A Disruption-Resistant MAC Layer for Multichannel Wireless Networks*

Henry Tan, Chris Wacek, Calvin Newport, and Micah Sherr

Georgetown University
Washington, DC, USA
{ztan,cwacek,cnewport,msherr}@cs.georgetown.edu

Abstract. Wireless networking occurs on a shared medium which renders communication vulnerable to disruption from other networks, environmental interference, and even malicious jammers. The standard solution to this problem is to deploy coordinated spread spectrum technologies that require pre-shared secrets between communicating devices. These secrets can be used to coordinate hopping patterns or spreading sequences. In this paper, by contrast, we study the local broadcast and unicast problems in a disrupted multichannel network with *no* pre-shared secrets between devices. Previous work in this setting focused on the special case of a single pre-designated sender in a single hop network topology. We consider in this paper, for the first time, upper and lower bounds to these problems in multihop topologies with multiple senders. To validate the potential real world application of our strategies, we conclude by describing a general purpose MAC protocol that uses the algorithms as key primitives, and validates its usefulness with a proof-of-concept implementation that runs the protocol on commodity hardware.

Keywords: wireless, broadcast, jamming.

1 Introduction

Wireless networks operate over a shared medium. This leaves them vulnerable to message loss due to (often unpredictable) disruption, such as contention from other networks, unrelated electromagnetic noise, and in some cases even malicious jamming. The standard solution to these issues in real deployments is to use *coordinated spread spectrum strategies*, such as frequency hopping spread spectrum (FHSS) [18, 27, 28], in which devices evade disruption by hopping between channels, and direct-sequence spread spectrum (DSSS) [5, 10], in which devices modulate their signal over additional frequencies to gain robustness.

A key property of these existing spread spectrum strategies is that they require the communicating devices to use pre-shared secrets (i.e., to synchronize hopping or signal spreading). In recent years, however, researchers from both the theory and systems communities have noted the need for reliable spread spectrum strategies that work in

* This work is supported in part by NSF grants CNS-1149832, CNS-1064986, CNS-1223825, CNS-1445967 and CCF 1320279, and the Ford Motor Company University Research Program. The findings and opinions described in this paper are those of the authors, and do not necessarily reflect the views of the funding parties.

M.K. Aguilera et al. (Eds.): OPODIS 2014, LNCS 8878, pp. 202–216, 2014.

the *absence* of pre-shared secrets (the so-called, *uncoordinated shared spectrum* setting) [6, 9, 14–16, 19, 24–26]. Such uncoordinated algorithms are useful to the increasingly common case where an ad hoc collection of devices need to reliably coordinate in a crowded environment. The existing work cited above studies how to generalize both FHSS and DSSS strategies to work without pre-shared secrets. In most cases, however, it focuses on the scenario of a single sender in a single hop network. Generalizing these strategies to work in general network topologies with arbitrary message arrivals—e.g., what would be required to implement a general-purpose MAC layer—was identified as an open question. This paper answer it.

Results. We prove new upper and lower bounds for two key communication primitives: *broadcast* (a device must deliver a message to all its neighbors in the network topology) and *unicast* (a device must deliver a message to a single known neighbor). As in existing work [6, 19, 25, 26], we model a crowded band of the wireless spectrum with n nodes having access to $\mathcal{C} \geq 1$ channels. In each round, each node can participate on a single channel. An adversary can choose up t channels (for some fixed $t < \mathcal{C}$) to *disrupt* locally at each receiver, preventing communication. This adversary incarnates the diversity of different sources of unpredictable interference that plague real deployments. We model the network topology with a graph, where the nodes correspond to the devices and edges to links. We assume that for both broadcast and unicast, messages arrive at arbitrary nodes at arbitrary times, and we require that randomized algorithms solve the relevant problem with high probability, abbreviated to w.h.p., in n.

We begin by describing a randomized broadcast algorithm that delivers a message, with high probability, from a sender to its (unknown) neighbors in the network in $O\big(\frac{\mathcal{C}}{\mathcal{C}-t}\Delta(\log(\Delta/\mathcal{C})+1)\log n\big)$ rounds, where Δ is the maximum degree in the topology graph (and therefore a measure of the *worst case* amount of nearby contention). The core strategy in this algorithm and its analysis is to use uncoordinated frequency hopping for a sufficient amount of time to ensure that nearby nodes have an opportunity to receive the message—regardless of the behavior of the disruption adversary. Note that we add 1 to the $\log(\Delta/\mathcal{C})$ factor in the asymptotic complexity to avoid a factor of 0 when $\Delta = \mathcal{C}$ (in our algorithm, the relevant term replaced with 1 when $\Delta = \mathcal{C}$).

Notice, in many cases, the actual amount of nearby contention might be much smaller than the worst-case. Motivated by this reality, we proceed with our primary technical result: a randomized *adaptive* broadcast algorithm that assumes a natural geographic constraint on the topology (see Section 2), and in exchange guarantees to solve broadcast for each node u in $O\big(\frac{\mathcal{C}t}{\mathcal{C}-t}\delta_u \log^3 n(\log(\Delta/C)+1)\big)$ rounds, where δ_u describes the actual amount of contention local to u (as noted: in practice δ_u might be much smaller than Δ). The core strategy in this algorithm is to have broadcasting devices participate in repeated iterations of local leader election competitions. If a device succeeds in becoming a leader, its local contention is small, and it can terminate confident that it successfully delivered its message. To obtain the δ_u factor in the time complexity, we prove that the leader election competition is fair—a given sender's probability of winning is inversely proportional to the number of nearby competing senders.

We then describe a randomized unicast algorithm that guarantees delivery of a message from u to a known neighbor v in $O\big(\frac{\mathcal{C}t'}{\mathcal{C}-t'}\log\Delta\log\mathcal{C}\log n\big)$ rounds, where t' bounds the *actual* amount of disruption at v. This algorithm uses a similar

uncoordinated frequency hopping strategy as our non-adaptive broadcast solution. The algorithm adapts to the actual amount of disruption (and not the worst-case) by testing different estimates. This strategy works because the algorithm is sending to a known receiver, and it can therefore use acknowledgments to know when it succeeded. In the interest of space, we defer many of our proofs and pseudocode to the full version [8].

