Manuscript submitted to
AIMS' Journals
Volume **X**, Number **0X**, XX **200X**

doi:10.3934/xx.xx.xx.xx

pp. **X–XX**

THE ONE-OUT-OF-K RETRIEVAL PROBLEM AND LINEAR NETWORK CODING

Abstract

In this paper we show how linear network coding can reduce the number of queries needed to retrieve *one specific message* among k distinct ones replicated across a large number of randomly accessed nodes storing one message each. Without network coding, this would require k queries on average. After proving that no scheme can perform better than a straightforward lower bound of $0.5k$ average queries, we propose and asymptotically evaluate, using mean field arguments, a few example practical schemes, the best of which attains $0.794k$ queries on average. The paper opens two complementary challenges: a systematic analysis of practical schemes so as to identify the best performing ones and design guideline strategies, as well as the need to identify tighter, nontrivial, lower bounds.

1. **Introduction.** This paper introduces a new problem, which we call *one-out-of-k retrieval*. Suppose there are k distinct messages $X = \{x_1, ..., x_k\}$, where $x_i \in \{0,1\}^m$ $\forall i \in [1,k]$ and $i \in \mathbb{Z}^+$. Such messages are assumed to be stored in a possibly distributed *repository*, which we loosely refer to as *sender*. A *receiver* is interested to learn all m bits of one *specific* target message, $x_r \in X$, out of the k available ones. For simplicity, in this paper, we will assume that the message x_r of interest for the receiver is uniformly drawn from the set of available messages. To retrieve such message, the receiver is entitled to receive from the sender exactly one message, or one linear combination of such messages, at every discrete amount of time Δt which we refer to as *round*. Clearly, if the received were able to specify, via an explicit query to the sender (repository), the specifically desired message, one round would trivially suffice to retrieve the message. Rather, the problem becomes interesting when we further assume that i) the receiver cannot specify which message it desires, and ii) the sender cannot store any memory of the past messages sent to a given receiver.

This scenario can be practically encountered in several data dissemination problems. For instance, consider a Delay Tolerant Networks (DTN) where we have a set of users that hold, in their finite memory storage, one message or a linear combination of the available messages, and a receiver that is interested in one specific message. In this scenario, the repository (sender) is the set of all nodes traveling in the network, and a round is a physical encounter of the receiver with another random user in the network. Clearly, if we assume lack of coordination among nodes and random mobility of the target receiver, there is no way for the sender to keep track and/or schedule a desired sequence of messages delivered to the receiver, nor there is possibility for the user to retrieve the desired message by querying it

2010 *Mathematics Subject Classification.* Primary: 58F15, 58F17; Secondary: 53C35.

Key words and phrases. Delay Tolerant Network, Linear Network Coding, Fluid Approximations, Lower Bounds, Mean Field Arguments.

This research is supported by NSF award CCF-1217506 and by the Israel Science Foundation (grant number 1696/14). Keren Censor-Hillel is a Shalon Fellow.

explicitly (the message retrieved at each round is the one stored in the memory of the encountered node).

In essence, the *one-out-of-k retrieval* problem investigated in this paper can be modeled as a data retrieval process of $x_r \in X$: at each round we randomly extract (with replacement) a data from a set Y, until we disclose the value of x_r. The design question revolves around how to construct Y starting from the elements in X in order to minimize the average retrieval delay (measured in number of round) to fetch x_r.

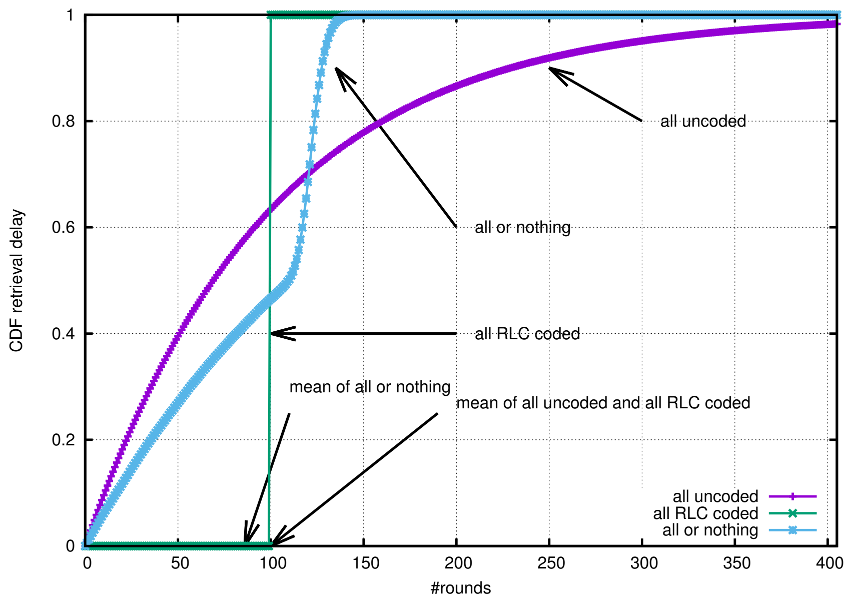


FIGURE 1. Delay distribution for the case of $k = 100$ and three different strategies: uncoded messages, fully coded messages, and "*all or nothing*" (see Section 4)

One obvious approach is to set $Y = X$, i.e. at each round a message x is randomly extracted from the set $X = \{x_1, ..., x_k\}$, and the retrieval process terminates whenever $x = x_r$. It is trivial to show that, with such strategy, owing to the assumption of uniform distribution of the desired message x_r, the average number of rounds required to retrieve the desired message is equal to k. Another obvious alternative approach is to set Y equal to all the possible linear combinations of *all* the messages comprising the set X (in practice, use Random Linear Coding over a large field). However, also in this case[1], the *average* retrieval delay does not change with respect to the first strategy (uncoded messages): as graphically shown in Figure 1, what differs with respect to the uncoded strategy is the distribution of the retrieval delay, but the average remains the same.

Interestingly, as we will show in this work, *there are* coding strategies which permit to *reduce the average delay below k*; in other words, we can construct specific sets Y, modeled by the distribution of the number of message $x \in X$ coded inside

[1]Assuming a large field, i.e. that the probability to receive two linearly dependent codewords is negligible

each element $y \in Y$ (we call this *degree distribution* and formalize it in definition 2.1) that allow the receiver to learn x_r with less than k interactions on average. For instance, Figure 1 shows both delay distribution and relevant average for a strategy which we will later on denote "*all or nothing*" and which involves an appropriate mix between uncoded messages (the original messages $X = \{x_1, ..., x_k\}$) and "fully coded" messages (the random linear combination of all the messages x_i as defined in Definition 4.2). With such strategy, not only the shape of the delay distribution changes, but also its average reduces (specifically, it reduces to the value $0.86k$, see Theorem 4.3).

