Gossiping in a Multi-channel Radio Network
An Oblivious Approach to Coping with Malicious Interference (Extended Abstract)

Shlomi Dolev[1], Seth Gilbert[2], Rachid Guerraoui[3], and Calvin Newport[4]

[1] Ben-Gurion University
dolev@cs.bgu.ac.il
[2] MIT CSAIL, EPFL IC
seth.gilbert@epfl.ch
[3] EPFL IC
rachid.guerraoui@epfl.ch
[4] MIT CSAIL
cnewport@mit.edu

Abstract. We study oblivious deterministic gossip algorithms for multi-channel radio networks with a malicious adversary. In a multi-channel network, each of the n processes in the system must choose, in each round, one of the c channels of the system on which to participate. Assuming the adversary can disrupt one channel per round, preventing communication on that channel, we establish a tight bound of $\max\left(\Theta\left(\frac{(1-\epsilon)n}{c-1}+\log_c n\right), \Theta\left(\frac{n(1-\epsilon)}{\epsilon c^2}\right)\right)$ on the number of rounds needed to solve the ϵ-gossip problem, a parameterized generalization of the all-to-all gossip problem that requires $(1-\epsilon)n$ of the "rumors" to be successfully disseminated. Underlying our lower bound proof lies an interesting connection between ϵ-gossip and extremal graph theory. Specifically, we make use of Turán's theorem, a seminal result in extremal combinatorics, to reason about an adversary's optimal strategy for disrupting an algorithm of a given duration. We then show how to generalize our upper bound to cope with an adversary that can simultaneously disrupt $t < c$ channels. Our generalization makes use of selectors: a combinatorial tool that guarantees that any subset of processes will be "selected" by some set in the selector. We prove this generalized algorithm optimal if a maximum number of values is to be gossiped. We conclude by extending our algorithm to tolerate traditional Byzantine corruption faults.

1 Introduction

Malicious adversaries pose a particular threat to radio networks. Due to the shared nature of the communication medium, an adversary can prevent *any* information exchange between honest processes by jamming the channel with noise. The first attempts to tackle this problem assumed that the malicious adversary could only corrupt honest processes, but not interfere with communication [1, 2, 3]. Another approach assumed that the adversary interferes only in a probabilistic manner, causing either random transient message corruption [4], or random permanent process corruptions [5]. More recent work allows malicious interference—but bounds the number of times that the adversary can disrupt communication [6, 7].

In this paper, we place no such restrictions on the adversary. Instead, we shift our focus to multi-channel radio networks in which each process can make use of any one of c available channels in each round. The adversary can disrupt communication by broadcasting concurrently on a channel with an honest process, causing a collision. This setting is appealing because of its practicality. Almost every major commercial/industrial/military radio device—including sensor motes, laptops running 802.11, and bluetooth-enabled devices—has the capability to switch between multiple communication channels. These multi-channel networks have been studied previously in the context of communication capacity and throughput (e.g., [8, 9]). They have also been studied in the field of Cognitive Radio Networks [10, 11, 12], where algorithms attempt to adaptively compensate for (semi-permanently) disrupted communication channels.

The Gossip Problem. We study the fundamental problem of *gossip* in which processes attempt to exchange *rumors*. Variants of this problem have been well-studied in (synchronous) single-channel radio networks: for broadcast in a fault-free network, see, for example, [13, 14, 15, 16]; for omissions and crash failures, see, for example, [17, 18]. We introduce a parameterized version of the gossip problem, called ϵ-gossip, in which $(1-\epsilon)n$ rumors must be disseminated to at least $n-1$ processes. (As we later show, it is impossible to disseminate even a single value to all n receivers in this setting). The ϵ-gossip problem is a generalization of classical *all-to-all gossip* (0-gossip) that allows for flexibility in the number of rumors that need to be spread—a desirable feature for many applications (e.g., when only a majority vote is needed).

Basic Setting. We assume that honest processes maintain no shared secrets (e.g., information unknown to the adversary). The honest processes could use such information to derive a transmission pattern that appears random to an outside observer. (Military communication systems, like those used by the MILSTAR satellite system, use such a scheme to evade eavesdroppers). We omit this possibility for three reasons. First, we are interested in deterministic solutions that guarantee correctness even in the worst case. Second, for low-resource devices—such as RFID tags or tiny sensor motes—cryptographic calculations (particularly of the public-key variety) and secure key dissemination may be prohibitively expensive. Third, shared secrets are hard to maintain when the adversary can corrupt and hijack honest devices (addressed in Section 9).

With such devices in mind, we focus on oblivious algorithms—those in which a process's decision to transmit or listen in a given round on a given channel is a function only of its unique identifier, the round number, and the number of processes and channels in the network. Oblivious algorithms are appealing because they are considered easy to construct and deploy. (In Section 10, we briefly discuss randomized and adaptive solutions.)

Results. We prove that $\max\left(\Theta\left(\frac{(1-\epsilon)n}{c-1}+\log_c n\right), \Theta\left(\frac{n(1-\epsilon)}{\epsilon c^2}\right)\right)$ rounds are necessary and sufficient to solve ϵ-gossip in a setting where the adversary can disrupt one channel per round, and n is the total number of processes. We demonstrate necessity by first reducing ϵ-gossip to a graph-theoretic game—(n, ϵ)-*clique destruction*—in which a player tries to remove enough edges from an n-clique to destroy any clique of size greater than ϵn, and then proving a lower bound on this game by appealing to Turán's Theorem [19], a seminal result in extremal combinatorics.

To demonstrate sufficiency, we describe a matching deterministic oblivious algorithm that proceeds in two phases. During the first phase, a sufficient number of values are disseminated to a distinguished group of *listeners* distributed among the channels. The resulting construction, when considered in the context of extremal graph theory, produces a Turán Graph. During the second phase, these values are disseminated to increasingly larger sets of processes until $n-1$ have learned the total requisite knowledge.