We conclude our theoretical analysis by proving that our broadcast bounds are optimal within polylogarithmic factors for large and small δ (contention) values. We then turn our attention to our claimed practical motivation: that these primitives can aid the design of uncoordinated but reliable MAC layers. To validate this claim, we describe a general purpose MAC protocol that uses our algorithms as key primitives. The MAC protocol implements a name service that reliably discovers nearby devices, and then provides broadcast and unicast communication. It also guarantees a form of link layer authentication that does not rely on a public key infrastructure: once a pair of neighboring honest devices begin to communicate, a malicious node cannot spoof messages on this link. We then describe a proof-of-concept implementation of this link layer protocol using commodity 802.11 hardware, and the Click Modular Router [11] and FreeMAC [23] (a modified Atheros 802.11 driver) software. Our testbed evaluation validates that our algorithmic strategies can be implemented in practice and yield reliability (at the cost of performance).

Related Work. The most relevant related work on uncoordinated shared spectrum protocols focuses on delivering messages from a designated source to receivers in a single-hop version of our disrupted multi-channel network model. This research direction began with Strasser et al.'s UFH algorithm [25], which delivers k small message fragments from a single sender to one or more local receivers in $O\big(\frac{\mathcal{C}^2}{\mathcal{C}-t}k\log(n)\big)$ rounds with high probability. Erasure coding and clever use of the channels when t is small improved this cost to $O\big(\frac{\mathcal{C}t}{\mathcal{C}-t}(k+\log(n))\big)$ [24, 26], while our recent work improved this result further to $O\big(\frac{\mathcal{C}}{\mathcal{C}-t}(k+\log(n))\big)$ (under certain assumptions) by recruiting successful receivers to help propagate the message faster [6].

A related problem in the theory literature studies reliable local communication in a network with a *single* channel and a *resource-bounded* adversary causing disruption. Constant-competitive throughput for local communication is possible in this setting for both single hop [1, 7, 20] and multihop [21] networks. These results leverage different techniques than those used in this paper (which focuses on uncoordinated frequency hopping), but are motivated by the same real world issues surrounding shared spectrum.

2 Model and Problems

We model contended shared spectrum using the *t-disrupted* model, which is parameterized by $0 \leq t < \mathcal{C}$, and describes a wireless network with $\mathcal{C}$ communication channels, up to t of which might be *disrupted* locally at each receiver in each round, preventing communication at that receiver on those channels. As detailed below, the disruption decisions are made by a bounded adversary the incarnates the diversity of unpredictable interference possible in shared spectrum settings.

Network Topology. To describe the network topology, we fix an undirected graph $G = (V, E)$ with diameter D and a maximum degree upper-bounded by a known parameter

Δ, where the vertices in V correspond to the $n = |V|$ wireless devices, which we call *nodes*. Let $N(u)$, for $u \in V$, be the neighbors of u in G. We assume nodes know only a polynomial upper bound on n and do not know G. The t-disrupted model is parameterized by $\mathcal{C}$ (the number of available channels) and t (the number of channels that can be concurrently disrupted), where $0 \leq t < \mathcal{C}$, and both parameters are known to the nodes. To simplify our asymptotic notation in some of the results that follow, we assume without loss of generality that $\Delta \geq \mathcal{C}$.[1]

Executions. We assume executions in our model proceed in synchronous rounds labeled $1, 2, 3....$ To capture unpredictable disruption we assume an adversary can disrupt up to t channels per node, per round. Formally, for each round r and node u, let $adv(u, r)$ be an array of size $\mathcal{C}$ describing how the adversary behaves on each channel with respect to u in r. Let $[\mathcal{C}] = 1, 2, ..., \mathcal{C}$ be the set of available channels. For each $c \in [\mathcal{C}]$: $adv(u, r)[c] = \perp$ indicates the adversary does not affect c with respect to u in this round; $adv(u, r)[c] = \pm$, on the other hand, indicates that the adversary disrupts c. We similarly define $disp(u, r) = \{c : adv(i, r)[c] \neq \perp\}$. We require that $|disp(u, r)| \leq t$. If $c \in disp(u, r)$, we say c is *disrupted* w.r.t. u and r.

To define communication behavior, fix a node u and round r. At the beginning of r, u chooses a channel from $[\mathcal{C}]$ to participate on, deciding either to *transmit* or *listen*. If u decides to transmit a message, then it cannot also receive a message in r (i.e., the channels are half-duplex). If u listens, the outcome depends on the adversary and its neighbors. In more detail, if $adv(u, r) = \pm$, u receives nothing. If $adv(u, r) = \perp$ and exactly one neighbor of u transmits on c during r, u receives this message. Otherwise, if multiple neighbors transmit, u receives one of the messages, chosen arbitrarily, or nothing (concurrent broadcasts may be lost to undetectable collision). It follows, therefore, that nodes in this model must handle *both* contention from their own network and disruption from outside. When modeling unicast communication from a node u to a single known neighbor v in the graph, we assume the presence of link layer acknowledgements that allow u to discover when v has successfully received its message.[2]

We define $t' \leq t$, with respect to an execution, to be the maximum value of $|disp(u, r)|$ over all u and r. That is, t' is the actual amount of concurrent disruption experienced in an execution, whereas t is the worst-case possibility. We typically assume nodes do not know t' in advance. Similarly, when studying executions of *communication algorithms* (i.e., algorithms in which nodes are passed messages to communicate to nearby nodes), we define δ_u for each u, to be the total number of nodes in $N(u)$ that are passed a message to communicate in the execution. Clearly, $\delta_u \leq |N(u)| \leq \Delta$.

Finally, we bound our adversary's power by assuming it is an arbitrary randomized algorithm that generates $adv(u, r)$ for all u at the beginning of each r. When defining this array, the adversary can leverage knowledge of G, the algorithm being run by the

[1] We say this holds w.l.o.g. because Δ is an upper bound. To explicitly handle the case of smaller Δ in our time complexity results, it is necessary only to replace the linear Δ factor in the broadcast bound with the slightly more messy notation: $\max\{\Delta, \mathcal{C}\}$.

[2] We omit in this model the case where v receives a message but u does not receive the corresponding acknowledgment due to disruption—we assume a full channel is disrupted or it is not disrupted at all. That is, we always assume the worst-case, that if there is any disruption for a given transmission, everything is lost. This simplifies the analysis of our algorithms.

nodes, and the history of the execution through $r-1$. It does not, however, know in advance u's random choices for r.

