A Better Wake-up in Radio Networks

Bogdan S. Chlebus *   Dariusz R. Kowalski †

June 4, 2004

Abstract

We present an improved algorithm to wake up a multi-hop ad-hoc radio network. The goal is to have all the nodes activated, when some of them may wake up spontaneously at arbitrary times and the remaining nodes need to be awoken by the already active ones. The best previously known wake-up algorithm was given by Chrobak, Gąsieniec and Kowalski [11], and operated in time $\mathcal{O}(n^{5/3} \log n)$, where n is the number of nodes. We give an algorithm with the running time $\mathcal{O}(n^{3/2} \log n)$. This also yields better algorithms for other synchronization-type primitives, like leader election and local-clocks synchronization, each with a time performance that differs from that of wake-up by an extra factor of $\mathcal{O}(\log n)$ only, and improves the best previously known method for the problem by a factor of $n^{1/6}$. A wake-up algorithm is a schedule of transmissions for each node. It can be represented as a collection of binary sequences. Useful properties of such collections have been abstracted to define a (radio) synchronizer. It has been known that good radio synchronizers exist and previous algorithms [17, 11] relied on this. We show how to construct such synchronizers in polynomial time, from suitable constructible expanders. As an application, we obtain a wake-up protocol for a multiple-access channel that activates the network in time $\mathcal{O}(k^2 \text{ polylog } n)$, where k is the number of stations that wake up spontaneously, and which can be found in time polynomial in n. We extend the notion of synchronizers to universal synchronizers. We show that there exist universal synchronizers with parameters that guarantee time $\mathcal{O}(n^{3/2} \log n)$ of wake-up.

1 Introduction

Radio network is defined by a property that messages arriving simultaneously at a node cannot be received due to mutual interference. Such networks have been typically considered in a fully synchronous mode, when all the nodes are active, their local clocks are ticking at the same rate, and they produce the same round numbers. Recent work on multi-hop radio networks has concentrated mainly on broadcasting and gossiping in such networks. In this paper we study synchronization-type primitives in radio networks. We consider the problem how to wake up a network, which is known to be applicable to the tasks of synchronizing local clocks and electing a leader.

A radio network is modeled as a directed strongly-connected graph $G = (V, E)$ with $n = |V|$ nodes. A diameter D denotes the maximum, among pairs of nodes, of the length of a shortest directed path joining them. The case of general topology is referred to as a multi-hop network, while a special case of $D = 1$ is called a single-hop network or a multiple-access channel. The networks we consider are ad-hoc, which means that communication protocols do not rely on the nodes to know the topology. Communication is assumed to be synchronous in the following sense: all the nodes are running local clocks ticking at the same rate, but no global clock is assumed to be available. A node can perform at most one transmission per round. Initially all the nodes are asleep. Eventually some nodes wake up spontaneously and begin to run a protocol to activate

*Computer Science and Engineering University of Colorado at Denver, Campus Box 109, Denver, CO 80217, USA. E-mail: `Bogdan.Chlebus@cudenver.edu`. The work of this author is supported by the NSF Grant 0310503.

†Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbruecken, Germany, and Instytut Informatyki, Uniwersytet Warszawski, Banacha 2, 02–097 Warszawa, Poland. E-mail: `darek@mpi-sb.mpg.de`. The work of this author is supported in part by the KBN Grant 4T11C04425.

the whole network. When a node is asleep, it can be awoken by incoming messages. We start counting global rounds from the moment when the first node wakes up, which defines round zero. This numbering of rounds is by an external observer and serves the purpose to analyze the performance of communication protocols. Each node uses its local clock that is zeroed at a round when the node either wakes up spontaneously or is awoken by some other already active node.

A message delivered to a node of a radio network is not necessarily received successfully. The semantics of radio communication is defined by the following properties:

1. When a node performs a transmission, it sends the same message to all its neighbors, and each of these messages is delivered in the same round.
2. A message delivered to a node at a round is received successfully, or is *heard by the node*, if and only if exactly one neighbor of this node was performing a transmission at this round.
3. A sleeping node is awoken by a message only if it can hear it.

We do not assume a collision-detection mechanism. See [1, 4, 7] for an early work on radio networks that has established this model.

The wake-up problem for radio networks is defined as follows: activate all the nodes, when some of them start spontaneously at arbitrary times, while others join only after hearing a message. We consider only oblivious algorithms, in which the value of the local clock of a node determines if the node performs a transmission. This means that a wake-up algorithm is a collection of binary sequences that determine a schedule of transmissions for each node. A node performs a transmission at step i if the i-th bit in its sequence is a 1. The problem of synchronizing local clocks is to have all the local clocks set to the same value, when at the starting point they have these values set arbitrarily, but are ticking at the same rate. The leader election problem is a classical one in distributed computing: we need to have a single node distinguished, while in the beginning all the nodes are similar.

Known efficient solutions of the wake-up problem, as given in [11, 17], rely on the existence of combinatorial structures called *radio synchronizers*. Informally, an (n, k, m)-synchronizer is a wake-up protocol for a single-hop radio networks of n nodes, which operates in time m and is correct if at most k nodes wake up spontaneously. More formally, such a synchronizer is a collection of n sequences of 0-1 values, arranged in an array with rows of length m, and with a property that if one selects at most k sequences and shifts each of them arbitrarily but by at most m positions, then there is a column with only one occurrence of the value 1 in it and in the selected rows. This formal property is intended to capture the fact that there is one station among k that transmits alone at the round determined by this column.

Our results We extend the notion of radio synchronizers to universal synchronizers and show that such objects exists with parameters that allow to obtain runtime $O(n^{3/2} \log n)$. Our deterministic wake-up algorithm can be plugged into synchronization and leader election algorithms, as described in [11], to improve the previously known best solutions of these problems by a factor of at least $\Omega(n^{1/6})$. Prior to this work, it was only known that radio synchronizers of certain parameters exist; we show how to construct comparable radio synchronizers in polynomial time, relying on suitable constructible expanders.

We now overview our contributions in more detail.

I. We show that there is a deterministic wake-up algorithm with runtime $\mathcal{O}(n^{3/2} \log n)$. The fastest previously known algorithm had runtime $\mathcal{O}(n^{5/3} \log n)$. To develop such an algorithm, we extend the notion of radio synchronizers to universal synchronizers and show how to apply them in a wake-up protocol. Next we show that such objects exists with parameters good enough to obtain the runtime $\mathcal{O}(n^{3/2} \log n)$ of a wake-up algorithm.

II. We show that there are (n, k, m)-synchronizers with $m = \mathcal{O}(k^2 \text{ polylog } n)$, that can be constructed in time polynomial in n. The best non-constructive (n, k, m)-synchronizers, that are known to exist, have the value of m equal to $\mathcal{O}(k^2 \log n)$ and were used in [11] as components of a wake-up algorithm of runtime equal to $\mathcal{O}(n^{5/3} \log n)$. Radio synchronizers can be used directly in wake-up algorithms for single-hop radio networks. Our synchronizer determines a constructive wake-up algorithm with runtime

$\mathcal{O}(k^2 \text{ polylog } n)$, in which k is an upper bound on the number of nodes that awake spontaneously. Another application is a wake-up algorithm for networks of diameter D that operates in time $\mathcal{O}(Dn^{1+\varepsilon})$, for an arbitrary $\varepsilon > 0$, and which can be constructed in a quasi-polynomial time $2^{\text{polylog } n}$.

III. We show that there are a deterministic synchronization algorithm and a deterministic leader-election algorithm, each with runtime $O(n^{3/2} \log^2 n)$. The best previously known algorithms for these problems ran in time $\mathcal{O}(n^{5/3} \log^2 n)$.

