Weak Communication in Radio Networks*

Tomasz Jurdziński∗†, Mirosław Kutyłowski‡, and Jan Zatopiański†

∗ Institute of Computer Science, Technical University Chemnitz
†Institute of Computer Science, Wrocław University
‡Institute of Mathematics, Wrocław University of Technology and Dept. of Math. and Computer Science, A. Mickiewicz University, Poznań

Abstract

Quite often algorithms designed for no-collision-detection radio networks use a hidden form of collision detection. Namely, they assume that a station can simultaneously send and listen. Then, if it cannot hear its own message, then apparently the message has been scrambled by another station sending at the same time. Industrial standard IEEE Standard 802.11 says that a station can either send or listen to a radio channel, but not both. Therefore we consider a *weak* radio network model with no collision detection where a station cannot simultaneously send and receive signals. Otherwise we talk about *strong model*.

We show that computational power of weak and strong radio networks differ substantially in deterministic case: deterministic leader requires $\Omega(\log n)$ energy cost in the weak model and can be solved with $O(\sqrt{\log n})$ energy cost in the strong model. On the other hand, we present a very efficient simulation of strong radio networks by weak ones, with preprocessing that requires $O(n)$ steps and has $O(\log\log n)$ energy cost. radio network, initialization, leader election, power consumption

1 Introduction

Communication via radio channels became today an attractive alternative for wired networks. Luck of a fixed infrastructure and mobility allows to create self-adjusting network of independent devices that is useful for application areas such as disaster-relief, law-enforcement, and so on. The main advantages are dynamic topology and simple communication protocol. However, colliding messages create many challenging problems.

We are interested in networks of *stations* communicating via radio channels. The stations are hand-held bulk produced devices running on batteries. They have no ID's or serial numbers, some of the stations are switched off, but somehow the network has to organize itself, despite the lack of a central control.

It is often overlooked by designers of algorithms for radio networks that many current technologies do not provide capability of listening while a station is sending (IEEE Standard 802.11). On the other hand, simultaneous sending and listening for collisions is a feature used in many algorithms. Our goal is to check how does it influence complexity of problems on radio networks.

* partially supported by KBN, grants 7 T11C 032 21, 7 T11C 032 20, and DFG, grant GO 493/1-1

1.1 Model

A radio network (RN, for short) consists of several processing units called *stations*. A RN is synchronized by a global clock (for instance based on GPS). Communication is possible in time slots, called here *steps*. We assume that there is a single communication channel available to all stations (so we consider *single-channel, single-hop* RN's). If during a step stations may either send (broadcast) a message or listen to the channel, then we talk about *weak* RN. If both operations can be performed by a station, then we have to do with the *strong* RN. (In the literature, by a RN the authors usually mean the strong RN.) If exactly one station sends, then all stations that listen at this moment receive the message sent. We assume in this paper that if at least two stations send, then a *collision* occurs and the stations that listen do not receive messages and even cannot recognize that messages have been sent (*no collision-detection RN*, no-CD RN for short). If a station is the only station that sends a message during step i, then we say that the station *succeeds* and that step i is *successful*.

The stations of a RN fall into two categories. Some of them are switched off and do not participate in the protocol. The other stations are *active* and we assume that they stay active during the whole protocol. The set of all active stations is called *active set*. During a step an active station might be either *awake* or *asleep*. The first case occurs when a station sends a message or listens to the communication channel.

1.2 Complexity measures

There are two main complexity measures for RN algorithms: *time complexity* and *energy cost* [11]. Time complexity is the number of steps executed. Energy cost is the maximum over all stations of the number of steps in which the station is awake.

Low energy cost is a desirable feature: if a station is awake for a longer time, it may happen that it fails to operate due to batteries exhaustion [3]. So a station which is awake much longer than the other stations is likely to fail and break down the protocol.

For further discussion on the RN model and its complexity measures see e.g., [2, 8–13, 7].