Graph Restrictions. When studying multihop radio networks, it is common to assume some geographic constraint on the communication topology. For the adaptive broadcast algorithm in this paper, we assume the constraint introduced by Daum et al. [3], that generalizes many of the constraints typically assumed in the wireless algorithms literature. In more detail, let $\mathcal{R} = \{R_1, R_2, ..., R_k\}$ be a partition of the nodes in G into regions such that the sub-graph of G induced by each region R_i is a clique. The corresponding *clique graph* (or *region graph*) is a graph $G_{\mathcal{R}}$ with one node u_i for each $R_i \in \mathcal{R}$, and an edge between u_i and u_j iff $\exists v \in R_i, w \in R_j$ such that v and w are connected in G; we write $R(v)$ for the region that contains v. In this paper, when we say a graph G satisfies the *regional clique decomposition* property, we mean that it can be partitioned into cliques $\mathcal{R}$ such that the maximum degree of $G_{\mathcal{R}}$ is upper bounded by some constant parameter. Notice, this model generalizes many common geometric network models, including unit ball graphs with constant doubling dimension [12], which was shown [22] to generalize (quasi) UDGs [2, 13].

The Broadcast and Unicast Problems. In this paper, we study upper and lower bounds for the *broadcast* and *unicast* problems in the t-disrupted model. Both problems assume the presence of a *message process* that passes broadcast/unicast messages to the network nodes. We place no restrictions on this message process besides the requirement that it waits for a node to indicate it is done processing its current message before passing the next message (i.e., we do not address queuing issues in this paper).

The *broadcast* problem requires a node u, when passed a message m from the message process, to attempt to deliver m to the nodes in $N(u)$. When it is done, it indicates this to the message process. We say a given algorithm implements a broadcast service with latency T rounds if the following two properties hold *with high probability* (i.e., with probability at least $1 - n^{-c}$, for a provided constant $c \geq 1$) for each time a node u is passed a message m: (1) u finishes transmitting m within T rounds of being passed m; (2) every node in $N(u)$ receives m during this interval.

The *unicast* problem requires a node u, when passed a message m and node $v \in N(u)$, to deliver m to v. The definition of u implementing a unicast service with latency T rounds is the same as for broadcast except it only need deliver the message to v.

3 Upper and Lower Bounds

In this section we present our upper and lower bounds for the broadcast (both non-adaptive and adaptive variants) and unicast problems. Due to space constraints, we defer some of the proofs and pseudocode to the full version.

3.1 Non-adaptive Broadcast Algorithm

We call our first broadcast algorithm *non-adaptive* as its time complexity is defined with respect to the worst-case contention. It works as follows: Each node u groups rounds into *phases*, each of which contains $\max(\lceil \log(\Delta/\mathcal{C}) \rceil, 1)$ rounds. When u is

passed a message m to send, it waits until the beginning of the next phase, then considers itself *active* for the next $T_p = \Theta(\frac{\mathcal{C}}{\mathcal{C}-t}\Delta \log n)$ phases (we define the constants for T_p later). After these phases are done, u considers its broadcast complete and returns to *inactive* status. During each round r, node u, regardless of whether it is active or inactive, chooses a channel on which to participate with uniform independent randomness. If u is active, it decides to transmit m with probability $\frac{1}{2^k}$, where $k = (r \mod \max(\lceil \log(\Delta/\mathcal{C}) \rceil, 1)) + 1$; otherwise it listens. If u is inactive, it always listens. When u receives a broadcast message m' from another node for the first time, it passes it up to the higher layer message process.

Analysis. In the following analysis, and those that follow, we make use of these basic probability facts: (1) for $p \leq \frac{1}{2} : (1-p) \geq (\frac{1}{4})^p$; and (2) for $p > 0 : (1-p) < e^{-p}$. Fix some node u and some phase q. Let A_u be the set of active nodes that neighbor u in q (notice, A_u is fixed over q). Let p_u be the probability that u receives a message during q. We start by bounding p_u, treating separately the case where A_u is large (and therefore, many channels are likely to be occupied by active neighbors), and A_u is small (and therefore, few channels are occupied—requiring more time to find active neighbors).

Lemma 1. *Assume $|A_u| > \mathcal{C}$. It follows: $p_u \geq \frac{\mathcal{C}-t}{8\mathcal{C}}$.*

Lemma 2. *Assume $|A_u| = \mathcal{C}^{1-\epsilon}$, for $0 \leq \epsilon \leq 1$. It follows: $p_u \geq \frac{\mathcal{C}-t}{8\mathcal{C}^{1+\epsilon}}$.*

We now combine our lemmas to prove our main theorem.

Theorem 1. *Our algorithm implements a broadcast service with a latency of $O\big(\frac{\mathcal{C}}{\mathcal{C}-t}\Delta(\log(\Delta/\mathcal{C}) + 1)\log n\big)$ rounds.*

Proof. The latency follows from the definition of our algorithm, which attempts to deliver a message for $T_p = O(\frac{\mathcal{C}}{\mathcal{C}-t}\Delta \log n)$ phases, each containing $\max(\lceil \log(\Delta/\mathcal{C}) \rceil, 1)$ rounds. We are left to show that during this active period the message is delivered to all neighbors with probability at least $1 - 1/n$.

Fix some node v that is passed message m to broadcast. We will show that during the T_p phases in which v is active with m, every neighbor of v receives m, with high probability. To do so, we start by analyzing a particular neighbor u. For each of the T_p phases during which v is active with m, we call the phase *crowded* if u has more than $\mathcal{C}$ active neighbors. Similarly, we call the phase *sparse* if u has $\mathcal{C}^{1-\epsilon}$ active neighbors, for some $0 \leq \epsilon \leq 1$. Notice, it must be the case that either: (1) at least half of v's T_p active phases are crowded; or (2) at least half of v's T_p active phases are sparse.

Case 1: At least half the active phases are crowded. It follows that there are $T'_p \geq T_p/2 = \Theta(\frac{\mathcal{C}}{\mathcal{C}-t}\Delta \log n)$ phases where $|A_u| > \mathcal{C}$. In each such phase, Lemma 1 tells us that u receives *some* message with probability at least $\frac{\mathcal{C}-t}{8\mathcal{C}}$. Because active nodes behave uniformly, u receives v's message in particular with probability at least $\frac{\mathcal{C}-t}{8\mathcal{C}|A_u|} \geq \frac{\mathcal{C}-t}{8\mathcal{C}\Delta}$. We conclude that node u fails to receive v's message in all T'_p crowded phases with probability p_f, bounded as: $p_f \leq \big(1 - \frac{\mathcal{C}-t}{8\mathcal{C}\Delta}\big)^{T'_p} < (1/e)^{(c_1/8)\log n}$.

