay buffer only for a duration $\tau_e$, whereas a copy of an ACK is retained in the relay buffer throughout the cycle.
  - The cycle lasts for a total duration
$$\tau_i := t_{M_i} + \tau_{i,S} + \tau_{i,W},$$
where $t_{M_i}$ denotes the time at which the $M_i$th RLC is sent by the source. At the end of the cycle: (i) all the relays drop the copy of the RLC or ACK they have, and (ii) the source considers the minimum of the missing DoFs indicated by all the ACKs it has received during the cycle. Let the minimum of the missing DoFs indicated by the ACKs be $j$.
- **Update:** $i \longleftarrow j$.
- **End While**.

### B. RLC and ACK Replication

For the sake of brevity, we call a cycle, which begins with $i$ missing DoFs, an *$i$-cycle*. For example, the first cycle begins with $M$ missing DoFs, i.e., the first cycle is an $M$-cycle. In a $i$-cycle, the source sends $M_i$ RLCs back to back. Each time an empty relay meets the source, the source gives a new RLC to the relay until $M_i$ RLCs have been sent in the current $i$-cycle. Each of the $M_i$ RLCs is spread by the source only once. The source makes use of the first $M_i$ transmission opportunities to send $M_i$ different RLCs.

In any $i$-cycle, we index the RLCs by $k$, $k = 1, 2, \ldots, M_i$, and index the ACKs by $l$, $l = 0, 1, \ldots, i-1$. Note that, *ACK $l$ contains the information that the destination still needs $l$ DoFs to recover the $M$ information packets*. Let $t_k$ denote the time at which RLC $k$ is sent by the source. When a relay with a copy of RLC $k$ meets with an empty relay during $(t_k, t_k+\tau_{i,S}]$, the empty relay gets a copy of RLC $k$. Thus, each RLC, after being sent by the source, is spread for a duration $\tau_{i,S}$, after which it is not replicated any more. Furthermore, a copy of an RLC is retained in a relay buffer only for a buffer time-out period of duration $\tau_e$. An empty relay, which had an RLC (or an ACK) earlier but dropped the RLC (or the ACK) due to buffer time-out or cycle time-out, is also allowed to receive, carry and spread another RLC (or even the same RLC) or ACK subsequently, i.e., we allow *re-infection*. This is desirable from the point of view of implementation, since the relays do not have to remember the history of the RLCs and the ACKs they have already carried. When two nodes, which have different RLCs, meet, then there is no exchange.

When the destination receives an RLC, it updates the missing DoFs, generates an ACK indicating the missing DoF, and the RLC in the relay gets replaced with the latest ACK. When the destination is in a state with $l$ missing DoFs, $l = 0, 1, \ldots, i-1$, it gives ACK $l$ to all the relays it meets, be they empty or not, except to those who already have ACK $l$. Note that, upon meeting an empty relay, the destination does not generate an ACK if it is in a state with $i$ missing DoFs, i.e., if it has not received any useful RLC in the current $i$-cycle. When a relay with ACK $l$ meets an empty relay, the empty

relay gets a copy of ACK $l$. When a relay with ACK $l$ meets another relay with ACK $l'$, $l' > l$, then ACK $l'$ is replaced by ACK $l$, since an ACK indicating a smaller number of missing DoFs provides more recent and more accurate information to the source. ACK 0 also replaces the RLCs, since ACK 0 indicates complete reception of the file and no more RLCs are required to reach the destination. Replication of the RLCs occur only during the RLC-spread phase, whereas replication of the ACKs continues throughout the cycle.

At the end of each cycle, the relays drop the copy of the RLC or ACK they have. This choice is to maintain the stability of the network as follows. Dropping the RLCs at the end of each cycle makes room for the RLCs and the ACKs of the same flow in subsequent cycles as well as for other flows. More importantly, the relays would not know when the transfer is complete. Thus, if the RLCs and the ACKs are not dropped at the end of each cycle, they would remain in the network, unnecessarily, for a very long time. The RLCs would ultimately be dropped due to buffer expiry, but the ACKs will not. We will show in Section V that for longer buffer expiry time-outs, the file transfer completes faster. In such cases, the RLCs would stay in the relay buffer much after the file transfer is complete. Also, the ACKs of the flow under consideration must be dropped, ultimately. Due to these practical considerations, we propose to drop the RLCs and the ACKs from the relay buffers at the end of each cycle.

