Beeping a Maximal Independent Set

Yehuda Afek[1], Noga Alon[1,2], Ziv Bar-Joseph[3],
Alejandro Cornejo[4], Bernhard Haeupler[4], and Fabian Kuhn[5]

[1] The Blavatnik School of Computer Science, Tel Aviv University, 69978, Israel
[2] Sackler School of Mathematics, Tel Aviv University, 69978, Israel
[3] School of Computer Science, Carnegie Mellon Univ., Pittsburgh, PA 15213, USA
[4] CSAIL, Massachusetts Institute of Technology, MA 02139, USA
[5] Faculty of Informatics, University of Lugano, 6904 Lugano, Switzerland

Abstract. We consider the problem of computing a maximal independent set (MIS) in an extremely harsh broadcast model that relies only on carrier sensing. The model consists of an anonymous broadcast network in which nodes have no knowledge about the topology of the network or even an upper bound on its size. Furthermore, it is assumed that nodes wake up asynchronously. At each time slot a node can either beep (i.e., emit a signal) or be silent. At a particular time slot, beeping nodes receive no feedback, while silent nodes can only differentiate between none of its neighbors beeping, or at least one neighbor beeping.

We start by proving a lower bound that shows that in this model, it is not possible to locally converge to an MIS in sub-polynomial time. We then study four different relaxations of the model which allow us to circumvent the lower bound and compute an MIS in polylogarithmic time. First, we show that if a polynomial upper bound on the network size is known, it is possible to find an MIS in $\mathcal{O}(\log^3 n)$ time. Second, if sleeping nodes are awoken by neighboring beeps, then we can also find an MIS in $\mathcal{O}(\log^3 n)$ time. Third, if in addition to this wakeup assumption we allow beeping nodes to receive feedback to identify if at least one neighboring node is beeping concurrently (i.e., sender-side collision detection) we can find an MIS in $\mathcal{O}(\log^2 n)$ time. Finally, if instead we endow nodes with synchronous clocks, it is also possible to compute an MIS in $\mathcal{O}(\log^2 n)$ time. We remark that the last two algorithms essentially match the bit complexity of the classic distributed MIS algorithms by Alon, Babai, and Itai [2], and by Luby [14].

1 Introduction

An MIS is a *maximal* set of nodes in a network such that no two of them are neighbors. Since the set is maximal every node in the network is either in the MIS or a neighbor of a node in the MIS. The problem of distributively selecting an MIS has been extensively studied in various models [2, 5, 20, 11, 13, 12, 14, 17, 15, 24] and has many applications in networking, and in particular in radio sensor networks. Some of the practical applications include the construction of a backbone for wireless networks, a foundation for routing and for clustering of nodes, and generating spanning trees to reduce communication costs [20, 24].

This paper studies the problem of computing an MIS in the discrete beeping wireless network model of [6]. The network is modeled as an undirected graph and time progresses in discrete and synchronous rounds, each being a time slot. In each round a node can either transmit a "jamming" signal (aka beep) or detect whether at least one neighbor beeps. We believe that such a model is minimalistic enough to be implementable in many real world scenarios. For example, it can easily be implemented using carrier sensing alone, where nodes only differentiate between silence and the presence of a signal on the wireless channel. Further, it has been shown that such a minimal communication model is strong enough to efficiently solve non-trivial tasks [1, 6, 18, 23]. The model is interesting from a practical point of view since carrier sensing typically uses less energy to communicate and reaches larger distances when compared with sending regular messages.

While this model is clearly useful for computer networks, it is also partially motivated by biological processes which are often more robust and adaptable than current computational systems. In biological systems, cells communicate by secreting certain proteins that are sensed ("heard") by neighboring cells [5]. This is similar to a node in a radio network transmitting a carrier signal which is sensed ("heard") by its neighbors. Such physical message passing allows for an upper bound on message delay. Thus, for a computational model based on these biological systems, we can assume a set of synchronous and anonymous processors communicating using beeps [6] in an arbitrary topology. We have recently shown that a variant of MIS is solved by a biological process, sensory organ precursor (SOP) selection in flies, and that the fly's solution provides a novel algorithm for solving MIS [1]. Here we extend algorithms for this model in several ways as discussed below.

This paper has two parts, we first prove a lower bound that shows that in a beeping model with adversarial wake-up it is not possible to locally converge on an MIS in sub-polynomial time. In the second part we present several relaxations of this model under which polylogarithmic MIS constructions are possible.

The lower bound shows that if nodes are not endowed with any information about the underlying communication graph, and their wake-up time is under the control of the adversary, any (randomized) distributed algorithm to find an MIS requires at least $\Omega(\sqrt{n/\log n})$ rounds. We remark that this lower bound holds much more generally. We prove the lower bound for the significantly more powerful radio network model with arbitrary message size and collision detection, and is therefore not an artifact of the amount of information which can be communicated in the beeping model.

Following the lower bound, in the second part of this paper four weaker adversarial models are considered and a polylog algorithm for MIS construction is presented for each. First, we present an algorithm that uses a polynomial upper bound on the size of the network, to compute an MIS in $\mathcal{O}(\log^3 n)$ rounds with high probability. Our next two algorithms assume that nodes are awakened by incoming messages (aka wake-on-beep). We present a $\mathcal{O}(\log^2 n)$ rounds algorithm in the beeping model with sender collision detection (aka sender CD) which we

believe that is the fastest MIS algorithm that does not require any information about the network. Next, we present a $\mathcal{O}(\log^3 n)$ that works without sender collision detection in the same wakeup model. Finally, we show that even when nodes are only awakened by an adversary (and not by incoming messages) and without any information about the network, its possible to use synchronous clocks to compute an MIS in $\mathcal{O}(\log^2 n)$ time. We highlight that all the upper bounds presented in this paper compute an MIS eventually and almost surely, and thus only their running time is randomized.

	Assumptions	Running Time
Section 4	-	$\Omega(\sqrt{n/\log n})$
Section 5	Upper bound	$\mathcal{O}(\log^3 n)$
Section 6	Wake-on-Beep + Sender CD	$\mathcal{O}(\log^2 n)$
Section 7	Wake-on-Beep	$\mathcal{O}(\log^3 n)$
Section 8	Synchronous Clocks	$\mathcal{O}(\log^2 n)$

2 Related work

The computation of an MIS has been recognized and studied as a fundamental distributed computing problem for a long time (e.g., [2, 3, 14, 19]). Perhaps the single most influential MIS algorithm is the elegant randomized algorithm of [2, 14], generally known as Luby's algorithm, which has a running time of $\mathcal{O}(\log n)$. This algorithm works in a standard message passing model, where nodes can concurrently reliably send and receive messages over all point-to-point links to their neighbors. Métivier et al. [15] show how to improve the bit complexity of Luby's algorithm to use only $O(\log n)$ bits per channel ($O(1)$ bits per round). For the case where the size of the largest independent set in the neighborhood of each node is restricted to be a constant (known as bounded independence or growth-bounded graphs), Schneider and Wattenhofer [22] presented an algorithm that computes an MIS in $\mathcal{O}(\log^* n)$ rounds. This class of graphs includes unit disk graphs and other geometric graphs that have been studied in the context of wireless networks.