Contribution

Most network coding research has focused on retrieving and decoding *all* the messages in a set, instead of *one specific* message of the set. If the goal is to retrieve the whole set of k messages, and only one message per round can be retrieved, there is obviously no strategy that permits to reduce the retrieval delay below k rounds (and perhaps for this reason this problem has been mostly neglected in the literature). Conversely, when the goal changes, and becomes that of retrieving one specific message in the set, in principle delay could reduce to as little as a single round (assuming the possibility to query the desired message). The natural question (duly addressed in this paper) is the extent to which the retrieval delay can be reduced using *only* network coding strategies, i.e. without having the possibility to perform selective queries from the receiver side, or to schedule message transmissions on the sender side, as previously discussed.

In this paper, we first show that such an average delay reduction can be attained via proper network coding techniques. Indeed, the trivial example shown in section 2.1, for $k = 2$ has the goal to show that concrete coding schemes with average delay strictly below k do indeed exist. This raises some questions: how much reduction in average numbers of messages from Y can we gain? And with which practical constructions?

We then present a straightforward lower bound of $0.5k$ for the average number of rounds required for obtaining the message by the receiver. Furthermore, we propose some initial example schemes where the selection of the probability distribution over Y results in a lower average number of requests than the naive average of k messages needed from the set Y. We then provide a general methodology to analyze such schemes. We specifically show how to apply mean field arguments to derive the asymptotic performance of the proposed approaches. We concretely apply the methodology to three example schemes, the best of which attains an average of $0.794k$ rounds of communication. The distance between the performance of our best scheme and the loose bound opens new interesting theoretical questions: how to tighten the lower bound and/or how to approach it via improved coding strategies?

Related Work

Previous work on network coding in DTNs has not considered the problem of decoding *one* out of k messages. For instance, LT codes [11] are designed with the different goal of optimizing the decoding procedures. Many papers [14], [13], [16], [2], and [5] investigate routing protocols in DTNs. It has been proved in [16], [5] that Random Linear Nework Coding (RLNC) shows substantial performance advantage over simple replication as it copes with the coupon collection problem. However, most of these papers attempt to decode all the messages, as opposed to just *one* of

k, or start from different hypothesis. For instance [6] presents a scenario where a data collector, who can appear anywhere in the network, querying any k randomly displaced storage nodes, can retrieve *all* the k data packets of interest. Conversely, despite the similarity on the scenario, we are not interested in collecting all the data, but just one out of k data, by querying -on average- the minimum number of storage nodes. This difference leads us to investigate a different analytic approach as the problem does not appear anymore in the form of a classic erasure channel coding problem.

Authors of [5] addressed the problem of *rumor mongering* with network coding: in their problem statement there is a network where each node holds some messages and at each round each node communicates with a random other node with the goal of diffusing the information. They show that using Random Linear Coding can increase the time it takes for *all* the nodes in the network obtain *all* the messages. In this paper's model we have that each node is interested in only one message.

In [16] uses Random Linear Coding on a Delay Tolerant Network. The goal in [16] is also similar to this one: reduce the average message distribution delay. Moreover, we are interested in one-out-of-k packets, while in [16], authors tackle the problem of delivery several packets from different sources to different destinations that can be somehow considered as similar. However, the goal of [16] is to decide how the sources should encode the packet to improve unicast communication. In [16] the authors consider combining all the packets in their buffers, or only the packets destined to the same destination, or only the packets belonging to the same source-destination pair. In other words, [16] analyze the dynamic of a network where each node should decide which operation to perform at each time slot. Instead we focus on the distribution over sent messages. We optimize the initial state, as opposed to improving the diffusion protocols.

A similar difference can be found also in [4] where authors use fountain coding for improving the delay of spreading a message in a DTN.

The paper [10] presents a protocol, called E-NCP, for data dissemination in DTNs that exploits network coding. In particular, the authors of [10] investigated the information-theoretic optimum number of data transmissions and a protocol able to approach to this limit.

This paper is similar to the network coding Spray and Wait work [13], where the authors diffuse some coded data in the network (called a *spray phase*) and than wait for each node data collection (called a *wait phase*).

Several other works, such as [15] or [8], are also focused on protocol design as well: in [8] authors show a collection protocols for sensor networks based on Random Linear Coding, while in [15] the authors show how network coding can help in designing more efficient DTN data diffusion for a collection of protocols that, if compared to epidemic routing, exhibit an higher degree of reliability of packet delivery because of a better nodes buffers usage.

By contrast, in this work we investigate the optimal deployment strategy for displacing data inside a network in order to make a user retrieve one-out-of-k messages with the minimum number of interaction possible. We neglect the real diffusion protocol operations and several other details (such as the impact of the coding vectors size, tackled in [8]) hence focusing more on the coding structure and on the theoretical fundamentals than on the practical implementation and the adaptive network protocol rules.

2. **Network Model and Problem Statement.**

In this paper model there are k messages $X = \{x_1, ..., x_k\}$, each of which can be represented by a binary vector of length m bits. There is a receiver node, r, which wants to know the contents of one message, denoted x_r. In each round, the receiver node r receives exactly one coded message, $y \in Y$, chosen over a given distribution on the set Y. The goal is to construct distributions over the set Y that result in an efficient decoding time for the message x_r. The messages $y \in Y$ are limited to be linear combinations of messages over some large field F_f.

As a simplification hypothesis, in this paper the field is assumed to be large enough so that probability of having two different codewords with the same linear combination (collision) is negligible. Moreover the size of the additional information needed for transmitting the coefficients of the random linear combinations is neglected with respect of the size of the messages.

The physical interpretation in the DTN scenario is the following: suppose having a set of nodes inside an area each of which can store a linear combination $y \in Y$ of some data X. Then suppose having a receiver node that move inside the area, searching for a specific data $x_r \in X$ not known a priori. The receiver gets information from other nodes it contacts in close proximity acquiring a new codeword, and it contact exactly one new node every round. We want to deploy the codewords to all the nodes in the network to minimize the time required by the received to get the data it wants.

Definition 2.1. The type (or degree) of a coded message is the number of messages linearly combined in that data message.

These linear combinations are stored with header data that specifies which messages were summed with what multiplicative constants.

Definition 2.2. Solving for message x_j means determining all m bits in the message x_j.

Definition 2.3. The *one-out-of-k* retrieval problem is a problem of determining what coding scheme produces the lowest expected time for r to solve for x_r, where a coding scheme is the proportion $p_1, p_2, \ldots, p_k$ of the codeword degrees distributed in the network.

In other words, we want to find $p_1, \ldots, p_k$ that minimize the time for retrieving only one message, given that the receiver collects at each round an uncoded message with probability p_1, a "pair" (codeword with degree 2) with probability p_2, a "triplet" with probability p_3, etc.

