Time-efficient randomized multiple-message broadcast in radio networks

Majid Khabbazian[*]
Department of Applied Computer Science
University of Winnipeg, Canada
email: m.khabbazian@uwinnipeg.ca

Dariusz R. Kowalski[†]
Department of Computer Science
University of Liverpool, United Kingdom
email: D.Kowalski@liverpool.ac.uk

ABSTRACT

Multiple-message broadcast, or k-broadcast, is one of the fundamental problems in network communication. In short, there are k packets distributed across the network, each of them has to be delivered to all other nodes. We consider this task in the model of multi-hop radio network, in which n nodes interact by transmitting and receiving messages. A message transmitted at a round reaches all neighbors of the transmitter at the end of the same round, but may not be successfully received by some, or even all, of these neighbors. More specifically, a node receives a message at a round if this is the only message that has reached this node in this round. Due to this specific interference-prone nature of radio networks, many communication tasks become more challenging and more costly than in other types of networks, especially in ad-hoc setting in which each node knows only its own id and linear estimates on the basic network parameters, such as the number of nodes n, diameter D and maximum node degree Δ. We design a new randomized k-broadcast algorithm combining the best of two worlds: efficient randomized transmission schedules and network coding. We show that our algorithm accomplishes multi-broadcast in $O(\log \Delta)$ amortized number of communication rounds per packet, with high probability. This improves over the best previous solution of Bar-Yehuda, Israeli and Itai [5], which guarantees only $O(\log \Delta \log n)$ of amortized number of rounds per packet, with high probability.

*The work of this author was supported by the Natural Sciences and Engineering Research Council of Canada.
†The work of this author was supported by the Engineering and Physical Sciences Research Council [grant numbers EP/G023018/1, EP/H018816/1].

PODC'11, June 6–8, 2011, San Jose, California, USA.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Computations on discrete structures*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Network problems*

General Terms

Algorithms, Theory

Keywords

radio networks, multiple-message broadcast, randomized algorithm

1. INTRODUCTION

In this work we study the problem of simultaneous broadcasting of k packets, distributed arbitrarily across the network, to all nodes in the underlying radio network. This problem is called *multiple-message broadcast*. The underlying radio network is represented by an undirected graph $G(V, E)$. Let $|V| = n$, $n \geq 2$, be the number of nodes in G, D be the network diameter, and Δ be the maximum node degree. We assume that every node knows only its own id and the basic network parameters: n, Δ and D.[1] Let k, $k \geq 1$, be the number of packets to be delivered to all other nodes in the network; this parameter is not a priori known to the nodes.[2] We denote by b the maximum size of a packet, measured in the number of bits. Since each packet specification includes at least one ID, we assume that $b \geq \log n$ and the length of each transmitted message is $\mathcal{O}(b)$. We assume that every node that initially possesses a packet wakes up at time 0, that is, before the first round. Other nodes wake up as soon as they receive a message.

Radio network is one of the classic models of wireless networks. In this setting, n nodes interact with each

^{1}In fact, throughout the paper we use slightly weaker assumptions that nodes only need to know a polynomial upper bound on n and Δ, and a linear upper bound on D.
^{2}We implicitly assume that k is bounded from above by some polynomial in n, for the sake of simplicity of algorithm design and analysis. If the real number of packets is larger than some fixed polynomial, nodes locally split the initially stored packets and apply the solution for polynomial number of packets separately for each group.

other by transmitting and receiving messages. A message transmitted at a round reaches all neighbors of the transmitter at the end of the same round, but may not be successfully received by some, or even all, of these neighbors. More specifically, a node receives a message at a round if this is the only message that has reached this node in this round, i.e., exactly one neighbor of this node has transmitted at the current round. This implies that we do not assume a collision detection mechanism hardwired at nodes, in the sense that the system does not provide a feedback to a node that allows to distinguishing between the following two cases: none of the neighbors has transmitted, or at least two neighbors have transmitted.

Our contribution. In this paper, we propose a multiple-message broadcast algorithm and show that using this algorithm, every node receives all k packets by time $\mathcal{O}(k\log\Delta + (D+\log n)\log n\log\Delta)$, with high probability. Our solution efficiently combines the advantages of randomized techniques and simple coding in the context of radio networks. This improves over the best known result of Bar-Yehuda, Israeli and Itai [5], which guaranteed only $\mathcal{O}(k\log n\log\Delta + (D+n/\log n)\log n\log\Delta)$ completion time in expectation. In terms of per-packet amortized complexity, our solution guarantees packet delivery in amortized number of $\mathcal{O}(\log\Delta)$ rounds, with high probability, which is a substantial improvement over the previous $\mathcal{O}(\log\Delta\log n)$ amortized number of rounds per packet, in expectation. Our solution could be efficiently used as a building block of various applications, including update of routing tables, learning topology of the underlying network (in order to benefit from efficiency of centralized solutions), aggregating functions in sensor networks, etc. In all these cases, the inherited average cost per amount of information is only $\mathcal{O}(\log\Delta)$, with high probability.

Previous and related work. Algorithmic aspects of radio communication have been widely studied in the last two decades. The formal model for multi-hop radio networks was introduced in the context of broadcasting by Chlamtac and Kutten [6]. In the early stage, the research in this area focused on basic communication primitives, such as already mentioned broadcast [1, 4] and point-to-point communication [5].

