Can Uncoordinated Beeps tell Stories?

Fabien Dufoulon, Janna Burman, Joffroy Beauquier

Can Uncoordinated Beeps tell Stories?

FABIEN DUFOULON, Faculty of Industrial Engineering and Management, Technion
JANNA BURMAN and JOFFROY BEAUQUIER, Université Paris-Saclay, CNRS, LRI

The *beeping model* is an extremely restrictive communication model. Nodes communicate in discrete rounds using *beeps*—simple bursts of energy—and carrier sensing. Simultaneous beeps produce (non-destructive) collisions, resulting in information loss. Such communication differs greatly from the traditional communication mechanisms in distributed systems, like message-passing or shared memory. Indeed, a beep is a *unary signal* that communicates no information (e.g., no message content, nor sender information) beyond its own presence. As a result, in a round of beeping communication, hearing a beep means only that some (unknown) neighboring node is communicating in this very round, whereas silence (i.e., hearing no beeps) means only that no neighboring node is communicating.

Most previous works assume that nodes all start (wake up) at the same time. In this (synchronous starts) setting nodes have synchronized local clocks and thus synchronized round numbers. Via these round numbers, information can be *extrinsically* conveyed in a round of beeping communication. For example, the parity of the round number allows to convey letters in $\{0, 1\}$. In contrast, we consider here that nodes start in an uncoordinated manner. Thus two nodes may have arbitrarily different round numbers and no information can be extrinsically conveyed in a single round. In the present paper, we show how *non-trivial information*—e.g., letters from a binary alphabet—can be conveyed through several rounds instead. Applying tools from coding theory and additive number theory, we propose *communication schemes*—binary words of length l specifying how a node communicates during l consecutive rounds—allowing nodes to convey letters from an alphabet of size h, for any constant $h \geq 2$ (known by all nodes). Direct application of such schemes allows to implement a message-passing primitive with an exponential (in the number of message bits) multiplicative overhead. Here, in contrast, we design an exponentially more efficient message-passing primitive.

First, we use these communication schemes to implement a *2-hop beep* communication primitive, simulating beeping communication on the square of the communication graph. Building upon this primitive, we present the first solution to the *2-hop desynchronization* problem in the beeping model with uncoordinated starts. This is a fundamental interference control problem, in which nodes must compute (periodic) infinite communication schemes, disjoint from those chosen by the nodes at distance one and two on the communication graph (thus, the schemes are said to be *2-hop desynchronized*). That way, nodes may communicate while avoiding collisions and information loss. Finally, we show how nodes can use these 2-hop desynchronized schemes to simulate a convenient, general (i.e., not limited to any predefined alphabet size h) and efficient (i.e., with an overhead linear in the number of bits of the message) message-passing abstraction layer.

CCS Concepts: • **Theory of computation → Distributed algorithms**; *Error-correcting codes.*

Additional Key Words and Phrases: beeping model, uncoordinated starts, local message broadcast, interference control

1 INTRODUCTION

In the present paper we study radio networks composed of devices with severely restricted communication capabilities. These devices do not exchange information via messages but instead communicate using bursts of energy (i.e., *beeps*) and carrier sensing. To formally analyze algorithms in these networks, we consider the *discrete beeping model* [6]. In this model, time is discrete and divided into synchronous rounds. During each round, a node can beep (signal) or listen. If it beeps, it cannot know whether or not any of its neighbors beeped during the round (*sender-side collision*, if at least another neighbor beeps). If it listens, it can distinguish if any of its neighbors beeped or none (*receiver-side collision*, if at least two neighbors beep).

Authors' addresses: Fabien Dufoulon, dfabien@campus.technion.ac.il, Faculty of Industrial Engineering and Management, Technion; Janna Burman, janna.burman@lri.fr; Joffroy Beauquier, joffroy.beauquier@lri.fr, Université Paris-Saclay, CNRS, LRI.

The beeping model is a weak distributed computing model, making little demands on the communication devices. As a result, it is very general and its solutions apply to most types of radio networks. Previous works have emphasized the model's relevance to radio networks with reduced network stacks [9] as well as its usefulness for studying distributed systems emerging from natural phenomena (e.g., firefly swarms [10] or biological cellular networks [1]).

However, designing algorithms in the beeping model is challenging. Indeed, unlike most communication mechanisms in distributed computing, beeps are unary signals, and communicate no message content, nor sender information. Moreover, beeps suffer from *interference* (i.e., collisions) which causes information loss (simultaneous beeps are received as a single beep). For that reason, providing efficient *communication primitives* is important for developers. However, the only information *intrinsically* conveyed in a round of beeping communication is whether some unknown neighboring node is communicating in this very round. In other words, a round of beeping communication intrinsically conveys *non-zero but trivial* information (i.e., at most a letter from an alphabet of size 1). Therefore, any communication primitive must build upon a technique that allows beeping communication to *extrinsically* convey *non-trivial information* (i.e., letters from some alphabet of size at least 2), or in other words, that allows nodes to locally broadcast (meaningful) messages using beeping communication. Techniques prior to this work can be separated into two different approaches. On the one hand, nodes can use a 2-hop symmetry-breaking problem to compute communication schemes that avoid collisions (e.g., 2-hop coloring [2]) and thus convey information. Alternatively, nodes can use coding techniques to compute communication schemes that convey information despite collisions (e.g., superimposed codes [8]). However, [2, 8] require strong synchronization between neighboring nodes: it is assumed that nodes all start at the same time (or at least in some coordinated way).

Contributions. The present paper deals with the *uncoordinated starts setting*, in which nodes wake up in an uncoordinated manner (i.e., with arbitrary time offsets). We show how nodes can (efficiently) locally broadcast meaningful information (messages)—through several rounds of beeping communication—even in this more difficult setting. To do so, we use a combination of the above-mentioned approaches. First, we introduce coding techniques suited to the uncoordinated starts setting—uncoordinated superimposed codes. Such codes can be used to locally broadcast (at most k per neighborhood) messages of B bits in $O(\exp(c \cdot B \cdot k))$ rounds, for some constant $c \geq 1$. Then, using such codes, we give a solution to the 2-hop desynchronization problem allowing for more efficient local broadcasts of messages. In this problem, every node has to compute, from some round onwards, an infinite communication scheme—that is, an infinite binary word specifying for each round whether a node beeps or not—disjoint from those of its 2-hop neighbors. Additionally, the communication scheme should be periodic (with period length $T = O(\Delta_{up}^4)$, where Δ_{up} is a known upper bound on the maximum degree of the communication graph). The time complexity of our solution is $O(\Delta_{up}^4 \log n)$ rounds w.h.p. (i.e., with probability at least $1 - O(\frac{1}{n})$). Such a solution allows nodes to avoid collisions (both sender-side and receiver-side collisions) in the uncoordinated starts setting[1]. Given this solution, we present a local message broadcast primitive, allowing a node to send a message of B bits in $O(\Delta_{up}^4 \cdot B)$ rounds.

1.1 Roadmap

First, we introduce *uncoordinated superimposed codes* (Section 3.1), a variant of superimposed codes [11]. Unfortunately, constructions for superimposed codes (and other similar combinatorial structures, see the discussion in the related

[1]In contrast, other symmetry-breaking problems, such as 2-hop coloring, may not allow to avoid collisions in this setting. With 2-hop coloring, although nodes get unique colors in their 2-hop neighborhoods, they have no common view of time and thus nodes cannot use these colors to compute communication schemes disjoint from those of their 2-hop neighbors.

work) cannot be straightforwardly translated to constructions for uncoordinated superimposed codes. Thus, we also provide such a construction, by exploiting properties of *Sidon sets*—see Section 3.2.

To solve 2-hop desynchronization using uncoordinated superimposed codes, we implement a *2-hop beep communication primitive* (Section 4). This primitive allows nodes to communicate to their 2-hop neighbors. As the primitive is quite general, it can also be used to solve other problems (e.g., translate a maximal independent set algorithm to a 2-hop maximal independent set solution) in the beeping model with uncoordinated starts. The complete 2-hop desynchronization solution is given in Section 5.

Finally, we present a local message broadcast primitive (Section 6). Nodes compute 2-hop desynchronized communication schemes, which they can then use to locally broadcast messages bit by bit in a reliable manner.

1.2 Related Work

On the Beeping Model. To the best of the authors' knowledge, [1, 6] are the only works assuming arbitrary wake-ups and a multi-hop communication graph in the beeping model. Similarly to the current work, these two results require some a priori knowledge on the communication graph: the maximum degree Δ or the number of nodes n, or upper bounds on these values. Interestingly, [1, 6] show that non-zero but trivial information—the only information that can be intrinsically conveyed in a round of beeping communication—is enough to design probabilistic solutions to both the maximal independent set and 1-hop desynchronization problems.

In [1], a probabilistic solution is given for maximal independent set (MIS). Nodes know an upper bound N on the number of nodes n in the graph, and decide whether they are in the MIS or not in $O(\log^2 N \log n)$ rounds w.h.p.
Cornejo and Kuhn [6] give a probabilistic algorithm solving the *1-hop desynchronization* problem (also called interval coloring). In this problem, every node is required to compute, from some round onwards, a (periodic) infinite communication scheme—with a period of (known) length $T = O(\Delta)$—disjoint from those of its neighbors. The time complexity of the solution in [6] is $O(T \log n)$ rounds w.h.p. However, such a solution does not allow nodes to avoid receiver-side collisions. Indeed, two neighbors of a node, at distance 2 of each other, can (correctly) compute non-disjoint communication schemes. Consequently, the communication schemes obtained with this solution are not 2-hop desynchronized, nor do they allow nodes to simulate communication on the square communication graph—a crucial part of any 2-hop desynchronization solution.

On Superimposed Codes. Such codes, and in particular *zero-false-drop* (of order k) *superimposed codes* (abbreviated as ZFD_k codes), were introduced in [11]. A ZFD_k code is a set of codewords with the *k-cover-free* property, which ensures that every superposition (bitwise OR) of at most k codewords (from this set) does not contain (or cover) any other codeword. Such codes can allow to deal with simple interference (i.e., collisions) in wireless communications, in a decentralized manner. Indeed, a ZFD_k code C with h codewords of length l can be used to specify the communications of h users over (periodic phases of) l rounds. If at most k of these users (with unique codewords in $C = \{c_1, \ldots, c_h\}$) communicate during these l rounds—according to their codewords, that is, only in codeword-defined rounds—then the resulting interference can be overcome by all k users. In the beeping model with synchronous starts, [8] solves leader election efficiently by encoding messages using ZFD_k codes.

Similar combinatorial structures, such as *selective families* [3] and *radio synchronizers* [5], have been used for the broadcast and wake-up problems in radio networks [3–5]. More specifically, a k-selective family can also be used to specify communications over (periodic phases) of l rounds, but provides weaker "interference-tolerance" guarantees than a ZFD_k code. Radio synchronizers provide the same level of guarantees, but extended to arbitrarily started

transmissions of *unique* codewords. In contrast, the *uncoordinated superimposed codes* introduced here have stronger properties, extending the k-cover-free property to the more general case of arbitrarily started transmissions of *non-unique* codewords. In the construction of the proposed 2-hop communication primitives, these properties are absolutely necessary.