For any constant $c \geq 1$, we can choose a sufficiently large constant c_1 for the definition of T_p (and thus T'_p) such that this failure probability is no more than n^{-c}. (We will fix the particular c we need later in the proof.)

Case 2: At least half the active phases are sparse. In each of these T'_p phases, $|A_u| = \mathcal{C}^{1-\epsilon}$ for some $0 \leq \epsilon \leq 1$. The problem here is that this ϵ value can change from phase to phase as neighbors of u become active and inactive. To simplify notation, for the r^{th} sparse phase, let ϵ_r be the definition of ϵ and let A^r_u be the set of u's active neighbors. Following the same general approach as for case 1, we now apply Lemma 2 to determine that in phase r, u will receive *some* message with probability at least $\frac{\mathcal{C}-t}{8\mathcal{C}^{1+\epsilon_r}}$. In particular, due to uniformity, u will receive v's message with probability at least: $\frac{\mathcal{C}-t}{8\mathcal{C}^{1+\epsilon_r}|A^r_u|} = \frac{\mathcal{C}-t}{8\mathcal{C}^{1+\epsilon_r}\mathcal{C}^{1-\epsilon_r}} = \frac{\mathcal{C}-t}{8\mathcal{C}^2}$. Therefore, u fails to receive v's message in all T'_p phases with probability p_f, bounded as: $p_f \leq \left(1 - \frac{\mathcal{C}-t}{8\mathcal{C}^2}\right)^{T'_p} < (1/e)^{\frac{c_1\Delta}{8\mathcal{C}}\log n}$.

Since we assumed in Section 2 that $\Delta \geq \mathcal{C}$, we can replace $\Delta/\mathcal{C}$ with some $c' \geq 1$, and conclude by choosing c_1 to be sufficiently large such that this failure probability is no more than n^{-c}, for any constant c. We can now tweak our constants in T_p to ensure that in both cases above, our failure probability is no more than n^{-2}. It follows that u receives m with probability at least $1 - n^{-2}$. A union bound over v's neighbors tells us that every neighbor of v receives m with probability at least $1 - n^{-1}$, as needed.

3.2 Adaptive Broadcast Algorithm

The broadcast algorithm presented in the previous section has a time complexity that depends on the worst-case amount of local contention (as captured by Δ). In practice, Δ might be large compared to the actual amount of contention at a node u (i.e., δ_u). Here we present a broadcast algorithm that requires the network topology satisfy the regional clique decomposition property defined in our model section, and in return is able to replace a Δ with a δ_u factor in its complexity. It does so, however, at the cost of an additional factor of $t\log^2 n$. The adaptive solution, therefore, is applicable when the actual amount of contention is at least a factor of $t\log^2 n$ improved over the worst-case.

Algorithm Description. Our adaptive broadcast algorithm has each node u divide time into *iterations*. An iteration I is composed of 2 phases, a *knockout* phase followed by a *cleanup* phase. Nodes begin *inactive*. When a node obtains a message m for broadcast, it waits until the beginning of the next iteration before becoming *active*.

In the knockout phase of I, active nodes compete to become local leaders. To simplify our analysis, we present the subroutine run in this phase as a combination of a subroutine designed for a disruption-free single channel (DFSC) model (i.e., our model with $\mathcal{C} = 1$ and $t = 0$), and a simulator capable of simulating any DFSC subroutine in our more general t-disrupted setting. The DFSC knockout subroutine works as follow. Every node u divides time into $\max(\lceil\log(\Delta/\mathcal{C})\rceil, 1)$ epochs, each consisting of $\alpha\log n$ rounds, where α is a constant fixed in the proof below. For each epoch $e \in \{1, ..., \max(\lceil\log(\Delta/\mathcal{C})\rceil, 1)\}$, if u is active, it transmits with probability $p_e = 2^{-(\log\Delta - e + 1)}$ for each round in e. (i.e., there is one epoch for each probability in the sequence $1/\Delta, 2/\Delta, ..., 1/\mathcal{C}$.) In every round, if u chooses not to transmit, including the case where it is inactive, it listens. If u receives a message from another node, it becomes *inactive* for the rest of the iteration and restarts active in the next iteration.

The simulator strategy we use to simulate the DFSC knockout subroutine in our t-disrupted model works as follows. The simulator uses only the first $\hat{\mathcal{C}} = \min\{2t, \mathcal{C}\}$ channels. For each node u and simulated round, the simulator uses $T_s = \frac{\mathcal{C}t}{\mathcal{C}-t}\beta\log n$

real rounds, for a constant β we fix in the proof below. It begins by determining u's behavior for the simulated round according to the routine being simulated; i.e., it decides whether u transmits, and if so, what message m to transmit. It then spends T_s rounds executing u's decision. In each of these real rounds, u chooses a channel with uniform randomness. If it decided to transmit, it transmits m with probability 1, otherwise, it listens. At the end of T_s, the simulator simulates u's reception behavior. There are three cases: if u transmitted in this simulated round, then u simulates receiving nothing. If u decided to listen in this simulated round, and does not receive any messages during the T_s real rounds, it simulates receiving nothing. If u decided to listen in this simulated round, and received at least 1 message during the T_s real rounds, then it simulates receiving one of these messages (chosen arbitrarily if there are multiple).

During the subsequent cleanup phase in a given iteration, each node u runs the non-adaptive broadcast algorithm described in Section 3.1 with $\Delta = \mathcal{C}$, participating as an active node only if it began this phase active. If u begins the clean-up phase active, then after this phase concludes, it completes its broadcast and becomes inactive.