Previous work The wake-up problem for radio networks was first considered by Gąsieniec, Pelc and Peleg [17]. They developed deterministic and randomized wake-up protocols for multiple-access channels. For such networks the algorithmic problems of clock synchronization and leader election are equivalent to the wake-up one, in that their complexities are of the same order of magnitude. Randomized wake-up protocols for multiple-access channel were also studied by Jurdziński and Stachowiak [20], and constructive protocols by Indyk [18]. The wake-up problem for general radio networks was first considered by Chrobak, Gąsieniec and Kowalski [11]. The best known general-case deterministic wake-up algorithm was given in [11]; it is non-constructive and its runtime is $\mathcal{O}(n^{5/3} \log n)$. That paper also showed how to synchronize local clocks and how to solve leader election in radio networks, based on a solution to the wake-up problem, while incurring only a logarithmic-time overhead.

The technique used in [11] relied on the notion of a radio synchronizer, which was implicitly used in [17]. It was shown in [11] that, for fixed n, k, where $k \leq n$, there exists (n, k, m)-synchronizer with $m = \mathcal{O}(k^2 \log n)$. Synchronizers with parameters (n, n, m), for $m = \mathcal{O}(n \log^2 n)$, were obtained in [17]. Such synchronizers give deterministic wake-up algorithms with runtime $\mathcal{O}(n \log^2 n)$ for single-hop radio networks. A $(n, n, \mathcal{O}(n^{1+\varepsilon}))$-synchronizer that can be constructed in a quasi-polynomial time $2^{\text{polylog } n}$ was given by Indyk [18], .

Clementi, Monti and Silvestri [14] showed a lower bound $\Omega(n \log D)$ on broadcasting, which is also a lower bound for the wake-up problem. A lower bound $\Omega(n)$ for wake-up in single-hop radio network was independently proved in [17].

Related work Broadcasting in radio networks is a special case of the wake-up problem, where initially only the source node is awoken, the remaining nodes join in disseminating the source message after having received a wake-up message from the source, which also brings the contents to be disseminated. Broadcasting problems have been considered in various computational settings, including centralized [1, 7, 8], distributed deterministic [6, 9, 10, 12, 14, 15, 21, 22, 23], and distributed randomized [4, 15, 23, 24]. A lower bound $\Omega(\log^2 n)$ for broadcasting was given by Alon *et al.* [1], it holds even with a bounded diameter and centralized protocol of complete knowledge. A lower bound $\Omega(D \log n)$ was shown by Bruschi and Del Pinto [6]. It was later improved to $\Omega(n \log D)$ by Clementi, Monti and Silvestri [14].

Problems regarding multiple concurrent communication were studied by Clementi, Monti and Silvestri [13] and Bar-Yehuda, Israeli, and Itai [5]. Radio protocols that are efficient in terms of energy use were given for a wake-up-like problem by Zheng, Hou and Sha [27], and for leader election by Jurdziński, Kutyłowski and Zatopiański [19].

A variant of a wake-up problem in asynchronous shared memory was studied by Fischer, Moran, Rudich and Taubenfeld [16]. A general approach to facilitate running synchronous algorithms in asynchronous message-passing model was proposed by Awerbuch [2]. Any specific solution following this approach is also called a synchronizer, see a book by Peleg [25] for more on this.

Organization The paper is organized as follows. We discuss the notions of a radio synchronizer and a universal (radio) synchronizer in Section 2. A deterministic wake-up algorithm, based on universal synchronizers, is presented and analyzed in Section 3. We show in Section 4 how to construct (n, k, m)-synchronizers in time polynomial in n and next discuss algorithmic applications. Section 5 contains a proof of the main technical result of this paper, which is about the existence of universal synchronizers with a small delay. Section 6 contains final remarks.

2 Radio synchronizers

An execution of a wake-up algorithm is determined by a pattern of spontaneous wake-ups. We model it by a function $\alpha \geq 0$, where $\alpha(v) = t$ means that the node v is to wake up spontaneously at round t to join the action at round $t+1$. Without loss of generality, one may assume that there is a node v such that $\alpha(v) = 0$. We say that *node v is active at step t* when either $\alpha(v) < t$, which means that v has woken up spontaneously, or v was awoken by a neighbor by step t. Let $\omega(v)$ denote the global step when node v becomes active.

An oblivious wake-up protocol $\mathcal{W}$ is determined by a collection $\mathcal{S} = \{\mathcal{S}^v\}_{v \in V}$, where each element $\mathcal{S}^v$ is a binary sequence $\mathcal{S}^v = S_1^v S_2^v \ldots S_m^v$ of length m. An interpretation is that node v transmits at step $\omega(v) + i$ precisely when $S_i^v = 1$. The parameter m denotes the runtime of the protocol.

Following Chrobak, Gąsieniec and Kowalski [11], we say that a protocol $\mathcal{S}$ is a (n, k, m)*-synchronizer* if, for an arbitrary set X of at most k nodes that wake up spontaneously according to an arbitrary schedule, there is a node in X that transmits alone within at most m steps after the first node in X wakes up. In terms of our notation, a synchronizer can be defined as follows: For any set $X \subseteq V$ of a size at most k, and for any schedule ω, there exists a number t, where $\min_{v \in X} \omega(v) < t \leq \min_{v \in X} \omega(v) + m$, such that $\sum_{v \in X} S^v_{t-\omega(v)} = 1$. We extend the range of function S^v to have $S_i^v = 0$, for $i < 0$, and we always assume that $\min_{v \in X} \omega(v) = 0$.

Next we recall two relevant known facts about synchronizers.

Fact 1 ([17]) *For every positive integer n there exists an (n, n, m)-synchronizer with $m = Cn \log^2 n$, where $C > 0$ is a sufficiently large constant.*

Fact 2 ([11]) *For every positive integers n and $k \leq n$ there exists an (n, k, m)-synchronizer with $m = Ck^2 \log n$, where $C \geq 22$ is a constant.*

These results are existential, as the synchronizers with these parameters were proved only to exist by the probabilistic method. Fact 1 was shown before synchronizers were defined in [11], it is reformulated here in terms of this notion.

In this paper we extend the notion of (n, k, m)-synchronizer to an (n, g)-universal synchronizer. Let n be a positive integer and g be a non-decreasing function from $[n] = [1..n]$ to positive integers. Let $\mathcal{S} = \{\mathcal{S}^v\}_{v \in V}$ be a collection of binary sequences $\mathcal{S}^v = S_1^v S_2^v \ldots S_m^v$ of length m. We extend the range of S^x to have $S_i^v = 0$, for $i < 0$. We say that $\mathcal{S}$ is a (n, g)*-universal synchronizer* if, for any schedule ω and integer $1 \leq k \leq n$, when the set $X = \{v \in [n] : \omega(v) < g(k)\}$ has size at most k, then there exists $t < g(k)$, such that $\sum_{v \in X} S^v_{t-\omega(v)} = 1$. The parameter m is called the *length* and the function $g \leq g(n) \leq m$ is the *delay* of this universal synchronizer. Notice that an (n, g)-universal synchronizer is a $(n, k, \Theta(g(k)))$-synchronizer for any $k \leq n$.

Theorem 1 *For each n there exists an (n, g)-universal synchronizer, with delay $g(k)$ upper bounded by the function $Ck \min\{k, \sqrt{n}\} \lg n$, for some constant $C > 0$.*

Notation $\lg x$ denotes a binary logarithm, while log is used when the base of a logarithm does not matter. An efficient wake-up algorithm that we develop is based on Theorem 1; see Section 5 for a proof.

3 Wakeup protocol

In leader-election and synchronization applications, one usually assumes that names are from some polynomial range, while the difference between the maximum and minimum-value names is $\mathcal{O}(n)$. To simplify an exposition, we assume that n nodes have unique names in the interval $[1..Bn]$, for a constant $B \geq 1$.