2 MODEL AND DEFINITIONS

2.1 Preliminaries

The *communication network* is represented by a static connected undirected graph $G = (V, E)$, where V is the node set and E—the edge set. The *network size* $|V|$ is denoted by n and the maximum degree by Δ. The *square of the communication graph* is denoted by $G^2 = (V_2, E_2)$, where $V_2 = V$ and $E_2 = E \cup \{\{v_1, v_2\} \in V^2 | \exists u \in V \setminus \{v_1, v_2\}$, s.t. $\{v_1, u\}, \{u, v_2\} \in E\}$. For any given node v, its one-hop neighborhood in G is denoted by $\mathcal{N}(v) = \{v\} \cup \{u \in V | \{v, u\} \in E\}$ and its 2-hop neighborhood (i.e., its 1-hop neighborhood in G^2) by $\mathcal{N}_2(v) = \{v\} \cup \{u \in V | \{v, u\} \in E_2\}$. Node v is included in both sets. For any variable var, var_v denotes its value in node v.

We use the terminology of formal language theory and focus on the binary alphabet $\Sigma = \{0, 1\}$. The free monoid on Σ—the set of finite binary words —is denoted by Σ^* and Σ^ω is the set of infinite words on Σ. The empty word is denoted by ϵ. The *length* of a word $x \in \Sigma^*$ is denoted by $|x|$. For any word x in Σ^* or Σ^ω, $x[j]$ denotes the j^{th} bit of x and $x[i, j]$ the *factor* of x, from the i^{th} to the j^{th} bit. For any two finite (respectively, infinite) binary words x and y, x and y are said to be *disjoint* if $\forall i \in \{1, \ldots, \min\{|x|, |y|\}\}$ (resp., $\forall i \in \mathbb{N}^*$), $\neg(x[i] = 1 \text{ and } y[i] = 1)$. Similarly, for any positive integer and two finite (respectively, infinite) binary words x and y, x and y are said to be *k-disjoint* if $\forall i, j \in \{1, \ldots, \min\{|x|, |y|\}\}$ (resp., $\forall i, j \in \mathbb{N}^*$) such that $|i - j| \leq k$, $\neg(x[i] = 1 \text{ and } y[j] = 1)$. For any positive integer i and word $u \in \Sigma^*$, u^i denotes the i-fold concatenation of u (where $u^0 = \epsilon$) and u^ω denotes the infinite binary word $uuu \cdots$. A word $x \in \Sigma^\omega$ is said to be *periodic*, of *period* u and *period length* p, if there is a finite word u of minimum length p such that $x = u^\omega$.

The following (superposition) operations are illustrated in Figures 1 and 2. The operator $\vee$ is for the *logical disjunction* (i.e., binary OR) on Σ. For any two words $x, y \in \Sigma^*$ of the same length, we define the (bitwise OR) *superposition* of x and y, $\vee(x, y)$, as the binary word w of length $|w| = |x|$ such that $\forall i \in \{1, \ldots, |w|\}$, $w[i] = x[i] \vee y[i]$. We naturally extend the superposition (and its operator) to arbitrary sets $\{x_1, \ldots, x_c\}$ of finite binary words of the same length. Additionally, for any two words $x, y \in \Sigma^*$ of the same length, y is said to *contain* (or *cover*) x (alternatively, x is said to be *included* in y) if $\forall i \in \{1, \ldots, |x|\}$, $x[i] = 1 \Rightarrow y[i] = 1$.

Finally, we consider superpositions of arbitrarily shifted words (in Σ^*). First, we define a *shifted word* $s = \sigma(x, t)$, for any word $x \in \Sigma^*$ and shift $t \in \{-|x| + 1, \ldots, |x| - 1\}$ as a binary word of length $|s| = |x|$ such that $\forall i \in \{1, \ldots, |s|\}$, $s[i] = x[-t + i]$ if $-t + i \in \{1, \ldots, |x|\}$ and $s[i] = 0$ otherwise. We also define an *extended shifted word* $s = \overline{\sigma}(x, t)$, for any word $x \in \Sigma^*$ and shift $t \in \{-|x| + 1, \ldots, |x| - 1\}$, as the binary word $s = 0^{|x|-1+t}\ x\ 0^{|x|-1-t}$ (of length $3|x| - 2$). Then, we define the *uncoordinated superposition* of a set $\{(x_1, t_1), \ldots, (x_c, t_c)\}$ of finite binary word (of the same length) and shift pairs, as the superposition of the set of shifted words $\{\sigma(x_1, t_1), \ldots, \sigma(x_c, t_c)\}$. Similarly, we define the *extended uncoordinated superposition* of a set $\{(x_1, t_1), \ldots, (x_c, t_c)\}$ of finite binary word (of the same length) and shift pairs, as the superposition of the set of extended shifted words $\{\overline{\sigma}(x_1, t_1), \ldots, \overline{\sigma}(x_c, t_c)\}$.

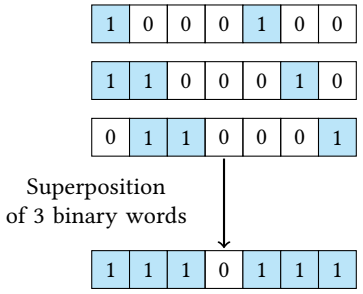

Fig. 1. Superposition of the binary words 1000100, 1100010 and 0110001.

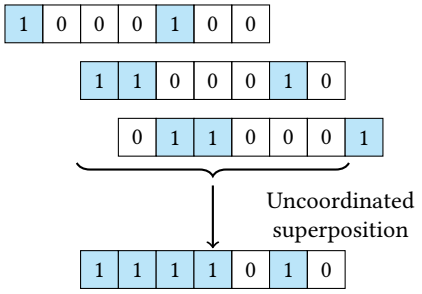

Fig. 2. Uncoordinated superposition of the set of shifted words $X = \{(1000100, -2), (1100010, 0), (0110001, 1)\}$.

2.2 Model Definitions

We consider the *beeping model* as originally defined in [6], that is, in the uncoordinated starts setting. Direct communication is only possible between neighboring nodes in the static communication graph G, and nodes have no knowledge of this graph. Time is divided into discrete time intervals, called (*global*) *rounds*. The first global round is the round in which the first node wakes up (the node itself is not aware of that). Any other node wakes up spontaneously at an *arbitrary* round (arbitrary time offset). From this *wake-up round* onwards, the node is said to be *awake*. In each subsequent round (and synchronously with other awake nodes) the node executes the following steps. First, it beeps (instruction *BEEP* in algorithms) or listens (*LISTEN* in algorithms). Beeps are transmitted to all (awake) neighbors of the beeping node during the round. Then, if the node listens (in the previous step of the same round), it knows whether or not at least one of its neighbors beeped (during the previous step of the same round). Finally, the node performs local computations.

Besides the static communication graph G, we also define the *awake communication graph* $G^a(r)$, for any given global round r, induced by the vertex set $V^a(r) = \{u \in V |\ u \text{ is awake in } r\}$. Then, for any awake node v in global round r, we define the *reachable 1-hop neighborhood* of v, denoted by $\mathcal{N}^a(v, r)$, as the 1-hop neighborhood of v in $G^a(r)$ and the *reachable 2-hop neighborhood* of v, denoted by $\mathcal{N}^a_2(v, r)$, as the (reachable) 2-hop neighborhood of v in $G^a(r)$.

Local Variables. For any given global round r, an awake node v (in r) knows only the *local* round number r_v (round number relative to node v's wake-up round). For any local round r_v, v is unaware of the global round. Thus, any two nodes u and v may have uncoordinated local clocks (i.e., arbitrarily different local round values). For the sake of analysis, for any given node v, a function g_v is defined such that for any local round $r_v \geq 1$, $g_v(r_v)$ is the global round corresponding to r_v. Additionally, g_v has an inverse function, denoted by g_v^{-1}.

For any node v, $\mathcal{H}_v$ denotes the *history* of v, defined as an infinite binary word s.t. $\mathcal{H}_v[r_v] = 1$ if in some local round r_v, v or one of its neighbors beeped, and $\mathcal{H}_v[r_v] = 0$ otherwise. Similarly, for any node v, $\mathcal{B}_v$ denotes the *beep history* of v, defined as an infinite binary word s.t. $\mathcal{B}_v[r_v] = 1$ if in some local round r_v, v beeped, and $\mathcal{B}_v[r_v] = 0$ otherwise. We assume that every node has 2 corresponding history variables denoted also by $\mathcal{H}$ and $\mathcal{B}$ (by abuse of notation). They are used to store successive prefixes of the (infinite) histories $\mathcal{H}$ and $\mathcal{B}$ of an execution (by registering the details of incoming and outgoing beeping communications).

We define a (finite) *communication scheme* of length l (for some integer $l \geq 1$) as a binary word of length l. It indicates how a node should communicate during l consecutive rounds. More precisely, a node v is said to *beep* (according to) a (single) communication scheme s starting in round r_v if $\forall i \in \{1, \ldots, |s|\}$, v beeps in round $r_v + i - 1$ if and only if

$s[i] = 1$. In other words, v beeps (respectively, listens) according to the 1's (resp, 0's) in s. If a node beeps multiple schemes at the same time, then it beeps if and only if at least one scheme indicates that it should beep (i.e., even if some other scheme indicates that it should listen). As a natural extension, we define an *infinite communication scheme* s' as an infinite binary word. Two (possibly infinite) communication schemes s_u and s_v, beeped by some nodes u and v (possibly $u = v$) starting respectively in rounds r_u and r_v, are said to be *disjoint* if the two binary words $0^{g_u(r_u)}\ s_u$ and $0^{g_v(r_v)}\ s_v$ are disjoint.

2.3 Problem Definition

In the *2-hop desynchronization* problem, every node has to compute, from some round onwards, an infinite communication scheme disjoint from those of its 2-hop neighbors. Moreover, the communication scheme should be *periodic* (of period length T). A communication scheme satisfying these conditions is said to be *2-hop desynchronized*. Notice that nodes can only compute such schemes by using their local view of time (local rounds).

We say that a (probabilistic) distributed algorithm solves the 2-hop desynchronization problem in $\mathcal{R}$ rounds (i.e., has time complexity $\mathcal{R}$) if $\mathcal{R}$ rounds after all nodes have woken up, all nodes compute 2-hop desynchronized communication schemes (w.h.p.).

3 USING UNCOORDINATED SUPERIMPOSED CODES TO CONSTRUCT COMMUNICATION SCHEMES FOR THE UNCOORDINATED STARTS SETTING

A single round of beeping communication intrinsically conveys little information. A natural way to (extrinsically) convey more is to use multiple rounds. In particular, a node may transmit some message (i.e., binary word) m (starting in some round r) by beeping the communication scheme m starting in round r. However, such message transmissions raise new issues. Multiple nodes may transmit messages simultaneously (not necessarily starting in the same round), and since beeps suffer from interference (i.e., simultaneous beeps are received as a single beep), these messages (and the corresponding communication schemes) must be carefully chosen to successfully convey information. In particular, a node may transmit two different messages m_1 and m_2 starting in the same round, and in doing so, also beep according to some third communication scheme m_3 contained in the bitwise OR superposition of m_1 and m_2. Since the node did not transmit m_3, it is said to *falsely transmit* m_3. In the present work, we use a coding-based approach to obtain communication schemes that allow nodes to avoid false transmissions and correctly decode superpositions, even in the uncoordinated starts setting.