While several methods were suggested for solving MIS in the case of symmetric processors, these methods have always assumed that nodes know something about the local or global topology of the network. Most previous methods assumed that nodes know the set of active neighbors each has at each stage of the execution. The first effort to design a distributed MIS algorithm for a wireless communication model in which the number of neighbors is not known is by Moscibroda and Wattenhofer [16]. They provided an algorithm for the radio network model with a $\mathcal{O}(\log^9 n/\log\log n)$ running time. This was later improved [17] to $\mathcal{O}(\log^2 n)$. Both algorithms assume that the underlying graph is a unit disk graph (the algorithms also work for somewhat more general classes of geometric graphs). In addition, while the algorithms solve MIS selection in multi-hop networks with asynchronous wake up, they still assume that an upper bound on the number of nodes in the network is known. In addition to the upper bound assumption their model allows for (and their algorithm uses) messages whose size is a function of the number of nodes in the network.

The use of carrier sensing (a.k.a. collision detection) in wireless networks has e.g. been studied in [4, 9, 23]. As shown in [23], collision detection can be powerful and can be used to improve the complexity of algorithms for various basic problems. Scheideler et al. [21] show how to approximate a minimum dominating set in a physical interference (SINR) model where in addition to sending messages, nodes can perform carrier sensing. In [8], it is demonstrated how to use carrier sensing as an elegant and efficient way for coordination in practice.

The present paper is not he first one that uses carrier sensing alone for distributed wireless network algorithms. A similar model to the beep model considered here was first studied in [7, 18]. As used here, the model has been introduced in [6], where it is shown how to efficiently obtain a variant of graph coloring that can be used to schedule non-overlapping message transmissions. In [1] a variant of beeping model, called there *the fly model* was considered. The fly model made the three additional assumptions, which do not necessarily hold for biological systems: that all the processors wake up together at the same synchronous round, that a bound on the network size is known to the processors, and that processors can detect collisions. That is, processors can listen on the medium while broadcasting (as in some radio and local area networks). In addition to the work from [1] the most related work to this paper are results from [23]. In [23], it is shown that by solely using carrier sensing, an MIS can be computed in $O(\log n)$ time in growth-bounded graphs (a.k.a. bounded independence graphs). Here, we drop that restriction and study the MIS problem in the beeping model for general graphs.

3 Model

Following [6], we consider a synchronous communication network modeled by an arbitrary graph $G = (V, E)$ where the vertices V represent processors and the edges represent pairs of processors that can hear each other. We denote the set of neighbors of node u in G by $N_G(u) = \{v \mid \{u, v\} \in E\}$. For a node $u \in V$ we use $d_G(u) = |N_G(u)|$ to denote its degree (number of neighbors) and we use $d_{\max} = \max_{u \in V} d_G(u)$ to denote the maximum degree of G.

Instead of communicating by exchanging messages, we consider a more primitive communication model that relies entirely on carrier sensing. Specifically, in every round a participating process can choose to either beep or listen. If a process at node v listens in round t it can only distinguish between silence (i.e., no process $u \in N_{G_t}(v)$ beeps in round t) or the presence of one or more beeps (i.e., there exists a process $u \in N_{G_t}(v)$ that beeps in round t). Observe that a beep conveys less information than a conventional 1-bit message, since in the latter it is possible to distinguish between no message, a message with a one, and a message with a zero.

Initially all processes are asleep, and a process starts participating at the round in which it is woken up by an adversary. We denote by $G_t \subseteq G$ the subgraph induced by the processes which are participating at round t.

Given an undirected graph H, a set of vertices $I \subseteq V(H)$ is an independent set of H if every edge $e \in E$ has at most one endpoint in I. An independent set $I \subseteq V(H)$ is a maximal independent set of H, if for all $v \in V(H) \setminus I$ the set

$I \cup \{v\}$ is not independent. An event is said to occur with high probability, if it occurs with probability at least $1 - n^{-c}$ for any constant $c \geq 1$, where $n = |V|$ is the number of nodes in the underlying communication graph. For a positive integer $k \in \mathbb{N}$ we use $[k]$ as short hand notation for $\{1, \ldots, k\}$. In a slight abuse of this notation we use $[0]$ to denote the empty set $\varnothing$ and for $a, b \in \mathbb{N}$ and $a < b$ we use $[a, b]$ to denote the set $\{a, \ldots, b\}$.

We say a (randomized) distributed algorithm solves the MIS problem in T rounds, if any node irrovocably decides to be either inactive or in the MIS after being awake for at most T rounds. Furthremore, no two neighboring nodes decide to be in the MIS, and every node which decided to be inactive has at least one neighbor which decided to be in the MIS.

4 Lower Bound for Uniform Algorithms

In this section we show that without any additional power or priori information about the network (e.g., an upper bound on its size or maximum degree), any fast-converging (randomized) distributed algorithm needs at least polynomial time to find an MIS with constant probability. In some ways, this result is the analog of the polynomial lower bound [10] on the number of rounds required for a successful transmission in the radio network model without collision detection or knowledge of n.

We stress that this lower bound is not an artifact of the beep model, but a limitation that stems from having message transmission with collisions and the fact that nodes are required to decide (but not necessarily terminate) without waiting until all nodes have woken up. Although we prove the lower bound for the problem of finding an MIS, this lower bound can be generalized to other problems (e.g., minimal dominating set, coloring, etc.).

Specifically, we prove the lower bound for the stronger communication model of the local message broadcast with collision detection. In this communication model a process can choose in every round either to listen or to broadcast a message (no restrictions are made on the size of the message). When listening a process receives silence if no message is broadcast by its neighbors, it receives a collision if a message is broadcast by two or more neighbors, and it receives a message if it is broadcast by exactly one of its neighbors. The beep communication model can be easily simulated by this model (instead of beeping send a 1 bit message, and when listening translate a collision or the reception of a message to hearing a beep) and hence the lower bound applies to the beeping model.