We then generalize our algorithm for the multi-channel adversary that can simultaneously disrupt up to $t < c$ channels. This models a network with t adversarial processes, each potentially disrupting a different channel. Our algorithm presented for $t = 1$ extends naturally to this scenario: the generalized first phase, in fact, relates to a conjectured hypergraph-generalization of Turán's Theorem; the second phase uses *selectors*, a combinatorial device introduced by [20], to generalize the dissemination schedule. We show this solution to be optimal for the natural case of $\epsilon = t/n$ (i.e., trying to gossip the maximum possible number of values). We conclude by describing how to modify our algorithm to tolerate traditional Byzantine corruption faults.

2 Model

We consider a system of n honest processes, each assigned a unique identifier in the range $[1..n]$. The processes inhabit a single-hop radio network comprised of c communication channels: $1 < c \leq n$. We first assume the presence of a malicious adversary that can corrupt one channel per round. We then consider the general case where it can simultaneously disrupt $t < c$ channels in each round.

Synchronous Rounds. Executions proceed in synchronous rounds. In each round, each process chooses a single channel on which to participate: it can either transmit or receive on that channel. In each round the adversary chooses up to t channels to disrupt. If exactly one process transmits a message m on channel k in a round, and the adversary does not disrupt channel k, then every process receiving on channel k receives message m. When two processes broadcast in the same round on the same channel, the message is lost. (We do not assume that the processes have the capacity to detect collisions.)

Deterministic and Oblivious. We consider deterministic, oblivious gossip algorithms. An oblivious algorithm is one in which the broadcast schedule is determined in advance. Formally, a *deterministic oblivious algorithm* is a sequence $\mathcal{A} = \langle A_1, \ldots, A_r \rangle$ where each $A_r : [1..n] \rightarrow \{\mathsf{trans}, \mathsf{recv}, \bot\} \times [1..c]$ is a function describing the behavior of the processes in round i. For example, when $A_r(i) = \langle \mathsf{trans}, k \rangle$, it indicates that in round r, process i transmits on channel k; when $A_r(i) = \langle \mathsf{recv}, k \rangle$, then process i receives on channel k. Without loss of generality, for the lower bound we consider *full information protocols* in which processes always transmit their entire state. Also without loss of generality, we assume each initial value is unique.

3 The Gossip Problem

Each process begins with an initial value (or "rumor") which it attempts to disseminate to the other processes. In this paper we consider the (ϵ, δ)-gossip problem in which all but δn processes must receive a common set of all but ϵn of the initial values. This definition is a generalization of commonly considered communication primitives: for example, all-to-all gossip is $(0, 0)$-gossip and one-to-all broadcast is $(1 - \frac{1}{n}, 0)$-gossip. Formally, algorithm $\mathcal{A}$ *solves* the (ϵ, δ)-gossip problem if and only if, for all possible adversarial choices, at least $(1 - \delta)n$ honest processes successfully receive a common set of at least $(1 - \epsilon)n$ initial values.

In this setting, it is clearly impossible to ensure that *all* n honest processes successfully receive even one common initial value. Assume there existed such an $(\epsilon, 0)$-gossip algorithm $\mathcal{A}$, and consider the adversarial strategy $\mathcal{C}$ in which the adversary always disrupts the channel on which process 1 either transmits or receives. Under these conditions, process 1 can never successfully transmit or receive any value: it neither learns the initial value of any other process, nor does any other process learn its initial value, implying that $\mathcal{A}$ does not solve $(\epsilon, 0)$-gossip. By the same argument, it is impossible to solve $(0, \delta)$-gossip.

Hence, we focus on solving the $(\epsilon, t/n)$-gossip problem where $t/n \leq \epsilon \leq 1$, that is, the problem in which all but ϵn initial values are disseminated to all but t processes. (Considering larger values of δ does not allow for significantly faster termination for most values of ϵ.) For the remainder of the paper, we refer to the $(\epsilon, \frac{t}{n})$-gossip problem where $t/n \leq \epsilon \leq 1$ simply as: ϵ-gossip.

Roadmap. In Sections 4, 5 and 6, we address the case where $t = 1$. In Section 4 we present a lower bound, in Section 5 we present a matching algorithm, and in Section 6 we outline the proof. In Section 7, we extend our algorithm to tolerate a multi-channel adversary that can block an arbitrary $t < c$ channels. In Section 9, we consider a more general model in which the adversary can corrupt honest players (rather than simply disrupting communication). We conclude with a discussion of open questions in Section 10.

4 A Lower Bound for ϵ-Gossip Where $t = 1$

Let $\mathcal{A}$ be an arbitrary deterministic oblivious algorithm for ϵ-gossip where $t = 1$. We show that $\mathcal{A}$ requires

$$\max\left(\Theta\left(\frac{(1-\epsilon)n}{c-1} + \log_c n\right), \Theta\left(\frac{n(1-\epsilon)}{\epsilon c^2}\right)\right)$$

rounds to terminate. The first term dominates when $\epsilon \geq 1/c$ (i.e., only a small number of values need to be disseminated), and it follows from the observation that at most $c - 1$ values can be broadcast in each round. In this section, we focus predominantly on the (more interesting) case where $\epsilon < 1/c$. For every execution α of algorithm $\mathcal{A}$, let round $R^{\mathsf{trans}}(\alpha)$ be the minimum round such that the following is true: at least $(1-\epsilon)n$ of the honest processes have broadcast without being disrupted by the adversary in the prefix of α through round $R^{\mathsf{trans}}(\alpha)$. We show that for some execution α of $\mathcal{A}$, $R^{\mathsf{trans}}(\alpha) \geq \Omega(\frac{n(1-\epsilon)}{\epsilon c^2})$. It follows that the protocol cannot terminate in α prior to round $R^{\mathsf{trans}}(\alpha)$—implying the second term of our bound.

We prove this result by exploiting an interesting connection between oblivious ϵ-gossip and graph theory. An oblivious algorithm can be imagined as a sequence of edge removals from an n-clique, and the adversary's optimal strategy can be described by the largest clique that remains after these removals. Accordingly, we turn to the field of extremal combinatorics and apply Turán's Theorem [19] to argue precisely about the size of the cliques that remain in a graph, which in turn tells us the adversary's best strategy.