Bar-Yehuda et al. [5] were the first who considered a multiple-message broadcast in radio networks. They designed the algorithm accomplishing this task in

$$\mathcal{O}(k\log n\log\Delta + (D+n/\log n)\log n\log\Delta)$$

rounds, in expectation. Khabbazian et al. [16] presented a modular approach to broadcasting in radio networks, and devised a multiple-message broadcast algorithm in this framework working in $\mathcal{O}((k\Delta\log n + D)\log\Delta)$ rounds, with high probability. The best known lower bound for randomized solutions is $\Omega(k+\log(n/D))$, in expectation [9, 19]. The best known deterministic solutions to the problem of multiple-message broadcast in ad-hoc radio networks works in time

$$\mathcal{O}(\min\{(k+n)n^{1/2}\log^3 n, (k+n^{5/2})\log^3 n\})$$

[10, 14]. On the other hand, the best known lower bound for the time complexity of a deterministic solution is $\Omega(k+n\log n)$ [9, 10, 12, 15, 17]. A lower bound $\Omega(n\log n)$ on the length of broadcast schedule in case $k=n$ was proved by Gasieniec and Potapov [15], and it holds for any, even randomized algorithms (though this result holds under assumption that the nodes cannot look into the content of the packets).

Bar-Yehuda et al. [4] designed a randomized broadcast algorithm and showed that it accomplishes broadcasting of a single packet in $\mathcal{O}((D+\log n)\log\Delta)$ rounds, with high probability. This result was later improved to $\mathcal{O}((D+\log n)\log(\min\{\Delta, n/D\}))$ independently by Kowalski and Pelc [17] and Czumaj and Rytter [13]. The matching lower bound was proved by Kushilevitz and Mansour [19]. Chlebus et al. [9] considered a related problem of many-to-many group communication, in which a subset of awaken nodes needs to exchange messages. In ad-hoc setting they showed

$$\mathcal{O}\big((d+\log p)\log^2 n + p\log p\big)$$

time complexity, with high probability, where p is the number of awaken nodes and d is the maximum distance between any two such nodes. Related problems of point-to-point communication and all-to-all communication in radio networks were also considered in both deterministic and randomized setting, c.f., [5, 11, 12, 13].

1.1 Mathematical Preliminaries

In the analysis of our main result we will use the following three facts. The first two lemmas estimate the probability of concentration of the sum of independent Bernoulli trials and random variables of geometric distribution, respectively (we will refer to them as Chernoff-type inequalities). The third lemma estimates the probability of achieving a full rank by a matrix with randomly chosen binary entries. These results can be easily derived from other similar results, such as presented in [2], but for the sake of completeness we attach their proofs in Appendix A.

LEMMA 1. **(Chernoff-type bound for the sum of Bernoulli trials)** *Let $Y_q, q=1,\ldots$ be a collection of independent $\{0,1\}$-valued random variables, each equal to 1 with probability $p>0$. Let d and τ be nonnegative reals, $d\geq 1$. Let $r=\lfloor\frac{1}{p}(3d+2\tau)\rfloor$. Then*

$$Pr(\sum_{q=1}^{r} Y_q < d) \leq e^{-\tau}.$$

LEMMA 2. **(Chernoff-type bound for the sum of geometric random variables)** *For $i=1,\ldots,k$, let X_i be independent geometric random variables with parameter p_i. Further, we define $X=\sum_{i=1}^{k} X_i$, $\mu=\mathbb{E}[X]=$*

$\sum_{i=1}^{k} 1/p_i$, *and* $p_{\min} = \min_{i\in[k]} p_i$. *For every* $\epsilon > 0$, *it holds that*

$$\Pr\left(X \geq 2\mu + \frac{4\ln(1/\epsilon)}{p_{\min}}\right) \leq \epsilon.$$

LEMMA 3. **(Rank of a random binary matrix)** *Let* l *and* w *be positive integers. Suppose* $l \geq 2(w+2) + 8\ln(\frac{1}{\epsilon})$. *Let* $A = (a_{i,j})$ *be a* $(l \times w)$*-binary-matrix with elements independently and identically distributed as:*

$$Pr(a_{i,j} = 0) = Pr(a_{i,j} = 1) = \frac{1}{2}.$$

Then, A *has full rank with probability at least* $1 - \epsilon$.

We will often use term *with high probability* (or *w.h.p.* for short) to denote probability $1 - n^{-c}$, for a sufficiently large constant $c > 1$.

2. MULTIPLE-MESSAGE BROADCAST ALGORITHM

Our proposed multiple-message broadcast algorithm consists of the following consecutive stages:

Stage 1: Leader election:
In this stage, among the nodes that have at least one packet, the node with the highest ID is selected as a leader.[3] This stage will take

$$\mathcal{O}((D + \log n)\log n \log \Delta)$$

rounds. By the end of this stage a leader is elected, w.h.p.

Stage 2: Construction of distributed BFS:
In this stage, a BFS tree with the leader as the root is constructed. The stage consists of D phases of $\mathcal{O}(\log n \log \Delta)$ rounds each. By the end of each phase d, where $1 \leq d \leq D$, every node at distance d from the root determines its parent in the tree as well as its distance to the root, w.h.p.