First, we use a simplified communication scenario to illustrate how a coding-based approach allows nodes to deal with interference. We consider the following *coordinated transmission* scenario: several neighbors of a given node v (and possibly also node v) transmit (each node possibly multiple) codewords of length l starting in the same round $g_v(r_v)$, and node v wants to deduce, given its (resulting) history factor[2] (of beeps and silences) $\mathcal{H}_v[r_v, r_v + l - 1]$, which codewords were transmitted by its neighbors.[3] The scenario is said to have *conflict* χ if at most χ different codewords were transmitted in v's neighborhood. Importantly, the history factor $\mathcal{H}_v[r_v, r_v + l - 1]$ is the bitwise OR superposition of codewords transmitted in v's neighborhood. The problem of extracting the original transmitted codewords from their superposition can be solved using *superimposed codes* [11] (see Definition 1). Such a code satisfies the k-cover-free property (for some integer k) on superposition of codewords, which guarantees that, for a scenario with conflict $\chi \leq k$,

[2]Defined in Section 2: the history of v limited to l consecutive rounds, starting in local round r_v.
[3]If node v transmits some codewords, then v only wants to deduce the transmitted codewords which are different from those it transmitted itself.

only transmitted codewords are contained in that history factor. In particular, this implies that no node falsely transmits codewords.

Definition 1. *A zero-false-drop (of order k) superimposed code C of length l is a set of $h \geq k$ binary codewords of length l such that every superposition of k or less different codewords of C does not contain any other codeword from C.*

However, in the uncoordinated starts setting, nodes may have arbitrarily different round numbers. Since these round numbers do not allow nodes to coordinate and start transmissions simultaneously, the communication cannot build upon superimposed codes. For that reason, we introduce (in Section 3.1) an extension of such codes—uncoordinated superimposed codes—which allows to extract the originally transmitted words in the uncoordinated scenario. In the following we say that these codes *solve an (uncoordinated) transmission scenario*. In such a scenario, several neighbors of a given node v (and possibly also node v) transmit codewords of length l in arbitrary rounds (without coordination) and yet for any local round r_v, the history factor $\mathcal{H}_v[r_v, r_v + l - 1]$ only contains codewords that have been transmitted (starting exactly in global round $g_v(r_v)$) in v's neighborhood. A single node is allowed to start multiple codeword transmissions (where a *codeword transmission* is now defined by the codeword and its starting round), and the scenario is said to have conflict χ if there were at most χ different codeword transmissions (where a node's multiple codeword transmissions are added up) started within $l-1$ rounds of r_v (i.e., in rounds $\{r_v - l + 1, \ldots, r_v + l - 1\}$) in v's neighborhood. In Section 3.2, we present a construction method for uncoordinated superimposed codes.

3.1 Uncoordinated Superimposed Codes

We start by defining uncoordinated superimposed codes as a natural extension of superimposed codes. This is done by extending the k-cover-free property to extended uncoordinated superpositions (i.e., to superpositions of (at most k) extended shifted codewords; defined in Section 2.1). Crucially for this work, the proposed codes guarantee the k-cover-property even if the superposition contains the same codeword arbitrarily shifted multiple times.[4]

Definition 2. *An (h, k)-uncoordinated superimposed code or USI(h, k)-code C of length l is a set of h binary codewords $\{c_1, \ldots, c_h\}$ of length l such that every extended uncoordinated superposition of a set of codeword and shift pairs $X_1 = \{(c'_1, t_1), \ldots, (c'_k, t_k)\}$, where $|X_1| \leq k$ and $\forall i \in \{1, \ldots, k\}$ $c'_i \in C$ and $t_i \in \{-l + 1, \ldots, l - 1\}$, does not contain any other extended shifted codeword (i.e., $\overline{\sigma}(c', t)$ such that $(c', t) \notin X_1$).*

In an uncoordinated transmission scenario, the history factor (of length l) is the uncoordinated superposition of a set of codeword and shift pairs, or in other words, the superposition of shifted codewords (and not of extended shifted codewords). However, a USI(h, k)-code, according to Definition 2, does not necessarily guarantee the k-cover-free property for uncoordinated superpositions of shifted codewords.[5] For that reason, we present a second—equivalent (see Theorem 1)—definition for uncoordinated superimposed codes below. It states that USI(h, k)-codes guarantee that any shifted codeword $\sigma(c', 0)$ (i.e., simply a codeword c') is not covered by the uncoordinated superposition of a set of (at most k) codeword and shift pairs $X_2 \not\ni (c', 0)$. Although the construction of uncoordinated superimposed codes—in Section 3.2—uses Definition 2, we rely on Definition 3 to deal with an uncoordinated transmission scenario (see Theorem 2).

[4]Note that the codeword and shift pairs (c, t'_1) and (c, t'_2) correspond to different extended shifted codewords (i.e., $\overline{\sigma}(c, t'_1) \neq \overline{\sigma}(c, t'_2)$) if and only if $t'_1 \neq t'_2$.

[5]A USI(h, k)-code, by Definition 2, only guarantees the k-cover-free property for superpositions of extended shifted codewords. In particular, notice that if some codeword c begins (respectively, ends) with s zeroes (for some integer $s \geq 1$), then for any $t \in \{l - s, \ldots, l - 1\}$ (resp., $t \in \{-(l-1), \ldots, -(l-s)\}$), the shifted codeword $\sigma(c, t)$ is 0^l, which is contained by any superposition of shifted codewords.

Definition 3. *A USI(h, k)-code C is a set of h binary codewords $\{c_1, \ldots, c_h\}$ of length l such that every uncoordinated superposition of a set of codeword and shift pairs $X_2 = \{(c'_1, t_1), \ldots, (c'_k, t_k)\}$, where $|X_2| \leq k$ and $\forall i \in \{1, \ldots, k\}$ $c'_i \in C$ and $t_i \in \{-l+1, \ldots, l-1\}$, contains a shifted codeword $\sigma(c', 0)$ (i.e., the codeword c') if and only if the codeword and shift pair $(c', 0) \in X_2$.*

Theorem 1. *Definitions 2 and 3 are equivalent.*

Proof. First, we consider a USI(h, k)-code C, as defined in Definition 2. Then, we show that it satisfies Definition 3. Let $X_2 = \{(c'_1, t_1), \ldots, (c'_k, t_k)\}$ be an arbitrary set of codeword and shift pairs where $|X_2| \leq k$ and $\forall i \in \{1, \ldots, k\}$ $c'_i \in C$ and $t_i \in \{-l+1, \ldots, l-1\}$. If $(c', 0) \in X_2$ for some codeword c', then the uncoordinated superposition of X_2 obviously contains c'. Now, consider that the uncoordinated superposition of X_2 contains some codeword c'. Then the extended uncoordinated superposition of X_2 contains the extended shifted codeword $\overline{\sigma}(c', 0)$ and due to the cover-free property of USI-codes, the codeword and shift pair $(c', 0) \in X_2$.

Now, we consider a USI(h, k)-code C, as defined in Definition 3. Then, we show that it satisfies Definition 2. Let $X_1 = \{(c'_1, t_1), \ldots, (c'_k, t_k)\}$ be an arbitrary set of codeword and shift pairs where $|X_1| \leq k$ and $\forall i \in \{1, \ldots, k\}$ $c'_i \in C$ and $t_i \in \{-l+1, \ldots, l-1\}$, and let (c', t') be an arbitrary codeword and shift pair not in X_1. We construct another set of codeword and shift pairs X_2 from X_1 in the following way: for every $(c'_i, t_i) \in X_1$, $(c'_i, t_i - t') \in X_2$ if $t_i - t' \in \{-l+1, \ldots, l-1\}$. Then, $|X_2| \leq |X_1|$ and if $t' = 0$, then $X_2 = X_1$. By construction, $(c', 0) \notin X_2$. Thus, the uncoordinated superposition of X_2 does not contain c' and the extended uncoordinated superposition of X_2 does not contain the extended shifted codeword $\overline{\sigma}(c', 0)$. Then, due to X_2's construction, the extended uncoordinated superposition of X_1 does not contain the extended shifted codeword $\overline{\sigma}(c', t')$. □

Theorem 2. *A USI(h, k)-code C solves an* uncoordinated transmission scenario *if it has a conflict $\chi \leq k$.*

Proof. Consider that for any given node v and any local round r_v, there are at most $\chi \leq k$ (not necessarily different) codeword transmissions started in node v's neighborhood within $l-1$ rounds of r_v. Then, one can define a set of codeword and shift pairs X as follows: X is the set of pairs (c, t) such that the transmission of the codeword c starts in $r' \in \{r_v - l + 1, \ldots, r_v + l - 1\}$ in node v's neighborhood, with a shift of $t = r' - r_v$ (thus in $\{-l+1, \ldots, l-1\}$). Notice that the multiple codeword transmissions of a single node result in different codeword and shift pairs in X. The set X is of cardinality at most k and the history factor $\mathcal{H}_v[r_v, r_v + l - 1]$ is the uncoordinated superposition of X. Since the communication builds upon the USI-code C, then by Definition 3 the history factor $\mathcal{H}_v[r_v, r_v + l - 1]$ contains a codeword c' if and only if $(c', 0) \in X$, or in other words, if and only if c' was transmitted in node v's neighborhood, starting in $g_v(r_v)$. □

Theorem 2 is used in the design of 2-hop communication primitives (in Section 4.2).

3.2 Using Sidon Sets to construct USI-Codes

Sidon sets are defined and used as building blocks for USI-codes. We employ the Sidon set construction (Theorem 3) given in [7]. This construction uses prime numbers. By the Bertrand-Chebyshev theorem, for every positive integer k, there is a prime number p such that $k \leq p \leq 2k$. As a result, for any integer k, a construction of USI(2,k)-codes of length $O(k^2)$ can be obtained. This construction gives a USI(2,$(\Delta_{up}+1)^2$)-code of length $O(\Delta_{up}^4)$ for any given Δ_{up} (Corollary 8) and is used in the 2-hop communication primitives presented in Section 4.2.

Definition 4 (Sidon Set). *An $(l, \mathcal{K})$ Sidon set is a subset D of $\{0, \ldots, l-1\}$ of size $\mathcal{K}$, such that all pairwise sums of elements in D are different.*

Theorem 3 ([7]). *Let p be an odd prime number such that $\mathcal{K} \leq p \leq 2\mathcal{K}$. Then the set of $\mathcal{K}$ integers $\{d_0, \ldots, d_{\mathcal{K}-1}\}$, where $\forall i \in \{0, \ldots, \mathcal{K}-1\}$ $d_i = 2ip + (i^2 \bmod p)$, is an $(l, \mathcal{K})$ Sidon set with $l \leq 2p^2 \leq 8\mathcal{K}^2$.*

Notice that Sidon sets are also *Golomb rulers* (Lemma 4), which are sets of integers such that all pairwise differences are different. Although both mathematical objects are in fact equivalent, we only prove one implication here. The distinct pairwise differences property is used later to construct a codeword of a USI-code.