1.3 Fundamental tasks

Due to the lack of ID numbers, uncertainty about the number of active stations, switching off the stations, and so on, certain problems that are trivial in the wired networks become hard for RN's. Some of such basic tasks are the following:

Leader election: active stations have to choose a single station, called *leader*, among themselves. That is, at the end of the protocol a single station has status *leader* and all other active stations have status *non-leader*.

Initialization: consecutive ID numbers are to be assigned to active stations with no ID's.

Renumeration: consecutive ID numbers are to be assigned to active stations having some unique ID's, but in the range bigger than the number of stations.

2 Previous results

Due to the role of energy cost, there is a lot of interest for algorithms that are time and energy efficient. However, research has been focused on the strong model.

For leader election, the first energy efficient algorithm has been designed in [10]. The authors present a randomized algorithm that for n stations elects a leader (n must be known to the stations) in time $O(\log f)$ and energy $O(\log\log f + \frac{\log f}{\log n})$ with probability $1 - 1/f$ for any $f \geq 1$. Moreover, they get $O(\log n)$ energy cost and $O(\log^2 n)$ time with probability $1 - \frac{1}{n}$, if the number of active stations n is unknown. [6] presents an algorithm which achieves $O(\log n)$ time and $O(\log^* n)$ energy cost with probability $1 - \frac{1}{n}$ when n is known to the stations. If n is unknown, a general algorithm that approximates the value of n within a constant multiplicative factor can be applied [5]. It uses time $O(\log^{2+\varepsilon} n)$ and energy cost $O((\log\log n)^{\varepsilon})$ with probability $1 - \frac{1}{n}$.

It is impossible to elect a leader deterministically when stations are indistinguishable, due to symmetry. So we apply deterministic algorithms when the stations have unique ID's, say in the range $[1..n]$ (but any subset of these ID's may correspond to active stations). A simple leader election deterministic algorithm for this case is given in [10, 11]. Its energy cost is $\log n + O(1)$, run time is $n + O(1)$. A recursive procedure that works in time $O(n)$ and energy $O(\log^{\varepsilon} n)$ is given in [6].

An energy efficient solution to initialization problem is proposed in [11]: with probability at least $1 - \frac{1}{n}$ its energy cost is $O(\log\log n)$ and execution time is $O(n)$. However, the number of stations n must be known beforehand. The algorithm can be easily generalized to the case when an approximation of n within a constant multiplicative factor is known. Hence using size-approximation algorithm from [5], the same time and energy efficiency can be obtained for an unknown n.

3 New results

The main result of this paper is a randomized protocol for simulating algorithms for strong RN's by weak RN's.

Theorem 1. *An algorithm for strong RN with runtime T and energy cost E can be simulated by a randomized weak RN with runtime $O(T)$ and energy cost $O(\max(E, T/n))$. A preprocessing is required with runtime $O(n)$ and energy cost $O(\log\log n)$.*

Before we construct the simulation algorithm we show in Section 4 that the situation is much different in deterministic setting:

Theorem 2. *Leader election has energy cost $\Omega(\log n)$ on deterministic weak RN's.*

This result is interesting since on the other hand we present a practical algorithm with energy cost $\Theta(\sqrt{\log n})$ and time $O(n)$ (an asymptotically better solution and a double logarithmic lower bound for the strong model is presented in [6]).

4 Gap between weak and strong models

In this section we assume that the stations have distinct identifiers in the range $\{1, \ldots, n\}$, but the number of active stations might be arbitrary up to n.

4.1 Leader election: lower bound for energy cost in weak model

In this section we prove Theorem 2. We construct sets $A_0, A_1, \ldots$ such that for every i, if $I \subseteq A_i$ is the active set, $j \in I$ and $|A_i| > 1$, then station j cannot decide if it is the leader after step i. Moreover, we impose some other properties that guarantee that $|A_i| > 0$ and the equality $A_i = \{j\}$ implies that station j is awake at least $\Omega(\log n)$ times up to step i.