Stage 3: Packet collection:
In each phase of this stage nodes estimate the total number of packets (i.e., k) and unicast their packets to the root (along the BFS path) with some random delays. Recall that Stage 2 guarantees that each node knows its parent in the BFS path to the root. We show that in $\mathcal{O}(k + (D + \log n)\log n)$ rounds, the root collects all k packets, w.h.p.

Stage 4: Packet dissemination:
Recall that in the beginning of this stage, the root has all the packets, w.h.p. In this stage, the root broadcasts all k packets in time

$$\mathcal{O}(k \log \Delta + D \log n \log \Delta)$$

using network coding.

[3]The reason we choose a leader from the nodes who have at least one packet is to avoid waking up other nodes if not necessary. Note however that some nodes with no packet can be involved in the next stages of the protocol to relay messages if necessary.

In the remainder of this section we describe the details of each stage of our algorithm.

2.1 Stage 1: Leader election

We say a multiple-access channel has capability of collision detection if devices using that channel can distinguish the case of no transmission from the case of multi-transmissions (i.e., transmission of more than one neighbor). Bar-Yehuda et al. [3] showed how to emulate one round of an algorithm specified for a multiple-access channel with capability of collision detection on a multi-hop radio network without collision detection, w.h.p. Using this emulation, a deterministic binary search algorithm based on collision detection can be used to select a node with maximum ID (c.f., [17]). The binary search algorithm requires $\log n$ rounds when run on a multiple-access channel with collision detection. Combining this algorithm with the emulation protocol from [3], and using union bound with respect to low probability of unsuccessful emulation of a single round on a multiple-access channel, we obtain the following result.

FACT 1. *There is a randomized algorithm electing a leader in* $\mathcal{O}((D + \log n)\log n \log \Delta)$ *rounds in any* n*-node network with diameter at most* D*, with high probability.*

2.2 Stage 2: Construction of distributed BFS

We use the algorithm proposed in [4]. It proceeds in D phases, each consisting of $\mathcal{O}(\log n \log \Delta)$ rounds. At each phase d, $0 \leq d \leq D-1$, only nodes with distance d from the leader (the root of the constructed BFS tree) participate in transmitting BFS construction messages. Each construction message includes the ID of the transmitter and its distance to the leader. At each phase, the set of participating nodes use the Decay algorithm [4] to transmit its construction message. If a node receives a construction message for the first time, it selects the sender of this message as its parent in the BFS tree and sets its distance equal to the distance of its parent plus one.

THEOREM 1. *[4] The distributed BFS algorithm terminates in* $\mathcal{O}(D \log n \log \Delta)$ *rounds, and at the end each node knows its parent in the constructed BFS tree and its distance to the root, w.h.p.*

2.3 Stage 3: Packet collection

The objective of this stage is to collect all the packets at the leader selected in Stage 1. Recall that the leader is also a root of the BFS tree. This stage is partitioned into phases, and each phase is further split into two epochs called grabbing epoch and alarming epoch, respectively. Inside epochs, two different subroutines are used: $\mathcal{GRAB}$ and $\mathcal{ALARM}$; they will be described in detail later.

In the *grabbing epoch*, nodes with at least one packet to be delivered, unicast all their packets to the root along the BFS path using sub-routine $\mathcal{GRAB}$. In the *alarming epoch*, the nodes that have not received an acknowledgment for at least one of their packets, execute procedure $\mathcal{ALARM}$.

Since k, the total number of the packets, is not known in advance. Therefore, the nodes estimate it, starting with value $(D + \log n) \log n$ set up in the beginning of the first phase. If a node receives an alarm message in a phase, it doubles its estimate of k in the beginning of the next phase. The collection stage is finished at the end of a phase in which no alarm message is received.

2.3.1 Two main sub-routines

We define two sub-routines, which are executed in the grabbing and alarming epochs of each phase.

Sub-routine $\mathcal{GRAB}$. Before specifying sub-routine $\mathcal{GRAB}$, we first present its main building block — procedure *One_Shot_Partial_Gather* ($\mathcal{OSPG}$ for short) — parameterized by x. In $\mathcal{OSPG}(x)$ each node that has at least one packet to be sent (except the root), chooses a random number between 1 and $6x$ for each of its packets. If the number assigned to a packet is r, then at round r of the grabbing epoch, the node starts unicasting the message towards the root along the BFS path; more precisely, in each step of the unicast a node that has received the packet in the previous round from its child, transmits it with the information that it is addressed to its parent (recall that after Stage 2, there is a BFS tree such that each node knows its parent in this tree, w.h.p.). If two or more packets stored at a node are assigned the same number, the node unicasts only one of them, selected arbitrarily. This part of procedure takes $6x + D$ rounds.

After round $6x + D$, the root unicasts acknowledgments for packets received in the current epoch, one after another, with delay of 3 to ensures that they do not collide with each other while being forwarded along their BFS paths. Each intermediate node along the path forwards an acknowledgment to the child node in the BFS tree from which it has received the corresponding packet in the grabbing epoch. This will last at most $3 \cdot (6x + D) + D$ rounds because the maximum distance to the root is D, and the root can collect at most $6x+D$ packets in that epoch (more precisely, all of them arrived during the first $6x + D$ rounds of the epoch, each in a different round). Therefore, we bound the length of procedure $\mathcal{OSPG}(x)$ by $(6x+D)+(3 \cdot (6x + D) + D) = 24x + 5D$ rounds.