### C. Implementation Issues

The nodes can implement our reliable transport scheme without being time synchronous. First, the source and the destination must agree on the values of the number of information packets, $M$, and the coding field size, $q$, by a handshaking mechanism. This is similar to the "connection set-up" phase of TCP in which basic variables are exchanged and agreed upon. The cycle time-out $\tau_i$ and the spreading time $\tau_{i,S}$ are included in each RLC generated by the source, and are kept, as is, in every copy of the RLC. The buffer expiry time-out $\tau_e$ is generated afresh by each relay at the time of receiving a copy of the RLC, since $\tau_e$ is local to each relay. An RLC is spread for a duration $\tau_{i,S}$, and is dropped from the relay buffer at the earliest of the time-outs $\tau_i$ and $\tau_e$. Since the destination generates ACKs only after receiving an RLC in the current cycle, the cycle time-out $\tau_i$ is copied into the destination's buffer and subsequently included in the ACKs as well. A final "connection release" mechanism is required in which the source informs the destination to clear all the variables corresponding to the flow under consideration.

In the remainder of the paper, we shall not discuss the connection set-up and release mechanisms any further.

## IV. Analytical Modeling and Optimization

### A. Modeling the Network Dynamics during One Cycle

We model the dynamics of an $i$-cycle to study the spreading of the $M_i$ RLCs and the $i$ ACKs. We can model the network dynamics of any $i$-cycle independent of the earlier cycles, since the relays drop the copy of the RLC or ACK they have at the end of each cycle. Thus, we reset the time variable $t$ to zero in the beginning of each cycle and study the network dynamics in an $i$-cycle for $t \in [0, \tau_i]$.

To model the drop of RLCs in the relay buffers due to expiry, we approximate the constant buffer expiry time-out $\tau_e$ by an exponentially distributed time-out with the same mean $\tau_e$. We shall see that this is a very good approximation in the sense that the resulting analytical model provides accurate predictions of the file transfer delay. Note, however, that *in the simulations, we actually use a constant buffer expiry time-out*. Let $\beta_e := 1/\tau_e$ denote the rate of expiry.

Let $X_k^{(N)}(t)$, $k = 1, 2, \ldots, M_i$, denote the number of relays that have a copy of RLC $k$ at time $t$. Let $Y_l^{(N)}(t)$, $l = 0, 1, \ldots, i-1$, denote the number of relays that have a copy of ACK $l$ at time $t$. The superscript $N$ emphasizes the fact that the evolution of the $X_k^{(N)}(t)$'s and the $Y_l^{(N)}(t)$'s depend on the total number of relays, $N$. Let $P_{X_k}(t)$ denote the probability that the destination has received RLC $k$ by time $t$, and let $P_{Y_l}(t)$ denote the probability that the source has received ACK $l$ by time $t$. Let $Q_l^{(i)}(t)$ denote the probability that the destination has received $i - l$ DoFs in the current $i$-cycle by time $t$. Note that, $Q_l^{(i)}(t)$ also represents the probability that the number of missing DoFs at the destination at time $t$ is $l$. Assuming that the reception of any RLC decreases the number of missing DoFs at the destination by one, $Q_l^{(i)}(t)$ is given by Equation (1) (placed at the top of next page). In reality, reception of any RLC will not decrease the missing DoF by one, and hence, Equation (1) is an approximation. Note, however, that *in the simulations, we actually check if a received RLC indeed decreases the missing DoFs*.

Applying Theorem 3.1 of [19], we observe that, for large $N$, $\forall k$, $k = 1, 2, \ldots, M_i$, $\forall l$, $l = 0, 1, \ldots, i-1$, the expectations $\mathsf{E}\Big(X_k^{(N)}(t)\Big)$ and $\mathsf{E}\Big(Y_l^{(N)}(t)\Big)$ are well-approximated by $Nx_k(t)$ and $Ny_l(t)$, respectively, where $x_k(t)$ and $y_l(t)$, are the unique solution of the ODEs

$$\frac{dx_k(t)}{dt} = \begin{cases} 0 \quad \text{for } 0 \le t \le \frac{k-1}{\lambda}, \\ (\beta + \lambda x_k(t))(1 - x(t) - y(t)) - \beta x_k(t) \\ -\beta_e x_k(t) - \lambda x_k(t) y_0(t) \\ \text{for } \frac{k-1}{\lambda} < t \le \frac{k}{\lambda}, \\ \lambda x_k(t)(1 - x(t) - y(t)) - \beta x_k(t) \\ -\beta_e x_k(t) - \lambda x_k(t) y_0(t) \\ \text{for } \frac{k}{\lambda} < t \le \frac{k}{\lambda} + \tau_{i,S}, \\ -\beta x_k(t) - \beta_e x_k(t) - \lambda x_k(t) y_0(t) \\ \text{for } \frac{k}{\lambda} + \tau_{i,S} < t \le \tau_i. \end{cases}$$