Thus, Y is the set of all linear combinations of the k messages in X. Each coded message, $y \in Y$, is a linear combination of n messages and has a probability $\frac{p_n}{\binom{k}{n}}$ of being sent to the receiver.

2.1. **A trivial example: the case $k = 2$.**

Consider the simple case where we have only two kinds of different messages that we call A and B. If we do not use coding ($p_1 = 1, p_2 = 0$) it is trivial to show that the average time spent by the receiver for collecting A (or equivalently B) is 2. Similarly, if all nodes carry a random linear combination of both A and B ($p_1 = 0, p_2 = 1$) the expected retrieval time is *exactly* 2 encounters, so once again the average is 2. Now, let AB be the random linear combination of A and B so that at each encounter the receiver can collect A with probability $p/2$, B with probability $p/2$, and AB with probability $1 - p$.

If we are interested in A (equivalently, B), the following cases occur:

- A is retrieved in 1 round if and only if the received message is A (probability $p/2$);
- A is retrieved in 2 rounds if and only if the received message was *not* A, and either:
  - the first message was AB, in this case any further message (i.e., either A, B, or AB) provides the second linear equation needed to determine A;
  - the first message was B and the second message is either A or AB;
- A is retrieved in 3 rounds (and, more generally, in i rounds) if and only if the two previous messages (more generally, $i-1$) were B, and the last encounter is either A or AB;

The average delay to retrieve message A (equivalently, item B) is thus expressed as:

$$\text{Delay} = 1 \cdot \frac{p}{2} + 2 \cdot \left((1-p) \cdot 1 + \frac{p}{2} \cdot \left((1-p) + \frac{p}{2}\right)\right) +$$
$$+ \sum_{i=3}^{\infty} i \cdot \left(\frac{p}{2}\right)^{i-1} \cdot \left((1-p) + \frac{p}{2}\right)$$

After algebraic simplifications, we obtain:

$$\text{Delay} = 1 - p + \frac{1}{1-p/2}$$

It is trivial to show that when $p = 2-\sqrt{2}$ the expected time to retrieve A is minimized and equals to $2\sqrt{2}-1 \approx 1.828$, i.e., about 9% lower than both previous cases. Hence this problem is solved adopting the coding scheme $p_1 = 2-\sqrt{2}, p_2 = \sqrt{2}-1$. The delay versus p is shown in Figure 2.

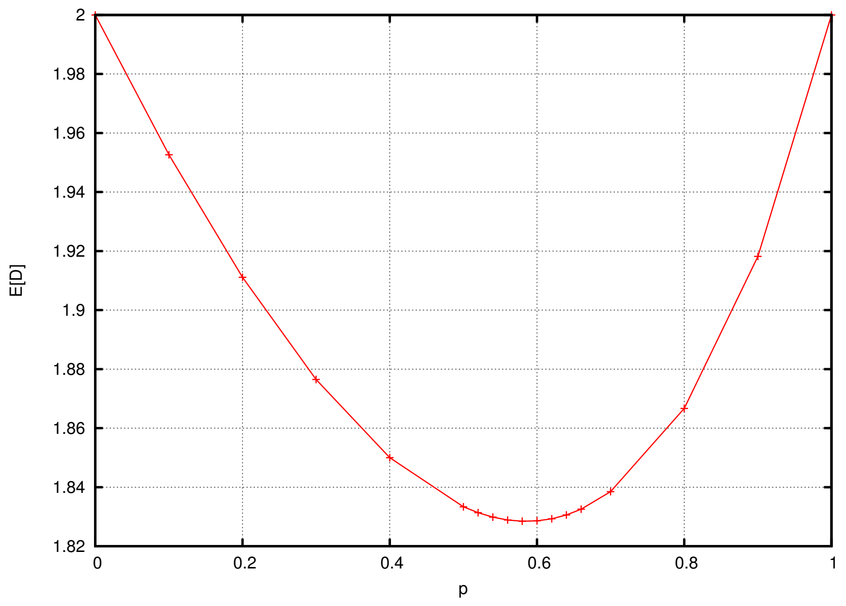


FIGURE 2. Average retrieval delay for the case of $k = 2$

2.2. **Lower bound.** As will be clear, finding a tight lower bound for the *one-out-of-k* is far from being trivial. However, a simple and straightforward lower bound of $k/2$ can be easily achieved in the following way. If we consider k different messages, there exists no coding scheme that allows to decode a generic message before $k/2$ rounds on average.

Theorem 2.4. *A lower bound for the* one-out-of-k *retrieval problem is* $\frac{1}{2}k$.

Proof. Let us consider the process of decoding all the k messages, where the delay of retrieving the first message is $D(x_1)$ and for the i-th message is $D(x_i)$.

Letting $P(i = r)$ be the probability of retrieving the i-th message, the average retrieving delay of a generic message x_r is:

$$E[D(x_r)] = \sum_{i=1}^{k} P(i = r)E[D(x_i)]$$

Given that at each round at most one message can be retrieved, all the k messages are retrieved within at least k rounds, so the average number of rounds to retrieve the i-th message is greater than i.

Finally, considering $P(i = r) = 1/k$ due to the uniform choice of the selected message over all the k messages, we have:

$$E[D(x_r)] \geq \sum_{i=1}^{k} \frac{1}{k} i = \frac{k}{2} + \frac{1}{2}$$

That for large k approaches $k/2$. □

3. **Methodology.** Determining whether the set of received messages fully specifies the target *one-out-of-k* message, is the major difficulty. Since messages are retrieved at random, differently coded messages are collected (e.g. uncoded messages, linear combination of two messages, linear combination of all k messages, and so on depending on the construction). The set of collected messages also depends on time, requiring a *transient* stochastic process to model a chosen strategy, which usually exhibits a non-trivial space state.

To avoid such stochastic modeling complexity, the methodology employed hereafter consists of three steps: i) model a proposed coding strategy via a discrete time (vector) stochastic process; this is arguably the most complex step, as discussed later on; ii) approximate the proposed coding strategy's transient solution with the deterministic mean trajectory specified by the drift (vector) differential equation of a conveniently rescaled stochastic process, and iii) derive the average number of queries needed to retrieve the target message from a relevant probability distribution, which is derived from the knowledge of the drift equation solutions.

The approximation in step (ii) above is motivated by the fact that practical values of k are relatively large. It consists of using mean field techniques widely established in the literature since [9], which have been successfully applied to a variety of problems [1, 3], and which guarantee asymptotic convergence to *exact* results for finite state space systems under mild assumptions (see e.g., [3]). This paper's results show a very accurate matching with simulation even for relatively small values of k.