Note that some of the packets propagated towards the root in the first part of procedure $\mathcal{OSPG}(x)$ may get lost somewhere in the path to the root due to a collision, as we do not implement any mechanism to recover from collisions during procedure $\mathcal{OSPG}(x)$.

However, if a packet is successfully delivered to the root, the corresponding acknowledgement will be propagated backwards without any collision, and thus every successfully received packet at the root will also be acknowledged at the originator of the packet. We will also show later in this section that the root will receive at least half of all packets with high probability if x is large enough. See the proof of Lemma 4 for details. Further, we define a procedure *Multi_Shot_Partial_Gather* ($\mathcal{MSPG}(x, z)$ for short), which works similar to $\mathcal{OSPG}(x)$, with the exception that every packet selects z rounds (instead of one) from $[1, 6x]$ and a copy of the packet is propagated in each of the selected rounds.

Using procedures $\mathcal{OSPG}$ and $\mathcal{MSPG}$, we define subroutine $\mathcal{GRAB}(x)$, in which $\mathcal{OSPG}(y)$ is called several times for specific values of y and followed by one execution of $\mathcal{MSPG}$ according to the sequence:

$$\mathcal{OSPG}(x), \mathcal{OSPG}\left(\frac{x}{2}\right), \ldots, \mathcal{OSPG}\left(\frac{x}{2^i}\right), \ldots,$$
$$\mathcal{OSPG}\left(c \log n\right), \mathcal{MSPG}\left(c^2 \log^2 n, c \log n\right)$$

for a sufficiently large constant c. Sub-routine $\mathcal{GRAB}$ is used in the grabbing epoch of each phase. Note that if procedure $\mathcal{OSPG}(y)$, for any y, guarantees that at least half of the remaining y packets will be collected by the root w.h.p., as we stressed above, then the resulting sub-routine $\mathcal{GRAB}(x)$ should reduce the number of non-collected packets first to at most $c \log n$, and then to 0, w.h.p. The latter reduction is based on the fact that each copy of a packet has a constant probability to be sent by its source in a unique round (and thus to be delivered to the root without a collision), and hence, every packet is delivered w.h.p., by Chernoff bound for sufficiently large c. Note that the length of sub-routine $\mathcal{GRAB}(x)$ is bounded by $(24x + 5D) + (12x+5D)+\ldots+(24c \log n+5D)+(24c^2 \log^2 n+5D) = \mathcal{O}\big(x + D \log x + \log^2 n\big)$.

Sub-routine $\mathcal{ALARM}$. The second subroutine we use is called $\mathcal{ALARM}$, and is executed in the alarming epoch of each phase. Each node that has at least one non-acknowledged packet, initializes a broadcast in order to distribute an alarm message (a single bit 1). The purpose of this broadcast is to inform all other nodes that the collection has not been completed. The BGI broadcast protocol from [4] is used in order to deliver an alarm message, consisting of a single bit 1, to all nodes in the network. Note that the original BGI protocol was analyzed in [4] in the scenario with exactly one source node, while in our alarming epoch there may be many nodes which want to initialize the process of sending the alarm message (i.e., many sources). The progress is made based on a specific procedure Decay, which guarantees that if a node has at least one neighbor who already got the source message, gets the message in $\mathcal{O}(\log \Delta)$ rounds with constant probability. In our case we have possibly many sources but still one message, therefore a simple transformation from the original graph with many alarm sources into the same graph with additional node connected to all sources justifies that the process of broadcasting alarm message in the original graph is not longer than the process of broadcasting a single message from the unique source in the latter graph of $n+1$ nodes and diameter at most $D+1$. Therefore, the BGI algorithm guarantees that each node receives an alarm message, if any, within asymptotically the same number of rounds $\mathcal{O}((D+\log n)\log\Delta)$ as in the classical setting with $n+1$ nodes and diameter $D+1$, w.h.p. Therefore, the number of rounds available in the alarm epoch of every phase is set to $\mathcal{O}((D+\log n)\log\Delta)$.

2.3.2 Analysis of Stage 3

Lemma 4. *If $x \geq k$, the root receives all the packets using $\mathcal{GRAB}(x)$ within designated time of*

$$\mathcal{O}\big(x + D\log x + \log^2 n\big)$$

rounds, w.h.p.

Proof. We have already estimated the length of subroutine $\mathcal{GRAB}$ right after describing it above. It remains to prove that if $x \geq k$ then the root collects all packets w.h.p.