Analysis of the Adaptive Broadcast Algorithm. Our analysis below requires that we show certain properties hold throughout a full execution. For technical reasons, therefore, we must assume that executions are bounded by a number of rounds polynomial in n: say, $O(n^k)$ for some constant $k \geq 1$. In the following, let $\mathcal{R} = \{R_1, R_2, ..., R_\ell\}$ be the set of $\ell \leq n$ non-empty regions provided by our assumed regional clique decomposition property on G. When analyzing the knockout subroutine in the DFSC model, we make use of the following helpful notation. For round r, let e_r be the epoch number from $\{1, ..., \max(\lceil \log(\Delta/\mathcal{C}) \rceil, 1)\}$ associated with that round, and let $A(r)$ be the nodes active in the network at the beginning of round r (i.e., nodes broadcasting a message). For region $R_i \in \mathcal{R}$, let $W_i(r) = \sum_{v \in A(r) \cap R_i} p_{e_r}$; i.e., the sum of transmission probabilities of active nodes in R_i in round r (recall $p_j = 2^{-(\log \Delta - j + 1)}$ is the transmission probability used in the j^{th} epoch. Finally, let $N^+(u) = N(u) \cup \{u\}$.

We begin by studying the behavior of our DFSC knockout subroutine when executed in the DFSC model. Lemmas 3, 4 and 5 all apply to this scenario. Recall in the following statements that α is a constant used in defining the epoch length for this subroutine. We begin below by adapting a strategy introduced in [17] to prove that for each region R_i, W_i self-regulates to never grow too large.

Lemma 3. *For sufficiently large α, in an execution of the DFSC knockout subroutine, the following holds w.h.p.: for every round r and region $R_i \in \mathcal{R}$: $W_i(r) < 2$.*

We now leverage this result to show that every active node survives an execution of the knockout subroutine with probability proportional to its local contention.

Lemma 4. *For all $\alpha \geq 1$, the following holds for each node u starting a given execution of the DFSC knockout subroutine active: the probability $\hat{p}_u$ that u transmits alone before any node in $N(u)$,* or *that no node in $N(u)$ transmits, is bounded as $\hat{p}_u \in \Omega(1/\delta_u)$.*

Proof. Let u be some node that starts an execution of this subroutine active, C be the subset of u's neighbors that are also active, $C^+ = C \cup \{u\}$, and $x = |C^+|$. We cannot argue directly about the behavior of nodes in C during this execution because

their transmission behavior might also depend on their own neighbors (e.g., if a node in C is knocked out by one of u's two-hop neighbors, then this obviously affects its transmission probability). We will instead argue about their behavior in the absence of other nodes. In particular, for each $v \in C^+$ let b_v be the binary string where bit ℓ, indicated $b_v[\ell]$, equals 1 if and only if v would broadcast in round ℓ of iteration I, given its random bits,[3] under the assumption that v has not received any messages through the preceding $\ell - 1$ rounds of the iteration. Let $B^+ = \{b_v \mid v \in C^+\}$.

We now bound the probability that certain properties of this set of strings, each generated with independent randomness, hold. In particular, let $r_p = \min\{r' \mid \exists v \in C^+ : b_v[r'] = 1\}$; the first round with a transmission. If all strings in B^+ contain only 0's, we say that r_p is *undefined*. If r_p *is* defined, we say that its value is *good* if it occurs in a round corresponding to a broadcast probability less than $1/x$ (recall $x = |C^+|$), and is *bad* if it occurs in a higher probability round. We can now partition the space of possible outcomes for the generation of B^+ into three mutually exclusive portions: (1) r_p is undefined; (2) r_p is defined and good; (3) r_p is defined and bad.

We first prove that the size of the outcome space that satisfies condition 3 is small. In particular, for r_p to be defined for a bad epoch means that there was some previous epoch, $\hat{e}$, for which $p_{\hat{e}} = 1/(kx)$, for some $1 < k \leq 2$, and yet no node generated a 1 for the corresponding rounds in its bit string. Given that there are x nodes, the probability that no node generates a 1 for all $\alpha \log n$ rounds corresponding to this epoch $\hat{e}$ is $(1 - \frac{1}{kx})^{x\alpha \log n} \leq \frac{1}{n^{\alpha/2}}$. In other words: it is a small probability.

Now consider the case where r_p is defined and good. We want to calculate the size of the portion in the outcome space where $b_u[r_p] = 1$, and $b_v[r_p] = 0$ for all $v \in C$. To do so, we note that by the definition of *good*, the probability associated with r_p can be expressed as $1/(kx)$ for some $k > 1$. Let p_1 be the probability that exactly *one node* in C^+ selects 1 in a round associated with this probability $p_1 = \binom{x}{1}\frac{1}{kx}(1 - \frac{1}{kx})^{x-1} > \frac{x}{kx}(\frac{1}{4})^{\frac{x}{kx}} = \frac{1}{k}(\frac{1}{4})^{\frac{1}{k}}$. Similarly, let p_i be the probability that exactly i nodes in C^+ transmit in a round associated with this good probability. We upper bound this value as follows: $p_i = \binom{x}{i}\left(\frac{1}{kx}\right)^i\left(1 - \frac{1}{kx}\right)^{x-i} < x^i\left(\frac{1}{kx}\right)^i = \frac{1}{k^i}$ Let p_{2+} be the probability of more than 1 node in C_+ transmits in this round: $p_{2+} \leq \sum_{i=2}^{x} \frac{1}{k^i} \leq \frac{1}{k(k-1)}$.

Finally, we consider both the case where k is small and large.
For $1 \leq k \leq 2$: Define p_0 as the probability that no nodes in C^+ transmit in r_p. We note that $p_0 = (1 - \frac{1}{kx})^x \geq (\frac{1}{4})^{\frac{1}{k}} \geq \frac{1}{4}$, $p_1 \geq \frac{1}{4}$ and $p_{2+} = 1 - p_1 - p_0 \leq \frac{1}{2}$.
For $k > 2$: $p_1 \geq \frac{1}{2k}$ and $p_{2^+} \leq \frac{1}{k}$ (by our above equation).

It follows that for both possibilities for the value of k, the probability of two or more transmitters in r_p is no more than twice as likely as a single transmitter. Given that there is at least a single transmitter, $p_1 \geq 2p_{2+}$ implies that $p_1 \geq \frac{1}{3}$. Finally, if there is a single transmitter, by the uniform probabilities used in each round, the probability that this transmitter is u is $1/x$. We have shown, therefore, that in the portion where r_p is defined and good, $\frac{1}{3x}$ of this portion has u transmitting alone before any node in C.