Details and a simple example of the proposed methodology are presented bellow.

3.1. **Explanation of Details.** Let us assume a discrete time scale, clocked by message arrivals, i.e., time $n \in \{1, 2, \cdots\}$ is defined as the time of arrival of the n-th element. Let us now identify a model for the *receiver state.* This is a critical step (as will appear in the construction examples discussed later on), as the relation between receiver state and the different "types" of messages collected (and how many) is in general not trivial and specific for every scheme considered; for instance, the reception of two different "types" of coded message, say a linear combination of messages A and B (called "pair"), and an uncoded message A (called "singleton") yields the decoding of message B, and suggests to use as state variables the number of message "types" resulting *after* decoding, in this case the two singletons A and B, rather than the actually received message types (a pair an a singleton).

In most generality, the status of the receiver at an arbitrary discrete time n, $n = 1, 2, \ldots$, is summarized by means of a state vector:

$$\overline{\psi}(n) = \{\psi_1(n), \psi_2(n), \cdots\} \tag{1}$$

where $\psi_i(n)$ is defined as the number of messages of "type" i stored by the receiver at time n.

Under the assumption of independent random messages being retrieved at each time step, and appropriate choice of the space state, $\overline{\psi}(n)$ introduced in (1) is a discrete-time Markov chain. Let us write the relevant time-dependent state transition probabilities as functions $f(\cdot)$ of the vector state components normalized with respect to k, i.e.:

$$P\left\{\overline{\psi}(n+1)|\overline{\psi}(n)\right\} = f_{\overline{\psi}(n+1)}\left(\frac{\overline{\psi}(n)}{k}\right) \tag{2}$$

The conditional expectation, namely the *drift* $\bar{d}(\cdot)$ of the considered Markov chain, is readily given by the vector

$$E\left[\overline{\psi}(n+1) - \overline{\psi}(n)|\overline{\psi}(n)\right] = \sum_{\bar{v}} \left(\bar{v} - \overline{\psi}(n)\right) f_{\bar{v}}\left(\frac{\overline{\psi}(n)}{k}\right) = \bar{d}\left(\frac{\overline{\psi}(n)}{k}\right), \tag{3}$$

where we conveniently express the state vector components as normalized with respect to k. We now introduce a new stochastic process which is a *doubly-rescaled* version of (1) in terms of both state (normalized with respect to k, i.e., a *density process* [1]) as well as time (also normalized with respect to k, i.e. $t = n/k$):

$$\bar{\sigma}(t) = \frac{\overline{\psi}\,(t \cdot k)}{k}$$

The conditional expectation (3) is readily rewritten for the rescaled process as:

$$E\left[k \cdot \bar{\sigma}(t+1/k) - k \cdot \bar{\sigma}(t)|\bar{\sigma}(t)\right] = \frac{E\left[\bar{\sigma}(t+1/k) - \bar{\sigma}(t)|\bar{\sigma}(t)\right]}{1/k} = \bar{d}\left(\bar{\sigma}(t)\right) \tag{4}$$

For large k, and under quite general assumptions (it suffices the drift $\bar{d}(.)$ to be a Lipschitz vector field [3]), the density process $\bar{\sigma}(t)$ converges in probability to a deterministic trajectory, computed by solving the system of differential equations obtained by replacing the left hand side of equation (4) with the derivative $\bar{\sigma}'(t)$:

$$\bar{\sigma}'(t) = \bar{d}\left(\bar{\sigma}(t)\right). \tag{5}$$

The average number of messages needed to decode the target message can be readily computed from the knowledge of $\bar{\sigma}(t)$.

3.2. **Example.** In order to clarify this approach, we present a trivial example.

Let us consider the simplest possible case of all messages being uncoded (singletons).

The first step is to define a convenient state space. In this case, the obvious state variable is the number $S(n)$ of distinct singletons received at time n. The process $S(n)$ is a discrete time Markov chain, with the only non null transition probabilities being $P\{S(n+1)=S(n)|S(n)\}=S(n)/k$ (probability that the new retrieved singleton message is already stored), and $P\{S(n+1)=S(n)+1|S(n)\}=1-S(n)/k$ (probability that the retrieved message is a new one). Hence, the drift of the chain is given by $E[S(n+1)-S(n)|S(n)]=1-S(n)/k$.

The second step consists of rescaling the process, and write, for the resulting density process $s(t)=S(tk)/k$, the differential drift equation $s'(t)=1-s(t)$. Since, at time $t=0$, no messages are received, the differential equation shall be solved with the initial condition $s(0)=0$, which yields $s(t)=1-e^{-t}$.

Finally, in order to derive the average number of messages needed to retrieve a randomly chosen target message, we note that $s(t)$ is the fraction of messages retrieved at time t, and hence can be interpreted as the cumulative probability distribution function of the random variable X representing the retrieval (rescaled) time. Thus, $E[X]=\int_0^\infty [1-s(t)]\,dt=\int_0^\infty e^{-t}dt=1$. Rescaling back to the original discrete time scale, we get the final result of k average messages needed to retrieve the target one.

4. **Practical Example Cases.** In order to understand the *asymptotic* nature of the gain, and show how the proposed methodology can be concretely applied we show two example constructions. In both cases, we compare analytical results with simulation.

All-or-nothing scheme. This scheme is extremely simple in terms of states, permits a simple analysis, and can be used as a reference to gauge the improvements brought about by more complex schemes. The *all-or-nothing* scheme comprises only two possible types of messages, defined below.

Definition 4.1. A *singleton* is a message x_i for $i \in [1,k]$ sent in plain text.

Definition 4.2. A *fully coded message* is a random linear combination $\sum_{i=1}^{k} \alpha_i x_i$ of all k messages over a large field size $\mathbb{F}$, with $\alpha_i \in \mathbb{F}$.

We assume that all messages x_i, with $i \in [1,k]$, are equiprobable. Under this assumption, the *all-or-nothing* scheme is characterized by a single parameter p, where p is the singleton reception probability and $1-p$ is the complementary fully coded message reception probability. The state space thus comprises two state variables: i) the number of singletons received at a given time, and ii) the number of fully coded messages received at the same time.

Theorem 4.3. *The* all-or-nothing *scheme achieves the best possible performance of* $0.86k$*; which corresponds to the value* $p \approx 0.626412$*.*

Proof. Using the methodology presented above, let us define the following two density processes:

- $s(t) \in (0,1)$ is the fraction of singletons accumulated until time t;
- $d(t) \in (0,1)$ is the fraction of fully coded messages accumulated until time t.