Definition 1 (The (n, ϵ)-Clique Destruction Game). *Let $G = (V, E)$ be a graph describing a clique on n nodes. We say that a subset of edges $S \subseteq E$ is a* solution *to the (n, ϵ)-clique destruction game if and only if the graph $G' = (V, E - S)$ contains no clique of size greater than ϵn.*

Turán's theorem relates the largest clique in a graph and the number of edges in a graph:

Theorem 1 (Turán's Theorem [19]). *If graph $G = (V, E)$ has no subgraph that is a clique of size $k + 1$, then $|E| \leq (1 - 1/k)\left(n^2/2\right)$.*

From this we derive an immediate corollary:

Corollary 1. *Fix $S \subseteq E$ a solution to the (n, ϵ)-clique destruction game. Then $|S| \geq n(1 - \epsilon)/(2\epsilon)$.*

Proof. By Theorem 1 where $k = \epsilon n$: subtract from the $\binom{n}{2}$ edges in an n-clique the maximum number of edges in a graph with no cliques of size $\epsilon n + 1$, as per Theorem 1, to get the minimum size of S.

We next connect the (n, ϵ)-clique destruction game to the ϵ-gossip problem:

Lemma 1. *If for every execution α of algorithm $\mathcal{A}$, $R^{\mathsf{trans}}(\alpha) \leq r$, then there exists a solution S to the (n, ϵ)-clique destruction game such that $|S| \leq c^2 r$.*

Proof. Without loss of generality, assume that $\mathcal{A}$ assigns exactly one process to transmit on each channel in each round. Construct S as follows: add edge (a, b) to S if, for some round $r' \leq r$, processes a and b both broadcast in round r'. Since for each $r' \leq r$ there are at most $\binom{c}{2}$ such pairs, we conclude that $|S| \leq c^2 r$.

Suppose, for contradiction, that nodes $V' \subseteq V$ form a clique of size $> \epsilon n$ in the residual graph. We construct the following strategy to thwart $\mathcal{A}$: whenever a process in V' attempts to broadcast on channel k, the adversary disrupts channel k. This is always possible as no two processes in V' broadcast in the same round (by the construction of S). This violates the correctness of $\mathcal{A}$, implying a contradiction.

We combines these two lemmas to obtain our final bound:

Theorem 2. *For any deterministic oblivious algorithm, $\mathcal{A}$, that solves ϵ-gossip,*

$$|\mathcal{A}| \geq \max\left(\Theta\left(\frac{(1-\epsilon)n}{c-1} + \log_c n\right), \Theta\left(\frac{n(1-\epsilon)}{\epsilon c^2}\right)\right) .$$

Proof. If $\mathcal{A}$ solves ϵ-gossip, then for every execution α of algorithm $\mathcal{A}$, there exists a round r such that $R^{\mathsf{trans}}(\alpha) \leq r$. We begin by establishing the first term of the bound. Since at most $c-1$ values can be transmitted without disruption in each round, it is clear that $r \geq (1-\epsilon)n/(c-1)$. Let r' be the round in which the last of these $(1-\epsilon)n$ values is transmitted. For each of the $n-1$ processes that ultimately learn all $(1-\epsilon)n$ values, there must be some round $\geq r'$ in which it listens on a non-disrupted channel. Let r'' be the latest of these rounds. We know $r'' \geq r' + \log_c n - 1$, as over the first $\log_c n - 1$ rounds there are only $c^{\log_c n-1} < n-1$ different channel-listening patterns a process can follow; by the pigeonhole principle this results in two processes listening on the same channels for these first $\log_c n - 1$ rounds, allowing the adversary to block both processes. Together these two pieces form the first term of the lower bound.

For the second term of the lower bound we turn to our Turán-derived results. From Lemma 1 we know there exists a set of edges S that solves the (n,ϵ)-clique destruction game, such that $c^2 r \geq |S|$. By Corollary 1 we know: $|S| \geq n(1-\epsilon)/(2\epsilon)$. This implies $r \geq \Theta\left(\frac{n(1-\epsilon)}{\epsilon c^2}\right)$.

5 An Upper Bound for ϵ-Gossip Where $t = 1$

In this section we describe a deterministic oblivious algorithm for solving the ϵ-gossip problem when $\epsilon < 1/c$. In Section 8, we discuss the (simpler) case where $\epsilon \geq 1/c$. For the sake of concision, we make a few simplifying assumptions. First, we assume that $n > 6c$. For smaller values of n, the algorithm can simply restrict itself to a subset of the channels. Second, we assume that the number of channels c is even. For odd c, the algorithm can restrict itself to $c-1$ channels. Finally, we assume that $\epsilon \geq \frac{2c+1}{n}$ (slightly larger than the trivial $\epsilon \geq 1/n$ lower bound for ϵ). In Section 5.3 we describe how to remove this last assumption.

5.1 The *Gossip* Protocol

The gossip protocol (see Figure 1) constructs an oblivious algorithm that solves the ϵ-gossip problem for $\epsilon < 1/c$. Recall, an oblivious algorithm is a sequence $\mathcal{A} = A_1, A_2, \ldots$ where each A_i is a function from processes $[1 \ldots n]$ to actions $\{\mathsf{trans}, \mathsf{recv}\}$ and channels $[1 \ldots c]$. Throughout the description, we use the notation $\mathsf{divide}(S, k)$ to refer to a partition of the set S into $\lceil |S|/k \rceil$ sets of size k, one of which may have fewer than k elements if $|S|$ does not divide evenly by k. Also, $i[b]$ refers to the b^{th} bit of the binary representation of i. Our protocol proceeds in two parts.

Part I: Initially, the processes are divided into two sets: a set of $2c$ *listeners*, consisting of processes $P_\ell = \{1, \ldots, 2c\}$, and a set of (at least $4c$) *transmitters* P_{tran}, consisting of the remaining processes. Each channel is assigned a pair of two listeners. Next, the $\mathsf{InfoTransfer}(\epsilon')$ routine is used to transfer all but $\epsilon' n + 2c$ of the initial values to some pair of listeners. (The additive $2c$ represents the listeners' values that are not transmitted.) By choosing $\epsilon' = \epsilon - 2c/n$, this ensures that all but ϵn initial values are known to some pair of listeners (in Section 5.3 we discuss how to allow the listeners to participate).