Lemma 4 (Distinct pairwise differences). *Let D be an $(l, \mathcal{K})$ Sidon set. Then every non-zero element of $\{0, \ldots, l-1\}$ can be expressed at most once as a difference $d_1 - d_2 \bmod l$ of two elements $d_1, d_2 \in D$ (i.e., D is a Golomb ruler).*

Proof. Suppose by contradiction that D is not a Golomb ruler. Then there must exist elements $d_1, d_2, d_3, d_4 \in D$ with $d_2 - d_1 = d_3 - d_4$. However, then $d_2 + d_4 = d_3 + d_1$ which is a contradiction, since D is a Sidon set. □

Given a Sidon set, a USI-code with a single codeword can be obtained (Theorem 5). The codeword construction leverages the Sidon set's distinct pairwise differences property to guarantee the cover-free property of the resulting USI-code. Notice that the cardinality $\mathcal{K}$ of the Sidon set used in the construction determines the maximum number of concurrent transmissions that the USI-code tolerates: $k = \mathcal{K} - 1$.

Theorem 5. *Let D be an $(l, \mathcal{K})$ Sidon set. Define the set S as the set of integers $\{d+1, d \in D\}$, and the codeword c as a binary word of length l such that $c[p] = 1$ if $p \in S$. Then $\{c\}$ is a USI(1, $\mathcal{K}-1$)-code of length l.*

Proof. To prove Theorem 5, we first consider Lemma 6, which states that if an $(l,\mathcal{K})$ Sidon set is used to construct the codeword c of a USI(1,k)-code (with $k = \mathcal{K} - 1$), then any two extended shifted words $\overline{\sigma}(c, t)$ and $\overline{\sigma}(c, t')$ have at most one position where their bits are both 1.

Lemma 6. *Let D be an $(l, \mathcal{K})$ Sidon set. Define the set S as the set of integers $\{d+1, d \in D\}$, and the codeword c as a binary word of length l such that $c[p] = 1$ if $p \in S$. There exists at most one $j \in \{1, \ldots, 3l-2\}$ such that $\overline{\sigma}(c, t)[j] = \overline{\sigma}(c, t')[j] = 1$.*

Proof. By contradiction, assume that there exist $j, j_2 \in \{1, \ldots, 3l-2\}$, $j \neq j_2$ such that $\overline{\sigma}(c, t)[j] = \overline{\sigma}(c, t')[j] = 1$ and $\overline{\sigma}(c, t)[j_2] = \overline{\sigma}(c, t')[j_2] = 1$. Then, there exist $d_1, d_2, d_1', d_2' \in D$, where $d_1 \neq d_1'$, $d_2 \neq d_2'$, $d_2 > d_1$ and $d_2' > d_1'$, such that $d_2 - d_1 = d_2' - d_1'$. That contradicts the fact that D is a Sidon set. □

As a result of Lemma 6, the extended uncoordinated superposition s of k extended shifted words $\overline{\sigma}(c, t_i)$ has at most k positions with bit 1 in common with a different (arbitrary) extended shifted word $a' = \overline{\sigma}(c, t')$. However, a' has $\mathcal{K} > k$ positions with bit 1, and at most k of these in common with s. Thus s cannot contain a'. This guarantees the k-cover-free property of the USI(1,k)-code (Definition 2). □

It is apparent (by slightly modifying the proof of Theorem 5) that a USI-code with multiple codewords can be obtained by dividing the single codeword from the previous construction, into multiple smaller codewords, resulting in Theorem 7. Then, Corollary 8 results from Theorems 3 and 7.

Theorem 7. *Let D be an $(l, h \cdot \mathcal{K})$ Sidon set. Define the set S as the set of integers $\{d+1, d \in D\}$, the sets $S_1, \ldots, S_h$ as partitions of S where each S_i is of size $\mathcal{K}$, and for any $i \in \{1, \ldots, h\}$ the codeword c_i as a binary word of length l such that $c_i[p] = 1$ if $p \in S_i$. Then $\{c_1, \ldots, c_h\}$ is a USI(h,$\mathcal{K}$−1)-code of length l.*

Corollary 8. *For any given h and k, one can construct a USI(h,k) of length $l \leq 8h^2(k+1)^2$. Thus, for any given k, one can construct a USI(2,k) of length $l \leq 32(k+1)^2$.*

4 IMPLEMENTING 2-HOP COMMUNICATION PRIMITIVES

In this section, the BEEP2H and LISTEN2H primitives are presented. When executed in G, they simulate the effect of BEEP and LISTEN on the square communication graph G^2, albeit with some time delay δ (in number of rounds).

Just as invocations of BEEP and LISTEN define histories $\mathcal{B}$ and $\mathcal{H}$, invocations of BEEP2H and LISTEN2H define the 2-hop histories $\mathcal{B}^2$ and $\mathcal{H}^2$. These 2-hop histories correspond (exactly) to $\mathcal{B}$ and $\mathcal{H}$ on the square graph G^2 (of awake nodes). Implementations of the BEEP2H and LISTEN2H primitives compute the 2-hop histories in variables denoted (by abuse of notation) by $\mathcal{H}^2$ and $\mathcal{B}^2$. These variables should be *coherent* within 2-hop neighborhoods, that is, they should satisfy the following condition for any node v in some local round r_v:

$$\mathcal{H}_v^2[r_v] = 1 \Leftrightarrow (\mathcal{B}_v^2[r_v] = 1 \text{ or } \exists\, u \in \mathcal{N}_2^a(v, g_v(r_v)) \text{ s.t. } \mathcal{B}_u^2[g_u^{-1}(g_v(r_v))] = 1).$$

From the above condition, we derive a specification of BEEP2H and LISTEN2H (Definition 5 below) that allows a time delay $\delta > 1$ for 2-hop beeps (to satisfy the condition). The specification requires that the computed variables $\mathcal{H}^2$ and $\mathcal{B}^2$ of all nodes are *coherent within a delay of δ rounds*. More precisely, for any global round r, the permanent factors (i.e., factors that no longer change from round r onwards according to the specification) of the computed 2-hop history variables, $\mathcal{H}_v^2[1, g_v^{-1}(r-\delta)]$ and $\mathcal{B}_v^2[1, g_v^{-1}(r-\delta)]$ for any node v awake in global round $r-\delta$, are coherent. We present algorithms for the two primitives (Algorithms 1 and 2) satisfying this specification for some even integer $\delta > 1$ (see Theorem 12).

Definition 5 (Specification of BEEP2H and LISTEN2H). *A node v implements the BEEP2H and LISTEN2H primitives—through its variables $\mathcal{H}_v^2$ and $\mathcal{B}_v^2$ (each initialized to 0^ω)—with some delay $\delta \geq 1$ if:*

- *v invokes BEEP2H or LISTEN2H in every local round r_v.*
- *If v invokes BEEP2H in local round r_v, then v sets $\mathcal{B}_v^2[r_v] := 1$ and $\mathcal{H}_v^2[r_v] := 1$ at the end of round r_v.*
- *If v invokes LISTEN2H in local round r_v, then:*
  - *If $\exists\, u \in \mathcal{N}_2^a(v, g_v(r_v))$ invoking BEEP2H in $g_v(r_v)$, v sets $\mathcal{H}_v^2[r_v] := 1$ at the latest in round $r_v + \delta$.*
  - *If $\nexists\, u \in \mathcal{N}_2^a(v, g_v(r_v))$ invoking BEEP2H in $g_v(r_v)$, v never sets $\mathcal{H}_v^2[r_v]$ to 1.*

Implementing communication with nodes at distance 2 (on the square of the communication graph) raises several issues. Communicating to a non-neighboring node (within distance 2) requires coordinating with an awake neighboring node relaying the communication. Moreover, nodes must be careful to relay communication up to distance 2 and no further. Algorithms 1 and 2 deal with these issues. In what follows, Section 4.1 presents the high-level intuition behind the proposed algorithms. Then, a detailed description is given in Section 4.2 and the analysis in Section 4.3.

4.1 High-level Description of the BEEP2H and LISTEN2H Algorithms

The BEEP2H and LISTEN2H algorithms rely on some USI(2,k)-code $C = \{c_1, c_2\}$, for some well-chosen integer k. All nodes should know the *same two codewords* c_1 and c_2 (of length l). By encoding the distance from any node executing BEEP2H using c_1 (distance 0: source) and c_2 (distance 1: relay), nodes communicate over distance 2 with a delay of $\delta = 2l$ rounds. The *source* s of a 2-hop beep transmits c_1. Due to the k-cover-free property of C, collisions between (at most k) uncoordinated transmissions of codewords do not affect the decoding. Thus, neighbors of node s decode (by simply checking whether c_1 is contained in some history factor $\mathcal{H}_v[r, r+l-1]$, defined in Section 2.) the c_1 transmission,

which they *relay* by transmitting c_2. When a node decodes a c_1 or c_2 transmission, it learns that it is at most 2 hops away from a 2-hop beep's source.

4.2 Algorithms for the 2-hop Communication Primitives

Recall that Δ denotes the maximum degree of the communication graph and that an upper bound $\Delta_{up} = O(\Delta)$ is known by all nodes. Moreover, we assume the common knowledge of some integer $f \geq 1$. Thus, each node can compute the same USI(2,k)-code $C = \{c_1, c_2\}$ of length $l = O(\Delta_{up}^4)$ with $k \geq f(\Delta_{up}+1)^2$.

Algorithms implementing BEEP2H and LISTEN2H are given below (Algorithms 1 and 2). Both rely on Algorithm 3, which manages the transmission of codewords c_1, c_2 of the given USI(2,k)-code and their decoding. Upon wake-up, a node v sets $\mathcal{H}_v^2$ and $\mathcal{B}_v^2$ to 0^ω—the infinite binary word composed only of 0's. After which, v invokes either BEEP2H or LISTEN2H in each local round $r_v \geq 1$, but cannot invoke BEEP2H more than f times within $2l$ rounds.

- On the one hand, v 2-hop beeps in some round r_v by invoking BEEP2H in r_v. It is said that v starts a 2-hop beep in (global) round $g_v(r_v)$ and v conveys this information to its neighbors by transmitting the codeword c_1 (beeping according to the ones in the binary word c_1).
- On the other hand, v listens to 2-hop beeps (and relays them if necessary) in the following manner. In each local round r_v, if the history factor $\mathcal{H}_v[r_v - l, r_v - 1]$ of v contains c_1 (resp. contains c_2 and $r_v \geq 2l+1$), v knows that a neighboring node started a 2-hop beep in $r_v - l$ (resp. for c_2, knows some 2-hop reachable node started a 2-hop beep in $r_v - 2l$) and sets $\mathcal{H}_v^2[r_v - l]$ to 1 (resp. for c_2, sets $\mathcal{H}_v^2[r_v - 2l]$ to 1). Furthermore, v transmits another codeword c_2 (relaying the information that one of its neighbors transmitted c_1 previously) to relay the 2-hop beep (resp. transmits nothing).

Each invocation of BEEP2H is stored in the 2-hop beep history variable $\mathcal{B}^2$—the beep history of the (simulated) communication on the square communication graph. More precisely, for some node v, assume v invokes BEEP2H in some local round r_v. Then v sets $\mathcal{B}_v^2[r_v]$ to 1 at the end of round r_v. This also allows v to know how it should transmit c_1 in the $l-1$ following rounds.

In the analysis of Algorithms 1, 2 and 3 (in Section 4.3), we prove that the 2-hop histories variables computed by all nodes are coherent within a delay of $2l$ rounds (see Theorem 12). For that, it is important to impose some (frequency) constraint on 2-hop beeps—that is, on the $\mathcal{B}^2$ variables—which is described in the following remark. Lemma 9 directly follows from Remark 1 and the fact that a node v transmits c_1 in round r_v if and only if v starts a 2-hop beep in r_v.