We can now pull together the pieces. The event whose probability we are bounding with $\hat{p}_u$ contains the fraction of the outcome space in which condition (1) from above

[3] To define b_v we treat randomness here such that v has a pre-determined collection of random bits from which it extracts the needed randomness for its probabilistic choices.

holds, as well as the fraction where condition (2) holds and u broadcasts alone before any of its neighbors. We proved that the fraction where condition (3) holds is small, i.e., $\leq n^{-\alpha/2}$, and therefore most of the outcome space (e.g., at least a constant fraction, $\geq 1 - n^{-1}$ for $\alpha > 2$) is dominated by conditions (1) and (2). Furthermore, we proved that at least a $\frac{1}{3x}$ fraction of the condition (2) portion has u broadcast alone before its neighbors. Combined, $p_{(1)} + p_{(2)} * \frac{1}{3x} \geq (p_{(1)} + p_{(2)}) * \frac{1}{3x}$ and $p_{(1)} + p_{(2)} \geq 1 - n^{-1}$ provide our lemma statement.

We now show that if u transmits alone in its local neighborhood, it has a constant probability of finishing the subroutine execution active (as there is a constant probability that this message knocks out your neighbors for the remainder of the execution).

Lemma 5. *For sufficiently large α, if node u transmits in an execution of the DFSC knockout subroutine before any other node in $N(u)$ transmits, then with constant probability u ends this execution active.*

We now shift our focus from the behavior of the knockout subroutine in the DFSC model to our broadcast algorithm as a whole. If a node ends the knockout phase of a given iteration active, it has a constant probability of successfully completing broadcast in the subsequent clean-up phase. The key insight is that it is unlikely that more than $\approx \mathcal{C}$ nearby nodes finish this phase active, allowing the non-adaptive algorithm, run with maximum contention $\mathcal{C}$, to succeed.

Lemma 6. *If u is active at the end of the knockout phase of some iteration I of the non-adaptive broadcast algorithm, then with high probability every node that neighbors u receives u's message by the end of I.*

Our final step before our final theorem, is to prove that our simulator routine can successfully simulate a subroutine designed for the DFSC setting. The core insight in the below proof is that we use a significantly large number of real rounds per simulated round to ensure that if a single neighbor v of some u decides to broadcast in a simulated round (the key case), u will hear from v with high probability during the corresponding T_s real rounds.

Theorem 2. *With high probability, the DFSC simulation subroutine correctly simulates the DFSC model using $O(\frac{\mathcal{C}t}{\mathcal{C}-t} \log n)$ rounds for each simulated round.*

We can now pull together the pieces for our final theorem statement. The key insight is that there are two good cases with respect to an active node u coming out of the knockout phase: (1) no node in u's neighborhood transmitted (implying that there are only a small number of such nodes), or (2) u transmitted alone before any of its neighbors (implying, by our above lemma, that u is the only active node in its neighborhood with constant probability). In either of these cases, the clean-up phase has a good chance of helping u succeed. Because we previously proved that one of these two conditions occurs with probability in $\Omega(1/\delta_u)$, we can show that $O(\delta_u \log n)$ iterations is enough to ensure success for u with high probability.

Theorem 3. *The adaptive broadcast algorithm implements a broadcast service in the general model with latency of $O(\frac{\mathcal{C}t}{\mathcal{C}-t}\delta_u \log^3 n(\log(\Delta/\mathcal{C}) + 1))$ rounds.*

3.3 Broadcast Lower Bounds

We now prove that our broadcast algorithms are close to optimal (which we define to mean *within polylogarithmic factors in the network size*) for most parameter values. We begin with a bound that shows our non-adaptive broadcast algorithm is close to optimal for a node u when δ_u is large (i.e., close to Δ). We then adapt a result from [3] (see Theorem 4.1) to prove that our adaptive algorithm is close to optimal when δ_u is small (i.e., constant). A requirement of the algorithm is that it is regular, a definition which we take from [3]. An algorithm is regular is there exists a sequence of pairs (F_i, b_i) for $i \in \{1, 2, ...\}$ where each F_i is a probability distribution over $[\mathcal{C}]$ and b_i is a probability s.t. for each node u and local round r, u will select a frequency from F_r to transmit on with probability b_i, until it receives a message. Once it receives message, its behavior is no longer constrained.

Theorem 4. *Fix some algorithm $\mathcal{A}$ that implements a broadcast service with latency $f(n, \delta_u, \mathcal{C}, t)$, for each node u. It follows that $f(n, \delta_u, \mathcal{C}, t) = \Omega(\frac{\mathcal{C}}{\mathcal{C}-t}\delta_u)$.*

Theorem 5 (Adapted from [3]). *Fix some regular algorithm $\mathcal{A}$ that implements a broadcast service with latency $f(n, \delta_u, \mathcal{C}, t)$, for each node u. It follows that $f(n, \delta_u, \mathcal{C}, t) = \Omega(\frac{\mathcal{C}t}{\mathcal{C}-t}\log n)$.*

3.4 Unicast Algorithm

In the unicast problem, a node u attempts to deliver a message to a single *known* neighbor v. Because the destination is now known, we assume u can leverage *link layer* acknowledgments to determine when it is successful in its delivery. This added power allows us to adapt the time complexity to the actual amount of local disruption (captured by t') as opposed to the worst case (captured by t).

Algorithm Description. As in non-adaptive broadcast, we assume that each node u groups rounds into *phases* of size $\lceil \log \Delta \rceil$ rounds. It then divides these phases into *groups*, each containing $\lceil \log \mathcal{C} \rceil$ phases. When node u is passed a pair (m, v), indicating that it should send message m to neighbor v, it becomes *active* at the beginning of the next phase. It will remain active until it receives an acknowledgment from v (in practice we might also add a timeout equal to the worst case broadcast latency from Section 3.1). During round r of phase k of some group, u chooses a channel with uniform probability from the first $\min\{2^k, \mathcal{C}\}$ channels. If u is active, it transmits (m, v) with probability $1/2^r$, otherwise it listens. If i is *inactive*, it always listens. Larger k corresponds to larger guesses for t' (as it requires u to choose from among more channels). If at any point, u receives (m', u) from some neighbor v, u replies immediately with an acknowledgment.