Let $A_0 = \{1,\ldots,n\}$. For $i > 0$ let S_i (R_i resp.) be a set of stations that send a message (listen, resp.) during step i, if the active set is A_{i-1}. For definition of A_i we use an auxiliary „weight" $w_i(j)$ of station j at step i. It fulfills the following invariant: $\sum_{j\in A_i} w_i(j) = n$ for $i \in \mathbb{N}$. We start with $w_0(j) = 1$ for $i = 1,\ldots,n$. We put $A_i = A_{i-1}\backslash R_i$ if $\sum_{j\in S_i} w_i(j) > \sum_{j\in R_i} w_i(j)$ and $A_i = A_{i-1}\backslash S_i$ otherwise. If $A_i = A_{i-1}\backslash R_i$, then

- $w_i(j) = w_{i-1}(j)$ for every $j \in A_i\backslash S_i$
- $w_i(j) = w_{i-1}(j) + v_j$ for every $j \in S_i$, where $v_j = \frac{w_{i-1}(j)}{\sum_{l\in S_i} w_{i-1}(l)} \cdot \sum_{l\in R_i} w_{i-1}(l)$.

If $A_i = A_{i-1}\backslash S_i$, then the weights are defined similarly. Obviously, the weights defined in such a way satisfy the invariant. Note that above rules guarantee that $w_i(j) \le 2w_{i-1}(j)$ for every i, j. Moreover, $w_i(j) > w_{i-1}(j)$ if and only if j is awake at step i (for active sets contained in A_i).

Proposition 1. *Let $j \in A_i$. Then station j does not hear any message up to step i for any active set $I \subseteq A_i$ such that $j \in I$.*

Proof. The proof is by induction on i. The case $i = 0$ is obvious. Assume that the property holds for $i-1$. If $A_i = A_{i-1}\backslash R_i$, then all awake stations from A_i are sending and nobody listens. So no information is reaching anybody. If $A_i = A_{i-1}\backslash S_i$, then all awake stations are listening, but nobody sends a message at step i, so no awake station of A_i gets any information. □ Proposition 1

Proposition 2. *If the active set equals $I \subseteq A_i$ and $j \in I$, then $w_i(j) \le 2^l$, where l is the number of steps among steps 1 through i during which station j was awake.*

Proof. We show it by induction with respect to i. The case $i = 0$ is obvious. For $i > 0$, it is enough to observe that the new factor added to the weight is smaller than the „old" weight and the weight may increase only when a station is awake. □ Proposition 2

By Proposition 1, if $|A_i| > 1$ and $j \in A_i$, then station j is unable to distinguish after i steps between active sets $\{j\}$ and A_i. Let l be the leader elected for active set A_i and $j \in A_i\backslash\{l\}$. Then station j cannot decide if it is the leader after step i (it becomes the leader if the active set is equal to $\{j\}$ and it is not the leader if the active set is A_i). We conclude that if $|A_i| > 1$, then the algorithm cannot terminate after step i. Since $|A_i| > 0$ for every $i > 0$, the algorithm can terminate after step i for the active set A_i if and only if $|A_i| = 1$. Let j be the only element of such A_i. Then $w_i(j) = n$ and by Proposition 2, station j is awake at least $\log n$ times during steps $1,\ldots,i$.

4.2 Sublogarithmic deterministic leader election in the strong model

In order to present the difference between the strong and weak RN's we construct now a sublogarithmic deterministic leader election algorithm for the strong RN.

Theorem 3. *There is a deterministic algorithm with energy cost $O(\sqrt{\log n})$ electing a leader in a RN consisting of n stations with unique identifiers 1 through n, where the subset of active stations might be arbitrary.*

Proof. Let us recall the algorithm with logarithmic energy cost from [10, 11], called henceforth *Basic Algorithm*: assume that $n = 2^h$ and consider a complete binary tree with n leaves. Let the ith leaf correspond to a station with label i. We assign unique labels $1,\ldots,2^h-1$ to the inner nodes of the tree, so that the label of a parent is bigger than the labels of its sons. Before step i, for each $j < i$ a leader is elected in the subtree with the root with label j. At step i the leader L

of the subtree rooted in the left son of node with label i sends a message and the leader R of the subtree rooted in the right son listens. If R can hear a message, it considers itself as non-leader of the subtree rooted at i. Otherwise it considers itself as a leader. Leader L, whenever exists, considers itself as the leader of the subtree rooted at i. It is easy to see that in this tricky way one leader is elected for the subtree rooted at i. We add an additional step during which L listens and R sends, so that the leader of the subtree not only knows that it is the leader, but also knows with whom it wins. This modification doubles execution time, but will be necessary for saving energy.