Remark 1. For any given node v and local round r_v: $|\{r'_v \in \{r_v, \ldots, r_v + 2l - 1\} \mid \mathcal{B}_v^2[r'_v] = 1\}| \leq f$.

Lemma 9. *For any global round r and for any node v awake in r, there are at most $f(\Delta_{up}+1)$ (respectively, $f(\Delta_{up}^2+1)$) 2-hop beeps, or equivalently c_1 transmissions, started in node v's neighborhood (resp., in node v's 2-hop neighborhood) within $l-1$ rounds of r.*

Algorithm 1. BEEP2H

1: **IN:** $\{c_1, c_2\}$: USI(2,k)-code of length l, f: maximum frequency, r: current (local) round
2: **INOUT:** $\mathcal{B}^2$: infinite binary word, $\mathcal{H}^2$: infinite binary word
3: **if** $|\{r' \in \{r - 2l + 1, \ldots, r - 1\} \mid \mathcal{B}^2[r'] = 1\}| < f$ **then**
4: $\quad \mathcal{B}^2[r] := 1, \mathcal{H}^2[r] := 1$ ▷ At most f BEEP2H invocations allowed within $2l$ rounds for a node.
5: CodewordTransmission($\{c_1, c_2\}$, r, $\mathcal{B}^2$, $\mathcal{H}^2$) ▷ Transfer updated 2-hop beep history $\mathcal{B}^2$ to Algorithm 3

Algorithm 2. LISTEN2H

IN: $\{c_1, c_2\}$: USI(2,k)-code of length l, f: maximum frequency, r: current (local) round
INOUT: $\mathcal{B}^2$: infinite binary word, $\mathcal{H}^2$: infinite binary word
CodewordTransmission($\{c_1, c_2\}$, r, $\mathcal{B}^2$, $\mathcal{H}^2$) ▷ Algorithm 3

Algorithm 3. CodewordTransmission

IN: $\{c_1, c_2\}$: USI(2,k)-code of length l, r: current (local) round
INOUT: $\mathcal{B}^2$: infinite binary word, $\mathcal{H}^2$: infinite binary word ▷ With the frequency constraint on $\mathcal{B}^2$.
// *Transmit a bit of codeword c_1, according to 2-hop beep history $\mathcal{B}^2$.*
for integer $j := 1$; $j \leq l$; j++ **do** ▷ At most $f \cdot c_1$ transmissions by the executing node.
  if $\mathcal{B}^2[r + 1 - j] = 1$ and $c_1[j] = 1$ **then** BEEP
for integer $i := 1$; $i \leq l$; i++ **do** ▷ Listen to beeps to detect bits of c_1.
  // *If c_1 is decoded (starting in $r - l - (i - 1)$), v relays information by transmitting c_2.*
  if the history factor $\mathcal{H}[r - l - (i - 1), r - 1 - (i - 1)]$ contains c_1 **then**
    if $c_2[i] = 1$ **then** BEEP
// *If c_1 or c_2 were decoded in the history factor $\mathcal{H}[r - l, r - 1]$, then update $\mathcal{H}^2$.*
if the history factor $\mathcal{H}[r - l, r - 1]$ contains c_1 **then** $\mathcal{H}^2[r - l] := 1$
else if the history factor $\mathcal{H}[r - l, r - 1]$ contains c_2 and $r \geq 2l + 1$ **then** $\mathcal{H}^2[r - 2l] := 1$

4.3 Analysis of the 2-hop Communication Primitives

The analysis of Algorithms 1, 2 and 3 is given below. The main result—Theorem 12—states that BEEP2H and LISTEN2H (as defined by Algorithms 1 and 2) satisfy Definition 5 given at the beginning of Section 4 (with a delay δ of $2l = O(\Delta_{up}^4)$), albeit with a constraint on the frequencies of BEEP2H invocations: a single node cannot invoke BEEP2H more than f times within $2l$ rounds when using Algorithms 1 and 2.

In Algorithms 1, 2 and 3, nodes use codewords to communicate the distance to the source of a 2-hop beep. This codeword-based communication can be reduced to an uncoordinated transmission scenario: nodes transmit codewords to send, listen to and relay 2-hop beeps in an uncoordinated manner.
First, we prove that in every round r and for any node v awake in r, there are at most $f(\Delta_{up}+1)^2$ codeword transmissions started in node v's neighborhood within $l - 1$ rounds of r (see Theorem 10). Thus the k-cover-free property (of the USI(2,k)-code with $k \geq f(\Delta_{up} + 1)^2$) guarantees that nodes correctly decode transmissions and their starting rounds, by simply checking whether codewords are contained in history factors of length l (see Lemma 11). Consequently, the 2-hop histories variables of all nodes, computed by Algorithm 3, are coherent within a delay of $2l$ rounds (Theorem 12).

Theorem 10. *For any global round r and for any node v awake in r, there are at most $f(\Delta_{up}+1)^2$ codeword transmissions started in node v's neighborhood within $l - 1$ rounds of r.*

Proof. We prove Theorem 10 by strong induction. First, consider the base case: $r = 1$. By the definition of Algorithm 3, no node starts transmitting c_2 in rounds 1 to l. Since by Lemma 9, there are at most $f(\Delta_{up} + 1)$ c_1 transmissions started in any given node v's neighborhood within rounds $\{1, \ldots, l\}$, then the induction hypothesis holds for $r = 1$.

Now, consider that the induction hypothesis holds for all rounds $r' \leq r$. Let v be any given node awake in round $r + 1$. On the one hand, by Lemma 9 there are at most $f(\Delta_{up} + 1)$ c_1 transmissions started in v's neighborhood within $l - 1$ rounds of $r + 1$. On the other hand, we upper bound, below, the number of c_2 transmissions started in v's neighborhood within $l - 1$ rounds of $r + 1$ by $f(\Delta_{up}^2 + 1)$. Combining both, there are at most $f(\Delta_{up}^2 + \Delta_{up} + 2) \leq f(\Delta_{up} + 1)^2$ codeword transmissions started in v's neighborhood within $l - 1$ rounds of $r + 1$.

To upper bound the number of c_2 transmissions started within $l-1$ rounds of $r+1$ (i.e., in $\{r-l+2,\ldots,r+l\}$), we first show that any given node u transmits c_2 starting in some round r'' (such that $r'' \leq r+l$) if and only if node u or its neighbors transmitted c_1 starting in round $r''-l$. By definition of Algorithm 3, any given node u transmits c_2 starting in any given round r'' if and only if c_1 is contained in its history factor $\mathcal{H}_u[g_u^{-1}(r''-l), g_u^{-1}(r''-1)]$. Additionally, since $r''-l \leq r$, then Lemma 11 (which relies on the induction hypothesis) applies in round $r''-l$ and c_1 is contained in the history factor $\mathcal{H}_u[g_u^{-1}(r''-l), g_u^{-1}(r''-1)]$ of any given node u if and only if node u or one of its neighbors transmitted c_1 starting in round $r''-l$.

Lemma 11. *For any global round $r' \leq r$ and for any node u awake in r', the history factor $\mathcal{H}_u[g_u^{-1}(r'), g_u^{-1}(r')+l-1]$ (of u) contains a codeword $c' \in C$ if and only if node u or one of its neighbors transmitted c' starting in round r'.*

Proof. The induction hypothesis holds for all rounds $r' \leq r$. Then, for any such round r' and for any node u awake in r', there are at most $f(\Delta_{up}+1)^2$ codeword transmissions started in node u's neighborhood within $l-1$ rounds of r'. For node u, this corresponds to an uncoordinated transmission scenario with conflict $\chi \leq f(\Delta_{up}+1)^2$. Since C satisfies the k-cover-free property for $k \geq f(\Delta_{up}+1)^2$, by Theorem 2, the lemma follows. □

Now, by Lemma 9 there are at most $f(\Delta_{up}^2+1)$ c_1 transmissions started in v's 2-hop neighborhood within $l-1$ rounds of $r+1-l$. Then, there are at most $f(\Delta_{up}^2+1)$ c_2 transmissions started in v's neighborhood within $l-1$ rounds of $r+1$. Notice that when multiple nodes hear the same c_1 transmission, the resulting (relay) c_2 transmissions are all started in the same round and are thus considered as a single c_2 transmission. □

Theorem 12. *For any given node v in some local round $r_v \geq 2l+1$ (accounting for the $\delta = 2l$ delay), the permanent factors of the computed 2-hop history variables, $\mathcal{H}_v^2[1, r_v-2l]$ and $\mathcal{B}_v^2[1, r_v-2l]$, are coherent. In other words, for any local round $r'_v \in \{1,\ldots,r_v-2l\}$, $\mathcal{H}_v^2[r'_v]=1 \Leftrightarrow$ ($\mathcal{B}_v^2[r'_v]=1$ or $\exists u \in \mathcal{N}_2^a(v, g_v(r'_v))$ s.t. $\mathcal{B}_u^2[g_u^{-1}(g_v(r'_v))]=1$).*

Proof. From the definition of Algorithms 1 and 3, $\mathcal{H}_v^2[r'_v]=1$ if and only if $\mathcal{B}_v^2[r'_v]=1$ (line 4 of Algorithm 1) or the history factor $\mathcal{H}_v[r'_v, r'_v+l-1]$ contains c_1 (line 11 of Algorithm 3) or the history factor $\mathcal{H}_v[r'_v+l, r'_v+2l-1]$ contains c_2 (line 12 of Algorithm 3).

By Lemma 11 (and Theorem 10) the history factor $\mathcal{H}_v[r'_v, r'_v+l-1]$ contains c_1 if and only if some node $u \in \mathcal{N}(v)$ (i.e., v or some neighboring node of v) transmits c_1 starting in $g_v(r'_v)$ (i.e., starts a 2-hop beep in $g_v(r'_v)$), that is if and only if $\exists u \in \mathcal{N}(v)$ such that $\mathcal{B}_u^2[g_u^{-1}(g_v(r'_v))]=1$.

Similarly, by Lemma 11 the history factor $\mathcal{H}_v[r'_v+l, r'_v+2l-1]$ contains c_2 if and only if some node $w \in \mathcal{N}(v)$ transmits c_2 starting in $g_v(r'_v+l)$. Let $r''=g_v(r'_v)$. By definition of Algorithm 3, w transmits c_2 starting in $r''+l$ if and only if c_1 is contained in its history factor $\mathcal{H}_w[g_w^{-1}(r''), g_w^{-1}(r''+l-1)]$. By Lemma 11 the history factor $\mathcal{H}_w[g_w^{-1}(r''), g_w^{-1}(r''+l-1)]$ of w contains c_1 if and only if some node $u \in \mathcal{N}(w)$ transmits c_1 starting in round r'' (i.e., starts a 2-hop beep in r''), that is if and only if $\exists u \in \mathcal{N}_2^a(v, r'')$ such that $\mathcal{B}_u^2[g_u^{-1}(r'')]=1$.

One now gets that $\mathcal{H}_v^2[r'_v]=1 \Leftrightarrow$ ($\mathcal{B}_v^2[r'_v]=1$ or $\exists u \in \mathcal{N}_2^a(v, g_v(r'_v))$ s.t. $\mathcal{B}_u^2[g_u^{-1}(g_v(r'_v))]=1$). □

Remark 2. For any global round r, if no node in the reachable 2-hop neighborhood of some node v 2-hop beeps in r, $\mathcal{H}_v^2[g_v^{-1}(r)]$ is never set to 1. That is also the case if some unreachable node $z \in \mathcal{N}_2(v)$ 2-hop beeps in r (i.e., $z \in \mathcal{N}_2(v) \setminus \mathcal{N}_2^a(v, r)$).