Part II: The goal of the second part is to disseminate the information acquired by each pair of listeners to all but one process. We say that a set is *knowledgeable* with respect to

```
Gossip($\epsilon$)
    ; Part I: Transfer info from transmitters to listeners.
    $P_\ell \leftarrow \{i \mid 1 \leq i \leq 2c\}$
    $P_{tran} \leftarrow \{i \mid 2c < i \leq n\}$
    channel-assignment $\leftarrow$ divide($P_\ell$, 2)
    InfoTransfer($\epsilon - \frac{2c}{n}$, $P_{tran}$, $P_\ell$, channel-assignement, $\langle A_1, \ldots \rangle$)

    ; Part II, Step 1: Create c knowledgable sets in two steps.
    Psets $\leftarrow$ divide($P_{tran}$, $\lceil P_{tran}/c \rceil$)
    for (chan $= 1$ to $c/2$) do
        disseminate2(channel-assignment[chan], Psets[chan], chan, chan$+c/2$, $\langle B_1, \ldots \rangle$)
    for (chan $= c/2+1$ to $c$) do
        L $\leftarrow$ channel-assignment[chan]
        disseminate2(channel-assignment[chan], Psets[chan], chan-$c/2$, chan, $\langle C_1, \ldots \rangle$)

    ; Part II, Step 2: Combine channels.
    $r \leftarrow 1$
    while ($|Psets| > 1$) do
        newPsets $\leftarrow \emptyset$
        $s \leftarrow \lfloor |Psets|/2 \rfloor$
        for ($i = 1$ to $s$) do
            $P \leftarrow$ combine(Psets[i], Psets[i+s], $i$, $i + c/2$, $\langle D_r, \ldots \rangle$)
            newPsets $\leftarrow$ newPsets $\cup P$
        ; If the size of Psets is odd, we let the last set pass through uncombined.
        if ($2s+1 = |Psets|$) then
            newPsets $\leftarrow$ newPsets $\cup$ Psets[$2s+1$]
        Psets $\leftarrow$ newPsets
        $r \leftarrow r + 6\lceil \log n \rceil + 36$
    return $\mathcal{A}.\mathcal{B}.\mathcal{C}.\mathcal{D}$
```

Fig. 1. An algorithm for solving the ϵ-gossip problem ($\epsilon < 1/c$)

some set of initial values if all but one process in the set has received the initial values. The second part of the gossip protocol proceeds in two steps. First, the transmitter processes are divided into c sets, one per channel. (The variable $Psets$ stores these c sets). The two listeners associated with each channel disseminate the information acquired in Part I to the set of transmitters assigned to that channel. This dissemination step uses two channels, and thus we can run $c/2$ instances in parallel: in the first $\lceil \log n \rceil$ rounds, we perform the dissemination for channels $1, \ldots, c/2$; in the next $\lceil \log n \rceil$ rounds, we perform the dissemination for channels $c/2 + 1, \ldots, c$. In each case, this dissemination is accomplished using the disseminate2 routine. At the end of this step, each set of processes in $Psets$ is knowledgeable with respect to the set of values known to the listeners on their channel.

In the second step, we repeatedly combine pairs of knowledgeable sets (via the combine routine) into larger knowledgeable sets in which the processes know values from both of the original sets. We continue combining sets until we are left with a single set in which the processes know the required $(1 - \epsilon)n$ values.

```
InfoTransfer($\epsilon'$, $P_{tran}$, $P_\ell$, channel-assignment, $\langle A_1, \ldots \rangle$)
  ; Assign listeners to channels.
  for every round $r$, for every channel $k$ do
    $\{a, b\} \leftarrow$ channel-assignment[$k$]
    $A_r(a) \leftarrow \langle$recv, $k\rangle$
    $A_r(b) \leftarrow \langle$recv, $k\rangle$
  ; Assign transmitters to channels.
  $r \leftarrow 0$
  for every $B \in$ divide($P_{tran} \cup P_\ell$, $1/\epsilon'$) do
    $B_{subs} \leftarrow$ divide($B$, $c/2$)
    for every $(S_1, S_2) \in B_{subs} \times B_{subs}$ do
      chan $\leftarrow 1$
      for every $i \in S_1 \cup S_2$
        $A_r(i) \leftarrow \langle$trans, chan$\rangle$
        chan $\leftarrow$ chan$+1$
      $r \leftarrow r+1$
```

Fig. 2. Routines to transfer information from transmitters to listeners

```
disseminate2($L$, $P$, $c_1$, $c_2$, $\langle A_1, \ldots \rangle$)
  $\{a_1, a_2\} \leftarrow L$
  for $b = 1$ to $\lceil \lg n \rceil$
    $A_b(a_1) \leftarrow \langle$trans, $c_1\rangle$
    $A_b(a_2) \leftarrow \langle$trans, $c_2\rangle$
  for $b = 1$ to $\lceil \lg n \rceil$
    for each $i \in P$
      if $i[b] = 0$ then
        $A_b(i) \leftarrow \langle$recv, $c_1\rangle$
      else if $i[b] = 1$ then
        $A_b(i) \leftarrow \langle$recv, $c_2\rangle$
```

Fig. 3. A routine that disseminates data from listeners to arbitrary sets

The Information Transfer Routine. The goal of the InfoTransfer routine (Figure 2) is to ensure that all but $\epsilon' n + 2c$ of the initial values are received by some pair of listeners. Each channel is assigned two listeners, and we assign transmitters to each channel in each round such that the resulting induced graph, as formulated in terms of Lemma 1 and the clique destruction game, forms a Turán Graph.[1]

We divide all processes into sets $\{B_1, B_2, \ldots, B_{\epsilon' n}\}$ of size $1/\epsilon'$; there are $\epsilon' n$ such sets. Our goal is to ensure that all but (at most) one transmitter in each set B_i succeeds in transmitting its value to a pair of listeners. We proceed as follows: For each set B_i, we sub-divide B_i into subsets of size $c/2$, and schedule each of the $\binom{(2/\epsilon' c)}{2}$ pairs of subsets to broadcast in a round (omitting listeners, which are already occupied).