At its core, our lower bound argument relies on the observation that a node can learn essentially no information about the graph G if upon waking up, it always hears collisions or silence. It thus has to decide whether it remains silent or beeps within a *constant* number of rounds. More formally:

Proposition 1. *Let A be an algorithm run by all nodes, and consider a fixed pattern $b \in \{\text{silent}, \text{collision}\}^*$. If after waking up a node u hears $b(r)$ whenever it listens in round r, then there are two constants $\ell \geq 1$ and $p \in (0, 1]$ that only*

depend on A and b such that either **a)** *u remains listening indefinitely, or* **b)** *u listens for $\ell - 1$ rounds and broadcasts in round ℓ with probability p.*

Proof. Fix a node u and let $p(r)$ be the probability with which node u beeps in round r. Observe that $p(r)$ can only depend on r, what node u heard up to round r (i.e., b) and its random choices. Therefore, given any algorithm, either $p(r) = 0$ for all r (and node u remains silent forever), or $p(r) > 0$ for some r, in which case we let $p = p(r)$ and $\ell = r$. □

We now prove the main result of this section:

Theorem 4.1. *If nodes have no a priori information about the graph G then any fast-converging distributed algorithm in the local message broadcast model with collision detection that solves the MIS problem with constant probability requires requires at least $\Omega(\sqrt{n/\log n})$ rounds.*

Proof. Fix any algorithm A. Using the previous proposition we split the analysis in three cases, and in all cases we show that with probability $1 - o(1)$ any algorithm runs for $o(\sqrt{n/\log n})$ rounds.

We first ask what happens with nodes running algorithm A that hear only silence after waking up. Proposition 1 implies that either nodes remain silent forever, or there are constants ℓ and p such that nodes broadcast after ℓ rounds with probability p. In the first case, suppose nodes are in a clique, and observe that no node will ever broadcast anything. From this it follows that nodes cannot learn anything about the underlying graph (or even tell if they are alone). Thus, either no one joins the MIS, or all nodes join the MIS with constant probability, in which case their success probability is exponentially small in n.

Thus, for the rest of the argument we assume that nodes running A that hear only silence after waking up broadcast after ℓ rounds with probability p. Now we consider what happens with nodes running A that hear only collisions after waking up. Again, by Proposition 1 we know that either they remain silent forever, or there are constants m and p' such that nodes broadcast after m rounds with probability p'. In the rest of the proof we describe a different execution for each of these cases.

CASE 1: (a node that hears only collisions remains silent forever) For some $k \gg \ell$ to be fixed later, we consider a set of $k-1$ cliques $C_1, \ldots, C_{k-1}$ and a set of k cliques $U_1, \ldots, U_k$, where each clique C_i has $\Theta(k \log n/p)$ vertices, and each clique U_j has $\Theta(\log n)$ vertices. We consider a partition of each clique C_i into k sub-cliques $C_i(1), \ldots, C_i(k)$ each with $\Theta(\log n/p)$ vertices. For simplicity, whenever we say two cliques are connected, they are connected by a complete bipartite graph.

Consider the execution where in round $i \in [k-1]$ clique C_i wakes up, and in round ℓ the cliques $U_1, \ldots, U_k$ wake up simultaneously. When clique U_j wakes up, it is is connected to sub-clique $C_i(j)$ for each $i < \ell$. Similarly, when clique C_i wakes up, if $i \geq \ell$ then for $j \in [k]$ sub-clique $C_i(j)$ is connected to clique U_j.

During the first $\ell-1$ rounds only the nodes in C_1 are participating, and hence every node in C_1 broadcasts in round $\ell + 1$ with probability p. Thus w.h.p. for

all $j \in [k]$ at least two nodes in sub-clique $C_1(j)$ broadcast in round ℓ. This guarantees that all nodes in cliques $U_1, \ldots, U_k$ hear a collision during the first round they are awake, and hence they also listen for the second round. In turn, this implies that the nodes in C_2 hear silence during the first $\ell - 1$ rounds they participate, and again for $j \in [k]$ w.h.p. there are at least two nodes in $C_2(j)$ that broadcast in round $\ell + 2$.

By a straightforward inductive argument we can show (omitted) that in general w.h.p. for each $i \in [k-1]$ and for every $j \in [k]$ at least two nodes in sub-clique $C_i(j)$ broadcast in round $\ell + i$. Therefore, also w.h.p., all nodes in cliques $U_1, \ldots, U_k$ hear collisions during the first $k - 1$ rounds after waking up.

Observe that at most one node in each C_i can join the MIS (i.e. at most one of the sub-cliques of C_i has a node in the MIS), which implies there exists at least one clique U_j that is connected to only non-MIS sub-cliques. However, since the nodes in U_j are connected in a clique, exactly one node of U_j must decide to join the MIS, but all the nodes in U_j have the same state during the first $k-1$ rounds. Therefore if nodes decide after participating for at most $k-1$ rounds, w.h.p. either no one in U_j joins the MIS, or more than two nodes join the MIS.

Finally since we have $n \in \Theta(k^2 \log n + k \log n)$ nodes, we can let $k \in \Theta(\sqrt{n/\log n})$ and the theorem follows.

CASE 2: (a node that hears only collisions remains silent forever) For some $k \gg m$ to be fixed later let $q = \lfloor \frac{k}{4} \rfloor$ and consider a set of k cliques $U_1, \ldots, U_k$ and a set of $m-1$ cliques $S_1, \ldots, S_{m-1}$, where each clique U_i has $\Theta(\log n/p')$ vertices, and each clique S_i has $\Theta(\log n/p)$ vertices. As before, we say two cliques are connected if they form a complete bipartite graph.

Consider the execution where in round $i \in [m-1]$ clique S_i wakes up, and in round $\ell + j$ for $j \in [k]$ clique U_j wakes up. When clique U_j wakes up, if $j > 1$ it is connected to every U_i for $i \in \{\max(1, j-q), \ldots, j-1\}$ and if $j < m$ it is also connected to every clique S_h for $h \in \{m-j, \ldots, m\}$.

During the first $\ell - 1$ rounds only the nodes in S_1 are participating, and hence every node in S_1 broadcasts in round $\ell + 1$ with probability p, and thus w.h.p. at least two nodes in S_1 broadcast in round $\ell + 1$. This guarantees the nodes in U_1 hear a collision upon waking up, and therefore they listen in round $\ell + 2$. In turn this implies the nodes in S_2 hear silence during the first $\ell - 1$ rounds they participate, and hence w.h.p. at least two nodes in S_2 broadcast in round $\ell + 2$.