5 SOLVING 2-HOP DESYNCHRONIZATION

Solving the 1-hop desynchronization problem [6] allows nodes to avoid sender-side collisions but not receiver-side collisions. Two neighbors of a node, at distance 2 of each other, may compute non-disjoint communication schemes

with a correct 1-hop desynchronization solution. In contrast, in the 2-hop desynchronization problem, nodes compute (periodic) communication schemes that must be disjoint from those of all nodes in their 2-hop neighborhood. This allows nodes to communicate in a predictable manner while avoiding both sender and receiver-side collisions. However, to compute a 2-hop desynchronization scheme, a node must communicate with other nodes at distance 2. That is the reason why the proposed 2-hop desynchronization solution is implemented by using the 2-hop communication primitives presented in Section 4.

Phases. In the following solutions, nodes consider consecutive sequences of T (local) rounds, starting from their wake-up round, called *phases*. The local phase number is denoted by $p \geq 1$ and the p^{th} local phase by $\mathcal{P}(p) = \{(p-1)\cdot T+1, \ldots, p\cdot T-1\}$. If $i \in \{1, \ldots, T\}$ is a round number inside phase p, the local round number corresponding to i is $\mathcal{P}(p, i)$. Notice that the local phases of different nodes may be different with respect to global rounds. For that reason, we give the following definitions. For any two nodes u and v (and respectively, phase numbers p_u and p_v), the phases $\mathcal{P}_u(p_u)$ and $\mathcal{P}_v(p_v)$ are said to *intersect* if $\{g_u(\mathcal{P}_u(p_u, 1)), \ldots, g_u(\mathcal{P}_u(p_u, T))\} \cap \{g_v(\mathcal{P}_v(p_v, 1)), \ldots, g_v(\mathcal{P}_v(p_v, T))\} \neq \emptyset$, or in other words, if they intersect in the global round structure. Furthermore, for any two nodes u, v, we denote the set of phases of u that intersect with any phase $\mathcal{P}_v(p_v)$ of v, by $\mathcal{I}(\mathcal{P}_v(p_v), u)$. This set has cardinality at most 2.

High-level Description of the 2-hop Desynchronization Solution. By leveraging our 2-hop communication primitives, we extend a simple 1-hop desynchronization solution to the 2-hop desynchronization problem. More precisely, we adapt a streamlined version of the 1-hop desynchronization solution presented in [6], but with period length $T = O(\Delta_{up}^4)$, to suit the 2-hop communication primitives and in particular, the resulting communication delay.

Briefly, the overall solution works as follows. During each phase, a node first computes a (finite) communication scheme $\mathcal{S} = 0^t\ 1\ 0^{T-1-t}$ (for some integer $t \in \{1, \ldots, T-1\}$) of length T. Then, the node beeps according to a jittered version—$\mathcal{S}'$—of $\mathcal{S}$, in which it jitters (i.e., delays its beep) by 1 round with probability $\frac{1}{2}$. This allows the node to detect sender-side collisions with probability $\frac{1}{2}$.
When computing $\mathcal{S}$, a node first computes with a *true-biased one-sided error* of $\frac{1}{2}$ (i.e., always computes `true` correctly but computes `false` correctly with probability at least $\frac{1}{2}$) whether its communication scheme during the previous phase was 2-disjoint[6] with the communication schemes $\mathcal{S}_u$ of all 2-hop neighbors. If so, then it keeps the communication scheme $\mathcal{S}$ from the previous phase (but computes a new jittered version $\mathcal{S}'$). Otherwise, it computes a new communication scheme $\mathcal{S}$, which is 2-disjoint with all those of its 2-hop neighbors (possibly also recomputed) with some constant probability. Eventually, a node no longer changes its finite communication scheme $\mathcal{S}$. As a result, the node computes the infinite periodic (of period $\mathcal{S}$) communication scheme $\mathcal{S}^\omega$, disjoint from those of its neighbors.

5.1 Description of the 2-hop Desynchronization Solution

It is assumed that nodes are given the same period length $T = \kappa(\Delta_{up} + 2)^4$, where κ is set to 86. Additionally, nodes compute the same USI(2,$(\Delta_{up}+1)^2$)-code C of length $l \leq 32(\Delta_{up}+2)^4$ ($l = O(\Delta_{up}^4)$)—which satisfies the conditions in Section 4.2 with $f = 1$. Next, we provide a precise description of the proposed 2-hop desynchronization solution (the algorithm is deferred to Appendix A.1) using the 2-hop communication primitives from Section 4.

Nodes decide, for each phase, on some positive integer $i^E \in A$ (where $A = \{2l+1, \ldots, T-2\}$ is the set of allowed indexes within the phase) and on some jitter bit J (chosen uniformly at random in $\{0, 1\}$)—see details below. Then, nodes compute, for each phase, a (finite) length T communication scheme $\mathcal{S} = 0^{i^E-1}\ 1\ 0^{T-i^E}$ and a corresponding

[6] A more precise statement is given in Section 5.1 using Definition 6.

jittered communication scheme $\mathcal{S}' = 0^{i^E+J-1}\ 1\ 0^{T-i^E-J}$ (except for the first phase, in which $\mathcal{S} = \mathcal{S}' = 0^T$). Importantly, during each phase a node 2-hop beeps—instead of beeping—according to $\mathcal{S}'$. In other words, a node (2-hop) listens for the first $i^E + J - 1 \geq 2l$ rounds of a phase, 2-hop beeps once and then (2-hop) listens again for the remaining rounds of the phase.

To account for the 2-hop communication primitives' delay and ensure nodes have a coherent 2-hop history variable $\mathcal{H}^2$ for the previous phase, all computations are done at the end of round $2l$ of the phase. Before that, no computation is necessary since the scheme $\mathcal{S}'$ always start with at least $2l$ 0's. Since any variable *var* changes at most once every phase per node, then by abusing the notation, the value of *var* computed at the end of round $2l$ of some phase $\mathcal{P}(p)$ is said to be the value of *var* in $\mathcal{P}(p)$ and is denoted by $var(p)$. Finally, for any node v in phase $\mathcal{P}_v(p_v)$ (for $p_v \geq 1$), we denote the local round $\mathcal{P}_v(p_v, i^E(p_v))$ by $\mathcal{L}_v(p_v)$ and the global round $g_v(\mathcal{P}_v(p_v, i^E(p_v)))$ by $\mathcal{G}_v(p_v)$ for simplification purposes (these rounds correspond to the 1 bit in $\mathcal{S}_v(p_v)$).

Computing i^E. Finally, we describe the computation of $i_v^E(p_v)$ by some node v in phase $\mathcal{P}_v(p_v)$ (for $p_v \geq 2$). First, node v computes the set of *occupied indexes* $O_v(p_v) = \{i^O \in \{1, \ldots, T\} |\ \mathcal{H}_v^2[\mathcal{P}_v(p_v - 1, i^O)] = 1\}$: indexes in which v heard 2-hop beeps during the previous phase. Then, v computes with a true-biased one-sided error of $\frac{1}{2}$ whether it was *good* (see Definition 6 below) in the previous phase $\mathcal{P}_v(p_v - 1)$ and stores the result in the variable $prevGood_v(p_v)$. More precisely, $prevGood_v(p_v) = ((\{i^E - 1, \ldots, i^E + 2\} \setminus \{i^E + J\}) \cap O = \emptyset)$.[7]
If $prevGood_v(p_v) = \texttt{true}$, then $i_v^E(p_v) = i_v^E(p_v - 1)$. Otherwise, v computes the set of *free indexes* $F_v(p_v) = \{i^F \in A \mid \forall i^O \in O_v(p_v), |i^F - i^O| > 2\}$, consisting of all indexes within A that are at least three rounds away from indexes in $O_v(p_v)$. Then, v chooses an index $i_v^E(p_v) \in F_v(p_v)$ uniformly at random. By doing so, it is good in phase $\mathcal{P}_v(p_v)$ with some constant probability (Lemma 19, deferred to Appendix A.2).

Definition 6. *For any node v and any phase $\mathcal{P}_v(p_v)$, v is said to be a* good node *in $\mathcal{P}_v(p_v)$ if for any node $u \in \mathcal{N}_2(v)$ in some (intersecting) phase $\mathcal{P}_u(p_u) \in \mathcal{I}(\mathcal{P}_v(p_v), u)$, $\mathcal{S}_v(p_v)$ and $\mathcal{S}_u(p_u)$ are 2-disjoint*[8]*, or equivalently, $|\mathcal{G}_v(p_v) - \mathcal{G}_u(p_u)| > 2$. Node v is* bad *in $\mathcal{P}_v(p_v)$ if it is not good.*

Importantly, notice that if a node v is good in some phase $\mathcal{P}_v(p_v)$ then its communication scheme $\mathcal{S}_v(p_v)$ is disjoint with all communication schemes $\mathcal{S}_u(p_u)$ of a 2-hop neighbor u in some intersecting phase $\mathcal{P}_u(p_u) \in \mathcal{I}(\mathcal{P}_v(p_v), u)$.[9] Furthermore, if v is good in some phase $\mathcal{P}_v(p_v)$ for which all of its neighbors are awake, then it remains good in all subsequent phases (see Lemma 13 below, its proof is deferred to Appendix A.2). Therefore, once a node v is good in some phase $\mathcal{P}_v(p_v)$ for which all of its neighbors are awake, it computes the 2-hop desynchronized communication scheme $\mathcal{S}_v(p_v)^\omega$. In the following section, we show that a node becomes good w.h.p. in $O(\log n)$ phases after all of its neighbors are awake (see Theorem 14).

Lemma 13. *Once a node v is good in some phase $\mathcal{P}_v(p_v)$ and all of its neighbors have woken up, it remains good in all following phases.*

5.2 Analysis

The proof of correctness (and of the upper bound on the time complexity) of the proposed solution is sketched below.

[7] The property of the computed value $prevGood(p_v)$ is proved in Lemma 17, which is deferred to Appendix A.2.
[8] For $\mathcal{S}_v(p_v)$ starting in $\mathcal{P}_v(p_v, 1)$ and $\mathcal{S}_u(p_u)$ starting in $\mathcal{P}_u(p_u, 1)$.
[9] See footnote 8.

Theorem 14. *$O(\Delta_{up}^4 \log n)$ rounds after all nodes wake up, all nodes compute a 2-hop desynchronized communication scheme w.h.p. Alternatively, for any given node v, $O(\Delta_{up}^4 \log n)$ rounds after all of its neighbors are awake, v computes a 2-hop desynchronized communication scheme.*

Notice that in the 2-hop desynchronization problem, any given node v can only compute a 2-hop desynchronized communication scheme after all of its neighbors have woken up. Indeed, before that, there might be an (as of yet) unreachable awake node u in $\mathcal{N}_2(v)$ (v is also awake). As u and v cannot yet communicate using 2-hop beeps (see Remark 2), neither u nor v can compute a 2-hop desynchronized scheme (if they decide on non-disjoint schemes S_u and S_v, these schemes will change when they become 2-hop connected). In contrast, with the 1-hop desynchronization solution of [6], each node computes a (1-hop) desynchronized communication scheme $O(\log n)$ phases after it wakes up.
Consequently, our results guarantee upper bounds on the time complexity—of computing a 2-hop desynchronized communication scheme—starting from the first round in which a node's whole neighborhood is awake. However, after that, a node only requires $O(\log n)$ phases (w.h.p.) to compute a 2-hop desynchronized communication scheme.