[1] A Turán Graph for value k (say, $k = \epsilon n + 1$), is the unique graph, as proved by Turán, to contain no cliques of size k and to contain the maximum number of edges for which this condition can be true, as established by the theorem of the same name.

```
combine($S_1$, $S_2$, $c_1$, $c_2$, $\langle A_1, \ldots \rangle$)
  $r \leftarrow 0$
  for ($i = 1$ to 2) do
    ; Use i and i+2 as witnesses for $S_1$ :
    $L \leftarrow S_1[i] \cup S_1[i+2]$
    $P \leftarrow S_1 \cup S_2 - L$
    disseminate2($L, P, c_1, c_2, \langle B_1, \ldots \rangle$)
    $A \leftarrow A . B$
  for ($i = 1$ to 3) do
    ; Use i and i+3 as witnesses for $S_2$ :
    $L \leftarrow S_2[i] \cup S_2[i+2]$
    $P \leftarrow S_1 \cup S_2 - L$
    disseminate2($L, P, c_1, c_2, \langle B_1, \ldots \rangle$)
    $A \leftarrow A . B$
  return $S_1 \cup S_2$
```

Fig. 4. A routine to combine knowledgeable sets

Since every pair of non-listeners in B_i broadcast together in some round, the adversary can block at most one non-listener in each set B_i from communicating its value to a listener. **Running Time:** $\lceil \epsilon' n \binom{2/\epsilon' c}{2} \rceil$.

The Disseminate Routines. The disseminate2$(L, P, c_1, c_2, \ldots)$ routine (Figure 3) disseminates all values known by *both* processes in the set L to all but (at most) one process in the set P, using channels c_1 and c_2. (We assume $|L| = 2$). First, we assign the two processes in L to transmit on channels c_1 and c_2 for $\lceil \log n \rceil$ rounds. In each round b, each process in P chooses a channel on which to receive based on its identifier i: if $i[b] = 0$, it chooses channel c_1; if $i[b] = 1$, it chooses channel c_2. Thus, for any pair of processes in P, there is some round in which they receive on different channels. **Running Time:** $\lceil \log n \rceil$.

The Combine Routine. The combine$(S_1, S_2, c_1, c_2, \ldots)$ routine (Figure 4) begins with two knowledgeable sets. (We assume that S_1 and S_2 each contain at least 6 processes.) Using two channels, the combine function creates a new knowledgeable set $S_1 \cup S_2$ such that all but (at most) one process in the combined set knows the shared information from *both* S_1 and S_2. The routine accomplishes this goal by running disseminate2 five times. The first two times, it uses pairs of witnesses from set S_1 to disseminate information to set S_2. Since at most one node in S_1 is not knowledgeable, we can conclude that one of these pairs of witnesses is knowledgeable. Hence after the first two calls to the disseminate2 routine, all but one node in $S_1 \cup S_2$ are knowledgeable with respect to the values from S_1. The next three times, it uses pairs of witnesses from set S_2 to disseminate information to set S_1. Notice that there may be two nodes in S_2 that are not fully knowledgeable: one node may not be knowledgeable about S_2's values, and one node may not be knowledgeable about S_1's values. Thus for one of the three pairs of witnesses, both are knowledgeable, and the dissemination succeeds in informing all but one node in $S_1 \cup S_2$. **Running Time:** $5\lceil \log n \rceil$.

5.2 Running Time of the *Gossip* Protocol

The running time for Gossip is calculated as:

$$[\mathit{InfoTransfer}] + 2[\mathit{disseminate2}] + \lceil \log c \rceil [\mathit{combine}] \, .$$

This equals: $\lceil \epsilon' n \binom{2/\epsilon' c}{2} \rceil + 2\lceil \log n \rceil + \lceil \log c \rceil (5\lceil \log n \rceil \rceil)$. We can simplify this to $\Theta(\frac{n}{\epsilon' c^2})$. Because $\epsilon < 1/c \leq 1/2$, this is equivalent to $\Theta(\frac{n(1-\epsilon)}{\epsilon' c^2})$. The modified algorithm, presented in the next section, improves the running time (marginally) to $\Theta(\frac{n(1-\epsilon)}{\epsilon c^2})$, exactly matching our lower bound for the case where $\epsilon < 1/c$. As mentioned, in Section 8 we provide an algorithm (matching within an additive factor of $\log^2 c$) for the less involved case of $\epsilon \geq 1/c$.

5.3 Achieving ϵ-Gossip for Small ϵ

The algorithm presented in the previous section assumes that $\epsilon \geq (2c+1)/n$. We now discuss modifying the algorithm to require only $\epsilon \geq 1/n$, the minimum value of ϵ for which the problem can be solved. The difficulty occurs in the InfoTransfer routine, where the listeners do not participate in transmitting their values. Unlike the transmitters, their initial values are known only to themselves, not to a pair of processes. The first step in our modification is to schedule the listeners, P_ℓ, as well as the transmitters, P_{tran}, to transmit their values during the InfoTransfer. If only one of the two listeners assigned to channel k is scheduled to transmit in a round, then it broadcasts on channel k, resulting in no difficulties.

Consider the problematic case where both listeners for channel k are scheduled to transmit in the same round. Since the division into sub-blocks of size $c/2$ is arbitrary, we can ensure that each of the listeners for channel k is in a different sub-block. Thus, there is only one round for which both listeners for a channel might be forced to broadcast. In this case, two "backup listeners" are recruited to monitor channel k during that round. Since there are only c processes scheduled to broadcast in that round, and only $2c$ listeners, there remain at least $2c$ processes to play the role of backup listener. Every channel may be forced to recruit one pair of backup listeners, resulting in $4c$ listeners and backup listeners whose values need to be propagated in the second part of the protocol; the described algorithm extends immediately to this case.