The protocol operates as follows. Take a (n, g)-universal synchronizer $\mathcal{S}$. Each processor v starts executing its sequence $\mathcal{S}^v \in \mathcal{S}$ immediately when it becomes active. We may assume that $g(k) = \mathcal{O}(k \min\{k, \sqrt{n}\} \log n)$, by Theorem 1.

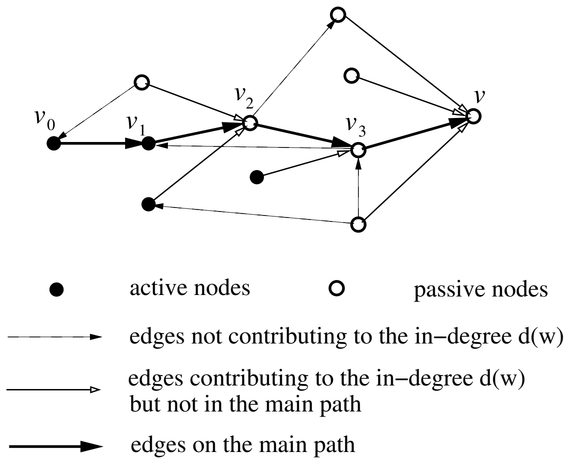

Figure 1: A sample main path from v_0 to v. The front consists of just the node v_3.

To analyze the protocol, fix a schedule α of spontaneous wake-ups. Let v_0 be among the nodes that become awoken spontaneously first. For every node $v \neq v_0$, consider a graph P_v defined as follows. It contains a shortest directed path from v_0 to v together with all the in-neighbors of nodes on the path and the in-going edges from them to the nodes on the path. Let us fix such a path $v_0, v_1, \ldots, v_a = v$ from v_0 to v and call it the *main path*, see Figure 1. Let $\delta(w)$, for every node w in the main path, denote the number of in-neighbors of node w in graph P_v. Additionally, let $d(w) \leq \delta(w)$, for every node w in the main path, denote the number of in-neighbors w' of node w in graph P_v such that $w' \neq v_{j'}$ and w' is not an in-neighbor of $v_{j'}$, for every $j < j' \leq a$. Notice that an in-neighbor contributes to the value of $d(w)$ iff it does not follow w on the main path nor is it joined to any node following w on the main path, in the order of growing indices.

For step t, we define the *front value* $f(t)$ to be equal to j iff the j-th node on the main path $v_0, v_1, \ldots, v_j, \ldots, v_a = v$ has the following properties:

a) Node v_j is not active, but one of its in-neighbors in P_v is active in step t, and

b) For every $j' > j$, node $v_{j'}$ is not active and none of its in-neighbors in P_v is active in step t.

Notice that $1 \leq j \leq a \leq D$, and that front values are non-decreasing.

Lemma 1 *If $f(t) < a$ and $d(v_{f(t)}) = k$, then $f(t + g(k)) > f(t)$.*

Proof: Let $t_0 \leq t$ be the first step such that $f(t_0) = f(t)$. Let $k = d(v_{f(t)})$. Consider the steps t_0 through $t_0 + g(k) - 1$. Notice that $f(t + g(k)) \geq f(t_0 + g(k))$, by monotonicity of function f, so it is sufficient to show that $f(t_0) < f(t_0 + g(k))$. There are two cases to consider.

Case 1. There is a node x such that either $\omega(x) < t_0 + g(k)$ and $x = v$ or x is an in-neighbor of some node v_j, for $j > f(t)$.

By definition of function f and assumption $f(t) < a$ we have that $f(t_0 + g(k)) \geq f(\omega(x)) > f(t_0)$.

Case 2. For every node x such that either $x = v$ or x is an in-neighbor of some node v_j, for $j > f(t)$, we have that $\omega(x) \geq t_0 + g(k)$.

By the assumption and by $d(v_{f(t)}) = k$, we have that the first in-neighbor of $v_{f(t)}$ that transmits does this at step t_0, and, during the next $g(k)$ steps, a total of at most k in-neighbors of $v_{f(t)}$ transmit in each step. These transmissions are according to an (n, g)-universal synchronizer, for schedule $\omega' = \omega - (t_0 - 1)$. According to ω', we have $\omega'(x) = 0$ for a in-neighbor x of $v_{f(t)}$ that transmits first. By the relevant properties of an (n, g)-universal synchronizer and Theorem 1, we obtain that there is a step $t_0 \leq t' \leq t_0 + g(k) - 1$ such that only one in-neighbor of $v_{f(t)}$ is transmitting in step t', which gives that $v_{f(t)}$ is active in step $t' + 1 \leq t_0 + g(k)$. Hence $f(t_0 + g(k)) \geq f(t' + 1) \geq f(t) + 1$, which concludes a proof of this case. $\square$

Theorem 2 *The algorithm wakes up a network of n nodes in time $\mathcal{O}(n^{3/2}\log n)$.*

Proof: Consider any node v and its graph P_v. It follows from Lemma 1 that node v is active in step

$$\sum_{1\le i\le a} g(d(v_i)) = \mathcal{O}(\sqrt{n}\log n \cdot \sum_{j=1}^{a} d(v_j)) \ .$$

Each node w in P_v contributes only once to the sum above, more precisely to $d(v_j)$, where x is an in-neighbor of v_j and x is not an in-neighbor of any $v_{j'}$ where $j' > j$. Hence an inequality $\sum_{1\le i\le a} d(v_i) \le n$ holds. We obtain that

$$\sum_{1\le i\le a} g(d(v_i)) = \mathcal{O}(n^{3/2}\log n) \ ,$$

which is the estimate sought. □

Combining Theorem 2 with a general method given in [11] to structure leader-election and synchronization protocols based on a wake-up one, we obtain the following.

Corollary 1 *For any positive integer n, there is a deterministic synchronization algorithm and a deterministic leader-election algorithm, for networks of n nodes, each with runtime $\mathcal{O}(n^{3/2}\log^2 n)$.*

4 Constructive synchronizers

In this section we show how to construct (n,k,m)-synchronizer in time polynomial in n.

Selective functions To simplify the exposition, assume that n is sufficiently large, and that $k \mid n$. Let P contain k different primes from the interval $[k\log n..3k\log n]$. The set P is well defined by the Chebyshev Theorem, see [3]. For a number $p\in P$, let π_p be a 0-1 sequence of length $3k^2\log n$ such that $\pi_p(j)=1$ when $p \mid j$ and $0<j\le 3k^2\log n$, and $\pi_p(j)=0$ otherwise. Let σ be a sequence of $3k^2\log n$ zeroes. Let d be the positive integer, that will depend on k and n in applications. Let h be a function from $V\times[d]$ to P. We define schedule S^v, for $v\in V$, as a concatenation of schedules $\pi_{h(v,1)}$, σ, $\pi_{h(v,2)}$, $\sigma,\ldots,\pi_{h(v,d)}$, σ, σ^{2d}, where σ^{2d} denotes a sequence of zeros of length $6dk^2\log n$. Function h is said to be *k-selective* iff for every set $W\subseteq V$ of size at most k, there is a number p in P such that $|h^{-1}(p)\cap(W\times[d])|=1$.

Lemma 2 *If function h is k-selective, then the set $\mathcal{S}=\{S^v\}_{v\in V}$ of schedules is an (n,k,m)-synchronizer with $m=\mathcal{O}(dk^2\log n)$.*

Proof: The length of each π_p and σ is $3k^2\log n$, hence $m=12dk^2\log n=\mathcal{O}(dk^2\log n)$. Fix a schedule ω. Let $X\subseteq V$ be a set of at most k nodes that are woken up. Let $W\subseteq X$ denote a set of awoken nodes in the interval $[1,\ldots,m/2]$. Let $k'=|W|$, which is at most k. By the property of k-selectiveness of function h, applied to the set $W\subseteq V$, we obtain that there is at least one $p\in P$ such that $|h^{-1}(p)\cap(W\times[d])|=1$. Since $v\in W$, we have that $\omega(v)\le m/2$, and hence the node v has enough time to execute the whole prefix $\pi_{h(v,1)},\sigma,\pi_{h(v,2)},\sigma,\ldots,\pi_{h(v,d)}$ by round m. Let $1\le j\le d$ be such that $h(v,j)=p$. Then, during time steps t_j+1 through $t_j+3k^2\log n$ according to its local clock, the node v executes the schedule π_p, where $t_j=\omega(v)+6(j-1)k^2\log n$.