Assume $x \geq k$. First we argue that procedure $\mathcal{OSPG}(y)$ assures that at least half of the packets are collected by the root w.h.p. when their number is at most y, for $c\log n \leq y \leq x$. Indeed, the probability that a packet is assigned a unique starting round within procedure $\mathcal{OSPG}(y)$ is at least $(1-1/(6y))^{y-1} \geq 1/2+1/4$, and since these events are independent, by the classical Chernoff bound yields that at least half of these events hold with probability at least $1-\exp(-y(1/4)^2/3)$, which is w.h.p. as $y \geq c\log n$ and c is chosen to be sufficiently large. It follows, by the union bound, that after executing

$$\mathcal{OSPG}(x), \ldots, \mathcal{OSPG}\left(\frac{x}{2^i}\right), \ldots, \mathcal{OSPG}\left(c\log n\right)$$

where c is a sufficiently large constant, the number of non-collected packets reduces to at most $(c/2)\log n$. Here we use the fact that if a packet has been successfully delivered to the root during the execution of some $\mathcal{OSPG}(y)$, the originating node is informed about it and it will never attempt to transfer this packet in the subsequent executions. We argue that $\mathcal{MSPG}(c^2\log^2 n, c\log n)$ assures that each copy of a packet is received by the root with constant probability, by the argument similar to the one in the analysis of procedure $\mathcal{OSPG}$ and by the fact that there are at most $c^2\log^2 n$ packet copies in total. Therefore, at least one of the copies of any given packet will be received by the root w.h.p., by the fact that each packet has $c\log n$ copies. Hence, by the union bound, all packets are successfully collected by the root at the end of sub-routine $\mathcal{GRAB}(x)$, w.h.p. $\square$

Lemma 5. *The collection stage finishes when all the packets are collected by the leader. This process takes $\mathcal{O}(k+(D+\log n)\log n)$ rounds, w.h.p.*

Proof. First note that once we run a phase for estimate y of k such that $y/2 < k \leq y$, we are guaranteed that all packets are collected at the leader and then successfully acknowledged, and thus no alarm message is launched and all nodes terminate, w.h.p.

Therefore it remains to estimate the number of rounds needed to reach the end of phase with such estimate y of the number of packets k. Each phase of the collection stage in which the estimate of k is set to x lasts

$$\begin{aligned}&\mathcal{O}\big(x + D\log x + \log^2 n\big) + \mathcal{O}((D+\log n)\log\Delta)\\ &= \mathcal{O}(x + (D+\log n)\log n)\ ,\end{aligned}$$

as $x \leq k$ and k is polynomial in n. Since our initial estimate of x is $(D+\log n)\log n$, and each phase it doubles, in order to reach the end of the phase with estimate y on k we need

$$\sum_{\{i:(D+\log n)\log n \leq 2^i \leq k\}} \mathcal{O}\Big(2^i + (D+\log n)\log n\Big)$$

rounds. Note that each of the formulas under the sum is upper bounded by $\mathcal{O}\big(2^i\big)$. Therefore we get the final formula $\mathcal{O}(k+(D+\log n)\log n)$ on the number of rounds in Stage 3 of the algorithm, which holds w.h.p. $\square$

2.4 Stage 4: Packet dissemination

At the end of the collection stage, the root will have all k packets, with high probability. In the dissemination stage, the root broadcasts all the packets using the following algorithm. Let $g = \lceil \frac{k}{\lceil\log n\rceil} \rceil$ be a positive integer. The dissemination stage is grouped into $g+3D$ phases, each consisting of $\mathcal{O}(\log n\log\Delta)$ rounds. The root divides packets into g groups each with $\lceil\log n\rceil$ packets (one of the groups may have less than $\lceil\log n\rceil$ packets if $\lceil\log n\rceil$ does not divide k). Recall that b is the upper bound on the size of a packet, measured in the number of bits, and that $b \geq \log n$. Let $\mathbb{F}$ be a finite field of size 2^b. Note that every packet/message of size at most b bits can be regarded as a number in $\mathbb{F}$, and XORing messages of size at most b is equivalent of adding their corresponding numbers in $\mathbb{F}$.

Following we describe a sub-routine called $\mathcal{FORWARD}$, which is based on the Decay procedure from [4]. $\mathcal{FORWARD}$ is used in each phase of dissemination stage by the set of nodes at some distance d, where $1 \leq d \leq D-1$, from the root to forward a set of at most $\lceil\log n\rceil$ messages to the nodes at distance $d+1$.

$\mathcal{FORWARD}$:
The sub-routine runs for $\mathcal{O}(\log n)$ epochs, each consisting of $\lceil\log\Delta\rceil$ rounds. A set T of nodes, where $|T| \geq 1$, participates. Let R be a non-empty set of nodes such that each node in R has at least one and at most Δ neighbors in T. Let $\mathcal{M}$ be a non-empty set of packets such that $|\mathcal{M}| \leq \lceil\log(n)\rceil$, and each packet in $\mathcal{M}$ has at least $\lceil\log n\rceil$ and at most b bits. Suppose every node in T knows all the messages in $\mathcal{M}$. Similar to the Decay algorithm [4], in each round $s = 1, ..., \lceil\log(\Delta)\rceil$ of

every epoch, every node in T transmits with probability $p_s = \frac{1}{2}, \frac{1}{4}, \ldots, \frac{1}{2^{\lceil \log(\Delta) \rceil}}$. Every time a node decides to transmit, it generates a new message from the set of messages $\mathcal{M}$ and transmits the new message. To generate the message, the node independently chooses each message from $\mathcal{M}$ with probability $\frac{1}{2}$ and adds their corresponding numbers in $\mathbb{F}$, where $\mathbb{F}$ is a finite field (of size 2^b) known by all the nodes in $T \cup R$. It then transmits the sum (which has b bits) together with a header of size $\lceil \log(n) \rceil$ bits indicating the set of selected messages in $\mathcal{M}$. Note that since every message in $\mathcal{M}$ has at least $\lceil \log(n) \rceil$ bits, the size of the new message is at most twice the size of any message in $\mathcal{M}$.