Proof of Theorem 14. The following results concern nodes whose neighbors are all awake (i.e., all of a node's 2-hop neighbors are in its reachable 2-hop neighborhood). By Lemma 13, once such a node is good then it remains good forever. On the other hand, if such a node is bad then it becomes good in the next phase with constant probability (Lemma 15) and thus good w.h.p. after $O(\log n)$ phases (Theorem 16). The proof of Lemma 15 is deferred to Appendix A.2.

Lemma 15. *A bad node v in phase $\mathcal{P}_v(p_v)$, after all of its neighbors have woken up, becomes good in phase $\mathcal{P}(p_v + 1)$ with constant probability.*

Theorem 16. *Once all of its neighbors have woken up, a bad node becomes good after $O(\log n)$ phases w.h.p.*

When a node becomes good, it computes an (infinite periodic) communication scheme $\mathcal{S}^\omega$—since its finite communication scheme $\mathcal{S}$ no longer changes—which is 2-hop desynchronized. Therefore, $O(\log n)$ phases after all nodes wake up, all nodes compute a 2-hop desynchronized communication scheme w.h.p. □

6 LOCAL MESSAGE BROADCAST

Solving the 2-hop desynchronization problem allows nodes to break symmetry despite their uncoordinated starts. If nodes communicate according to their computed 2-hop desynchronized communication schemes, they avoid both *sender-side* and *receiver-side* collisions. Consequently, nodes can simulate message-passing (i.e., *local message broadcast*) by sending (broadcasting) their messages bit by bit. More precisely, any given node v can send a B_v-bit message (for any individually chosen constant $B_v > 0$) using $O(\Delta_{up}^4 \cdot B_v)$ rounds (without collisions nor information loss, once the 2-hop desynchronized schemes have been computed).

The implementation of the corresponding SEND and RECEIVE primitives can be explained as follows. The two primitives implement local message broadcast (for messages of any size), $O(\Delta_{up}^4 \log n)$ rounds after all nodes wake up. They are built on top of the algorithm presented in Section 5 (Algorithm 4 in the appendix). Here, we assume that instead of a USI(2,k)-code, nodes communicate using codewords from a USI(6,k)-code $\{c_1, c_2, c_3, c_4, c_5, c_6\}$ (where $k \geq 2(\Delta_{up} + 1)^2$). The first two codewords c_1 and c_2 are reserved for the 2-hop communication primitives BEEP2H and LISTEN2H. The other four—c_3, c_4, c_5 and c_6—are used to transmit information by the SEND and RECEIVE primitives.

SEND. Consider some node v that invokes SEND(m) for some message m (i.e., a binary word). To convey m, v communicates one bit of the message m per phase $\mathcal{P}_v(p_v)$ (using overall $|m|$ consecutive phases), according to the period $\mathcal{S}_v(p_v)$ of the 2-hop desynchronized communication scheme computed by v. More precisely:

- Node v sends a 0 bit (respectively, a 1 bit) in some phase p_v by transmitting the codeword c_3 (resp., c_4) starting in round $\mathcal{L}_v(p_v)$.
- Node v starts (respectively, ends) a message in some phase p_v by transmitting the codeword c_5 (resp, c_6) starting in round $\mathcal{L}_v(p_v)$.

Notice that node v also transmits c_1 (i.e., starts 2-hop beeps) starting in round $\mathcal{L}_v(p_v)$ (or $\mathcal{L}_v(p_v)+1$) in Algorithm 4. Consequently, awake nodes in v's neighborhood also transmit c_2 (i.e., relay 2-hop beeps) starting l rounds later. The cover-free property of a USI(6,k)-code (with $k \geq 2(\Delta_{up}+1)^2$, instead of $k \geq (\Delta_{up}+1)^2$ as in Section 5) allows—and is necessary—to decode all codeword transmissions, both in the underlying 2-hop communication primitives BEEP2H and LISTEN2H, as well as in SEND and RECEIVE.

RECEIVE. Node v listens in all phases for any bit transmitted by neighboring nodes. More precisely, it decodes any transmissions of c_3, c_4, c_5 or c_6. Upon decoding a c_5 transmission, v concatenates the bits decoded from the codewords transmitted each T rounds (c_3 or c_4), until it decodes a c_6 transmission. Any other sequence of decoded transmissions is considered as a faulty message. Once v and all of its neighbors are good, any such concatenation of bits, m^*, is a message sent by some neighboring node. In particular, the bits of m^* were transmitted by the same node.

Notice that it is important, for the above property, that SEND and RECEIVE are built upon 2-hop desynchronization. Indeed, when some neighbor u of v transmits c_3, c_4, c_5 or c_6 (i.e., a bit), it does so according to its scheme $\mathcal{S}_u$. Once u is good, $\mathcal{S}_u$ is the period of a 2-hop desynchronized communication scheme $\mathcal{S}_u^{\omega}$, which is disjoint with those of all other nodes within u's 2-hop neighborhood. Thus, once v and u are good, then for v, u is the only neighboring node to transmit a codeword in $\{c_3, c_4, c_5, c_6\}$ according to $\mathcal{S}_u^{\omega}$. This ensures that, once v and all of its neighbors are good, the bits transmitted each T rounds—decoded by v—were all transmitted by a single node. Additionally, a good node transmits bits exactly every T rounds. Thus the concatenation of these bits is the message of some neighboring node.

REFERENCES

[1] Y. Afek, N. Alon, Z. Bar-Joseph, A. Cornejo, B. Haeupler, and F. Kuhn. 2013. Beeping a Maximal Independent Set. *Distributed Computing* 26, 4 (2013), 195–208.

[2] J. Beauquier, J. Burman, F. Dufoulon, and S. Kutten. 2018. Fast Beeping Protocols for Deterministic MIS and (Δ+1)-Coloring in Sparse Graphs. In *Proceedings of the 37th IEEE Conference on Computer Communications (INFOCOM 2018)*. 1754–1762.

[3] B. S. Chlebus, L. Gasieniec, A. Gibbons, A. Pelc, and W. Rytter. 2002. Deterministic broadcasting in ad hoc radio networks. *Distributed Computing* 15, 1 (2002), 27–38.

[4] B. S. Chlebus, L. Gąsieniec, D. R. Kowalski, and T. Radzik. 2005. On the Wake-Up Problem in Radio Networks. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP 2005)*. 347–359.

[5] M. Chrobak, L. Gąsieniec, and D. Kowalski. 2007. The Wake-Up Problem in Multi-Hop Radio Networks. *SIAM J. Comput.* 36, 5 (2007), 1453–1471.

[6] A. Cornejo and F. Kuhn. 2010. Deploying Wireless Networks with Beeps. In *Proceedings of the 24th International Symposium on Distributed Computing (DISC 2010)*. 148–162.

[7] P. Erdös and P. Turán. 1941. On a Problem of Sidon in Additive Number Theory, and on some Related Problems. *Journal of the London Mathematical Society* s1-16, 4 (1941), 212–215.

[8] M. Ghaffari and B. Haeupler. 2013. Near Optimal Leader Election in Multi-hop Radio Networks. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2013)*. 748–766.

[9] S. Gilbert and C. Newport. 2015. The Computational Power of Beeps. In *Proceedings of the 29th International Symposium on Distributed Computing (DISC 2015)*. 31–46.

[10] R. Guerraoui and A. Maurer. 2015. Byzantine Fireflies. In *Proceedings of the 29th International Symposium on Distributed Computing (DISC 2015)*. 47–59.

[11] W. H. Kautz and R. C. Singleton. 1964. Nonrandom Binary Superimposed Codes. *IEEE Transaction on Information Theory* 10, 4 (1964), 363–377.

A THE 2-HOP DESYNCHRONIZATION ALGORITHM AND THE REMAINING PROOFS OF ITS ANALYSIS

A.1 2-hop Desynchronization Algorithm

The 2-hop desynchronization solution is described in Section 5.1. Below, we give the corresponding algorithm—Algorithm 4. The discrete uniform distribution on a set S is denoted by $\mathcal{U}(S)$.

Algorithm 4. 2-hop Desynchronization

IN: κ: integer, Δ_{up}: upper bound on the maximum degree, C: USI-code of length l
$prevGood :=$ `false`, $T := \kappa(\Delta_{up}+2)^4$, $J := 0$, $i^E := 0$
$A := \{2l+1, \ldots, T-2\}$ ▷ The set of rounds in each phase in which v can beep.
$\mathcal{S} := 0^T$, $\mathcal{S}' := 0^T$ ▷ Initialize the phases' communication schemes.

// $\mathcal{H}^2$ and $\mathcal{B}^2$ are the 2-hop histories variables of communication in the square graph.
$\mathcal{H}^2 := 0^\omega$, $\mathcal{B}^2 := 0^\omega$ ▷ Infinite binary words (at initialization, composed only of 0's)
for local round $r := 1$; r++ **do**
  $p := 1 + r/T$ ▷ Current phase number (computed by integer division), for v.
  $i := (r-1)\ \%\ T + 1$ ▷ Round index in the current T-round phase.

  // Communicate on the square graph (according to $\mathcal{S}'$), with a $2l$ delay.
  if $p = 1$ or $i \leq 2l$ **then**
    // Listen in the first phase, and for the first $2l$ rounds (accounting for the 2-hop beeps' delay) of every phase.
    LISTEN2H(C,1,r,$\mathcal{B}^2$,$\mathcal{H}^2$)
  else if $i = i^E + J$ **then** BEEP2H(C,1,r,$\mathcal{B}^2$,$\mathcal{H}^2$)
  else LISTEN2H(C,1,r,$\mathcal{B}^2$,$\mathcal{H}^2$)

  // Local computation done once per phase $\mathcal{P}(p)$, for $p > 1$, at the end of the round $\mathcal{P}(p, 2l)$.
  // This ensures that all 2-hop beeps from the previous phase have been detected.
  if $p \geq 2$ and $i = 2l$ **then**
    // O contains the indexes of 2-hops beeps heard (these indexes are thus occupied*) during the previous phase.*
    $O := \{i^O \in \{1, \ldots, T\} \mid \mathcal{H}^2[\mathcal{P}(p-1, i^O)] = 1\}$ ▷ Where $\mathcal{P}(p-1, i^O) = r + i^O - i - T$.

    if $p > 2$ **then** ▷ i^E and J are correctly initialized now, after the second phase.
      // If the node was good in the previous phase (after all neighbors have woken up), then $prevGood =$ `true`.
      // If the node was bad, $prevGood =$ `false` *with probability at least $\frac{1}{2}$ (in part due to sender-side collisions).*
      $prevGood := (\{i^E - 1, \ldots, i^E + 2\} \setminus \{i^E + J\}) \cap O = \emptyset)$

      if not $prevGood$ **then** ▷ If the node detects that it was bad in the previous phase.
        $F := \{i^F \in A \mid \forall i^O \in O, |i^F - i^O| > 2\}$ ▷ All *free* indexes F are at least 3 rounds away from O.
        $i^E := \mathcal{U}(F)$, $\mathcal{S} := 0^{i^E - 1}\ 1\ 0^{T - i^E}$ ▷ Choose a new scheme.
      $J := \mathcal{U}(\{0,1\})$, $\mathcal{S}' := 0^{i^E + J - 1}\ 1\ 0^{T - i^E - J}$ ▷ Compute a new jitter bit and jittered communication scheme.