Theorem 6. *Our algorithm implements a unicast service with a latency of $O\big(\frac{\mathcal{C}t'}{\mathcal{C}-t'}\log \Delta \log \mathcal{C} \log n\big)$ rounds.*

4 Evaluation: A Disruption-Resistant Link Layer Protocol

We demonstrate the utility of our algorithms with a link layer protocol that is robust against unpredictable disruption and operates on commodity 802.11 hardware. Our protocol consists of (1) a *name service* that maintains the list of local neighbors, performs

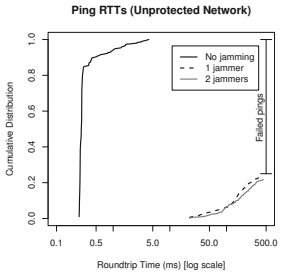

Fig. 1. Cumulative distribution of ICMP ping RTTs in an *unprotected* network

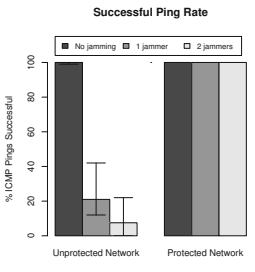

Fig. 2. Percentage of successful ICMP ping requests (with min/max ranges)

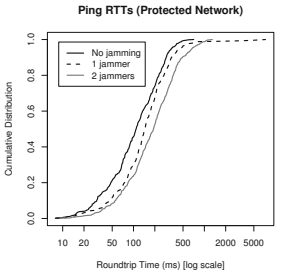

Fig. 3. Cumulative distribution of ICMP ping RTTs in a *protected* network

key exchange and key derivation, and serves as an address resolution protocol, based on the broadcast algorithm, and (2) an *authenticated communication service* that encrypts, signs, authenticates and sends messages, based on the unicast algorithm.

The disruption-resistant link layer leverages the broadcast protocol to conduct Diffie-Hellman (DH) [4] exchanges between nodes. Since the broadcast protocol guarantees, w.h.p., the reception of messages in the presence of a disrupting adversary, the adversary cannot prevent honest nodes' announced DH public keys from being received by their neighbors. When transmitting messages, a node includes a nonce that is encrypted with the receiver's public DH key. Only the intended receiver can decrypt the nonce and reply with an acknowledgment that contains the nonce, confirming that the message was received as no other node could have generated this response.

We developed a proof-of-concept implementation *using commodity 802.11 hardware* to demonstrate our protocol's efficacy as a robust communication primitive. Our implementation is built on top of the Click Modular Router [11] and FreeMAC [23], a modified Atheros 802.11 driver. We present the full details of our link layer protocol and its implementation in the full version. In the next section, we evaluate its ability to provide reliable communication even in the presence of malicious jammers.

4.1 Experimental Setup

Our testbed consists of 4 nodes $n1$ ... $n4$ and two *jammers* $j1$, $j2$. These reside in the same 4-by-6 meter room, run Linux 2.6, are equipped with PCMCIA Atheros 802.11b/g wireless controllers with AR5212 chipsets, and use the 802.11b bit rate of 2 Mbit/s.

Network Configurations. Since our protocol is a general link layer protocol, we compare it against standard 802.11 ad hoc mode (with acknowledgments and incremental backoff enabled), which we refer to as the *unprotected network*. We run our reliable MAC-layer protocols in the *protected network* setting. In both settings, we restrict the choice of channels to the non-overlapping frequencies: 2412Hz, 2437Hz, and 2462Hz.

Network Jamming. To maximize disruption, jammers continuously send packets. Jammers do not obey 802.11 backoff requirements in either network. We assume the adversary has knowledge of the communication protocols in use and adapts its jamming

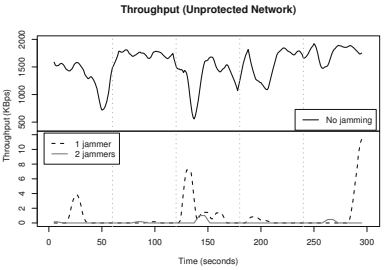

Fig. 4. Average throughput over time when operating in *unprotected* mode, with and without jammers

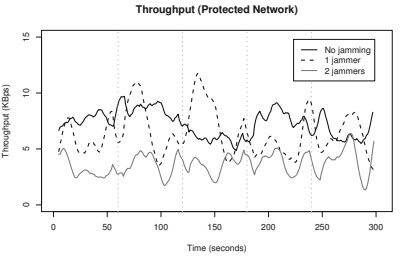

Fig. 5. Average throughput over time when operating in *protected* mode, with and without jammers

strategy accordingly. For the unprotected network, jammers operate on the channel used by the ad hoc network. For the protected network, jammers hop between the channels used by the protocol in a round robin fashion. When there are two jammers, they jam two of the three channels at any given time.

Metrics. The nodes periodically transfer large files over HTTP (i.e., via TCP) and send ICMP ping requests to measure roundtrip times (RTTs). Nodes $n1$ and $n3$ respectively receive large files over HTTP from nodes $n2$ and $n4$ in the presence of jammers $j1$ and $j2$. Ping measurements are conducted between node pairs $(n1, n2)$ and $(n3, n4)$.

We explore the performance of our disruption-resistant protocols by measuring the *roundtrip time* of ICMP-based ping measurements, the percentage of lost ping messages, and the effective *throughput* of the file transfers. To increase contention, the two pairs of nodes communicate simultaneously. All experiments are repeated five times.

4.2 Performance Results

Latency and Packet Loss. As shown in Figure 1, 802.11 ad hoc mode maintains ping RTTs of less than 0.250ms when there are no jammers. However, the performance of the network significantly degrades when even a single jammer is active. For instance, when only $j1$ is active, the minimum RTT of *successful* pings rises to 21ms, and the median increases to 153ms. The jammers are also able to cause significant packet loss in the unprotected network. As illustrated in Figure 2, with one (resp. two) active jammers, only 20% (resp. 5%) of pings are successful.

For comparison, the performance of ICMP pings in protected networks is shown in Figure 3. The protected network produces larger ping times – the median RTT without jamming is 109.5ms. However, performance degrades only slightly with jamming: in the worst case in which both $j1$ and $j2$ attempt to disrupt communication, the median RTT increases to 191.5ms. Importantly, as shown in Figure 2, *all pings are successful*.

Throughput. Figures 4 and 5 respectively show the throughput over time for the file transfer on the unprotected and protected networks, with and without jamming. Results are shown for 5 consecutive experiments (separated with dashed lines).