In this case, the drift differential equation reduces to two independent ordinary differential equations. Let us denote by p the ratio of the singletons to the total number of messages. Hence the probability that we extract a singleton in a round. For the case of singletons, operating in a similar way to the example in Section 3.1, the average increment in the $n+1$-th step is:

$$E[S(n+1)S(n)|S(n)] = p(1 - S(n)/k)$$

that is, we have an increment if a singleton arrives (probability p) and if that singleton does not hit the previous collected singletons (probability $S(n)/k$). By rescaling the process as described in Section 3.1 we obtain:

$$s'(t) = p\,(1 - s(t)) \tag{6}$$

which, when solved with initial conditions $s(0) = 0$, yields

$$s(t) = 1 - e^{-pt}. \tag{7}$$

For the case of fully coded messages, we have for each small Δt the average amount of fully coded messages received is constant and equal to $(1-p)$, hence:

$$d'(t) = (1-p) \qquad \text{with } d(0) = 0. \tag{8}$$

Therefore

$$d(t) = (1-p)t. \tag{9}$$

We now note that a target message is decoded when either the corresponding singleton is received, or when the number of received singletons plus the number of fully coded messages is equal to the total number k of distinct messages. In terms of density processes, this latter condition is expressed by the equation

$$s(t) + d(t) = 1, \text{ which gives} \qquad 1 - e^{-pt} + (1-p)t = 1. \tag{10}$$

Let we call t^* the solution of the previous transcendental equation. Then we can express t^* in closed form as

$$t^* = \frac{W(\frac{p}{1-p})}{p} \tag{11}$$

by introducing the Lambert W function defined as the solution W of the transcendental equation $g(W) = We^W$.

Finally, the average number of messages $E[X]$ needed to decode the target message can be computed as usual by integrating the Complementary Cumulative Density Function (CCDF) of the probability of retrieving the target message at time t in the interval $[0, \infty)$. The CCDF of the retrieval time $D(x_r)$ is given by:

$$P(D(x_r) > t) = \begin{cases} 1 - s(t) & \text{if } t \leq t^* \\ 0 & \text{otherwise} \end{cases}$$

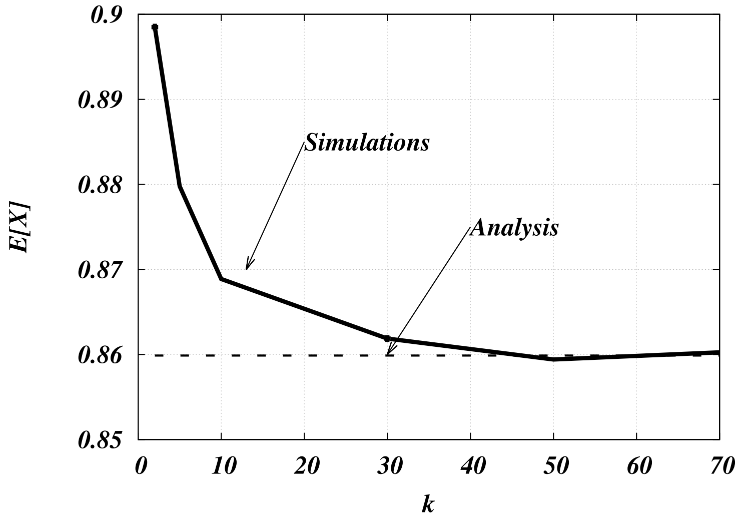
Indeed, for $t \leq t^*$ only the singletons count, while the contribution of the fully coded message received is given at time $t = t^*$ where all the remaining un-decoded messages will be decoded at once.

By combining equations 7 and 11, the resulting expression of average delay is than:

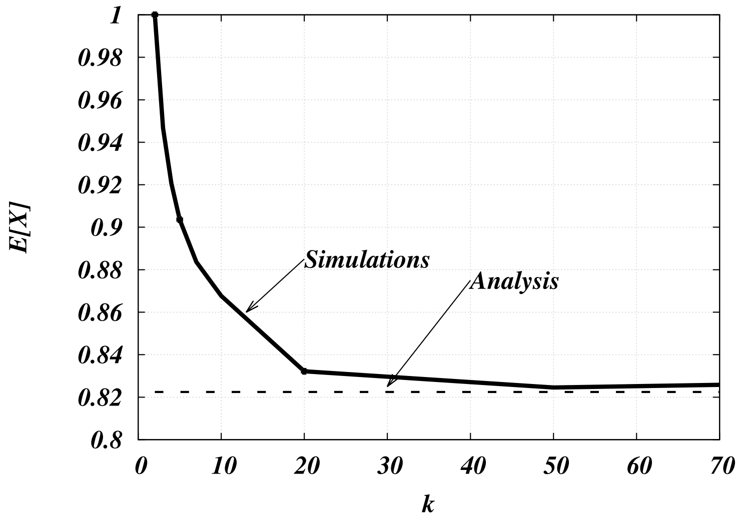
$$E[X] = \int_0^{t^*} 1 - s(\tau)d\tau = \frac{1 - e^{-W(\frac{p}{1-p})}}{p} \tag{12}$$

This expression is minimized when $p = 0.626412$, and yields a minimum (normalized) number of retrieved messages $E[X] = 0.859884$.

□



(a) All-or-Nothing



(b) Pairs

FIGURE 3. Average retrieval delay varying the number of messages: mean field approximation vs simulation.

In order to verify the correctness of the analysis, Figure 3-a shows that simulations vary the number of messages from $k = 2$ to $k = 70$. Note that the theoretical results have an asymptotic nature, hence the choice of running simulations with small values of k. Every point in the figure is the delay to retrieve a data message averaged on 50000 samples. Even though the proposed methodology obtains an exact solution only for large values of k, already after $k = 20$ the error is below 1%.

Pairs-only scheme. This scheme shows how the state space can become extremely complex (actually an infinite set of state variables) even when considering an apparently very simple approach. Moreover, it can be solved using an alternative methodology, because its emerging decoding structure can be cast as an Erdös-Rényi random graph; thus it permits us to verify that the methodology, despite being extended to the case of infinite state variables (hence violating the assumptions in [3]), nevertheless yields the same results derived in the relevant random graph literature [7].

As the name suggests, the *pairs-only* scheme includes only one type of coded message, namely the random linear combination of two randomly chosen messages. This type of message is called *pair* and is formally defined as follows.

Definition 4.4. A *pair* is a random linear combination of two randomly chosen messages over a field of large size $\mathbb{F}$ in the form $\{(\alpha x_i + \beta x_j) | i \neq j \text{ and } i, j \in [1, k]\}$ where $\alpha, \beta \in \mathbb{F}$.

In analyzing this scheme, the difficulty lies in defining an appropriate state space. Once this is done, the remaining analysis reduces to the conceptually straightforward application of the methodology. The state space definition and justification is presented in Section 5, along with the proof of the following theorem:

Theorem 4.5. *The* pairs-only *scheme achieves a performance of* $\frac{\pi^2}{12}k \approx 0.8224k$.