$$\begin{aligned} \frac{dy_l(t)}{dt} = {} & \lambda y_l(t)(1 - x(t) - y(t)) + \beta Q_l^{(i)}(t)(1 - y_l(t)) \\ & + \lambda y_l(t) \sum_{m>l} y_m(t) - \lambda y_l(t) \sum_{m<l} y_m(t) \\ & + \mathbf{1}_{\{l=0\}} \lambda y_l(t) x(t), \text{ for } 0 < t \le \tau_i, \end{aligned} \tag{2}$$

with initial conditions, $\forall k = 1, 2, \ldots, M_i$, $x_k(0) = 0$, and $\forall l = 0, 1, \ldots, i-1$, $y_l(0) = 0$, where $\lambda = N\beta$, $x(t) :=$

$$Q_l^{(i)}(t) = \sum_{E\subset\{1,\ldots,M_i\}\,:\,|E|=i-l} \prod_{m\in E} P_{X_m}(t) \prod_{m'\in\{1,\ldots,M_i\}\setminus E} (1-P_{X_{m'}}(t)) \quad (1)$$

$\sum_{k=1}^{M_i} x_k$ and $y(t) := \sum_{l=1}^{i} y_l$.

To understand Equation (2), the quantities $x_k(t)$ and $y_l(t)$ may be interpreted as the *mean fraction* of relays that have a copy of RLC $k$ and ACK $l$, respectively. With this interpretation, the quantity $x(t) = \sum_{k=1}^{M_i} x_k$ and $y(t) = \sum_{l=1}^{i} y_l$ denote the mean fraction of relays that have a copy of the RLCs and the ACKs, respectively, and $1 - x(t) - y(t)$ denotes the mean fraction of relays that are empty. Note that the mean fraction of relays that have a copy of RLC $k$ is equal to zero until the $k$th meeting of the source with any relay, which roughly occurs in the interval $[(k-1)/\lambda, k/\lambda]$, at which time the first copy of RLC $k$ appears in the network. The term $\beta(1-x(t)-y(t))$ corresponds to forwarding of the RLCs by the source, and this forwarding occurs for a mean duration $1/\lambda$, i.e., for a duration equal to one inter-meeting time between the source and any relay, and the mean number of RLCs forwarded by the source in that duration is equal to one. The terms $x_k(t)(1-x(t)-y(t))$ and $y_l(t)(1-x(t)-y(t))$ correspond to the spreading of the RLCs and the ACKs, respectively, to empty relays. The term $\beta x_k(t)$ corresponds to the replacement of an RLC by an ACK at the destination. The term $\beta_e x_k(t)$ corresponds to the dropping of the RLCs due to buffer expiry time-out. The terms $x_k(t)y_0(t)$ and $\mathbf{1}_{\{l=0\}}y_l(t)x(t)$ correspond to the replacement of the RLCs by ACK 0. The term $\beta Q_l^{(i)}(t)(1-y_l(t))$ corresponds to the spreading of the ACKs by the destination. The terms $\lambda y_l(t)\sum_{m>l} y_m(t)$ and $\lambda y_l(t)\sum_{m<l} y_m(t)$ correspond to the replacement of older ACKs by newer ACKs.

It remains to obtain the $P_{X_k}(t)$'s and the $P_{Y_l}(t)$'s, which denote the probability that the destination has received RLC $k$ and ACK $l$, respectively, by time $t$. To that end, let $D_{X_k^{(N)}}$ denote the delay of RLC $k$ and $D_{Y_l^{(N)}}$ denote the delay of ACK $l$. Let $\tilde{X}_k^{(N)}(t)$ denote the number of copies of RLC $k$ received by the destination up to time $t$. Viewing $\{\tilde{X}_k^{(N)}(t), t \geq 0\}$ as a non-homogeneous Poisson process with rate parameter $\beta X_k^{(N)}(t)$, we observe that the number of copies of RLC $k$ received by the destination in the interval $[0,T]$, conditioned on the natural filtration, $\mathcal{F}_{X_k^{(N)}}$, of the process $\{X_k^{(N)}(t), t \geq 0\}$, is a Poisson random variable with parameter $\beta\int_0^T X_k^{(N)}(t)dt = \lambda\int_0^T \frac{X_k^{(N)}(t)}{N}dt$. Then, the probability that the packet delivery delay, $D_{X_k^{(N)}}$, is larger than $T$, conditioned on $\mathcal{F}_{X_k^{(N)}}$, is given by

$$\begin{aligned} &P(D_{X_k^{(N)}} > T \mid \mathcal{F}_{X_k^{(N)}}) \\ &= P(\tilde{X}_k^{(N)}(T) = 0 \mid \mathcal{F}_{X_k^{(N)}}) \\ &= \exp\left(-\lambda\int_0^T \frac{X_k^{(N)}(t)}{N}dt\right). \end{aligned} \quad (3)$$