By a straightforward inductive argument we can show (omitted) that in general for $i \in [m-1]$ the nodes in S_i hear silence for the first $\ell - 1$ rounds they participate, and w.h.p. at least two nodes in S_i broadcast in round $\ell + i$. Moreover, for $j \in [k]$ the nodes in U_j hear collisions for the first $m - 1$ rounds they participate, and hence w.h.p. there are at least two nodes in U_j who broadcast in round $\ell + m + j - 1$. This implies that w.h.p. for $j \in [k - q]$ the nodes in U_j hear collisions for the first q rounds they participate.

We argue that if nodes choose weather or not to join the MIS q rounds after participating, then they fail w.h.p. In particular consider the nodes in clique U_j for $j \in \{q, \ldots, k - 2q\}$. These nodes will collisions for the first q rounds they

participate, and they are connected to other nodes which also hear beeps for the first q rounds they participate. Therefore, if nodes decide after participating for less or equal than q rounds, w.h.p. either a node and all its neighbors won't be in the MIS, or two or more neighboring nodes join the MIS.

Finally since we have $n \in \Theta(m \log n + k \log n)$ nodes, we can let $k \in \Theta(n/\log n)$ and hence $q \in \Theta(n/\log n)$ and the theorem follows. □

5 Using an upper Bound on n

To circumvent the lower bound, this section assumes that all nodes are initially given some upper bound $N > n$ (its not required that all nodes are given the same upper bound) on the total number of nodes participating in the system. The algorithm described in this section guarantees that $\mathcal{O}(\log^2 N \log n)$ rounds after a node wakes up, it knows weather it belongs to the MIS or if it is inactive (i.e. covered by an MIS node). Therefore, if the known upper bound is polynomial in n (i.e., $N \in \mathcal{O}(n^c)$ for a constant c), then this algorithm solves the MIS problem in $\mathcal{O}(\log^3 n)$ rounds.

Algorithm: If a node hears a beep while listening at any point during the execution, it restarts the algorithm. When a node wakes up (or it restarts), it stays in an inactive state where it listens for $c \log^2 N$ consecutive rounds. After this inactivity period, nodes enters a competing state and group rounds into $\log N$ phases of $c \log N$ consecutive rounds. Due to the asynchronous wake up and the restarts, in general phases of different nodes will not be synchronized. In each round of phase i with probability $2^i/8N$ a node beeps, and otherwise it listens. Thus by phase $\log N$ a node beeps with probability $\frac{1}{8}$ in every round. After successfully going through the $\log N$ phases of activity (recall that when a beep is heard during any phase, the algorithm restarts) a node assumes it has joined the MIS and into a loop where it beeps in every round with probability 1/8 forever (or until it hears a beep).

Algorithm 1 MIS with an upper bound N on the size of the network.

```
1: for c log^2 N rounds do listen                                   ▷ Inactive
2: for i ∈ {1, ..., log N} do                                       ▷ Competing
3:     for c log N rounds do
4:         with probability 2^i/8N beep, otherwise listen
5: forever with probability 1/2 beep then listen, otherwise listen then beep   ▷ MIS
```

Theorem 1. *Algorithm 1 solves the MIS problem in $O(\log^2 N \log n)$ time, where N is an upper bound for n that is a priori known to the nodes.*

This is another example which demonstrates that knowing a priori size information about the network, even as simple as its size, can drastically change the complexity of a problem.

Proof Outline. First, we leverage the fact that for two neighboring nodes to go simultaneously into the MIS they have to choose the same actions (beep or listen) during at least $c \log N$ rounds. This does no happen w.h.p. and thus MIS nodes are independent w.h.p. On the other hand, since nodes which are in the MIS keep trying to break ties, an inactive node will never start competing while it has a neighbor in the MIS, and even in the low probability event that two neighboring nodes do join the MIS, one of them will eventually and almost surely leave the MIS. The more elaborate part of the proof is showing that w.h.p., any node either joins the MIS or has one of its neighbors in the MIS after $O(\log^2 N \log n)$ consecutive rounds. This requires three technical lemmas. First we show that if the sum of the beep probabilities of a neighbor are greater than a large enough constant, then they have been larger than a (smaller) constant for the $c \log N$ preceding rounds. We then use this to show that with constant probability, when a node u hears or produces beep, none of its neighbors beeps at the same time (and therefore it joins the MIS). Finally, since a node hears a beep or produces a beep every $O(\log^2 N)$ rounds, $\mathcal{O}(\log^2 N \log n)$ rounds suffice to stabilize w.h.p.

We remark that this algorithm is very robust, and in fact it works as-is if we give the adversary the power to crash n arbitrary set of nodes at every round. For such a powerful adversary, the running time guarantees have to be modified slightly, since if an inactive node has a single MIS node which is then crashed by the adversary, we must allow the inactive node to start competing to be in the MIS again.

Knowing when you are done. We've argued hat with high probability every node will (correctly) be in the MIS or the inactive state $\mathcal{O}(\log^3 n)$ rounds after waking up, however observe that the algorithm provides no way for a node to determine/output when it has arrived at the correct state. This is not a flaw of the algorithm, but an inherent limitation of the model and assumptions in which it is implemented. To see why, observe that regardless of what algorithm is used, at every round there is a non-zero probability that all nodes which are in the same state make the same random choices (beep or listen), and remain in the same state on the next round. Although this probability will drop exponentially fast with n, this already means that its impossible for a node to distinguish with complete if it is executing alone, or if it has one or more neighbors.

If we are willing to tolerate a small probability of error, we can simply turn Algorithm 1 (which is a Las Vegas algorithm) to a Monte Carlo algorithm and have every node output their state $\mathcal{O}(\log^3 n)$ rounds after waking up. Theorem 1 would guarantee that with high probability the output would describe an MIS. Another alternative, would be to endow nodes with unique identifiers encoded in $\mathcal{O}(\log n)$ bits. Using these identifiers its possible to augment the last phase of the algorithm (i.e., line 5) to detect with certainty the case where two neighboring nodes are in the MIS state in asymptotically the same round complexity (we omit the details due to lack of space). Another alternative, which is described in detail in the next section, is to use sender-side collision detection.

6 Wake-on-Beep and Sender Collision Detection

This section considers a different relaxation of the model. Specifically, in addition to allowing the adversary to wakeup nodes arbitrarily, in this and the next section we assume that sleeping nodes wake up upon receiving a beep (aka wake-on-beep). Moreover this section we also assume that when a node beeps, it receives some feedback from which it can know if it beeped alone, or one of its neighbors beeped concurrently (aka sender collision detection). We will show that in such a model, its possible to compute an MIS in $O(\log^2 n)$ time, even if there is no knowledge of the network (including no knowledge of neighbors and / or any upper bound on the network size).