These results confirm those found in random graphs literature. However, the approach can be extended to coding schemes which cannot be directly cast as a random graph problem, such as, the combination of singletons and pairs, which yields a performance slightly below $0.8k$ (we postpone analysis to a later extended version of this work). Comparison with simulation results averaged over 50.000 realizations is reported in Figure 3-b. Again, results show that convergence to the asymptotic result is very fast, with an error lower than 1% for $k > 20$.

5. Pairs.

To avoid an overly long presentation, we directly operate over re-scaled state variables, i.e., densities (the transformation from discrete state variables to densities being readily performed as in the example presented in Section 3.2).

Since pairs are selected at random, the tracking of all the possible combination of messages would yield state space explosion. To circumvent such issue, we resort to the following convenient definition of an infinite, but numerable, set of state variables $s_i(t)$, where

- $s_1(t)$ is the fraction of messages (normalized with respect to k), which, at (normalized) time t, do *not* belong to any so far received pair;
- $s_2(t)$ is the fraction of messages which are covered by one and only one pair;
- $s_3(t)$ is the fraction of messages which belong to a group of three messages "connected" by two pairs[2];
- and, in most generality, $s_i(t)$ is the fraction of messages which belong to a group of i messages "connected" by $i-1$ pairs.

For an illustrative example, assume the node has so far received the pairs AB, AC, AD, EF, FG, HI, and JK. According to the definition, we have 1 group of 4 "connected" messages (A, B, C, D), 1 group of three connected messages (E,F,G), two groups of two connected messages (H,I) and (J,K), and all remaining messages

[2] we define x messages as "connected" by y pairs if each message is coded in at least one of the y pairs

not yet covered by any pair. Being k the total number of distinct messages, the state representation for the above example would be: $\{s_1(t) = 1 - 11/k, s_2(t) = 4/k, s_3(t) = 3/k, s_4(t) = 4/k, s_5(t) = 0, \cdots\}$. Note that $s_i(t) \cdot k/i$ yields the number of groups having cardinality i.

Suppose now that a pair AI is received: as a result, the two groups (A,B,C,D) and (H,I) merge in a new group of cardinality 6. This corresponds to the transition to the following state: $\{s_1(t) = 1 - 11/k, s_2(t) = 2/k, s_3(t) = 3/k, s_4(t) = 0, s_5(t) = 0, s_6(t) = 6/k, \cdots\}$.

For $k \to \infty$, the probability that a pair arrives in an already formed group *of finite size* vanishes; as such, a state transition can occur only because two different groups are merged via a random pair arrival. We can thus write the drift differential equations as follows:

$$\begin{array}{rcl}
s_1'(t) & = & -2s_1(t) \\
s_2'(t) & = & 2\left[-2s_2(t) + s_1(t)^2\right] \\
s_3'(t) & = & 3\left[-2s_3(t) + s_1(t)s_2(t) + s_2(t)s_1(t)\right] \\
s_4'(t) & = & 4\left[-2s_4(t) + s_1(t)s_3(t) + s_2(t)^2 + s_3(t)s_1(t)\right] \\
\cdots & = & \cdots \\
s_i'(t) & = & i\left[-2s_i(t) + \displaystyle\sum_{j=1}^{i-1} s_j(t)s_{i-j}(t)\right] \\
\cdots & = & \cdots
\end{array} \tag{13}$$

These equations are readily explained as follows. Let us first focus on the set of messages so far not yet covered by any pair, i.e. those accounted by the state variable $s_1(t)$. Let us also remark that, owing to the normalization, $s_1(t)$ *also* corresponds to the probability to pick one of such messages as a component of an arriving pair. A state transition involving s_1 thus comprises two possible cases: i) with probability $s_1(t)^2$, an arriving pair removes two of such messages and add them to the group of non overlapping pairs, namely those accounted in the state variable s_2, or ii) with probability $2s_1(t) \cdot (1 - s_1(t))$ only one of the messages is removed. This corresponds to a negative *drift* for the state variable $s_1(t)$ given by the *average* state variable decrement:

$$s_1'(t) = -2 \cdot s_1(t)^2 - 2s_1(t)(1 - s_1(t)) = -2s_1(t),$$

as stated by the first equation in the above system.

Let us now focus on the set of messages accounted by the state $s_2(t)$. We recall that these are messages covered by exactly one pair. On one hand, $s_2(t)$ can increase, with the addition of two new messages, only when an arriving pair covers two messages belonging to the set s_1 (this occurs with probability $s_1(t)^2$ as discussed above). On the other hand, it decreases of i) four messages, whenever a new arriving pair "hits" two messages in the set s_2 (hence "connects" the two pre-existing pairs forming a 4-messages group, this event has probability $s_2(t)^2$), or ii) connects one pair in s_2 with a message outside the set s_2, this event occurs with probability $2s_2(t) \cdot (1 - s_2(t))$. By averaging the resulting state variations, we obtain the second drift equation. The remaining equations are derived via identical considerations.

It only remains to solve this differential system, using initial conditions $s_1(0) = 1$, $s_i(0) = 0, \forall i > 1$. This is a purely calculus problem, not anymore related to this specific modeling problem, that can be addressed as follows. First, we note that equations can be solved recursively, starting from the top. The following set of solutions is readily obtained:

$$\begin{aligned} s_1(t) &= e^{-2t} \\ s_2(t) &= 2e^{-4t}t \\ s_3(t) &= 6e^{-6t}t^2 \\ s_4(t) &= \frac{64}{3}e^{-8t}t^3 \\ s_5(t) &= \frac{250}{3}e^{-10t}t^4 \\ s_6(t) &= \frac{1728}{5}e^{-12t}t^5 \\ &\cdots \end{aligned} \tag{14}$$

Where the general solution pattern for $i = 1, 2, \ldots$ can be easily determined, besides a multiplicative constant C_i, as:

$$s_i(t) = C_i e^{-2it} t^{i-1} \tag{15}$$

Given that we are interested in the sum of all the $s_i(t)$ we can easily recognize that:

$$\sum_{i=1}^{\infty} s_i(t) = t^{-1} \sum_{i=1}^{\infty} C_i \left(e^{2t}t^{-1}\right)^{-i} = t^{-1} C_z \left(e^{2t}t^{-1}\right) \tag{16}$$

where $C_z(e^{2t}t^{-1})$ is the Z-Trasform [12] of sequence C_i calculated in the point $e^{2t}t^{-1}$.