6 Analysis

We now outline an argument that the algorithm from Section 5 solves ϵ-gossip (when $\epsilon < 1/c$). We focus on some of the key invariants satisfied by the different components of the construction. First, we observe that the InfoTransfer routine guarantees that a sufficient number of values are transmitted without adversarial disruption:

Lemma 2. *During the first $r = \lceil \epsilon n \binom{2/\epsilon c}{2} \rceil$ rounds, all but ϵn of the processes transmit their values in some round $\leq r$ without disruption.*

This claim follows from the construction of the schedule: for each of the ϵn sets, all pairs of processes broadcast together in some round. We therefore conclude that the listeners receive a sufficient set of values:

Corollary 2. *At the end of the* InfoTransfer *routine, there exists some set of values V of size at least $(1-\epsilon)n$ such that each value $v \in V$ is known to some set of 2 listeners or backup listeners.*

We next observe that after the first step of Part II (i.e., after the disseminate2 routines) each set in $Psets$ is knowledgeable with respect to some subset of the values V:

Lemma 3. *There exists a partition $V_1, \dots V_c$ of the values in V such that after the* disseminate2 *routines (i.e., by line 15), each set $Psets[i]$ is knowledgeable with respect to V_i, $1 \leq i \leq c$.*

The proof for this claim follows from the fact that the disseminate2 routine successfully transmits the values from the listeners (or backup listeners) to the remaining nodes in the set; since every node's identifier is unique, there will be some round during the disseminate2 routine in which each pair of nodes is listening on a different channel, and hence will receive the appropriate set of values. Finally, we observe that the combine routine successfully merges knowledgeable sets, which concludes the proof:

Lemma 4. *If P_i and P_j are knowledgeable sets with respect to some sets of values V_i and V_j, then set $P \leftarrow$* combine$(P_i, P_j, \dots)$ *is knowledgeable with respect to $V_i \cup V_j$.*

This fact follows from the correctness of the disseminate2 routine. We thus conclude:

Theorem 3. *Let $\mathcal{A}$ be the deterministic oblivious algorithm constructed by* Gossip. *If $\epsilon < 1/c$, then $\mathcal{A}$ solves the ϵ-gossip problem in time $O(n(1-\epsilon)/\epsilon c^2)$.*

7 The Multi-channel Adversary ($t < c$)

The algorithm described in Section 5 tolerates an adversary that can disrupt one channel in each round. The algorithm naturally extends to an adversary that can disrupt $t < c$ channels. (Again, for the purpose of brevity, we focus on the case where $\epsilon < t/c$, as this is the more interesting case. The case of $\epsilon \geq t/c$ is described in Section 8.) The overall algorithm maintains the same structure as that presented in Section 5: in the first part, the nodes transmit their values to a set of listeners; in the second part, the listeners become transmitters and create ever-expanding knowledgeable sets.

Part I. First, we assign $t+1$ listeners to each channel, instead of 2 listeners. The InfoTransfer routine is modified as follows: The processes are divided into $\epsilon n/t$ sets of size t/ϵ (instead of ϵn sets of size $1/\epsilon$). Each of these sets is subdivided into subsets of size $c/(t+1)$ (instead of size 2). All $\binom{t(t+1)/c\epsilon}{t+1}$ combinations of subsets are scheduled. This ensures that any combination of $t+1$ nodes in a set broadcast in the same round, and thus that there are at most $t \cdot \epsilon n/t$ nodes that fail to transmit their value. The resulting running time is $O\left(\frac{ne^{t+1}}{c\epsilon^t}\right)$ (approximating the binomial and the fact that $t < c$). Notice that the resulting schedule can be reduced to a $(t+1)$-hypergraph, in the same manner that the schedules for $t = 1$ could be reduced to a graph. If this construction is optimal, then it corresponds to a hypergraph-generalization of a Turán Graph. Finding such an entity (and proving it optimal) remains an open problem in extremal graph theory.

Part II. In the second part, the listeners disseminate the information to groups of nodes. The basic routine here is a disseminate[t + 1] routine that uses $t + 1$ channels to distribute data from $t + 1$ listeners to a set that becomes knowledgeable. Each of $t + 1$ listeners transmits on one channel throughout the dissemination phase; the rest of the nodes in the set are scheduled to listen on different channels in different rounds such that any set of $t + 1$ nodes is scheduled in some round to listen on different channels.

To accomplish this, we need to introduce an additional tool: selectors, as introduced by Komlos and Greenberg [20] (the term "selector" was coined later by [21]). Let $\mathcal{S}$ be a family of sets, where each $S \in \mathcal{S}$ is a subset of $[1, \ldots, n]$. For integer $k \leq n$, we say that $\mathcal{S}$ is a $(n, k, 1)$-**selector** if for every set $A \subseteq [1, \ldots, n]$ where $|A| = k$, there exists a set $S \in \mathcal{S}$ such that $|S \cap A| = 1$. In [20], it was shown that there exist $(n, k, 1)$-selectors of size at most $O(k \log n/k)$, and [22] shows how to explicitly construct selectors of size $O(k\mathsf{polylog}(n))$. For this section, we use the existential bounds from [20], and assume that for all $k \leq n$, $\mathcal{S}_k$ is a family of selectors of size $O(k \log n/k)$.

The schedule is constructed recursively. We define $T(c')$ recursively to be the number of rounds needed to construct the schedule for c' channels. In the beginning, we are constructing a schedule for all $c' = t + 1$ channels. If $c' = 2$, then the recursion terminates: schedule the remaining nodes to listen on those two channels as per the disseminate2 routine, i.e., each node chooses a channel based on its identifier. This takes $T(2) = O(\log n)$ rounds. If $c' > 2$, then we use the family of selectors $\mathcal{S}_{c'}$: for each set $S \in \mathcal{S}_{c'}$, schedule the nodes in S to listen on one channel for $T(c' - 1)$ rounds, and recursively schedule the remaining nodes on the remaining $c' - 1$ channels. For each set in $\mathcal{S}_k$ this takes $T(c' - 1)$ rounds, and thus $T(c') = |\mathcal{S}_{c'}|T(c' - 1)$. Since selector $\mathcal{S}_{c'}$ is of size at most $O(c' \log n/c')$, we conclude that the entire schedule for $T(t + 1)$ is (roughly) $O((t + 1)^t \log^t n)$. To see that it satisfies the desired property, consider any subset of $t + 1$ nodes: by definition, exactly one node is selected by one of the sets in $\mathcal{S}_{t+1}$; at the next step of the recursion, one node is selected by a set in $\mathcal{S}_t$, one by a set in $\mathcal{S}_{t-1}$, and so on, until the recursion bottoms out at the simple two-channel case. Thus there exists some round in which all $t + 1$ nodes listen in the same round.