This algorithm is an improvement of the algorithm presented in [1], which used an upper bound on the size of the network. In this algorithm nodes go through several iterations in which they gradually decrease the probability of being selected. The run time of the algorithm is still $O(\log^2 n)$ as we show below. Compared to the algorithm in [1], in addition to eliminating the dependence on any topological information, the current algorithm tolerates asynchronous wakeups if we assume wake-on-beep.

The algorithm proceeds in phases each consisting of x steps where x is the total number of phases performed so far (the phase counter). Assuming all nodes start at the same round, step i of each phase consists of two exchanges. In the first exchange nodes beep with probability $1/2^i$, and in the second exchange a node that beeped in the first and did not hear a beep from any of its neighbors, beeps again, telling its neighbors it has joined the MIS and they should become inactive and exit the algorithm.

Asynchronous wakeup. Nodes that woke up spontaneously, or were awakened by the adversary, propagate a wave of wake-up beeps throughout the network. Upon hearing the first beep, which must be the wake up beep, a node broadcasts the wake up beep on the next round, and then waits one round to ensure none of its neighbors is still waking up. This ensures that all neighbors of a node wakeup either at the same round as that node or one round before, or after, that node. Due to these possible differences in wakeup time, we divide each exchange between 3 rounds. Nodes listen in all three rounds to incoming messages. During the second round of the first exchange each active node broadcasts a message to its neighbors with probability p_i (the value of p_i is given in the algorithm). The second exchange also takes three rounds. A node that has broadcast a message in the first exchange joins the MIS if none of its neighbors had broadcast in any of the three round of the first exchange. Such node broadcasts again a message in the second round of the second exchange telling its neighbors to terminate the algorithm. The algorithm is presented in Figure 2.

Safety properties. While the algorithm in [1] used a different set of coin flip probabilities, it relied on a similar two exchanges structure to guarantee the safety properties of the algorithm (when the algorithm terminates nodes are either MIS nodes or connected to a MIS node and no two MIS nodes are connected to each other). In that paper each exchange was only one round (since

Algorithm 2 MIS with wake-on-beep and sender-side collision detection

```
1: upon waking up (by adversary or beep) do beep to wake up neighbors
2: wait for 1 round; x ← 0                                  ▷ while neighbors wake up
3: repeat
4:     x ← x + 1                                            ▷ 2^x is current size estimate
5:     for i ∈ {0, ..., x} do                               ▷ log 2^x phases
6:         ** exchange 1 ** with 3 rounds
7:         listen for 1 round; v ← 0                        ▷ round 1
8:         with probability 1/2^i, beep and set v ← 1       ▷ round 2
9:         listen for 1 round                               ▷ round 3
10:        if received beep in any round of exchange 1 then v ← 0
11:            ** exchange 2 ** with 3 rounds
12:        listen for 1 round                               ▷ round 1
13:        if v = 1 then beep and join MIS                  ▷ round 2
14:        listen for 1 round                               ▷ round 3
15: until in MIS or received beep in any round of exchange 2
```

we assumed synchronous wakeup). We thus need to show that replacing each one round exchange with a three round exchange does not affect the MIS safety properties of our algorithm. We will thus start by proving that the termination lemma from [1], which relies on the fact that all neighbors are using the same probability distribution in each exchange, still holds.

Lemma 1. *All messages received by node j in the first exchange of step i were sent by processors using the same probability as j in that step (see [**?**] for proof).*

Note that a similar argument would show that all messages received in the second exchange of step i are from processors that are in the second exchange of that step. Since our safety proof only relies on the coherence of the exchange they still hold for this algorithm. Notice also that by adding a listening round at the beginning and end of each exchange the algorithm now works in the un-slotted model (with at most doubling the round complexity).

Runtime analysis. After establishing the safety guarantees, we next prove that with high probability all nodes terminate the algorithm in $O(\log^2 n)$ time where n is the number of nodes that participate in the algorithm. Let d_v be the number of *active* neighbors of node v. We start with the following definition [2]: a node v is Good if it has at least $d_v/3$ active neighbors u, s.t., $d_u \leq d_v$.

Lemma 4.4 from [2]: In every graph G at least half of the edges touch a Good vertex. Thus, $\sum_{v \in Good} d_v \geq |E|/2$.

Note that we need less than $O(log^2 n)$ steps to reach $x \geq \log n$ since each phase until $x = \log n$ has less than $\log n$ steps. When $x \geq \log n$, the first $\log n$ steps in each phase are using the probabilities: $1, 1/2, 1/4, ..., 2/n, 1/n$. Below we show that from the time $x = \log n$, we need at most $O(\log n)$ more phases to guarantee that all processors terminate the algorithms with high probability.

Lemma 2. *The expected number of edges deleted in a phase (with more than $\log n$ steps) is $\Omega(|E|)$*

Proof. Fix a phase j, and fix a Good vertex v. We show that the expected number of edges incident with v that are deleted in this phase is $\Omega(d_v)$. Assume that at the beginning of phase j, $2^k \leq d_v \leq 2^{k+1}$ for some $k < \log n$. If when we reach step $i = k$ in phase j at least $d_v/20$ edges incident with v were removed already we are done. Otherwise, at step i there are still at least $d_v/3 - d_v/20 > d_v/4 \geq 2^{k-2}$ neighbors u of v with $d_u \leq d_v$. Node v and all its neighbors u are flipping coins with probability $\frac{1}{2^k}$ at this step and thus the probability that at least one of them would broadcast is:

$$p(v \text{ or } u, \textit{ neighbor of } v \textit{ with } d_u \leq d_v, \textit{ broadcasts}) \geq 1-(1-\frac{1}{2^k})^{2^{k-2}} \cong 1-1/e^{1/4}$$

On the other hand, all such nodes u, and v, have less than 2^{k+1} neighbors. Thus, the probability that a node from this set that broadcasts a message does not collide with any other node is:

$$p(\textit{no collisions}) \geq (1-\frac{1}{2^k})^{2^{k+1}} \cong 1/e^2$$

Thus, in every phase a Good node v has probability of at least $(1-\frac{1}{e^{1/4}})\frac{1}{e^2} \geq \frac{1}{2^7}$ to be removed. Thus, the probability that v is removed is $\Omega(1)$ which means that the expected number of edges incident with v removed during this phase is $\Omega(d_v)$.
Since half the edges touch a Good node, by the linearity of expectation the expected number of edges removed in each phase is $\geq \frac{1}{2}\Omega(\sum_{v \in Good} d_v) = \Omega(|E|)$.