Taking expectation (over all the sample paths of the process $\{X_k^{(N)}(t), t \geq 0\}$) on both sides, we obtain

$$P(D_{X_k^{(N)}} > T) = \mathsf{E}\left(\exp\left(-\lambda\int_0^T \frac{X_k^{(N)}(t)}{N}dt\right)\right). \quad (4)$$

Since the right hand side of Equation (4) is bounded for all $N$, applying Dominated Convergence Theorem, we obtain [5]

$$\begin{aligned} P(D_{X_k} > T) &:= \lim_{N\to\infty} P(D_{X_k^{(N)}} > T) \\ &= \exp\left(-\lambda\int_0^T x_k(t)dt\right). \end{aligned} \quad (5)$$

Thus, for large $N$, the Cumulative Distribution Function (CDF) of $D_{X_k^{(N)}}$, $k = 1, 2, \ldots, M_i$, and the CDF of $D_{Y_l^{(N)}}$, $l = 0, 1, \ldots, i-1$, are well-approximated by the unique solution of the ODEs, $\forall t$, $0 < t < \infty$,

$$\begin{aligned} \frac{dP_{X_k}(t)}{dt} &= \lambda x_k(t)(1-P_{X_k}(t)) \\ \frac{dP_{Y_l}(t)}{dt} &= \lambda y_l(t)(1-P_{Y_l}(t)) \end{aligned} \quad (6)$$

with initial conditions $P_{X_k}(0) = 0$ and $P_{Y_l}(0) = 0$, where $x_k(t)$ and $y_l(t)$ are obtained by solving Equation (2).

### *B. Combining the Cycles Together*

We now proceed with the description of the sequence of cycles. Let $\Delta_n$ denote the number of DoFs missing at the destination in the beginning of the $n^{th}$ cycle. It is easy to see that, $\{\Delta_n, n \geq 1\}$ is a Markov chain with state space $\{0, 1, 2, \ldots, M\}$. The Markov chain $\{\Delta_n, n \geq 1\}$ begins with $\Delta_1 = M$, and gets absorbed in state 0. Let $P_{ij}$ denote the transition probability from state $i$ to state $j$.

As in [18], the transition probabilities can be expressed in terms of the *erasure probabilities* as seen by the source. In our context, given that a cycle begins with $i$ missing DoFs (from the point of view of the source), the erasure probabilities correspond to $(1-P_{Y_l}(\tau_i))$, $l = 0, 1, 2, \ldots, i-1$, i.e., the probability that the source has not received ACK $l$ by the end of the $i$-cycle. Thus, the transition probabilities $P_{ij}$, $j = 0, 1, \ldots, i-1$, are given by

$$P_{ij} = P_{Y_j}(\tau_i)\prod_{l=0}^{j-1}(1-P_{Y_l}(\tau_i)), \quad (7)$$

and

$$P_{ii} = 1 - \sum_{j=0}^{i-1} P_{ij}.$$

Let $T_i$, $i = 1, 2, \ldots, M$, denote the expected time to reach the state with 0 missing DoFs, starting from the beginning of an $i$-cycle. Clearly, $T_M$ represents the expected completion

time for the transfer of the whole file. By a renewal argument, we obtain, $\forall i$, $i = 1, 2, \ldots, M$,

$$T_i = \tau_i + \sum_{j=1}^{i} P_{ij} T_j, \tag{8}$$

which, in matrix form, can be written as

$$\boldsymbol{T} = \boldsymbol{\tau} + \boldsymbol{P}\boldsymbol{T} = (\boldsymbol{I} - \boldsymbol{P})^{-1}\boldsymbol{\tau} \tag{9}$$

where $\boldsymbol{T} = (T_1, T_2, \ldots, T_M)'$, $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_M)'$,

$$\boldsymbol{P} = \begin{bmatrix} P_{11} & 0 & 0 & \ldots & 0 \\ P_{21} & P_{22} & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{M1} & P_{M2} & P_{M3} & \ldots & P_{MM} \end{bmatrix},$$

$\boldsymbol{I}$ denotes the identity matrix (of appropriate dimension) and '$'$' represents the transpose.