Our algorithm is related to Basic Algorithm. We consider a complete tree with n leaves, height $\sqrt{\log n}$, where each internal node has $2^{\sqrt{\log n}}$ sons. The leader in a subtree rooted at node U is found by electing it from the leaders of the subtrees rooted at the sons of U. The computation, called later *Election Procedure*, consists of two phases:

Phase 1: Each leader corresponding to the sons of U sends a message. Simultaneously, it listens. If its message came through, then it considers itself as the leader in the subtree rooted at U and skips Phase 2.

Phase 2: The leaders execute Basic Algorithm to elect the leader among the sons of U.

At Phase 1 energy cost is 1, while during Phase 2 it is $O(\sqrt{\log n})$. If we would sum up the second energy cost over $\sqrt{\log n}$ Election Procedures, no gain would be achieved. An additional feature of the algorithm is that each station may collect *slaves*, they are active stations which already resigned to compete for leadership, but may be used to perform some operations in behalf of their *master*. For this reason, at Phase 2 the leader which wins gets new slaves: a station which has lost against the leader and all its slaves.

The key point is how to use the slaves. If a leader g has slaves $s_1, s_2, \ldots$, then they are used during Phase 2 as follows: first for some number of steps, g is simulated by the first slave of g, say $s_1(g)$. Then, in an extra step, $s_1(g)$ sends information of its computation history and the second slave $s_2(g)$ listens. This procedure is repeated with consecutive slaves until at the end g takes over. In that way the energy cost of g is distributed among its slaves.

Now we determine for how many steps each station has to work during execution of Phase 2. Consider a single Election Procedure. Each participating station u is either a leader of a subtree or a slave of a leader. Each such u has a current capacity $\mathcal{C}(u)$. The invariant we have is that for each leader s participating in Election Procedure

$$\sum\{\mathcal{C}(u) \,|\, u = s \vee u \textit{ is a slave of } s\} = \sqrt{\log n}. \tag{1}$$

During Phase 2 each u is used for simulating $\mathcal{C}(u)$ steps of the respective leader.

Initially, $\mathcal{C}(u) = \sqrt{\log n}$. When Phase 2 is not executed during Election Procedure, then the capacities of the participating stations remain unchanged. If Phase 2 is executed, then we elect a leader s_1 and choose a leader s_2 that have lost against s_1. Then we halve the capacities of s_1, s_2 and their slaves. Now, the sum of capacities of s_1 and its slaves becomes $0.5\sqrt{\log n} + 0.5\sqrt{\log n} = \sqrt{\log n}$. The first term is responsible for capacities of s_1 and its old slaves, the second term to s_2 and its old slaves.

We see that each station is awake $O(\sqrt{\log n})$ times. Indeed, for Phases 1 it is awake $O(\sqrt{\log n})$ times. For Phases 2 it is awake $v = O(\sqrt{\log n})$ times (the first execution of Phase 2), then $v/2$ times (the second execution of Phase 2 but with reduced capacities), and so on. We see that energy cost occurred at Phases 2 sum up to at most $2v = O(\sqrt{\log n})$. □

5 Algorithmic tricks for weak RN's

Although energy cost of deterministic leader election is different for weak and strong RN's, in the randomized case the best algorithms can be run on weak RN's [6], using tricks presented below. Moreover, a linear approximation of the number of active stations may be obtained efficiently by randomized algorithm, using a method designed for the strong model. This indicates that randomization may really help. In this section we make some observations that help to design "strong algorithms" in weak RN's.