Note that since the number of edges removed in a phase in a graph (V, E) is clearly always at most $|E|$, the last lemma implies that for any given history, with probability at least $\Omega(1)$, the number of edges removed in a phase is at least a constant fraction of the number of edges that have not been deleted yet. Therefore there are two positive constants p and c, both bounded away from zero, so that the probability that in a phase at least a fraction c of the number of remaining edges are deleted is at least p. Call a phase successful if at least a fraction c of the remaining edges are deleted during the phase.

By the above reasoning, the probability of having at least z successful phases among m phases is at least the probability that a binomial random variable with parameters m and p is at least z. By the standard estimates for Binomial distributions, and by the obvious fact that $O(\log |E|/c) = O(\log n)$, starting from $x = \log n$ we need an additional $O(\log n)$ phases to finish the algorithm. Since each of these additional $O(\log n)$ phases consists of $O(\log n)$ steps, and since as discussed above until $x = \log n$ we have less than $O(\log^2 n)$ steps, the total running time of the algorithm is $O(\log^2 n)$.

□

7 Wake-on-Beep with no collision detection

This section shows how to solve MIS in the wake-on-beep model with no collision detection. To extend our algorithm to a model with no collision detection we

increase the number of exchanges in each step from 3 to x (x is the same as in Algorithm 2 and represents current estimate of the network size). Prior to starting the exchanges in each step each *active* processor flips a coin with the same probability as in Algorithm 2. If the flip outcome is 0 (tail) the processor only listens in the next cx exchanges (for a constant c discussed below). If the flip outcome is 1 the processor sets $v = 1$ and sets, with probability 0.5, every entry in the vector X of size cx to 1 (the rest are 0). In the following exchanges the processor broadcasts a beep if $X(j) = 1$ where j is the index of that exchange and only listens if $X(j) = 0$. If at any of the exchanges it listens it hears a beep it sets $v = 0$ and stops broadcasting (even in the selected exchanges). If a node hears a beep during these exchanges it does not exit the algorithm. Instead, it denotes the fact that one of its neighbors beeped and sets itself to be inactive. If it does not hear a beep in any of the exchanges of a future phase it becomes active and continues as described above. Similarly, a node that beeped and did not hear any beep in a specific step (indicating that it can join the MIS) continues to beep indefinitely (by selecting half the exchanges in all future steps and following the algorithm above).

However, the guarantees we provide differ from those in the stronger collision detection model. Specifically, the algorithm guarantees that after a certain time (which depends on the network size and is derived below), all MIS members are fixed, and the safety requirements hold (all nodes not in the MIS are connected to a node in the MIS and no two MIS members are connected). Until this time nodes can decide to become MIS members and later drop from the set if they find out that one of their neighbors has also decided to join the MIS. Since nodes do not have an estimate of the network size the processors continue to perform the algorithm indefinitely. At the end of the section we describe a stopping criteria that could be used if an estimate of the network size were available.

The main difference between this algorithm and Algorithm 2 a set of competition exchanges that are added at the end of each coin flip. The number of competition exchanges is equal to the current phase counter (which serves as the current estimate of the network size). Initially the competition rounds are short and so they would not necessarily remove all collisions. We require that nodes that attempt to join continue to participate in all future competition rounds (when $v = 1$). Processors that detect a MIS member as a neighbor set z to 1 and do not broadcast until they go through one complete set of competition exchanges in which they do not hear any beep. If and when this happens they set $z = 0$ and become potential MIS candidate again.

While not all collisions will be resolved at the early phases, when $x \geq \log n$ each set of competition exchanges is very likely to remove all collisions. We prove below that once we arrive at such x values, all collisions are resolved with very high probability such that only one processor in a set of colliding processors remains with $v = 1$ at the end of these competition exchanges. From there, it takes another $O(\log n)$ phases to select all members of the MIS as we have shown for Algorithm 1. Since each such phase takes ($O(\log n)$ steps with each step

Algorithm 3 MIS with wake-on-beep *without* sender-side collision detection

```
1: upon waking up (by adversary or beep) do beep to wake up neighbors
2: wait for 1 round; x ← 0; v ← 0; z ← 0                    ▷ while neighbors wake up
3: repeat forever
4:     x ← x + 1
5:     for i ∈ {0, ..., x} do
6:         if v = 0 ∧ z = 0 then with probability 1/2^i set v ← 1
7:         X ← random 0/1-vector of length cx                ▷ c is a constant
8:         z ← 0
9:           ** cx competition exchanges **
10:        for k ∈ {1, ..., cx} do
11:            listen for 1 round
12:            if beep received then v ← 0; z ← 1        ▷ z = 1: conn. to node in MIS
13:            if v = 0 ∨ X[k] = 0 then
14:                listen for 1 round; if beep received then v ← 0; z ← 1
15:            else                                          ▷ v = 1 ∧ X[k] = 1
16:                beep for 1 round
17:            listen for 1 round; if beep received then v ← 0;
```

taking $O(\log n)$ rounds for the competition, the total run time of the algorithm is $O(\log^3 n)$.

Below we prove that if two neighbors in the network have $v = 1$ after the coin flip in step i in a phase with $x \geq \log n$, then with high probability one would set $v = 0$ at the end of step i of that phase and so at most one of them enters the MIS.

Lemma 3. *Assume processor y collided with one or more of its neighbors setting $v = 1$ in step i of phase $x \geq \log n$. Then the probability that y would still be colliding with any of its neighbors at the end of the cx competition exchanges for step i is $\leq \frac{1}{n^{c/3}}$.*

Proof. If at any of the exchanges in this step all neighbors of y have $v = 0$ we are done. Otherwise in each exchange, with probability at least $1/4$ y decided not to broadcast whereas one of its colliding neighbors decided to broadcast. Thus, the probability that y does not resolve its collision in a specific exchange is $\leq 3/4$. Since there are $(c \log n)$ exchanges in this step, the probability that y is still colliding at the end of these competition exchanges $\leq (\frac{3}{4})^{c \log n} \leq \frac{1}{n^c/3}$. □

Note that if a collision is not resolved in a specific exchange the colliding nodes continue to broadcast in the following phase. As we proved in the previous section, if all collisions are resolved in the $O(\log n)$ phases that follow the phase $x = \log n$ the algorithm will result in a MIS set with very high probability. Since we only need $O(\log^2 n) < n$ steps for this, and we have n nodes, the probability that there exists a step and a node in phase $x \geq \log n$ such that a node that collided during this step with a neighbor does not resolve this collision in that step is smaller than $\frac{1}{n^{c/3-2}}$. Thus, with probability $\geq 1 - \frac{1}{n^{c/3-2}}$ all collisions are detected and the MIS safety condition holds.