The sub-routine $\mathcal{FORWARD}$ lasts exactly one phase of the dissemination stage. To explain how $\mathcal{FORWARD}$ is used to disseminate messages, let us first consider a special case where $k \leq \lceil \log n \rceil$ (i.e., $g = 1$). In this case, in the first phase of the dissemination stage, the root transmits all the packets to its one-hop neighbors, one-by-one in k rounds. Since $k \leq \lceil \log n \rceil$, by the end of the first phase all the nodes with distance one from the root will receive all k packets. In the second phase of the dissemination stage, the set of nodes at distance one uses sub-routine $\mathcal{FORWARD}$ to forward all k packets to the set of nodes at distance two from the root. Note that each node with distance two has at least one and at most Δ neighbors with distance one. Therefore, as we will shortly prove in Lemma 6, all the nodes at distance two will receive all k messages by the end of the second phase, w.h.p. Similarly, in phase d, $3 \leq d \leq D-1$, nodes at distance $d-1$ use sub-routine $\mathcal{FORWARD}$ to send all k messages to the nodes at distance d. Therefore, after D phases, all the nodes in the network will receive all messages, with high probability.

In general, when $g \geq 1$, the above process is done for each group of messages. To avoid collision between transmissions associated with different groups of messages, the dissemination of each group starts 3 phases after the start of the dissemination of the previous group of messages.

We first prove a desired property of sub-routine $\mathcal{FORWARD}$, and then conclude the analysis of Stage 4.

Lemma 6. *Using $\mathcal{FORWARD}$, all nodes in R will receive all the messages in $\mathcal{M}$, with high probability.*

Proof. Let u be any node in R. In each epoch, u receives a message from one of its neighbors in T with a constant probability, as guaranteed by procedure Decay [4]. Therefore, after $\mathcal{O}(\log n)$ epochs, it will receive $\mathcal{O}(\log n)$ messages, w.h.p., by Chernoff bound, c.f., Lemma 1.

We show that u can use the set of received messages to extract the set of packets in $\mathcal{M}$. Each received message is a linear combination of packets in $\mathcal{M}$ with independent random binary coefficients, each taking value 0 (or 1) with probability $\frac{1}{2}$. To extract packets in $\mathcal{M}$, node u requires to solve a set of $\mathcal{O}(\log n)$ linear equations. In other words, node u can extract all the packets in $\mathcal{M}$ if the corresponding binary matrix of linear equations has a full rank. Let M be the matrix corresponding to the set of linear equations. The elements of M are independent random binary numbers, each equal to 0 with probability $\frac{1}{2}$. Further, the matrix M has at most $\log n$ columns (i.e., the number of packets in $\mathcal{M}$) and $\mathcal{O}(\log n)$ rows (i.e., the number of rounds in which messages have been sent during the current $\mathcal{FORWARD}$ execution), with high probability. Then, by Lemma 3, M has full rank, with high probability, and consequently the set of linear equations is solvable, w.h.p. $\square$

Lemma 7. *The root successfully broadcast all k packets stored at it in the beginning of Stage 4 during*

$$\mathcal{O}(D \log n \log \Delta + k \log \Delta)$$

rounds of that stage, w.h.p.

Proof. The proof is by induction on the distance from the root. The first phase of the stage takes k rounds and guarantees that each node of distance 1 from the root receives all k packets. Suppose that by the end phase d, for $1 \leq d \leq D-1$, each node of distance at most d from the root has received all k packets. Lemma 6 applied to the execution of sub-routine $\mathcal{FORWARD}$ in phase $d+1$ guarantees that all k packets are received by any node of distance $d+1$ from the root, and thus being a neighbor of at least one and at most Δ nodes of distance d from the root, during phase $d+1$. All D inductive steps hold with high probability, therefore we conclude that all k packets are successfully delivered to all nodes within D phases of Stage 4 w.h.p.

The whole dissemination process takes $D+3g$ phases, which consists of

$$(D+3g) \cdot \mathcal{O}(\log n \log \Delta) = \mathcal{O}(D \log n \log \Delta + k \log \Delta)$$

rounds. $\square$

2.5 Final analysis of multi-broadcast algorithm

Combining Fact 1 and Theorem 1 with Lemmas 5 and 7, we obtain the following result.

Theorem 2. *The multi-broadcast algorithm successfully broadcast all k packets stored in the system, for any $k > 0$, in $\mathcal{O}(k \log \Delta + (D + \log n) \log n \log \Delta)$ rounds, with high probability.*

3. CONCLUSIONS AND OPEN PROBLEMS

In this work we showed how to efficiently combine randomization techniques with network coding to obtain better multiple-message broadcast algorithm for radio networks. An interesting open question is whether a similar approach could improve design and analysis of efficient protocols in other models of wireless networks, such that geometric graphs, bounded-growth graphs or Signal-to-Interference-and-Noise-Ratio (SINR). In more practical scenario, packets appear at nodes dynamically; a challenging direction would be to adapt "static" solutions to various communication problems to such more dynamic setting.