Claim. There is a position t between t_j+1 and $t_j+3k^2\log n$ such that $S^v_{t-\omega(v)}=1$ and $S^w_{t-\omega(w)}=0$, for every $w\in X$.

Proof of the claim: Let two nodes be said to *collide* at a position when there is an occurrence of 1 at this position in both of their schedules. Between the positions t_j+1 and $t_j+3k^2\log n$, each node $w\in X$ may follow only a part of some schedule π_q, for $q\ne p$, and a part of a σ schedule, possibly empty. This is by the design of $\mathcal{S}$, and by the fact that each interval of length $3k^2\log n$ intersects only two consecutive sub-schedules from the concatenated sequence $\pi_{h(w,1)},\sigma,\pi_{h(w,2)},\sigma,\ldots,\pi_{h(w,d)}$, σ, σ^{2d}. The numbers q and p are different by the choice of p. Hence for every node $w\in X$ there may be at most one position t_w between

t_j+1 and $t_j+3k^2\log n$ such that v and w collide at this position t_w. If there were at least two such positions $t_w > t'_w$, then $p \mid (t_w - t'_w)$ and $q \mid (t_w - t'_w)$, which means that $t_w - t'_w \geq pq \geq k^2\log^2 n$. This would contradict $t_w - t'_w \leq 3k^2\log n < k^2\log^2 n$, for $n > 8$. Is follows that the node v, which has an occurrence of 1 on at least $k\log n$ positions between t_j+1 and $t_j+3k^2\log n$, may collide on at most $|X| \leq k$ of these positions with other nodes. Hence there is a position on which there is no collision with any other node in X. This completes a proof of the claim.

We obtain that a success we seek occurs at most at position $t_j + 3k^2\log n$, which is not larger than

$$\begin{aligned}\omega(v) + 6(j-1)k^2\log n + 3k^2\log n &< \omega(v) + 6jk^2\log n \\ &\leq m/2 + 6jk^2\log n \leq m\ ,\end{aligned}$$

by the facts that $\omega(v) \leq m/2$ and $6jk^2\log n \leq m/2$. This completes a proof. □

Selective graphs Selective families have been shown to be a useful building block to design broadcasting algorithms, see [9, 10, 14, 18]. Selective graphs are selective families represented by graphs. We consider bipartite graphs in which the nodes are partitioned into two sets V and P, and the function h assigns to a node $v \in V$ its j-th neighbor as $h(v,j)$, for $1 \leq j \leq d$. Assume that the incident edges of each node $v \in V$ are numbered from 1 through d. Define $h(v,j) = p$ when $p \in P$ is the jth neighbor of $v \in V$ in graph H. We say that $H = (V, P, h)$ is a (n,k,d)-*selective graph*, where each node in V is of a degree d, if for any set $W \subseteq V$ of a size at most k, there is a node $p \in P$ which has exactly one neighbor in the set W. If follows directly that such selectivity of graphs corresponds to selectivity of the function h.

Indyk [18] showed how to construct such selective graphs from dispersers. More precisely, for every $k' \leq k$, some d' and some $0 < \varepsilon < 1/2$, one can construct a (n,k,d)-selective graph, for $d = \mathcal{O}((d')^2\log n)$, in time polynomial in n, by relying on $(n,k',d',1/2-\varepsilon)$-dispersers. We say that a bipartite graph $H = (V,P,h')$ is a $(n,k',d',1/2-\varepsilon)$-disperser if the degree of each node in V is d' and, for every set $W \subseteq V$ of size k', the set of neighbors of W has size at least $(1/2+\varepsilon)|P|$.

Corollary 2 *There is a (n,k,m)-synchronizer with $m = \mathcal{O}(k^2 \text{ polylog } n)$ that can be constructed in time polynomial in n.*

Proof: We rely on a construction of $(n,k',d',1/2-\varepsilon)$-dispersers, for $d' = \mathcal{O}(\text{polylog } n)$ and for any constant $0 < \varepsilon < 1/2$, in time polynomial in size n, developed by Ta-Shma, Umans and Zuckerman [26]. Combine this with our construction, which is achievable in time linear in size of dispersers, and with Lemma 2. The time to construct an (n,k,m)-synchronizer is $\mathcal{O}(n^2)$ plus the time to construct the neighborhoods of nodes in the used disperser. □

Applications Synchronizers can be used directly to yield wake-up algorithms for single-hop radio networks. Applied this way, our synchronizer determines a constructive wake-up algorithm. It can be obtained in time polynomial in n, and operates with delay $\mathcal{O}(k^2 \text{ polylog } n)$. The parameter k is an upper bound on the number of nodes that awake spontaneously.

Another application regards wakeup algorithms for general networks of diameter D. An algorithm that operates in time $\mathcal{O}(n^{1+\varepsilon}D)$ can be constructed in a quasi-polynomial time $\mathcal{O}(2^{\text{ polylog } n})$. This algorithm can be obtained using an $(n,n,\mathcal{O}(n^{1+\varepsilon}))$-synchronizer as designed by Indyk [18]. The nodes running the algorithm transmit according to the corresponding sequences in this synchronizer. An analysis is by way of considering main-path graphs P_v. We argue that a new node on such a path is awoken within time $\mathcal{O}(n^{1+\varepsilon})$, while a total number of nodes to wake up is at most D. A non-constructive wake-up algorithm with the runtime $\mathcal{O}(nD\log^2 n)$ can be obtained by using a (n,n,m)-synchronizer, for $m = \mathcal{O}(n\log^2 n)$). Such a synchronizer is known to exist, as shown in [17].

Synchronizers were used to construct sub-quadratic wake-up algorithms for multi-hop radio networks [11]. Replacing a $(n, n^{1/3}, O(n^{2/3}\log n))$-synchronizer applied in [11], which was only proved to exist, by a $(n, n^{1/3}, \mathcal{O}(n^{2/3}\text{polylog } n))$-synchronizer, which is constructible in time polynomial in n, results in a wake-up protocol with runtime $\mathcal{O}(n^{5/3}\text{polylog } n)$. However the obtained protocol is not fully polynomial to construct, since the algorithm also uses another non-constructive part, which was only proved to exist by the probabilistic method.

5 Small-delay universal synchronizers exist

In this section we prove Theorem 1. Assume that n is a power of 4 and that $n \geq k \geq 2$. Define the quantity $C2^{\lceil \log k \rceil} \min\{2^{\lceil \log k \rceil}, \sqrt{n}\} \log n$ to be $g(k)$, where C is a constant that will be determined later. Observe that $g(k) = \mathcal{O}(k \min\{k, \sqrt{n}\} \log n)$. We use a counting argument, by way of estimating probabilities. For each $v \in V$ and t, such that $1 \leq t \leq g(n)$, set S^v_t equal to either 0 or 1 independently over v and t, where $S^v_t = 1$ with a suitable probability $p(v,t)$. These probabilities are chosen as follows:

$$p(v,t) = \begin{cases} 1 & \text{if } t \leq \log n, \\ 1/2 & \text{if } \log n < t \leq C4^2 \log n, \\ 1/2^j & \text{if } C4^j \log n < t \leq C4^{j+1} \log n, \text{ for some} \\ & \qquad j \text{ such that } 2 \leq j \leq 0.5 \log n - 1, \\ 1/n & \text{if } Cn \log n < t \leq Cn\sqrt{n} \log n, \\ 0 & \text{otherwise.} \end{cases}$$

The goal is to show that $\mathcal{S} = \{\mathcal{S}^v\}_{v \in V}$ is an (n,g)-universal synchronizer with a positive probability, for a certain value of C. We refer to a *position* to mean a global column number after shifting the universal synchronizer under construction by ω, that is, node v has value $S^v_{t-\omega(v)}$ in column t. If we refer to a *step*, then we mean a local column number, that is, node v in its step t has value S^v_t. We say that *there is a success at position* t if $\sum_{v \in [n]} S^v_{t-\omega(v)} = 1$. We may restrict our attention to a set $\mathcal{E}$ of such schedules ω that at least two nodes are awoken in the first step in ω, since otherwise a success occurs immediately.