Without jamming, ad hoc mode outperforms our protocol. However, *with even a single jammer, the median throughput of the unprotected network drops to zero*. In contrast, while the protected network experiences a modest decrease when jammers become active, it maintains the ability to transmit data effectively. Without jammers, median throughput for the protected network is 7.4KBps; with one and two jammers, the respective throughputs drops to 6.1 and 3.2KBps. Although our protocols do not perform as efficiently as standard 802.11 *in the absence of contention*, our MAC layer achieves significantly lower loss rates and reasonable throughput when jammers are present. In environments where reliability is of utmost importance (e.g., in first responder networks), our protocols provide strong delivery guarantees.

5 Conclusion

This paper describes and proves correct uncoordinated communication algorithms for general noisy shared spectrum networks. It also describes a reliable link layer protocol based on these primitives, and performs a preliminary testbed evaluation. This work indicates that algorithmic techniques can be used to enable reliable communication even in settings,e.g., low power, where such reliability is otherwise hard to achieve.

References

[1] Awerbuch, B., Richa, A., Scheideler, C.: A Jamming-Resistant MAC Protocol for Single-Hop Wireless Networks. In: PODC (2008)
[2] Barriére, L., Fraigniaud, P., Narayanan, L.: Robust position-based routing in wireless ad hoc networks with unstable transmission ranges. In: DIAL M, pp. 19–27 (2001)
[3] Daum, S., Gilbert, S., Kuhn, F., Newport, C.: Leader Election in Shared Spectrum Radio Networks. In: PODC (2012)
[4] Diffie, W., Hellman, M.E.: New Directions in Cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
[5] Foerster, J.: The Performance of a Direct-Sequence Spread Ultra-Wideband System in the Presence of Multipath, Narrowband Interference, and Multiuser Interference. In: IEEE Conference on Ultra Wideband Systems and Technologies (2002)
[6] Ghaffari, M., Gilbert, S., Newport, C., Tan, H.: Optimal Broadcast in Shared Spectrum Radio Networks. In: Baldoni, R., Flocchini, P., Binoy, R. (eds.) OPODIS 2012. LNCS, vol. 7702, pp. 181–195. Springer, Heidelberg (2012)
[7] Gilbert, S., Guerraoui, R., Newport, C.: Of Malicious Motes and Suspicious Sensors: On the Efficiency of Malicious Interference in Wireless Networks. Theoretical Computer Science 410(6-7), 546–569 (2009)
[8] Tan, H., Wacek, C., Newport, C., Sherr, M.: A Disruption-Resistant MAC Layer for Multichannel Wireless Networks, `http://people.cs.georgetown.edu/~cnewport/publications.html`
[9] Jin, T., Noubir, G., Thapa, B.: Zero Pre-Shared Secret Key Establishment in the Presence of Jammers. In: MOBIHOC (2009)
[10] Kavehrad, M., Ramamurthi, B.: Direct-Sequence Spread Spectrum with DPSK Modulation and Diversity for Indoor Wireless Communications. IEEE Transactions on Communications 35(2), 224–236 (1987)

[11] Kohler, E., Morris, R., Chen, B., Jannotti, J., Kaashoek, M.: The Click Modular Router. ACM Transactions on Computer Systems (TOCS) 18(3), 263–297 (2000)
[12] Kuhn, F., Moscibroda, T., Wattenhofer, R.: On the locality of bounded growth. In: PODC, pp. 60–68 (2005)
[13] Kuhn, F., Wattenhofer, R., Zollinger, A.: Ad hoc networks beyond unit disk graphs. Wireless Networks 14(5), 715–729 (2008)
[14] Liu, A., Ning, P., Dai, H., Liu, Y.: USD-FH: Jamming-Resistant Wireless Communication using Frequency Hopping with Uncoordinated Seed Disclosure. In: MASS (2010a)
[15] Liu, A., Ning, P., Dai, H., Liu, Y., Wang, C.: Defending DSSS-Based Broadcast Communication Against Insider Jammers via Delayed Seed-Disclosure. In: ACSAC (2010b)
[16] Liu, Y., Ning, P., Dai, H., Liu, A.: Randomized Differential DSSS: Jamming-Resistant Wireless Broadcast Communication. In: INFOCOM (2010c)
[17] Moscibroda, T., Wattenhofer, R.: Maximal independent sets in radio networks. In: PODC, pp. 148–157. ACM (2005)
[18] Navda, V., Bohra, A., Ganguly, S., Rubenstein, D.: Using channel hopping to increase 802.11 resilience to jamming attacks. In: INFOCOM (2007)
[19] Pöpper, C., Strasser, M., Čapkun, S.: Jamming-Resistant Broadcast Communication without Shared Keys. In: USENIX Security Symposium (2009)
[20] Richa, A., Scheideler, C., Schmid, S., Zhang, J.: Competitive and Fair Throughput for Co-Existing Networks Under Adversarial Interference. In: PODC (2012)
[21] Richa, A., Scheideler, C., Schmid, S., Zhang, J.: Competitive Throughput in Multi-Hop Wireless Networks Despite Adaptive Jamming. Distributed Computing 26(3), 159–171 (2013)
[22] Schmid, S., Wattenhofer, R.: Algorithmic models for sensor networks. In: Proc. 14th Int. Workshop on Parallel and Distributed Real-Time Systems, pp. 1–11 (2006)
[23] Sharma, A., Belding, E.M.: FreeMAC: Framework for Multi-channel MAC Development on 802.11 Hardware. In: ACM Workshop on Programmable Routers for Extensible Services of Tomorrow (PRESTO) (2008)
[24] Slater, D., Tague, P., Poovendran, R., Matt, B.: A Coding-Theoretic Approach for Efficient Message Verification over Insecure Channels. In: WiSec (2009)
[25] Strasser, M., Capkun, S., Popper, C., Cagalj, M.: Jamming-Resistant Key Establishment using Uncoordinated Frequency Hopping. In: IEEE Symposium on Security and Privacy (2008)
[26] Strasser, M., Pöpper, C., Čapkun, S.: Efficient Uncoordinated FHSS Anti-Jamming Communication. In: MOBIHOC (2009)
[27] Xu, W., Wood, T., Trappe, W., Zhang, Y.: Channel surfing and spatial retreats: Defenses against wireless denial of service. In: ACM Workshop on Wireless Security (2004)
[28] Xu, W., Trappe, W., Zhang, Y.: Channel Surfing: Defending Wireless Sensor Networks from Interference. In: ACM IPSN (2007)