By combining eq. (13) and eq. (15) we obtain:

$$\begin{aligned} C_i e^{-2it} t^{i-2}(i - 1 - 2it) &= \\ -2iC_i e^{-2it} t^{i-1} + e^{-2it} t^{i-2} i \sum_{j=1}^{i-1} C_j C_{i-j} \end{aligned}$$

After algebraic simplifications, the latter becomes:

$$C_i(i-1) = i \sum_{j=1}^{i-1} C_j C_{i-j}$$

We can transform this equation using the Z-Transform on i, so that:

$$-C_z - zC'_z = -z(C_z^2)'$$

and finally:

$$C_z = 2zC_zC'_z - zC'_z$$

Solving the above differential equation in z we have:

$$C_z = -\frac{1}{2} W\left(-\frac{2e^{-P}}{z}\right) \tag{17}$$

where W is the Lambert Function and P is a constant.
If we Z-antitransform the last expression, and after that we set the constant P (knowing that $C_1 = 1$), we have:

$$C_i = \frac{(2i)^{i-1}}{i!} \tag{18}$$

C_i assumes the following values : $1, 2, 6, 64/3, 250/3, 1728/5, 67228/45, 2097152/315, 1062882/35, 80000000/567\ldots$
We can then express $s_i(t)$ as:

$$s_i(t) = \frac{(2i)^{i-1}}{i!} t^{i-1} e^{-2it}$$

The average delay is readily obtained by integrating in the interval $[0, \infty)$ the probability that we do not have decoded the target message (CCDF). The CCDF corresponds to $\sum_{i=1}^{\infty} s_i(t)$ that is the fraction of message not yet decoded at time t. Thus, we can finally calculate the average delay $E[D]$ as:

$$E[D] = \sum_{i=1}^{\infty} \int_{t=0}^{\infty} s_i(t) = \sum_{i=1}^{\infty} \frac{1}{2i^2} = \frac{\pi^2}{12}$$

This is equal to $0.822467k$.

If we analyse the timeline of decoding process by observing the behavior of the CDF (or equivalently of the CCDF), we can notice a sharp threshold (corresponding to the receiving of $k/2$ pairs, i.e. $t = k/2$) that separates a phase in which the decoding of messages is negligible from a phase in which it is significant, as represented in Figure 4. Indeed, this problem can also be modelled as a random graph with k vertices where we randomly add edges upon receiving a new codeword. When we have a cycle we can decode all the messages corresponding to the vertices connected by the cycle. Initially, if the number of vertices is very large, the probability of having a cycle is negligible. However, as Erdős pointed out in his seminal paper [7], after that vertices reach a degree $c = 1$ there is the emerging of a "giant component" whose size in the supercritical part (i.e. $c > 1$) is $\sim y(c)k$ where y is the solution of $e^{-cy} = 1 - y$.

6. **Hybrid approach: singletons and pairs.**

By using the same methodology, we consider an hybrid approach where we mix singletons and pairs. We denote by p the ratio of the number of singletons divided by the total number of messages, hence $q = 1 - p$ represents the probability that the arriving codeword is a pair.

As in the previous case, we define a set of state variables $s_i(t)$ such that:

- $s_1(t)$ is the fraction of of messages (normalized with respect to k) which, at (normalized) time t, do not belong to any received pair or singleton;
- $s_2(t)$ is the fraction of messages which are covered by one and only one pair;
- $s_3(t)$ is the fraction of messages which belong to a group of three messages "connected" by two pairs; and, in most generality, $s_i(t)$ is the fraction of messages which belong to a group of i messages "connected" by $i - 1$ pairs

Next, we find the optimum combination that corresponds to the optimum value of p that minimizes the expected retrieval delay.
We start defining the drift equations as follows:

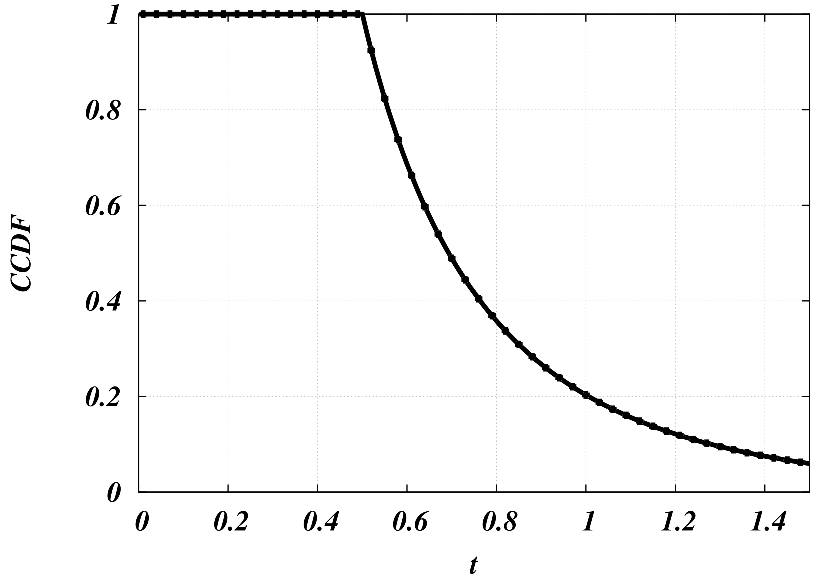


FIGURE 4. Behavior of the CCDF of the average retrieval delay with a pair-only scheme, corresponding to the expression $\sum_{i=1}^{\infty} s_i(t)$. Before $t = 1/2$ we decode very few messages, after that threshold we start decoding a consistently fraction of messages.

$$
\begin{array}{rcl}
s_1'(t) & = & -ps_1(t) - 2qs_1(t)^2 - 2qs_1(t)\left[1 - s_1(t)\right] \\
s_2'(t) & = & -2ps_2(t) - q\left\{4s_2(t)^2 + 4s_2(t)\left[1 - s_2(t)\right]\right\} + \\
 & & 2qs_1(t)^2 \\
s_3'(t) & = & -3ps_3(t) - q\left\{6s_3(t)^2 + 6s_3(t)\left[1 - s_3(t)\right]\right\} + \\
 & & 3q\left[s_1(t)s_2(t) + s_2(t)s_1(t)\right] \\
\cdots & &
\end{array} \tag{19}
$$

These equations describe the behavior of the state variables during the time, when a new singleton or a new pair arrives.

For instance $s_1(t)$: i) decreases by *one* data message when a degree-1 message arrives (this happens with probability p) and hits the fraction of uncovered messages (this happens with probability $s(t)$) ii) decreases by *two* data messages when a pair arrives (probability q) and both the elements of that pair hit the fraction of uncovered messages (probability $s_1(t)^2$), or iii) decreases by *one* data message when a pair arrives and only one of its element hits $s_1(t)$ (this happens with probability $2s_1(t)(1 - s_1(t))$).