4. REFERENCES

[1] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg, A lower bound for radio broadcast, *Journal of Computer and System Sciences* 43 (1991) 290-298.

[2] N. Alon and J.H. Spencer, "The probabilistic method," (2ed). New York: Wiley-Interscience (2000).

[3] R. Bar-Yehuda, O. Goldreich, and A. Itai, Efficient emulation of single-hop radio network with collision detection on multi-hop radio network with no collision detection, *Distributed Computing*, 5 (1991) 67 - 71.

[4] R. Bar-Yehuda, O. Goldreich, and A. Itai, On the time complexity of broadcast in radio networks: An exponential gap between determinism and randomization, *Journal of Computer and System Sciences*, 45 (1992) 104 - 126.

[5] R. Bar-Yehuda, A. Israeli, and A. Itai, Multiple communication in multi-hop radio networks, *SIAM Journal on Computing*, 22 (1993) 875 - 887.

[6] I. Chlamtac and S. Kutten, Tree-based broadcasting in multihop radio networks, *IEEE Transactions on Computers* 36 (1987) 1209 - 1223.

[7] B.S. Chlebus, L. Gasieniec, A.M. Gibbons, A. Pelc, and W. Rytter, Deterministic broadcasting in ad hoc radio networks, *Distributed Computing*, 15 (2002) 27 - 38.

[8] B.S. Chlebus, L. Gasieniec, A. Östlin, and J.M. Robson, Deterministic radio broadcasting, in *Proc., 27th International Colloquium on Automata, Languages and Programming (ICALP)*, 2000, LNCS 1853, pp. 717 - 728.

[9] B. S. Chlebus, D. R. Kowalski, and T. Radzik, Many-to-many communication in radio networks, *Algorithmica*, 54(1): 118 - 139, 2009.

[10] M. Christersson, L. Gasieniec, and A. Lingas, Gossiping with bounded size messages in ad-hoc radio networks, in *Proc., 29th International Colloquium on Automata, Languages and Programming (ICALP)*, 2002, pp. 377 - 389.

[11] M. Chrobak, L. Gasieniec, and W. Rytter, Fast broadcasting and gossiping in radio networks, in *Proc., 41st Symposium on Foundations of Computer Science (FOCS)*, 2000, pp. 575 - 581.

[12] A.E.F. Clementi, A. Monti, and R. Silvestri, Distributed broadcasting in radio networks of unknown topology, *Theoretical Computer Science*, 302 (2003) 337 - 364.

[13] A. Czumaj, and W. Rytter, Broadcasting algorithms in radio networks with unknown topology, in *Proc., 44th IEEE Symposium of Foundations of Computer Science (FOCS)*, 2003, pp. 492 - 501.

[14] L. Gasieniec, E. Kranakis, A. Pelc, and Q. Xin, Deterministic m2m multicast in radio networks, in *Proc., 31st International Colloquium on Automata, Languages and Programming (ICALP)*, 2004, pp. 670 - 682.

[15] L. Gasieniec, I. Potapov, Gossiping with unit messages in known radio networks, in *Proc., 2nd International Conference on Theoretical Computer Science (TCS)*, 2002, pp. 193 - 205.

[16] M. Khabbazian, F. Kuhn, D.R. Kowalski, and N.A. Lynch, Decomposing broadcast algorithms using abstract MAC layers, on *Proc., 6th International Workshop on Foundations of Mobile Computing (DIALM-POMC)*, 2010, pp. 13-22.

[17] D.R. Kowalski, and A. Pelc, Broadcasting in undirected ad-hoc radio networks, *Distributed Computing*, 18 (2005) 43 - 57.

[18] D.R. Kowalski, and A. Pelc, Time of deterministic broadcasting in radio networks with local knowledge, *SIAM Journal on Computing*, 33 (2004) 870 - 891.

[19] E. Kushilevitz, and Y. Mansour, An $\Omega(D\log(N/D))$ lower bound for broadcast in radio networks, *SIAM Journal on Computing*, 27 (1998) pp. 702 - 712.

APPENDIX

A. PROOFS FROM SECTION 1.1

Proof of Lemma 1: Let $\mu = rp$. Using Chernoff, we get:

$$Pr(\sum_{q=1}^{r} Y_q < d) \le \exp\left(-\frac{1}{2}\frac{(\mu-d)^2}{\mu}\right). \quad (1)$$

Note that the function $f(x) = \exp(-\frac{(x-d)^2}{2x})$ is non-increasing in x for $d \le x$. Also, since $d \ge 1$, we have

$$d \le 3d+2\tau-p = (\frac{1}{p}(3d+2\tau)-1)p \le \lfloor\frac{1}{p}(3d+2\tau)\rfloor p = rp = \mu.$$

Therefore,

$$\begin{aligned}\exp\left(-\frac{1}{2}\frac{(\mu-d)^2}{\mu}\right) &\le \exp\left(-\frac{1}{2}\frac{(3d+2\tau-p-d)^2}{3d+2\tau-p}\right)\\ &= \exp\left(-\frac{1}{2}\frac{(2d+2\tau-p)^2}{3d+2\tau-p}\right)\\ &\le \exp(-\tau).\end{aligned}$$