*Remark* IV.1. The matrix $\boldsymbol{P}$ is obtained by deleting the row and the column corresponding to state 0 in the transition probability matrix of the Markov chain $\{\Delta_n, n \geq 1\}$. Hence, $\boldsymbol{P}$ is a sub-stochastic matrix and $\boldsymbol{I} - \boldsymbol{P}$ is invertible.

### C. Optimization Procedure

Our objective is to minimize the mean time to transfer the complete file, i.e., the time until receiving an ACK indicating 0 missing DoFs. The optimization is performed over the parameters $\{M_i, \tau_{i,S}, \tau_{i,W}\}$, $i = 1, \ldots, M$. Although we do not focus on energy issues (related to RLC and ACK replication) in this paper, it is worth noting that parameters $\tau_{i,S}$ and $\tau_{i,W}$ can also be used to trade between delay and energy. We now briefly describe the optimization procedure.

The objective is to minimize the mean completion time for the transmission of all the $M$ packets, i.e., to minimize $T_M$. We can see from Equation (8) or Equation (9) that the optimization can be recursive. Indeed, we have

$$T_i = \frac{\tau_i + \sum_{j=1}^{i-1} P_{ij} T_j}{1 - P_{ii}}.$$

Since $T_i$, $i = 2, \ldots, M$, depends only on $j = 1, \ldots, i-1$, we can perform a three-dimensional optimization (over the parameters $\{M_i, \tau_{i,S}, \tau_{i,W}\}$) at each step $i$, starting at $i = 1$, and then substituting $T_j$, for $j = 1, \ldots, i-1$, by the values obtained in the previous steps as

$$\min(T_i) = \frac{\tau_i + \sum_{j=1}^{i-1} P_{ij} \min(T_j)}{1 - P_{ii}}.$$

We thereby obtain the optimal values of $\{M_i, \tau_{i,S}, \tau_{i,W}\}$, $i = 1, \ldots, M$, in a recursive manner.

For each step $i$, the three-dimensional optimization of the parameters $\{M_i, \tau_{i,S}, \tau_{i,W}\}$ pertains to the class of nonlinear optimization problems. Many general algorithms for solving such problems have been developed. We experimented with an algorithm called Differential Evolution (DE) [20]. DE is a robust optimizer for multivariate functions. We do not describe DE here, but only say that this algorithm is in part a hill climbing algorithm and in part a genetic algorithm.

The above optimization procedure does not have to be performed at the source node, but is rather performed offline, and the resulting optimal parameters for each possible $i$-cycle, $i = 1, 2, \ldots, M$, are stored in memory at the source node, so as to be used as needed. Note that our optimization procedure requires the knowledge of $\beta$ and $\lambda = N\beta$ which can be estimated by using the history of node meetings.

## V. Validation of the Analytical Model and Performance of the Optimal Procedure

To solve the ODE model, and in the optimal procedure, we take $M = 5$, $N = 100$, $\beta = 0.05$, and $\tau_e = 1, 2, 3, 4, 5, 6$. In the simulations, in addition to the above setting of parameters, we took $q = 1$, i.e., we generated RLCs with *binary* random coefficients and upon receipt of every RLC at the destination we actually checked if the received RLC indeed decreased the missing DoFs by one. Random inter-meeting intervals were generated using exponential distributions. Simulation results have been averaged over 1000 runs.

Notice that we have not specified the units of $\beta$ and $\tau_e$. One can think of $\tau_e = 2$ as $\tau_e = 2$ *sec*, in which case, $\beta = 0.05$ has to be $\beta = 0.05\ sec^{-1}$. Similarly, one can think of $\tau_e = 2$ as $\tau_e = 2\ min$ (resp. $\tau_e = 2\ hr$), in which case, $\beta = 0.05$ has to be $\beta = 0.05\ min^{-1}$ (resp. $\beta = 0.05\ hr^{-1}$).

### A. Validation of the Analytical Model

In Figures 1-6, we compare the delay CDF of the final ACK, ACK 0, obtained as the solution of Equation (6) with the delay CDF of ACK 0 obtained from simulations for a cycle which begins with $i = 3$ and the set of expiry times $\tau_e = 1, 2, 3, 4, 5, 6$, respectively. The values of $M_i$, $\tau_{i,S}$ and $\tau_{i,W}$ are set to the optimal setting as provided by the optimal program. A good match between analysis and simulation curves can be observed and validates our ODE model near the optimal settings where the network will operate. Note, however, that our analytical model is also accurate in other regimes, but since it is not possible to show the accuracy of our model in the entire parameter space, we have chosen the optimal settings to demonstrate the accuracy of our model.