The remaining generalizations of the algorithm from Section 5 are straightforward: the combine routine merges sets S_1 and S_2 as follows: first, it chooses $(t + 1)(t + 1)$ witnesses from set S_1 and runs $t + 1$ iterations of disseminate$[t + 1]$ to set S_2; it then chooses $(2t+1)(t+1)$ witnesses from set S_2 and runs $2t+1$ iterations of disseminate$[t+1]$ to set S_1. (By contrast, in the $t = 1$ case, there were two pairs of witnesses chosen for the first dissemination and three pairs of witness chosen for the second.) It should be noted that using selectors here too would result in improved performance. Thus each combine costs $O((t+1)^{t+1} \log^t n)$. Since each combine uses $t+1$ channels, it requires $\frac{c}{t+1} \log c$ iterations of the main loop to combine all c knowledgeable sets.

Noting that $t < c < n$, we thus conclude that the total running time of the gossip protocol is:

$$O\left(\frac{ne^{t+1}}{c\epsilon^t} + c(t + 1)^t \log^{t+1} n\right)$$

Lower Bound. Proving a matching lower bound for the general case remains an open question. We can prove that the result is optimal for the natural case of $\epsilon = t/n$, that is, trying to disseminate the maximum possible number of values. Specifically, we claim

$\binom{n}{t+1}/\binom{c}{t+1}$ rounds to be necessary for all but t nodes to transmit even once without being disrupted. The numerator follows from the observation that in order to transmit all but t values without interference, there must exist, for each combination of $t+1$ processes, a round in which all $t+1$ processes transmit concurrently (otherwise, the t adversaries can always interference when any of these processes transmit). The denominator follows from the observation that at most $\binom{c}{t+1}$ unique sets of $t+1$ processes can transmit concurrently on c channels during a single round. This fraction simplifies directly to $\Theta(\frac{n^{t+1}}{c^{t+1}})$, matching our above bound for InfoTransfer (within a factor of $O(e^t)$). Again, notice that proving a hypergraph-generalization of Turán's Theorem would result in an immediate lower bound. (See [23, 24] for more on hypergraph generalizations of Turán's Theorem.)

8 Achieving ϵ-Gossip for Large ϵ

In this section, we describe an algorithm to solve ϵ-gossip when $\epsilon > t/c$.

In the case where $t = 1$, we again divide the protocol into a transmission phase and a dissemination phase. In the transmission phase, we attempt to ensure that $(1-\epsilon)n$ values are known to a set of $6c$ processes. This is accomplished by assigning six listeners to each channel, and dividing the nodes into groups of size c; each group is assigned one round to broadcast for $(1-\epsilon)n/(c-1) \leq n/c$ rounds, ensuring that in each round $c-1$ nodes succeed. The listeners then exchange their values amongst themselves using the combine routine described in Section 5. The total running time of the transmission phase is $O((1-\epsilon)n/(c-1) + \log^2 c)$.

In the dissemination phase, we repeat the following twice, each time with a disjoint set of c "listeners" from the previous phase: each of the "listeners" broadcasts on one of the c channels, and each of the remaining nodes chooses a channel to listen on in each round using the "base-c" representation of its identifier. Since each identifier is unique, we can be sure that for any pair of nodes, in one of these rounds the two nodes choose different channels to listen on, and hence at least one receives the appropriate set of values. The total running time for the dissemination phase is $O(\log_c n)$. Thus the final overall running time of both phases is $O((1-\epsilon)n/(c-1) + \log_c n + \log^2 c)$.

When $t > 1$, this strategy generalizes in the natural way: we assign $(2t+1)(t+1)$ listeners in the transmission phase, and c nodes broadcast in each round; at least $c-t$ of them succeed, resulting in at least $(1-\epsilon)n$ values being received by listeners since $(1-\epsilon)n/(c-t) \leq n/c$; as before, the combine routine, generalized for the multi-channel adversary, is then used to combine the data. The total running time in this case is $(1-\epsilon)n/(c-t) + O(c(t+1)^t \log^{t+1} n)$.

9 Byzantine Adversary

The algorithms described in Sections 5 and 7 tolerate an adversary that can disrupt communication, but not an adversary that can directly corrupt an "honest" player. It is easy, however, to extend our algorithm to tolerate a Byzantine adversary that corrupts up to t honest players. Each corrupt player can either disrupt a channel or send a message in each round. (Thus, up to t channels can be disrupted, as in the previous case.)

The main modification involves the transmission phase: more listeners are needed on each channel, as some may be Byzantine. Instead of $t + 1$ listeners on each channel, we assign $(2t + 1)(t + 1)$. We then run the disseminate and combine routines $2t + 1$ times, each time with a different set of $t + 1$ listeners representing each channel. An honest process accepts a value as authentic only if it was received in at least $t + 1$ of the $(2t + 1)$ runs of disseminate and combine. The running time is increased by a factor of $\Theta(t)$. With further care, the number of listeners can be reduced. However, we conjecture that the problem is solvable only if $n = \Omega(tc)$, for example, in the case where $t = c - 1$.

10 Open Questions

The problems discussed introduce several new directions for future research. First, it remains to close the gap between the upper and lower bounds in the case of a multi-channel adversary. Such a result would have interesting connections to a hypergraph generalization of Turán's Theorem, an open problem in extremal graph theory.