7.1 Stopping criteria when an upper bound on network size exists

TODO: I would remark that if you use this stopping criteria you are turning the algorithm into monte carlo, since if you stop there is a small chance that your output is wrong. The above algorithm leads to a MIS set and does not require knowledge of the network size. However, the time it takes to reach the MIS set is a function of the size of the network and so if nodes do not have an estimate of this number they cannot terminate the algorithm and need to indefinitely listen to incoming messages. Note that, as the analysis above indicates, if a rough estimate on the network size n exists, nodes can terminate the algorithm when $x = 2\log n + 1$. At that phase we have a high probability that every collision that has occurred during the last $\log n$ phases has been resolved ($\geq 1 - \frac{1}{n^{c/3-2}}$) and as proved in the previous section when all collisions are resolved the algorithm terminates with very high probability.

8 Synchronized Clocks

For this section the only assumption we make on top of the beep model is that that nodes have synchronized clocks, i.e., know the current round number t.

Algorithm: Nodes have three different internal states: *inactive*, *competing*, and *MIS*. Each node has a parameter k that is monotone increasing during the execution of the algorithm. All nodes start in the inactive state with $k = 6$.

Nodes communicate in beep-triples, and synchronize by starting a triple only when $t \equiv 0 \pmod 3$. The first bit of the triple is the Restart-Bit. A BEEP is sent for the Restart-Bit if and only if $t \equiv 0 \pmod k$. If a node hears a BEEP on its Restart-Bit it doubles its k and if it is active it becomes inactive. The second bit sent in the triple is the MIS-Bit. A BEEP is sent for the MIS-Bit if and only if a node is in the MIS state. If a node hears a BEEP on the MIS-bit it becomes inactive. The last bit send in a triple is the Competing-Bit. If inactive, a node listens to this bit, otherwise it sends a BEEP with with probability 1/2. If a node hears a BEEP on the Competing-Bit it becomes inactive. Furthermore, if a node is in the MIS-state and hears a BEEP on the Competing-Bit it doubles its k. Lastly, a node transitions from inactive to active between any time t and $t+1$ for $t \equiv 0 \pmod k$. Similarly, if a node is active when $t = 0 \mod k$ then it transitions to the MIS state. In the sequel, we refer to this algorithm as Algorithm 2. The state transitions are also depicted in Figure 1.

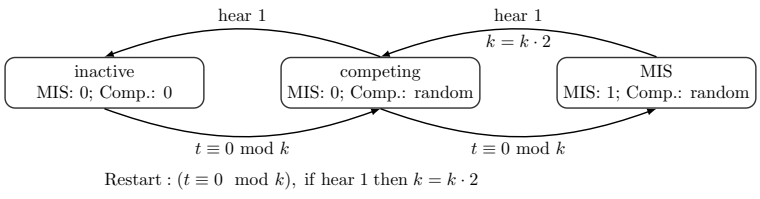

Fig. 1. State Diagram for Algorithm 2

Idea: The idea of the algorithm is to employ Luby's permutation algorithm in which a node picks a random $O(\log n)$-size priority which it shares with its neighbors. A node then joins the MIS if it has the highest priority among its neighbors, and all neighbors of an MIS node become inactive. Despite the fact that this algorithm is described for the message exchange model, it is straightforward to adapt the priority comparisons to the BEEP model. For this, a node sends its priority bit by bit, starting with the highest-order bit and using a BEEP for a 1. The only further modification is that a node stops sending its priority if it has already heard a BEEP on a higher order bit during which it remained silent because it had a zero in the corresponding bit. Using this simple procedure, a node can easily realize when a neighboring node has a higher priority. Furthermore, a node can observe that it has the highest-priority in its neighborhood which is exactly the case if it does not hear any BEEP .

Therefore, as long as nodes have a synchronous start and know n (or an upper bound) it is straightforward to get Luby's algorithm working in the beep model in $O(\log^2 n)$ rounds.

In the rest of this section we show how to remove the need for an upper bound on n and a synchronous start. We solely rely on synchronized clocks to synchronize among nodes when a round to transmits a new priority starts. Our algorithm uses k to compute an estimate for the required priority-size $O(\log n)$. Whenever a collision occurs and two nodes tie for the highest priority the algorithm concludes that k is not large enough yet and doubles its guess. The algorithm uses the Restart-Bit to ensure that nodes locally work with the same k and run in a synchronized manner in which priority comparisons start at the same time (namely every $t \equiv 0 \pmod{k}$). It is not obvious that either a similar k or a synchronized priority comparison is necessary but it turns out that algorithms without them can stall for a long time. In the first case this is because repeatedly nodes with a too small k enter the MIS state simultaneously while in the second case many asynchronously competing nodes (even with the same, large enough k) keep eliminating each other without one becoming dominant and transitioning into the MIS state.

Analysis: To proof the algorithm's correctness, we first show two lemmas that show that with high probability k cannot be super-logarithmic.

Lemma 4. *With high probability $k \in O(\log n)$ for all nodes during the execution of the algorithm.*

Proof. We start by showing that two neighboring nodes u, v in the MIS state must have the same k and transitioned to the MIS state at the same time. We prove both statements by contradiction.

For the first part assume that nodes u and v are in the MIS state but u transitioned to this state (the last time) before v. In this case v would have received the MIS-bit from u and become inactive instead of joining the MIS, a contradiction.

Similarly, for sake of contradiction, we assume that $k_u < k_v$. In this case, during the active phase of u before it transitioned to the MIS at time t it would

have set its Restart-bit to 0 at time $t - k_u$ and received a 1 from v and become inactive, contradicting the assumption that $k_u < k_v$.

Given this we now show that for a specific node u it is unlikely to become the first node with a too large k. For this we note that k_u gets doubled because of a Restart-Bit only if a BEEP from a node with a larger k is received. This node can therefore not be responsible for u becoming the first node getting a too large k. The second way k can increase is if a node transitions out of the MIS state because it receives a Competing-Bit from a neighbor v. In this case, we know that u competed against at least one such neighbor for k rounds with none of them loosing. The probability of this to happen is 2^{-k}. Hence, if $k \in \Theta(\log n)$, this does not happen w.h.p. A union bound over all nodes and the polynomial number of rounds in which nodes are not yet stable finishes the proof. □

Theorem 2. *If during an execution the $O(\log n)$ neighborhood of a node u has not changed for $\Omega(\log^2 n)$ rounds then u is stable, i.e., u is either in the MIS state with all its neighbors being inactive or it has at least one neighbor in the MIS state whose neighbors are all inactive.*

Proof. First observe that if the whole graph has the same value of k and no two neighboring nodes transition to the MIS state at the same time, then our algorithm behaves exactly as Luby's original permutation algorithm, and therefore terminates after $O(k \log n)$ rounds with high probability. From a standard locality argument, it follows that a node u also becomes stable if the above assumptions only hold for a $O(k \log n)$ neighborhood around u. Moreover, since Luby's algorithm performs only $O(\log n)$ rounds in the message passing model, we can improve our locality argument to show that in if a $O(\log n)$ neighborhood around u is well-behaved, then u behaves as in Luby's algorithm.