### B. Performance of the Optimal Procedure

In Figures 7-9, we plot the optimal values of the number of RLCs, $M_i$, the spreading time, $\tau_{i,S}$, and the waiting time $\tau_{i,W}$, respectively, for the set of expiry times $\tau_e = 1, 2, 3, 4, 5, 6$. In Figures 10-12, we plot the optimal values of the cycle time $\tau_i$, the mean completion time, $T_i$, starting from a missing DoFs $i$, and the file transfer time, $T_M$, respectively. Despite the erratic behavior of the optimal values of the parameters $M_i$, $\tau_{i,S}$ and $\tau_{i,W}$ at first sight, it is worth noting that, they give rise to a flat behavior of $\tau_i$, $T_i$, and $T_M$. The optimal procedure adjusts the parameters $M_i$, $\tau_{i,S}$ and $\tau_{i,W}$ so as to make $T_i$ as close as possible to $\tau_i$ (compare Figures 10 and 11 and compare $T_M$ and $\tau_M$ from analysis in Figure 12). This implies that the optimal procedure tries to provide settings such that, starting with any missing DoF $i$ in the beginning of a cycle, the file transfer completes in the same cycle with high probability. In

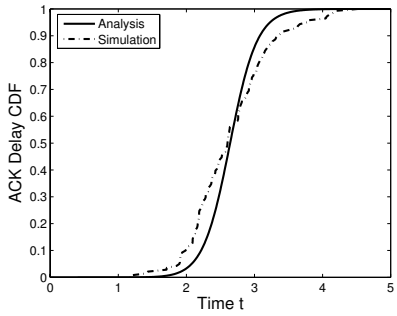


Fig. 1. Comparison of delay CDF of ACK 0 from analysis and simulation with expiry time 1 for a cycle which begins with missing DoF 3.

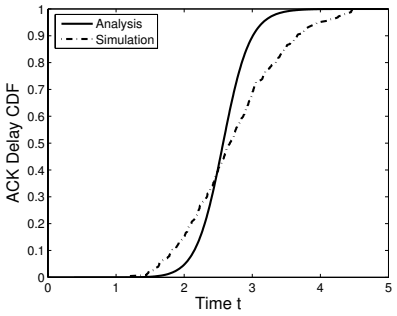


Fig. 2. Comparison of delay CDF of ACK 0 from analysis and simulation with expiry time 2 for a cycle which begins with missing DoF 3.

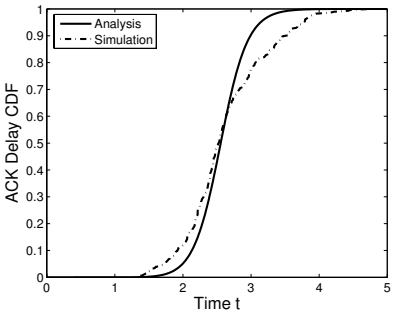


Fig. 3. Comparison of delay CDF of ACK 0 from analysis and simulation with expiry time 3 for a cycle which begins with missing DoF 3.

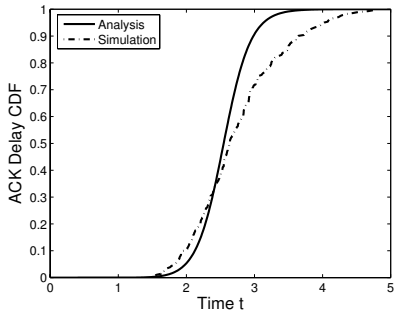


Fig. 4. Comparison of delay CDF of ACK 0 from analysis and simulation with expiry time 4 for a cycle which begins with missing DoF 3.

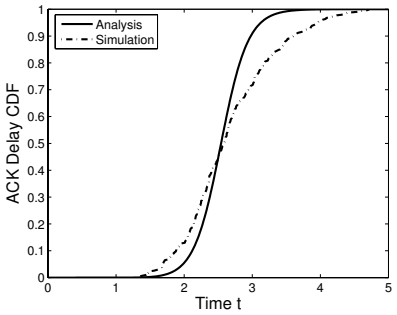


Fig. 5. Comparison of delay CDF of ACK 0 from analysis and simulation with expiry time 5 for a cycle which begins with missing DoF 3.