Let $p_v(t)$, for node v and position t, denote a probability that v chooses 1 in position t, that is, in its step $t - \omega(v)$. More precisely, we set $p_v(t) = p(v, t - \omega(v))$ if $t > \omega(v)$, and $p_v(t) = 0$ for $t \leq \omega(v)$. Notice that the variable t represents a (local) step of node v in a definition of probability $p(v,t)$, while it stands for a (global) position t in a definition of probability $p_v(t)$. For a schedule ω and for any given position $t \leq g(n)$, let $X_t = \{v \in [n] : \omega(v) < t\}$ be a set of nodes that have awoken spontaneously by position t, unless woken up earlier. (The set $X_{g(k)}$ is referred to in the definition of a (m,g)-universal synchronizer.) We restrict our attention to those schedules $\mathcal{E}$ for which $|X_0| > 1$. Let Y_j denote an interval of positions t such that $2^j < |X_t| \leq 2^{j+1}$, for $0 \leq j \leq \log n - 1$. These sets Y_j are a partition of positions 1 through $g(n)$, for a schedule ω. Let t_j be a position that is smallest in Y_j.

Next partition $\mathcal{E}$ into sets $\mathcal{E}_j$, for $0 \leq j \leq \log n - 1$, as follows. For the range $0 \leq j \leq 0.5 \log n - 1$, we put $\omega \in \mathcal{E}_j$ if $|Y_j| \geq C4^{j-1} \log n$ and $|Y_{j'}| < C4^{j'-1} \log n$, for every $0 \leq j' < j$. For the remaining range $0.5 \log n \leq j \leq \log n - 1$, we put $\omega \in \mathcal{E}_j$ if $|Y_j| \geq C2^{j-1}\sqrt{n} \log n$, and $|Y_{j'}| < C4^{j'-1} \log n$, for every $0 \leq j' \leq 0.5 \log n - 1$, and $|Y_{j'}| < C2^{j'-1}\sqrt{n} \log n$, for every $0.5 \log n \leq j' < j$.

We also partition each $\mathcal{E}_j$ into *representatives*: one representative in $\mathcal{E}_j$ contains all $\omega \in \mathcal{E}_j$ with the same sequences of sets $X_0, X_1, \ldots, X_{g(2^j-1)}$. If there is a success on some position 1 through $g(2^j)$ in any schedule in a representative, then there is a success at the same position in all the schedules in this representative. There are at most $g(2^j)^{2^j}$ different representatives for each $\mathcal{E}_j$, as each is determined by a function from $[n]$ to a set of positions 0 through $g(2^j) - 1$.

Lemma 3 *Sets $\mathcal{E}_j$ make a partition of all the schedules in $\mathcal{E}$, for $0 \leq j \leq \log n - 1$.*

Proof: The sets $\mathcal{E}_j$ are all disjoint by definition. Suppose, to the contrary, that some $\omega \in \mathcal{E}$ is not is any $\mathcal{E}_j$. It follows that $|Y_j| < C4^{j-1} \log n$, for every $0 \leq j \leq 0.5 \log n - 1$, and $|Y_j| < C2^{j-1}\sqrt{n} \log n$, for every $j \geq 0.5 \log n$. Thus we have $g(n) = Cn\sqrt{n} \log^2 n = \sum_{j=0}^{\log n - 1} |Y_j|$, because the sets Y_j make a partition. This sum of numbers $|Y_j|$ is strictly smaller than

$$\sum_{j=0}^{0.5\log n-1} C4^{j-1}\log n + \sum_{j=0.5\log n}^{\log n-1} C2^{j-1}\sqrt{n}\log n$$
$$\le \sum_{j=0}^{\log n-1} C2^j\sqrt{n}\log n \le Cn\sqrt{n}\log n = g(n) \ ,$$

which is the contradiction. □

Let k denote the number of nodes that wake up spontaneously. The corresponding value $\gamma(k) = \lceil \lg k \rceil - 1$ turns out to be a critical upper bound on indices j in $\mathcal{E}_j$. We write γ instead of $\gamma(k)$, when k is understood. We often use only $\gamma = \gamma(k)$ in formulas, when it is sufficient to know, given $\gamma = \gamma(k)$, that $2^\gamma \le k < 2^{\gamma+1}$.

Suppose γ is small, by which we mean $1 \le \gamma \le 0.5\log n - 1$. Consider a schedule in $\mathcal{E}_\gamma$. Let T_γ denote the set of positions $t \in Y_\gamma$ such that $t \le g(2^\gamma)$, $p_v(t) \ne 1$ for every node $v \in [n]$, and $1 \le \sum_{v\in[n]} p_v(t) \le 13$.

Lemma 4 *If $1 \le \gamma \le 0.5\log n - 1$, then the size of T_γ is at least $(C/2)4^{\gamma-3}\log n$, for a sufficiently large constant $C > 0$.*

Proof: Let A denote the set X_{t_γ} and let B be the set $X_{t_{\gamma+1}-1} \setminus A$. The size of B is less than 2^γ, because $|A|+|B| \le 2^{\gamma+1}$ and $|A| > 2^\gamma$. For every node $v \in A$ we have $p_v(t_\gamma + C4^{\gamma-2}\log n) \le 1/2^{\gamma-2}$, since every node in A was woken up at least $C4^{\gamma-2}\log n$ steps before position $t_\gamma + C4^{\gamma-2}\log n$ and because $\gamma \le 0.5\log n - 1$. Therefore we may restrict the positions to those from $t_\gamma + C4^{\gamma-2}\log n$ through $t_\gamma + C4^{\gamma-1}\log n$. Denote the set of these positions as T', where $|T'| = 3C4^{\gamma-2}\log n$.

Our plan is to select a subset $T \subseteq T'$, then show that $|T| \ge C4^{\gamma-3}\log n$, and finally that $T \subseteq T_\gamma$, thus proving both upper and lower bounds for $\sum_{v\in[n]} p_v(t)$, for every $t \in T$.

Let $T \subseteq T'$ contain each $t \in T'$ such that $\sum_{v\in B} p_v(t) \le 9$. We show that T is sufficiently large by way of a counting argument. Let $q = \sum_{v\in B}\sum_{t\in T'} p_v(t)$. We can estimate it from above as follows:

$$q \ \le \ \sum_{v\in B}\sum_{j=0}^{\gamma-2}(C\log n\cdot 1/2^j)\cdot(4^{j+1}-4^j)$$
$$\le \ |B|\cdot 3C2^{\gamma-1}\log n \ < \ 6C4^{\gamma-1}\log n \ .$$

Also $|T| \ge C4^{\gamma-3}\log n$, since otherwise we would have that q is at least

$$9(|T'|-|T|) \ > \ 9(3C4^{\gamma-2}\log n - C4^{\gamma-3}\log n)$$
$$= \ 9(11C4^{\gamma-3}\log n) \ = \ 99C4^{\gamma-3}\log n \ > \ 6C4^{\gamma-1}\log n \ ,$$

which is strictly larger than q: clearly a contradiction. The inequality $\sum_{v\in B} p_v(t) \le 9$ holds, by definition of T. Similarly, we have the inequality $\sum_{v\in A} p_v(t) \le |A|\cdot 1/2^{\gamma-1} \le 4$. Therefore the number 13 is an upper bound on $\sum_{v\in X} p_v(t)$, for each $t \in T$.