Second, adaptive algorithms can likely achieve better performance than oblivious algorithms. It remains an interesting open question to determine how much efficiency adaptiveness provides. Similarly, it is possible to achieve better performance using a randomized algorithm, at the cost of some probability of failure. A trivial randomized algorithm can solve $1/n$-gossip in $O(n \log n)$ time (w.h.p.). Is it possible to do better?

Third, we believe the techniques developed here for single-hope networks extend to multi-hop networks. Fourth, prior research on the problem of malicious interference has assumed that the adversary can cause only a bounded number of collisions. It may be interesting to consider the possibility of bounded collisions in a multi-channel network.

Finally, this paper considers an adversary who wants to prevent communication. Other research (e.g., [25]) has considered an eavesdropper who wants to compromise the secrecy of the information (but who may not disrupt communication). This leaves open the question of whether it is possible to achieve reliable *and secret* communication in a multi-channel network in the presence of a malicious and disruptive adversary.

References

1. Koo, C.Y.: Broadcast in radio networks tolerating byzantine adversarial behaviour. In: Proceedings of the 23rd Symposium on Principles of Distributed Computing (PODC), pp. 275–282 (2004)
2. Bhandari, V., Vaidya, N.H.: On reliable broadcast in a radio network. In: Proceedings of the 24th Symposium on Principles of Distributed Computing (PODC), pp. 138–147 (2005)
3. Bhandari, V., Vaidya, N.H.: On reliable broadcast in a radio network: A simplified characterization. Technical report, University of Illinois at Urbana-Champaign (May 2005)
4. Pelc, A., Peleg, D.: Feasibility and complexity of broadcasting with random transmission failures. In: Proceedings of the 24th Symposium on Principles of Distributed Computing (PODC), pp. 334–341 (2005)
5. Bhandari, V., Vaidya, N.H.: Reliable broadcast in wireless networks with probabilistic failures. In: Proceedings of the 26th Conference on Computer Communications (Infocom), pp. 715–723 (2007)

6. Gilbert, S., Guerraoui, R., Newport, C.: Of malicious motes and suspicious sensors: On the efficiency of malicious interference in wireless networks. In: Shvartsman, A.A. (ed.) OPODIS 2006. LNCS, vol. 4305, pp. 215–229. Springer, Heidelberg (2006)
7. Koo, C.Y., Bhandari, V., Katz, J., Vaidya, N.H.: Reliable broadcast in radio networks: The bounded collision case. In: Proceedings of the 25th Symposium on Principles of Distributed Computing (PODC), pp. 258–264 (2006)
8. Kyasanur, P., Vaidya, N.H.: Capacity of multi-channel wireless networks: Impact of number of channels and interfaces. In: Proceedings of the 11th Annual International Conference on Mobile Computing and Networking (Mobicom), pp. 43–57 (2005)
9. Bhandari, V., Vaidya, N.H.: Connectivity and capacity of multi-channel wireless networks with channel switching constraints. Technical report, University of Illinois at Urbana-Champaign (January 2007)
10. Mitola, J.: Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio. PhD thesis, Royal Institute of Technology, Sweden (2000)
11. Krishnamurthy, S., Chandrasekaran, R., Mittal, N., Venkatesan, S.: Brief announcement: Synchronous distributed algorithms for node discovery and configuration in multi-channel cognitive radio networks. In: Dolev, S. (ed.) DISC 2006. LNCS, vol. 4167, pp. 572–574. Springer, Heidelberg (2006)
12. Krishnamurthy, S., Thoppian, M., Kuppa, S., Chanrasekaran, R., Venkatesan, S., Mittal, N., Prakash, R.: Time-efficient layer-2 auto-configuration for cognitive radios. In: Procedings of the International Conference on Parallel and Distributed Systems (PDCS), pp. 459–464 (2005)
13. Alon, N., Bar-Noy, A., Linial, N., Peleg, D.: A lower bound for radio broadcast. Journal of Computer and System Sciences 43(2), 290–298 (1992)
14. Bar-Yehuda, R., Goldreich, O., Itai, A.: On the time-complexity of broadcast in multi-hop radio networks: an exponential gap between determinism and randomization. Journal of Computer and System Sciences 45(1), 104–126 (1992)
15. Czumaj, A., Rytter, W.: Broadcasting algorithms in radio networks with unknown topology. In: Proceedings of the 44th Symposium on Foundations of Computer Science (FOCS), pp. 492–501 (2003)
16. Kowalski, D.R., Pelc, A.: Time of deterministic broadcasting in radio networks with local knowledge. SIAM Journal on Computing 33(4), 870–891 (2004)
17. Diks, K., Pelc, A.: Almost safe gossiping in bounded degree networks. SIAM Journal on Discrete Mathematics 5(3), 338–344 (1992)
18. Kranakis, E., Krizanc, D., Pelc, A.: Fault-tolerant broadcasting in radio networks. Journal of Algorithms 39(1), 47–67 (2001)
19. Turán, P.: On an extremal problem in graph theory. Matematicko Fizicki Lapok 48 (1941)
20. Komlos, J., Greenberg, A.: An asymptotically fast non-adaptive algorithm for conflict resolution in multiple access channels. Transactions on Information Theory 31(2), 302–306 (1985)
21. Bonis, A.D., Gasieniec, L., Vaccaro, U.: Optimal two-stage algorithms for group testing problems. SIAM Journal on Computing 34(5), 1253–1270 (2005)
22. Indyk, P.: Explicit constructions of selectors and related combinatorial structures, with applications. In: Proceedings of the 13th Symposium on Discrete Algorithms (SODA), pp. 697–704 (2002)
23. de Caen, D.: Extension of a theorem of Moon and Moser on complete subgraphs. Ars Combinatoria 16, 5–10 (1983)
24. Sidorenko, A.F.: What we know and what we do not know about Turán numbers. Graphs and Combinatorics 11(2), 179–199 (1995)
25. Miller, M.J., Vaidya, N.H.: Leveraging channel diversity for key establishment in wireless sensor networks. Technical report, U. of Illinois at Urbana-Champaign (December 2005)