A.2 Analysis of the 2-hop Desynchronization Algorithm: Remaining Proofs

The analysis of Algorithm 4 is sketched in Section 5.2. We complete that analysis here. First we provide the auxiliary Lemmas 17, 18 and 19. Lemma 17 states that a node v, in some phase $\mathcal{P}_v(p_v)$ for which all of its neighbors have woken up, computes with a true-biased one-sided error (of $\frac{1}{2}$) whether it was good in the previous phase $\mathcal{P}_v(p_v-1)$ and stores the result in $prevGood(p_v)$—see line 28 of Algorithm 4. Lemma 18 states that at least a constant fraction of any given phase is composed of free indexes (see line 31 of Algorithm 4). Finally Lemma 19, building upon Lemma 18, states that a node v, in some phase $\mathcal{P}_v(p_v)$, that detects that it was bad in the previous phase, is good with some constant probability in phase $\mathcal{P}_v(p_v)$—see line 32 of Algorithm 4.

Lemma 17. *Consider a node v in some phase $\mathcal{P}_v(p_v)$ (with $p_v \geq 3$), such that all of its neighbors have woken up. If v was good in phase $\mathcal{P}_v(p_v-1)$ then $prevGood(p_v) = \texttt{true}$. Otherwise, if v was bad in phase $\mathcal{P}_v(p_v-1)$ then $prevGood(p_v) = \texttt{false}$ with probability at least $\frac{1}{2}$.*

Proof. First, assume that v was good in phase $\mathcal{P}_v(p_v-1)$. Then for any node $u \in \mathcal{N}_2(v)$ in some (intersecting) phase $\mathcal{P}_u(p_u) \in \mathcal{I}(\mathcal{P}_v(p_v), u)$, $|\mathcal{G}_v(p_v) - \mathcal{G}_u(p_u)| > 2$. Therefore, u does not 2-hop beep in rounds $\{i^E_v(p_v)-1, \ldots, i^E_v(p_v)+2\}$, even with the jitter. Consequently, v heard no 2-hop beeps in rounds $\{i^E_v(p_v)-1, \ldots, i^E_v(p_v)+2\} \setminus \{i^E_v(p_v)+J_v(p_v)\}$ of phase $\mathcal{P}_v(p_v-1)$ and $prevGood(p_v) = true$.

Now, assume that v was bad in phase $\mathcal{P}_v(p_v-1)$. Then there exists a node $u \in \mathcal{N}_2(v)$ in some (intersecting) phase $\mathcal{P}_u(p_u) \in \mathcal{I}(\mathcal{P}_v(p_v), u)$, $|\mathcal{G}_v(p_v) - \mathcal{G}_u(p_u)| \leq 2$. Let $r' = \mathcal{G}_u(p_u) + \mathcal{J}_u(p_u)$ be the global round in which u beeps during its phase $\mathcal{P}_u(p_u)$. Since all neighbors of v have woken up, then v hears the 2-hop beep of u in round r'—see Remark 2—if it does not itself 2-hop beep in r'. We consider the following three cases:

- Assume that $|\mathcal{G}_v(p_v) - \mathcal{G}_u(p_u)| = 2$. Then, due to the jitter bit, $r' \in \{\mathcal{G}_v(p_v)-2, \mathcal{G}_v(p_v)+3\}$ with probability $\frac{1}{2}$ and v incorrectly computes $prevGood(p_v) = \texttt{true}$, or $r' \in \{\mathcal{G}_v(p_v)-1, \mathcal{G}_v(p_v)+2\}$ with probability $\frac{1}{2}$ and v correctly computes $prevGood(p_v) = \texttt{false}$.
- Assume that $|\mathcal{G}_v(p_v) - \mathcal{G}_u(p_u)| = 1$. Then, $\mathcal{G}_v(p_v) + J_v(p_v) = \mathcal{G}_u(p_u) + J_u(p_u)$ with probability $\frac{1}{4}$ and v incorrectly computes $prevGood(p_v) = \texttt{true}$, or $\mathcal{G}_v(p_v) + J_v(p_v) \neq \mathcal{G}_u(p_u) + J_u(p_u)$ with probability $\frac{3}{4}$ and v correctly computes $prevGood(p_v) = \texttt{false}$.
- Assume that $\mathcal{G}_v(p_v) = \mathcal{G}_u(p_u)$. Then, $\mathcal{G}_v(p_v) + J_v(p_v) = \mathcal{G}_u(p_u) + J_u(p_u)$ with probability $\frac{1}{2}$ and v incorrectly computes $prevGood(p_v) = \texttt{true}$, or $\mathcal{G}_v(p_v) + J_v(p_v) \neq \mathcal{G}_u(p_u) + J_u(p_u)$ with probability $\frac{1}{2}$ and v correctly computes $prevGood(p_v) = \texttt{false}$. Notice that this case corresponds to sender-side collision detection.

Therefore, v correctly computes $prevGood(p_v) = \texttt{false}$ with probability at least $\frac{1}{2}$. □

Lemma 18. *If $\kappa \geq 76$, then for any given node v and any given phase $\mathcal{P}_v(p_v)$ (such that $p_v \geq 2$), $|F_v(p_v)| \geq (1-\frac{76}{\kappa})T$.*

Proof. Since indexes that are within 2 rounds of $O_v(p_v)$ are not in $F_v(p_v)$, $|F_v(p_v)| \geq T - (2l+2) - 5(\Delta^2+1)$. As $l \leq \frac{32}{\kappa}T$, $|F_v(p_v)| \geq T(1-\frac{64}{\kappa}) - 5\Delta^2 - 7$. Following which, $|F_v(p_v)| \geq T(1-\frac{64}{\kappa}-\frac{12}{\kappa})$ (as $T = \kappa \cdot O(\Delta^4)$).
Finally, $|F_v(p_v)| \geq (1-\frac{76}{\kappa})T$ (and $1-\frac{76}{\kappa} \geq 0$ for $\kappa \geq 76$). □

Lemma 19. *Consider a bad node v in some phase $\mathcal{P}_v(p_v)$ with $prevGood_v(p_v+1) = \texttt{false}$. Then v is good in phase $\mathcal{P}_v(p_v+1)$ with probability at least e^{-1}.*

Proof. First, we emphasize that the following analysis does not consider whether v hears the 2-hop beeps started in its 2-hop neighborhood. Instead, once v knows that it was bad in the previous period, it can become good with some

constant probability without coordinating with its 2-hop neighborhood. Notice that v might not hear these beeps if some of its neighbors have not woken up—see Remark 2.

Since $prevGood_v(p_v+1) = \texttt{false}$, $i_v^E(p_v+1)$ is chosen uniformly at random in $F_v(p_v+1)$. Then, node v becomes good in $\mathcal{P}_v(p_v+1)$ unless some non-empty subset of nodes $N \subset \mathcal{N}_2(v)$ satisfies $\forall u \in N$, $|\mathcal{G}_v(p_v+1) - \mathcal{G}_u(p_u)| \leq 2$ for some phase $\mathcal{P}_u(p_u)$. Since a node's consecutive $\mathcal{G}_u$ values are at least $2l+2$ rounds apart, then for each node $u \in N$ there is at most one phase $\mathcal{P}_u(p_u)$ satisfying the previous condition. Therefore, the probability Q_u that a node $u \in N$ interferes with v is at most $\frac{5}{|F_u(p_u)|}$ for a single phase $\mathcal{P}_u(p_u)$. By Lemma 18, this is at most $\frac{5}{(1-\frac{76}{\kappa})T}$. Then, the probability that v is good in phase $\mathcal{P}_v(p_v+1)$ is $Q = \prod_{u\in N}(1-Q_u) \geq \prod_{u\in N}(1-\frac{5}{(1-\frac{76}{\kappa})T}) \geq \exp\frac{-10|N|}{(1-\frac{76}{\kappa})\kappa(\Delta_u p+2)^4} \geq \exp\frac{-10}{(1-\frac{76}{\kappa})\kappa}$. The middle inequality holds for $\frac{5}{(1-\frac{76}{\kappa})\kappa} \leq \frac{1}{2}$, or equivalently, for $\kappa \geq 86$. The last inequality holds because $\frac{|N|}{(\Delta_u p+2)^4} \leq 1$. Finally, the probability that v is good in phase $\mathcal{P}_v(p_v+1)$ is $Q \geq \exp\frac{-10}{(\kappa-76)} \geq e^{-1}$, since $\kappa = 86$. □

Finally, using Lemmas 17, 18 and 19, we provide the proofs of Lemmas 13 and 15.

Proof of Lemma 13. We prove that once a node v is good in some phase $\mathcal{P}_v(p'_v)$ and all of its neighbors have woken up, it remains good in the following phase $\mathcal{P}_v(p'_v+1)$, for any $p'_v \geq p_v$. Then, Lemma 13 follows by simple induction.

Now, consider some node v, after all of its neighbors have woken up, which is good in some phase $\mathcal{P}_v(p'_v)$ such that $p'_v \geq p_v$. By Lemma 17, $prevGood_v(p'_v+1) = \texttt{true}$ and thus $\mathcal{S}_v(p'_v+1) = \mathcal{S}_v(p'_v)$. Furthermore, consider an arbitrary two-hop neighbor of v, $u \in N_2(v)$, in some phase $\mathcal{P}_u(p_u) \in \mathcal{I}(\mathcal{P}_v(p'_v+1), u)$. Then, $\mathcal{P}_u(p_u)$ starts after the start of $\mathcal{P}_v(p'_v)$ (i.e., after $g_v(\mathcal{P}(p'_v, 1))$). Thus, when u computes $\mathcal{S}_u(p_u)$, its 2-hop history during the previous phase is complete and contains the 2-hop beep transmitted by v during its p''^{th}_v phase. Consequently, by the definition of Algorithm 4 (line 31) u chooses a round index $i_u^E(p_u)$ such that $\mathcal{S}_v(p'_v)$, starting in $\mathcal{P}_v(p'_v+1, 1)$, and $\mathcal{S}_u(p_u)$, starting in $\mathcal{P}_u(p_u, 1)$, are 2-disjoint. Since $\mathcal{S}_v(p'_v+1) = \mathcal{S}_v(p'_v)$, then $\mathcal{S}_v(p'_v+1)$, starting in $\mathcal{P}_v(p'_v+1, 1)$, and $\mathcal{S}_u(p_u)$, starting in $\mathcal{P}_u(p_u, 1)$, are 2-disjoint. Consequently, v is good in phase $\mathcal{P}_v(p'_v+1)$. □

Proof of Lemma 15. By Lemma 17, a bad node in phase $\mathcal{P}_v(p_v)$—such that all of its neighbors have woken up—computes $prevGood(p_v+1) = false$ with probability at least $\frac{1}{2}$. Then, by Lemma 19 v becomes good in phase $\mathcal{P}_v(p_v+1)$ with probability at least $\frac{1}{2e}$. □