Next we prove a lower bound on $\sum_{v\in X} p_v(t)$, for all $t \in T$. We start with an observation that, for every node $v \in A$ and time step $t \in T$ such that $t_\gamma + C4^{\gamma-2}\log n \le t < t_\gamma + C4^{\gamma-1}\log n$, the inequality $p_v(t) \ge 1/2^\gamma$ holds. Indeed, by a definition of γ, we have that

$$t \ < \ |Y_0| + |Y_1| + \ldots + |Y_{\gamma-1}| + C4^{\gamma-1}\log n$$
$$< \ (4^{-1} + 4^0 + \ldots + 4^{\gamma-2} + 4^{\gamma-1})C\log n \ < \ 2C4^{\gamma-1}\log n \ .$$

It follows that $\sum_{v\in A} p_v(t) \ge |A|\cdot 1/2^\gamma \ge 1$.

Next we show that any element of T is at most $g(2^\gamma)$. Indeed, $T \subseteq T'$, and every $t \in T'$ is smaller than $t_\gamma + C4^{\gamma-1}\log n$. Additionally, by the property that $\omega \in \mathcal{E}_\gamma$, we have that t_γ is at most

$$\sum_{j=0}^{\gamma-1}|Y_j| \ < \ \sum_{j=0}^{\gamma-1} C4^{j-1}\log n \ < \ C4^{\gamma-1}\log n \ ,$$

and so $t < t_\gamma + C4^{\gamma-1}\log n < C4^\gamma \log n = g(2^\gamma)$.

To complete a proof, it is sufficient to show that if we remove from T all the positions t such that there is $v \in [n]$ with the property $p_v(t) = 1$, then what remains is large enough. By definition, there are at most $\log n$ positions t such that $p_v(t) = 1$, for every node $v \in X_{2^{\gamma+1}-1}$. Since $|X_{2^{\gamma+1}-1}| \le 2^{\gamma+1}$, we remove at most $2^{\gamma+1}\log n$ positions from T. What remains is a subset of T_γ of a size at least $C4^{\gamma-3}\log n - 2^{\gamma+1}\log n \ge (C/2)4^{\gamma-3}\log n$, for a sufficiently large constant $C > 0$. $\square$

Suppose γ is large, by which we mean $0.5\log n \le \gamma \le \log n - 1$. Consider a schedule in $\mathcal{E}_\gamma$. Let U_γ denote a set of positions $t \in Y_\gamma$ such that $t \le g(2^\gamma)$, $p_v(t) \ne 1$ for every node $v \in [n]$, and $\frac{2^\gamma}{n} \le \sum_{v\in[n]} p_v(t) \le 19$.

Lemma 5 *The size of U_γ, for $0.5\log n \le \gamma \le \log n - 1$, is at least $(C/2)2^{\gamma-3}\sqrt{n}\log n$, for a sufficiently large constant $C > 0$.*

Proof: We proceed in a way similar to that in the proof of Lemma 4, with minor differences in formulas and calculations. Define U' to consist of positions t_γ through $t_\gamma + C2^{\gamma-1}\sqrt{n}\log n$. Clearly$|U'| = C2^{\gamma-1}\sqrt{n}\log n$. The plan is to select a suitable subset $U \subseteq U'$, then show that $|U| \ge 2^{\gamma-3}\sqrt{n}\log n$, and finally that $U \subseteq U_\gamma$, proving both upper and lower bounds for $\sum_{v\in[n]} p_v(t)$, for every $t \in U$.

Let $U \subseteq U'$ contain every position $t \in U'$ such that $\sum_{v\in[n]} p_v(t) \le 19$. We show that U is sufficiently large, using a counting argument. Let $q = \sum_{v\in[n]}\sum_{t\in U'} p_v(t)$. It can be estimated that q is at most:

$$
\begin{aligned}
&\sum_{v\in[n]} \Big(\sum_{j=0}^{0.5\log n-1} (C\log n)\cdot(4^{j+1}-4^j)\cdot\frac{1}{2^j} \;+ \\
&\qquad \max\{C2^{\gamma-1}\sqrt{n} - Cn, 0\}\log n \cdot \frac{1}{n}\Big) \;\le \\
&|X_{t_\gamma + C2^{\gamma-1}\sqrt{n}\log n - 1}| \Big(3C2^{0.5\log n}\log n + C2^{\gamma-1}\log n\frac{1}{\sqrt{n}}\Big) \\
&\quad < \; 3C2^{\gamma+1}\sqrt{n}\log n + C\frac{4^\gamma}{\sqrt{n}}\log n \\
&\quad \le \; 7C2^\gamma\sqrt{n}\log n \; .
\end{aligned}
$$

Notice that the derivation above involved only nodes in $X_{t_\gamma + C2^{\gamma-1}}$. The other nodes, by definition, are not active at the positions in U'. This is because if $v \notin X_{t_\gamma + C2^{\gamma-1}}$, then $\omega(v) \ge t_\gamma + C2^{\gamma-1}$. All this, together with $S^v_{t'<0} = 0$, implies $S^v_{t-\omega(v)} = 0$, for each $t \in U'$. The estimate $|U| \ge C2^{\gamma-3}\sqrt{n}\log n$ holds, since otherwise the quantity q would be at least

$$
\begin{aligned}
&(|U'| - |U|)\cdot 19 > 19\,(C2^{\gamma-1}\sqrt{n}\log n - C2^{\gamma-3}\sqrt{n}\log n) \\
&\quad = 57\,C2^{\gamma-3}\sqrt{n}\log n > 7\,C2^\gamma\sqrt{n}\log n > q \; ,
\end{aligned}
$$

which is a contradiction.

By the definition of U, the number 19 is an upper bound on $\sum_{v\in[n]} p_v(t)$, for each $t \in T$. Next we show a lower bound for $\sum_{v\in[n]} p_v(t)$, for each $t \in U$. Consider the nodes in the set X_{t_γ}, who are all active at any position in U. In every step $t \in U$, each processor in X_{t_γ} transmits with a probability at least $1/n$. It follows that

$$\sum_{v\in[n]} p_v(t) \;\ge\; |X_{t_\gamma}|\cdot 1/n \;>\; 2^\gamma/n \; .$$

Next we show that any element in U is at most $g(2^\gamma)$. Indeed, $U \subseteq U'$, and every $t \in U'$ is smaller than

$t_\gamma + C2^{\gamma-1}\sqrt{n}\log n$. Additionally, by the property that $\omega \in \mathcal{E}_\gamma$, we obtain that t_γ is at most

$$\begin{aligned}\sum_{j=0}^{\gamma-1}|Y_j| &< \sum_{j=0}^{0.5\log n-1} C4^{j-1}\log n + \sum_{j=0.5\log n}^{\gamma-1} C2^{j-1}\sqrt{n}\log n \\ &< \sum_{j=0}^{\gamma-1} C2^{j-1}\sqrt{n}\log n \ < \ C2^{\gamma-1}\sqrt{n}\log n\end{aligned}$$

hence $t < t_\gamma + C2^{\gamma-1}\sqrt{n}\log n < C2^\gamma\sqrt{n}\log n = g(2^\gamma)$.

To conclude, it is sufficient to show that if we remove from U all the positions t such that there is $v \in [n]$ satisfying $p_v(t) = 1$, then the remaining set is large enough. By definition, there are at most $\log n$ positions t such that $p_v(t) = 1$, for each node $v \in X_{2^{\gamma+1}-1}$. We remove at most $2^{\gamma+1}\log n$ positions from set U, because $|X_{2^{\gamma+1}-1}| \le 2^{\gamma+1}$, Therefore what remains is a subset of U_γ and its size is at least $C2^{\gamma-3}\sqrt{n}\log n - 2^{\gamma+1}\log n \ge (C/2)2^{\gamma-3}\sqrt{n}\log n$, for sufficiently large constant $C > 0$. □

The next technical lemma is used while applying Lemmas 4 and 5.