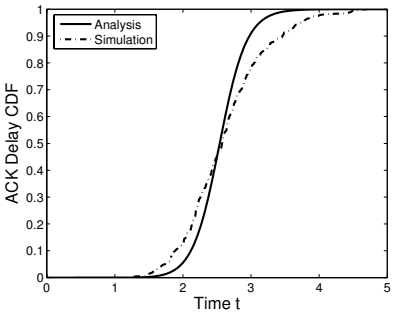


Fig. 6. Comparison of delay CDF of ACK 0 from analysis and simulation with expiry time 6 for a cycle which begins with missing DoF 3.

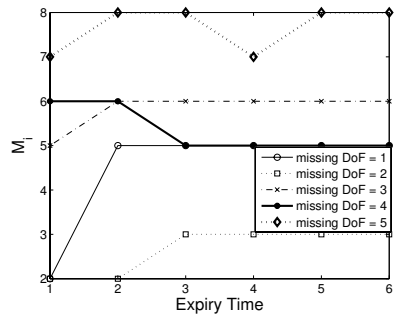


Fig. 7. Optimal values of $M_i$.

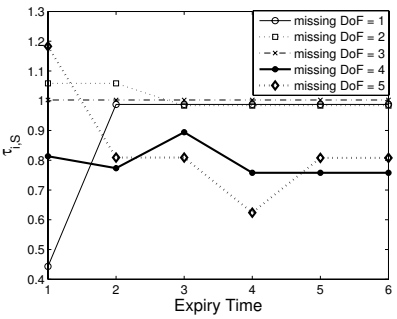


Fig. 8. Optimal values of $\tau_{i,S}$.

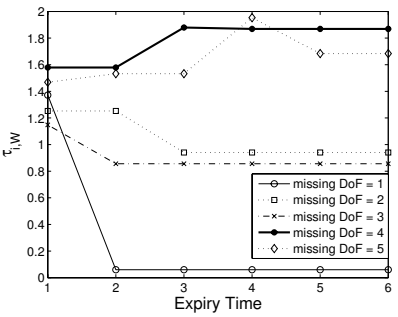


Fig. 9. Optimal values of $\tau_{i,W}$.

the simulations, we observed that this is indeed so under the optimal setting of parameters.

In Figure 12 we compare the mean file transfer time from simulations under the optimal settings with the file transfer time provided by the optimal procedure. It can be observed that the mean file transfer times from simulations, under the optimal settings of the parameters $M_i$, $\tau_{i,S}$ and $\tau_{i,W}$, are in excellent agreement with the optimal mean file transfer times. This validates our overall procedure of minimization of mean file transfer time based on our fluid-limit model.

## VI. Conclusion

In this work, we proposed a new reliable transport scheme for DTNs based on the use of ACKs and coding. We modeled the evolution of the network under our scheme using a fluid-limit approach. We obtained the CDF of ACK delays and showed, by comparing numerical analysis versus simulations, the validity of our models. Using our fluid-limit model, we optimized the number of outstanding RLCs to be sent back to back before the expiration of a time-out as well as the value of the time-out itself, so as to minimize the file transfer time. We accounted for the buffer expiry time-out, quantified its impact on the optimal values of our protocol parameters and also demonstrated the adaptability of our optimal procedure

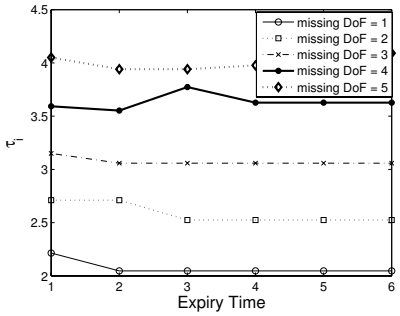


Fig. 10. Optimal values of the cycle time $\tau_i$.

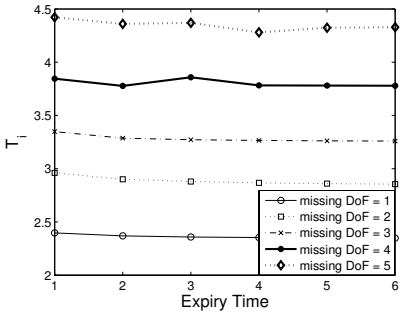


Fig. 11. Optimal values of the mean completion time $T_i$ starting with a missing DoFs $i$.

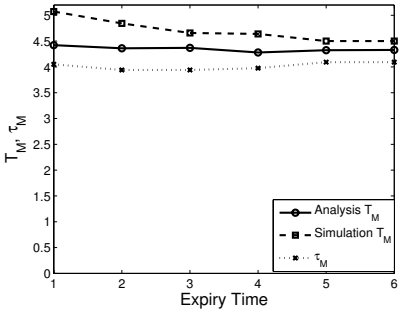


Fig. 12. Optimal values of the file transfer time $T_M$ and comparison with simulation.

to variations in expiry time-out.