□

Proof of Lemma 2: We prove the claim by using a Chernoff-type argument. For every $\gamma > 0$ and every $t \ge 0$, we have

$$\Pr(X \ge t) = \Pr\left(e^{\gamma X} \ge e^{\gamma t}\right) \le \frac{\mathbb{E}\left[e^{\gamma X}\right]}{e^{\gamma t}} = \frac{\prod_{i=1}^{k}\mathbb{E}\left[e^{\gamma X_i}\right]}{e^{\gamma t}}, \quad (2)$$

where the inequality follows by applying Markov's inequality. Let first derive a bound on $\mathbb{E}[e^{\gamma X_i}]$. Assume that $\gamma = p_{\min}/4 \le 1/4$, yielding $e^\gamma - 1 \le p_{\min}/3$. For all $i \in [k]$, we get

$$\begin{aligned}\mathbb{E}\left[e^{\gamma X_i}\right] &= \sum_{s=1}^{\infty} p_i(1-p_i)^{s-1}e^{\gamma s}\\ &= \frac{p_i e^\gamma}{1-(1-p_i)e^\gamma}\\ &\overset{(1+x\le e^x)}{\le} e^{\frac{e^\gamma-1}{p_i e^\gamma-(e^\gamma-1)}}\\ &\overset{(e^\gamma-1\le\frac{p_{\min}}{3})}{\le} e^{\frac{p_{\min}}{3p_i e^\gamma - p_{\min}}}\\ &\overset{(e^\gamma>1)}{\le} e^{\frac{p_{\min}}{2p_i}}.\end{aligned}$$

Plugging this into (2), we obtain

$$\Pr(X \ge t) \le e^{\frac{p_{\min}}{2}\cdot\sum_{i=1}^{k}\frac{1}{p_i} - \gamma t} = e^{\frac{p_{\min}}{2}\cdot\left(\mu-\frac{t}{2}\right)}.$$

The lemma now follows by using $t = 2\mu + 4\ln(1/\epsilon)/p_{\min}$. □

Proof of Lemma 3: Let V be the set of all binary vectors of size w. Consider the following game with possibly infinite number of rounds: At each round of the game a vector is randomly and uniformly selected from V. The game terminates if w linearly independent vectors are collected. For $i = 1, 2, \ldots, w$, let Y_i be a random variable equal to the round number at which i linearly independent vectors are collected for the first time. Let us define $X_1 = Y_1$, and $X_i = Y_i - Y_{i-1}$ for every $2 \le i \le w$. A set of i, $i \ge 1$, linearly independent vectors, span a vector space of dimension i and size 2^i. The probability that a uniformly selected vector from S avoid this space is $(1 - \frac{2^i}{2^w})$. Thus, for any positive integer x, and any integer i, $1 \le i \le w-1$, we get

$$\begin{aligned}Pr(X_{i+1} = x) &= Pr(Y_{i+1} - Y_i = x)\\ &= \sum_{y=1}^{\infty} Pr(Y_{i+1} = x+y | Y_i = y) Pr(Y_i = y)\\ &= \sum_{y=1}^{\infty}\left(\frac{2^i}{2^w}\right)^{x-1}\left(1-\frac{2^i}{2^w}\right)Pr(Y_i = y)\\ &= \left(\frac{2^i}{2^w}\right)^{x-1}\left(1-\frac{2^i}{2^w}\right)\sum_{y=1}^{\infty}Pr(Y_i = y)\\ &= \left(\frac{2^i}{2^w}\right)^{x-1}\left(1-\frac{2^i}{2^w}\right)\end{aligned} \quad (3)$$

Also,

$$Pr(X_1 = x) = Pr(Y_1 = x) = (\frac{1}{2^w})^{x-1}(1-\frac{1}{2^w}) \quad (4)$$

Thus, by (3) and (4), $X_1, \ldots, X_w$ are geometric random variables with parameters $p_i = 1 - \frac{2^{i-1}}{2^w}$, $1 \le i \le w$. By Lemma 2, we get

$$\Pr\left(\sum_{i=1}^{w} X_i \ge 2\mu + \frac{4\ln(1/\epsilon)}{p_{min}}\right) \le \epsilon,$$

where $p_{min} = 1 - \frac{2^{w-1}}{2^w} = \frac{1}{2}$ and

$$\mu = \sum_{i=1}^{w}\frac{1}{1-\frac{2^{i-1}}{2^w}} \le w+2.$$

Thus,

$$\Pr(\sum_{i=1}^{w} X_i \ge l) \le \epsilon,$$

hence

$$\begin{aligned}Pr(Y_w \ge l) &= Pr\left(\left(Y_1 + \sum_{i=2}^{w}(Y_i - Y_{i-1})\right) \ge l\right) \le \epsilon\\ &= Pr\left(\sum_{i=1}^{w} X_i \ge l\right)\\ &\le \epsilon.\end{aligned} \quad (5)$$

Note that each row of matrix A is a uniformly selected vector from V. Matrix A has full rank if it has w linearly independent rows. Thus, by (5), the probability that A has full rank is at least $1-\epsilon$. □