Lemma 6 *Let X be a set and let p_v, for every $v \in X$, be the positive number not greater than $1/2$. For any $a, b > 0$, if $a \le \sum_{v\in X} p_v \le b$, then $\sum_{v\in X} p_v \cdot \prod_{w\ne v}(1-p_w) \ge a \cdot 4^{-b}$.*

Proof: We can estimate the considered sum of products as follows:

$$\begin{aligned}\sum_{v\in X} p_v \prod_{w\ne v}(1-p_w) &\ge \Big(\sum_{v\in X} p_v\Big)\prod_{w\in X}(1-p_w) \\ &\ge \Big(\sum_{v\in X} p_v\Big)\prod_{w\in X}\Big((1-p_w)^{1/p_w}\Big)^{p_w} \\ &\ge \Big(\sum_{v\in X} p_v\Big)\prod_{w\in X} 4^{-p_w} \\ &\qquad\qquad \text{since } p_w \le 1/2\ , \\ &= \Big(\sum_{v\in X} p_v\Big)4^{-\sum_{w\in X} p_w}\ ,\end{aligned}$$

which is at most $a4^{-b}$. □

Let an event $\mathrm{E}(k, R)$ be defined to hold if there are a set X of a size k, such that $X = X_{g(k)}$, and a schedule $\omega \in \mathcal{E}_\gamma$, such that there is no success at positions 1 through R, where $2^\gamma < k \le 2^{\gamma+1}$.

Lemma 7 *If $0 \le \gamma \le 0.5\log n - 1$, then the probability of event $\mathrm{E}(k, C4^\gamma \log n)$ is at most 2^{-k}.*

Proof: The case $\gamma = 0$ is direct. Consider $\gamma > 0$ Let us fix $t \in T_\gamma$, where the set T_γ is as in Lemma 4. We estimate a probability that there is exactly one $v \in [n]$ such that $S^v_{t-\omega(v)} = 1$. By independence of the random variables S^v_t, this probability equals

$$\sum_{v\in[n]} p_v(t) \cdot \prod_{w\ne v}(1 - p_w(t))\ . \tag{1}$$

Lemma 6 is applicable, because for every $v \in X_t$ we have $p_v(t) \le 1/2$ and $1 \le \sum_{v\in[n]} p_v(t) \le 13$ by definition of T_γ. We obtain that the probability (1) is at least 4^{-13}.

We need to estimate the probability of the event that the sum $\sum_{v\in[n]} S^v_{t-\omega(v)}$ is different from 1 for all $t \in T_\gamma$. This is an intersection of the corresponding events for each such t. A success at position t, for any $t \in T_\gamma$, is determined by random variables $S^v_{t-\omega(v)}$, and is independent from successes at other positions in T_γ. A probability that there is no success with schedule ω during all the positions t from 1 through $C4^{\gamma+1}\log n$ is at most

$$\left(1-4^{-13}\right)^{|T_\gamma|} \le \left(1-4^{-13}\right)^{C4^{\gamma-3}\log n} ,$$

by our estimate of (1) and by Lemma 4. Consider set X such that $X = X_{g(k)} \subseteq X_{g(2^{\gamma+1})}$. Notice that $g(k) = g(2^{\gamma+1})$, by definition of the delay g. There are at most $2^\gamma\binom{n}{2^{\gamma+1}}$ such sets X. For each of them, there are at most $g(k)^k \le (C4^\gamma \log n)^{2^{\gamma+1}}$ representatives of schedules in $\mathcal{E}_\gamma$. It follows that a probability of the event $\mathrm{E}(k, C4^\gamma \log n)$ is at most

$$\begin{aligned} & 2^\gamma\binom{n}{2^{\gamma+1}} \cdot (C4^\gamma\log n)^{2^{\gamma+1}} \cdot (1-4^{-13})^{C4^{\gamma-3}\log n} \\ \le\ & \left(2\frac{ne}{k}\cdot Cn\log n\cdot(1-4^{-13})^{C2^{\gamma-4}\log n}\right)^{2^{\gamma+1}} \end{aligned}$$

which is less than $2^{-2^{\gamma+1}}$, and hence than 2^{-k}, for a sufficiently large constant $C>0$. □

A proof along the lines of the proof of Lemma 7, while using Lemma 5 instead of Lemma 4, shows the following:

Lemma 8 *If $0.5\log n \le \gamma \le \log n - 1$, then a probability of the event $\mathrm{E}(k, C2^\gamma\sqrt{n}\log n)$ is at most 2^{-k}.*

Proof: Fix a schedule ω and consider a set U_γ as in Lemma 5. Choose a $t \in U_\gamma$. We first estimate a probability that there is exactly one $v\in[n]$ such that $S^v_{t-\omega(v)} = 1$. Similarly as in the proof of Lemma 7, by independence of the variables S^v_t, this probability is equal to

$$\sum_{v\in[n]} p_v(t)\cdot\prod_{w\ne v}(1-p_w(t)) \ . \tag{2}$$

Lemma 6 is applicable, because $p_v(t)\le 1/2$ for every $v\in X_t$. Plugging in the bounds on $\sum_{v\in[n]} p_v(t)$ from the definition of U_γ, we obtain that the probability given in (2) is at least $x = 2^\gamma\cdot 4^{-19}/n$. A success at a position t means that $\sum_{v\in[n]} S^v_{t-\omega(v)} = 1$. Relying on the independence of the S^v_t, and hence also of the random variables $S^v_{t-\omega(v)}$, we obtain that a probability of failures for all $t\in U_\gamma$ is at most

$$\begin{aligned} (1-x)^{|U_\gamma|} &\le \left(1-\frac{2^\gamma}{n}\cdot 4^{-19}\right)^{C2^{\gamma-3}\sqrt{n}\log n} \\ &\le n^{-4\cdot 2^{\gamma+1}\log C} , \end{aligned}$$

for a sufficiently large constant C, by Lemma 5. The number $(1-4^{-19})^{C2^{\gamma-3}\sqrt{n}\log n}$ upper bounds the probability that no success occurs under schedule ω at any position t from 1 through $C2^\gamma\sqrt{n}\log n$. Consider set X of size k such that $X = X_{g(k)} \subseteq X_{g(2^{\gamma+1})}$. Note that $g(k) = g(2^{\gamma+1})$, by our definition of g.

There are at most $2^\gamma\binom{n}{2^{\gamma+1}}$ such sets X, and at most $g(k)^k \le (C2^\gamma\sqrt{n}\log n)^{2^{\gamma+1}}$ representatives of schedules in $\mathcal{E}_\gamma$ for each of them. It follows that a probability of the event $\mathrm{E}(k, C2^\gamma\sqrt{n}\log n)$ is at most

$$\begin{aligned} & 2^\gamma\binom{n}{2^{\gamma+1}}\cdot(C2^\gamma\sqrt{n}\log n)^{2^{\gamma+1}}\cdot n^{-3\cdot 2^{\gamma+1}\log C} \\ \le\ & \left(2\frac{ne}{k}\cdot Cn\log n\cdot n^{-4\log C}\right)^{2^{\gamma+1}} , \end{aligned}$$

for a suitable constant $C>0$, which is less than $2^{-2^{\gamma+1}}$, and so less than 2^{-k}. □

Putting this all together, we consider the following *Combined Event*, for a wake-up protocol $\mathcal{S}$ selected randomly according to the probabilities $p(v,t)$:

There are a schedule $\omega \in \mathcal{E}$, and a number k with $1 < k \leq n$, and a set X of size k with $X = X_{g(k)}$, such that there is no success at positions 1 through $C2^\gamma \min\{2^\gamma, \sqrt{n}\} \log n$.