Since the values for k are monotone increasing and propagate between two neighboring nodes u, v with different k (i.e., $k_u > k_v$) in at most $2k_u$ steps, it follows that for a node u it takes at most $O(k_u \log n)$ rounds until either k_u increases or all nodes v in the $O(\log n)$ neighborhood of u have $k_v = k_u$ for at least $O(k \log n)$ rounds. We can furthermore assume that these $O(k \log n)$ rounds are collision free (i.e, no two neighboring nodes go into the MIS), since any collision leads with high probability within $O(\log n)$ rounds to an increased k value for one of the nodes.

For any value of k, within $O(k \log n)$ rounds a node thus either performs Luby's algorithm for $O(\log n)$ priority exchanges, or it increases its k. Since k increases in powers of two and, according to Lemma 4, with high probability it does not exceed $O(\log n)$, after at most $\sum_i^{O(\log \log n)} 2^i \cdot 3 \cdot O(k \log n) = O(\log^2 n)$ rounds the status labeling around a $O(\log n)$ neighborhood of u is a proper MIS. This means that u is stable at some point and it is not hard to verify that the function of the MIS-bit guarantees that this property is preserved for the rest of the execution. □

We remark that as the algorithm of Section 5, this algorithm is also robust enough to work as-is with an adversary capable of crashing nodes (with the same caveats on the guarantees mentioned in Section 5).

References

[1] Y. Afek, N. Alon, O. Barad, E. Hornstein, N. Barkai, and Z. Bar-Joseph. A biological solution to a fundamental distributed computing problem. *Science*, 331(6014):183–185, 2011.

[2] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Journal of Algorithms*, 7(4):567–583, 1986.

[3] B. Awerbuch, A. V. Goldberg, M. Luby, and S. A. Plotkin. Network decomposition and locality in distributed computation. In *Proc. of the 30th Symposium on Foundations of Computer Science (FOCS)*, pages 364–369, 1989.

[4] B. Chlebus, L. Gasieniec, A. Gibbons, A. Pelc, and W. Rytter. Deterministic broadcasting in unknown radio networks. In *Prof. 11th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 861–870, 2000.

[5] J.R. Collier, N.A. Monk, P.K. Maini, and J.H. Lewis. Pattern formation by lateral inhibition with feedback: a mathematical model of delta-notch intercellular signalling. *J Theor Biol*, 183(4):429–46, 1996.

[6] A. Cornejo and F. Kuhn. Deploying wireless networks with beeps. In *Proc. of 24th Symposium on Distributed Computing (DISC)*, pages 148–162, 2010.

[7] J. Degesys, I. Rose, A. Patel, and R. Nagpal. Desync: self-organizing desynchronization and TDMA on wireless sensor networks. In *Prof. 6th Conf. on Information Processing in Sensor Networks (IPSN)*, page 20, 2007.

[8] R. Flury and R. Wattenhofer. Slotted programming for sensor networks. *Proc. 9th Conference on Information Processing in Sensor Networks (IPSN)*, 2010.

[9] D. Ilcinkas, D. Kowalski, and A. Pelc. Fast radio broadcasting with advice. *Theoretical Computer Science*, 411(14-15), 2010.

[10] T. Jurdziski and G. Stachowiak. Probabilistic algorithms for the wakeup problem in single-hop radio networks. *Proc. 13th International Symposium on Algorithms and Computation (ISAAC)*, 2002.

[11] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer. Fast deterministic distributed maximal independent set computation on growth-bounded graphs. In *Proceedings of the 19th International Symposium on Distributed Computing (DISC'05)*, pages 273 – 287, 2005.

[12] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. What cannot be computed locally! In *Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing, PODC 2004, St. John's, Newfoundland, Canada, July 25-28, (PODC)*, pages 300–309, 2004.

[13] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. The price of being near-sighted. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006 (SODA)*, pages 980–989, 2006.

[14] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 15:1036–1053, 1986.

[15] Y. Métivier, J. Michael Robson, N. Saheb-Djahromi, and A. Zemmari. An optimal bit complexity randomized distributed mis algorithm. *Proc.*

16th Colloquim on Structural Information and Communication Complexity (SIROCCO), 2009.

[16] T. Moscibroda and R. Wattenhofer. Efficient computation of maximal independent sets in structured multi-hop radio networks. *Proc. of 1st International Conference on Mobile Ad Hoc Sensor Systems (MASS)*, 2004.

[17] T. Moscibroda and R. Wattenhofer. Maximal Independent Sets in Radio Networks. *Proc. 24th Symposium on Principles of Distributed Computing (PODC)*, 2005.

[18] A. Motskin, T. Roughgarden, P. Skraba, and L. Guibas. Lightweight coloring and desynchronization for networks. In *Proc. 28th IEEE Conf. on Computer Communications (INFOCOM)*, 2009.

[19] A. Panconesi and A. Srinivasan. On the complexity of distributed network decomposition. *Journal of Algorithms*, 20(2):581–592, 1995.

[20] D. Peleg. *Distributed computing: a locality-sensitive approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000. ISBN 0-89871-464-8.

[21] C. Scheideler, A. Richa, and P. Santi. An $O(\log n)$ dominating set protocol for wireless ad-hoc networks under the physical interference model. *Proc. 9th Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, 2008.

[22] J. Schneider and R. Wattenhofer. A Log-Star Maximal Independent Set Algorithm for Growth-Bounded Graphs. *Proc. 28th Symposium on Principles of Distributed Computing (PODC)*, 2008.

[23] J. Schneider and R. Wattenhofer. What is the use of collision detection (in wireless networks)? In *Proc. of 24th Symposium on Distributed Computing (DISC)*, pages 133–147, 2010.

[24] P.-J. Wan, K. M. Alzoubi, and O. Frieder. Distributed construction of connected dominating set in wireless ad hoc networks. *Mobile Networks and Applications*, 9(2):141–149, 2004.