Similarly $s_2(t)$: i) decrease by *two* data messages if a singleton arrives on a already collected pair (probability $ps_2(t)$) because we have enough information to reconstruct both the messages encoded in that pair; otherwise, if we get a pair, we can have two different case: ii-a) a pair connects two other collected pairs and then we decrease $s_2(t)$ by *four* data messages or ii-b) a pair connects one collected pair with another element (probability $2s_2(t)(1-s_2(t))$) then we decrease $s_2(t)$ by *two* data message; finally we increase $s_2(t)$ by *two* messages if a new pair hits on two previously unassigned elements (probability $s_1(t)^2$).

By setting the initial conditions $s_1(0) = 1$ and $s_i(0) = 0, \forall i > 1$, we can recursively solve the equations 19, obtaining the following set of solutions:

$$\begin{aligned} s_1(t) &= e^{(-2+p)t} \\ s_2(t) &= -2e^{2(-2+p)t}(-1+p)t \\ s_3(t) &= 6e^{3(-2+p)t}(-1+p)^2t^2 \\ \cdots \end{aligned} \tag{20}$$

Then we can easily derive the generic form of the i-th state variable as:

$$s_i(t) = \frac{(2i)^{i-1}}{i!} e^{-i(2-p)t} \left[(1-p)t\right]^{i-1} \tag{21}$$

where we derive the coefficients as described in Section 5.

Finally, we can compute the average retrieval delay by integrating the time in the interval $[0, \infty)$ and summing over all the components i in $[1, \infty)$. The result is:

$$\begin{aligned} E[D] &= \sum_{i=1}^{\infty} \int_{t=0}^{\infty} s_i(t) \\ &= \frac{Li_2\left(\frac{2(p-1)}{p-2}\right)}{2(1-p)} \end{aligned} \tag{22}$$

Where Li_2 is the polylogarithm function of order 2.

Minimizing $E[D]$ with respect to p we find out the optimal mix for $p = 0.155474$ that corresponds to an average retrieval delay of $E[D] = 0.793933k$

7. **Conclusion.**

In this work we introduce a problem called *one-out-of-k retrieval*, and we show how network coding techniques can help in reducing the average retrieval delay for a desired message out of a set of k available ones. We specifically prove a straightforward lower bound of $0.5k$ and we propose practical coding constructions, the best of which asymptotically reaches an average delay equal to $0.794k$. The significant gap between the lower bound and the performance of the considered approaches calls for the need to identify tighter lower bounds or improved strategies, a challenging foundational problem which we leave to future work.

Many interesting avenues for investigation consist in relaxing the quite strict assumptions employed in this work. Indeed, mainly motivated by the desire to understand the basic foundational issues emerging in the *one-out-of-k retrieval*, the system model considered in this paper is very simplistic and abstract, and can be generalized in many directions and/or more closely cast to more specific network scenarios. First, this model assumes that exactly one message can be retrieved at each round, whereas, for instance in a concrete DTN setting, a way more realistic assumption is to permit nodes to store and exchange multiple messages, as well as to *relay* messages among nodes. Furthermore, we have assumed independent retrieval rounds whereas in a realistic settings either message relaying as well as contacts among nodes may entail some sort of graph structure which can be exploited using approaches similar to [16], although cast into our different *one-out-of-k retrieval* problem. Moreover, we assumed no memory or correlation in the retrieval of messages: by permitting (some level of) scheduling of the delivered messages, results are deemed to significantly improve; for instance, the $0.5k$ bound can be practically achieved by scheduling the delivery of messages in a strict sequence.

Finally, a complementary interesting research direction consists in retaining the basic system model, but generalizing it to account for a non uniform probability distribution of the receiver's interests, to better fit with more realistic content retrieval scenarios where popularity of content is is far from being uniform. Also, moving from the *one-out-of-k retrieval* to the *m-out-of-k retrieval* could be another interesting extension.

REFERENCES

[1] F. M. Buckley and P. K. Pollett. Limit theorems for discrete-time metapopulation models. *Probability Surveys*, 7:53–83, 2010.

[2] L. Chen, T. Ho, S.H. Low, M. Chiang, and J.C. Doyle. Optimization based rate control for multi-cast with network coding. In *Proc. IEEE Infocom*, 2007.

[3] R. W. R. Darlings and J. R. Norris. Differential equation approximations for markov chains. *Probability Surveys*, 5:37–79, 2008.

[4] F. De Pellegrini, R. El-Azouzi, and F. Albini. Interplay of contact times, fragmentation and coding in dtns. In *Modeling Optimization in Mobile, Ad Hoc Wireless Networks (WiOpt), 2013 11th International Symposium on*, pages 580–587, May 2013.

[5] S. Deb, M. Medard, and C. Choute. Algebraic gossip: a network coding approach to optimal multiple rumor mongering. *Information Theory, IEEE Transactions on*, 52(6):2486–2507, June 2006.

[6] A.G. Dimakis, V. Prabhakaran, and K. Ramchandran. Decentralized erasure codes for distributed networked storage. *Information Theory, IEEE Transactions on*, 52(6):2809–2816, June 2006.

[7] P. Erdős and A. Rényi. On the evolution of random graphs. In *Publication of the mathematical institute of the Hungarian Academy of Sciences*, pages 17–61, 1960.

[8] L. Keller, E. Atsan, K. Argyraki, and C. Fragouli. Sensecode: Network coding for reliable sensor networks. *ACM Trans. Sen. Netw.*, 9(2):25:1–25:20, April 2013.

[9] T. G. Kurtz. Solutions of ordinary differential equations as limits of pure jump markov processes. *Journal of Applied Probability*, 7(1):49–58, 1970.

[10] Y. Lin, B. Li, and B. Liang. Efficient network coded data transmissions in disruption tolerant networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 2180–2188, April 2008.

[11] M. Luby. LT codes. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 271–280, 2002.

[12] Alan V Oppenheim, Ronald W Schafer, John R Buck, et al. *Discrete-time signal processing*, volume 2. Prentice-hall Englewood Cliffs, 1989.

[13] L. Sassatelli and M. Medard. Inter-session network coding in delay-tolerant networks under spray-and-wait routing. *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 10:103 – 110, 2012.

[14] J. Widmer and J.Y. Le Boudec. Network coding for efficient communication in extreme networks. *ACM SIGCOMM workshop on Delay-tolerant networking*, pages 284–291, 2005.

[15] S. K. Yoon and Z. J. Haas. Application of linear network coding in delay tolerant networks. *IEEE Ubiquitous and Future Networks (ICUFN)*, pages 338–343, 2010.

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

E-mail address: giuseppe.bianchi@uniroma2.it
E-mail address: lorenzo.bracciale@uniroma2.it
E-mail address: ckeren@cs.technion.ac.il
E-mail address: andreali@mit.edu
E-mail address: medard@mit.edu