The quantity $C2^\gamma \min\{2^\gamma, \sqrt{n}\} \log n$ is at most $g(k)$. Notice that the Combined Event is a union of events $\mathrm{E}(k, g(k))$, for $k = 2, \ldots, n$, hence its probability is at most $\sum_{k=2}^{n} 2^{-k} < 1$, by adding the probabilities given by Lemmas 7 and 8 for all the possible values of k. By the probabilistic method, there exists a wake-up protocol $\mathcal{S}$ for which the Combined Event does not hold.

Next we show that if the Combined Event does not hold, for a wake-up protocol $\mathcal{S}$, then $\mathcal{S}$ is an (n, g)-universal synchronizer. A condition defining this event, when negated, translates into the following: for every $1 < k \leq n$ and X of a size k, and schedule $\omega \in \mathcal{E}$, if $X_{g(k)} = X$, which means that $\omega \in \mathcal{E}_\gamma$, then there is a success at a position between 1 and $C2^\gamma \min\{2^\gamma, \sqrt{n}\} \log n = g(2^\gamma)$, and hence by the position $g(k)$. Take any k such that $2 \leq k \leq n$ and schedule ω such that $|X_{g(k)}| \leq k$. There are two cases possible.

Case 1: $\gamma(|X_{g(k)}|) = \gamma(k)$.

In this case, by definition of function g, we have that $|X| = |X_{g(k)}| = |X_{g(|X|)}|$, so using $k' = |X|$ instead of k we obtain what is required in the definition of a universal synchronizer.

Case 2: $\gamma(|X_{g(k)}|) < \gamma(k)$.

In this case we have that $|X_{g(2^{\gamma(k)-1})}| \leq |X_{g(k)}| \leq 2^{\gamma(k)}$ and we can inductively rely on $k' = |X_{g(2^{\gamma(k)-1})}|$. Since $\gamma(k') < \gamma(k)$, there is $k^* \leq k' < k$ such as required in the definition of a universal synchronizer. The base of induction occurs for $\gamma(k) = 1$, which is covered by Case 1.

6 Conclusion and open problems

This paper is about efficient design of fundamental synchronization primitives in general ad-hoc radio networks. We present a wake-up algorithm that has $\mathcal{O}(n^{3/2} \log n)$ as a bound on its time performance, which improves a result from [11]. This also improves efficiency of algorithms for synchronization and leader election.

We give a polynomial-time construction of radio synchronizers, which results in a first constructive wake-up algorithm for a multiple-access channel with $\mathcal{O}(k^2 \text{ polylog } n)$ time performance, where k is the number of nodes that wake up spontaneously.

The optimal performance of a wake-up algorithm is open. The best known lower bound, given in [14], is $\Omega(n \log D)$, and it holds for broadcasting. It is also open if broadcasting can be performed faster than waking up a network. It is not known if waking up could be performed faster than in time $\mathcal{O}(nD \text{ polylog } n)$ in networks of diameter D, for $D = o(\sqrt{n})$.

Radio synchronizers seem to be a natural notion to study in the context of synchronizations problems in radio networks. It would be interesting to find a lower bound on the length m of (n, k, m)-synchronizers or on the delay function $g(k)$ of an (n, g)-universal synchronizer that would be better than $\Omega(k \log(n/k))$, which is derived from a bound on the size of (n, k)-selective-families given in [14]. Achieving a polynomial-time construction of universal synchronizers of a similar length to those shown to exist in this paper and in [11] is an open problem, especially for k close to n.

7 Acknowledgment

The authors thank Surender Baswana and Piotr Indyk for helpful discussions.

References

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

[2] B. Awerbuch, Complexity of network synchronization, *Journal of the ACM*, 4 (1985) 804 - 823.

[3] E. Bach, and J. Shallit, *"Algorithmic Number Theory,"* Vol. I, 1996, The MIT Press, Cambridge, Massachusetts.

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

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

[6] D. Bruschi, and M. Del Pinto, Lower bounds for the broadcast problem in mobile radio networks, *Distributed Computing*, 10 (1997) 129 - 135.

[7] I. Chlamtac, and S. Kutten, On broadcasting in radio networks - problem analysis and protocol design, *IEEE Transactions on Communications*, 33 (1985) 1240 - 1246.

[8] I. Chlamtac, and O. Weinstein, The wave expansion approach to broadcasting in multi-hop radio networks, *IEEE Transactions on Communications*, 39 (1991) 426 - 433.

[9] B.S. Chlebus, L. Gąsieniec, A. Gibbons, A. Pelc, and W. Rytter, Deterministic broadcasting in unknown radio networks, *Distributed Computing*, 15 (2002) 27 - 38.

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

[11] M. Chrobak, L. Gąsieniec, and D. Kowalski, The wake-up problem in multi-hop radio networks, in *Proc., 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2004, pp. 985 - 993.

[12] M. Chrobak, L. Gąsieniec, and W. Rytter, Fast broadcasting and gossiping in radio networks, *Journal of Algorithms*, 43 (2002) 177 - 189.

[13] A. Clementi, A. Monti, and R. Silvestri, Distributed multi-broadcast in unknown radio networks, in *Proc., 20th ACM Symposium on Principles of Distributed Computing (PODC)*, 2001, pp. 255 - 264.

[14] A. Clementi, A. Monti, and R. Silvestri, Selective families, superimposed codes, and broadcasting on unknown radio networks, in *Proc., 12th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2001, 709 - 718.

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

[16] M.J. Fischer, S. Moran, S. Rudich, and G. Taubenfeld, The wakeup problem, *SIAM Journal on Computing*, 25 (1996) 1331 - 1332.

[17] L. Gąsieniec, A. Pelc, and D. Peleg, The wakeup problem in synchronous broadcast systems, *SIAM Journal on Discrete Mathematics*, 14 (2001) 207 - 222.

[18] P. Indyk, Explicit constructions of selectors and related combinatorial structures, with applications, in *Proc., 13th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2002, pp. 697 - 704.

[19] T. Jurdziński, M. Kutyłowski, and J. Zatopiański, Efficient algorithms for leader election in radio networks, in *Proc., 21st ACM Symposium on Principles of Distributed Computing (PODC)*, 2002, pp. 51 - 57.

[20] T. Jurdziński, and G. Stachowiak, Probabilistic algorithms for the wakeup problem in single-hop radio networks, in *Proc., 13th International Symposium on Algorithms and Computation (ISAAC),* 2002, LNCS 2518, pp. 535 - 549.

[21] D.R. Kowalski, and A. Pelc, Deterministic broadcasting time in radio networks of unknown topology, in *Proc., 43rd IEEE Symposium of Foundations of Computer Science (FOCS),* 2002, pp. 63 - 72.

[22] D.R. Kowalski, and A. Pelc, Faster deterministic broadcasting in ad-hoc radio networks, in *Proc., 20th Symposium on Theoretical Aspects of Computer Science (STACS),* 2003, LNCS 2607, pp. 109 - 120.

[23] D.R. Kowalski, and A. Pelc, Deterministic broadcasting time in radio networks of unknown topology, in *Proc., 22nd ACM Symposium on Principles of Distributed Computing (PODC),* 2003, pp. 73 - 82.

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

[25] D. Peleg, *"Distributed Computing. A Locality-Sensitive Approach,"* 2000, SIAM, Philadelphia, Pennsylvania.

[26] A. Ta-Shma, C. Umans, and D. Zuckerman, Loss-less condensers, unbalanced expanders, and extractors, in *Proc., 33rd ACM Symposium on Theory of Computing (STOC)*, 2001, pp. 143 - 152.

[27] R. Zheng, J.C. Hou, and L. Sha, Asynchronous wakeup for power management in ad hoc networks, in *Proc., 4th ACM Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, 2003, pp. 35 - 45.