Of immense practical interest are the scenarios where: (1) multiple packets per flow are allowed in the relay buffers, and (2) the number of information packets per flow is of the same order of magnitude as the number of nodes in the network. Our reliable transport scheme may be extended to cover such scenarios by combining with suitable buffer management policies. This is part of our ongoing work.

In future, we will focus on the use of coding of packets belonging to multiple flows at the intermediate relays too so as to enable further gains, and hence, better delay performance. Another issue which we would like to investigate is the trade-off between reliability and energy consumption due to the replication of packets and ACKs in the relays.

## REFERENCES


[1] Z. Zhang, "Routing in Intermittently Connected Mobile Ad Hoc Networks and Delay Tolerant Networks: Overview and Challenges," *IEEE Communications Surveys and Tutorials*, vol. 8, no. 1, pp. 24–37, First Quarter 2006.

[2] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic Networking: Data Forwarding in Disconnected Mobile Ad Hoc Networks," *IEEE Communications Magazine*, vol. 44, no. 11, pp. 134–141, November 2006.

[3] G. Holland and N. Vaidya, "Analysis of TCP Performance Over Mobile Ad Hoc Networks," *ACM Wireless Networks*, vol. 8, no. 2, pp. 275–288, March 2002.

[4] M. Ramadas, S. Burleigh, and S. Farrell, "Licklider Transmission Protocol - Motivation," in *RFC-5325*, September 2008.

[5] E. Altman, F. D. Pellegrini, and L. Sassatelli, "Dynamic Control of Coding in Delay Tolerant Networks," in *Infocom*, March 2010.

[6] M. Luby, "LT Codes," in *IEEE FOCS*, 2002, pp. 271–282.

[7] M. A. Shokrollahi, "Raptor Codes," in *IEEE International Symposium on Information Theory*, July 2003.

[8] D. S. Lun, M. Médard, and M. Effros, "On Coding for Reliable Communication Over Packet Networks," in *Proc. 42nd Annual Allerton Conference on Communication, Control, and Computing*, September 2004, pp. 20–29.

[9] K. Scott and S. Burleigh, "Bundle Protocol Specification," in *RFC-5050*, November 2007.

[10] L. Wood, J. McKim, W. Eddy, W. Ivancic, and C. Jackson, "Saratoga: A Scalable File Transfer Protocol," November 13, 2009.

[11] M. Ramadas, S. Burleigh, and S. Farrell, "Licklider Transmission Protocol Specification," in *RFC-5326*, September 2008.

[12] CCSDS, "CCSDS File Delivery Protocol (CFDP)," in *CCSDS 727.0-B-4, Blue Book*, January 2007.

[13] G. Papastergiou, I. Psaras, and V. Tsaoussidis, "Deep-Space Transport Protocol: A Novel Transport Scheme for Space DTNs," *Computer Communications, Special Issue of Computer Communicationson Delay and Disruption Tolerant Networking*, vol. 32, no. 16, p. 17571767, October 2009.

[14] O. B. Akan, J. Fang, and I. F. Akyildiz, "TP-Planet: A Reliable Transport Protocol for InterPlanetary Internet," *IEEE Journal on Selected Areas in Communications(JSAC)*, vol. 22, no. 2, pp. 348–361, 2004.

[15] R. Groenvelt, P. Nain, and G. Koole, "The Message Delay in Mobile Ad Hoc Networks," *Performance Evaluation*, vol. 62, pp. 210–228, 2005.

[16] M. Ibrahim, A. A. Hanbali, and P. Nain, "Delay and Resource Analysis in MANETs in Presence of Throwboxes," *Performance Evaluation*, vol. 24, no. 9–12, pp. 933–945, October 2007.

[17] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance Modeling of Epidemic Routing," *Computer Networks*, vol. 51, pp. 2867–2891, 2007.

[18] D. Lucani, M. Stojanovic, and M. Médard, "Random Linear Network Coding for Time Division Duplexing: When to Stop Talking and Start Listening," in *Infocom*, April 2009.

[19] T. G. Kurtz, "Solutions of Ordinary Differential Equations as Limits of Pure Jump Markov Processes," *Journal of Applied Probability*, vol. 7, no. 1, pp. 49–58, 1970.

[20] K. Price and R. Storn, "Differential Evolution: A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces," *J. Global Optimiz.*, vol. 11, pp. 341–359, 1997.