Previous RN algorithms use very often the following *basic trick* ([4, 11, 10, 13]): given n active stations, each of them sends with probability $1/n$.Then exactly one station sends with probability $n \cdot \frac{1}{n} \cdot \left(1-\frac{1}{n}\right)^{n-1} \approx \frac{1}{e}$. In the strong model, the station that succeeds may recognize it. This experiment can be used for randomized leader election in the strong model: If we repeat the experiment $\ln n$ times, then the probability that every time nobody succeeds is approximately $e^{-\ln n} = 1/n$. Then Basic Algorithm for leader election over those stations that have been successful during $\ln n$ experiments (with ID's equal to numbers of successive steps) provides us a leader chosen from the set of n active and anonymous stations.

Pairing trick in the weak model The method described above cannot be implemented directly in the weak model: its key point is that the station that sends a message can immediately recognize a collision. In the weak model it is impossible. However, this problem can be eluded in the following way:

1. each station chooses uniformly at random to be either *sender* or *receiver*,
2. each sender transmits a message with probability $\frac{2}{n}$,
3. each receiver listens with probability $\frac{2}{n}$ and if a message is received, it responds with the same message,
4. each sender that have sent a message at step 2 listens. If it receives its a message now, then it was the only station sending at step 2.

In the above procedure it may happen that a station sending a message is successful but it does not receive a confirmation for the simple reason that there is more than one receiver sending the confirmation. Nevertheless, we show that the probability that one station sends a message and receives a confirmation is bounded from below by a constant close to $\frac{1}{e^2}$. First observe that the probability that there are more than $n/2+n^{2/3}$ senders is at most $e^{n^{1/3}/6}$ by Chernoff bound (3). For deviation d, such that $|d| < n^{2/3}$ the probability of receiving a confirmation takes the form

$$\left(\left(\tfrac{n}{2}+d\right)\cdot\tfrac{2}{n}\cdot\left(1-\tfrac{2}{n}\right)^{n/2-1+d}\right)\cdot\left(\left(\tfrac{n}{2}-d\right)\cdot\tfrac{2}{n}\cdot\left(1-\tfrac{2}{n}\right)^{n/2-1-d}\right)$$
$$\approx \frac{1}{e^2}\cdot\left(1-\frac{d^2}{(n/2)^2}\right) > \frac{1}{e^2}\cdot\left(1-\frac{4}{n^{2/3}}\right)$$

Partial initialization Now we describe the second trick which is very useful in converting algorithm running in the strong model into algorithms running in the weak model without loosing energy efficiency. Assume that among n stations we have already a fraction n/d stations that have assigned unique ID numbers in the range $1..n/d$ - we call them the *first group*. Then we are able to run an algorithm for the strong model on the remaining $n-n/d$ stations, called the *second group*, in the following way: step i of the algorithm is simulated by the steps $2i-1$ and $2i$. At step $2i-1$ the processors of the second group execute step i of the algorithm. Additionally, the station $1+i \bmod \frac{n}{d}$ from the first group listens. At step $2i$ it sends the message received (if any). In turn, all stations of the second group that have been sending during step $2i-1$ are listening. This enables them to recognize a collision, if it occurred at step $2i-1$.

Observe that for simulating n steps of an algorithm for strong model each of the stations of the first group is awake for $2d$ steps. If d is small, for instance $\log\log n$, then this energy cost might be smaller than the energy cost occurring for the stations of the second group, and therefore acceptable.

6 Initialization

The best known algorithm for initialization [11] works in the strong model only: its key point is that the stations assign themselves random ID's in the range $1,\ldots,m$ for some $m \in \mathbb{N}$ and verify their uniqueness in such a way that stations with ID equal to j send at step j for $j = 1,\ldots,m$ and check for collisions. In the weak model it does not work this way. The natural way is to re-use pairing trick, i.e. split stations randomly into senders and receivers, then each sender sends a message and each receiver listens in the step equal to picked number and they confirm uniqueness as in the pairing trick. However, it does not work well, if there are few stations choosing ID's from a big set – but just this situation leads to double logarithmic energy cost in [11].

Let us outline our initialization algorithm. It consists of three main phases:

1. Assign temporary unique ID's from the set $\{1,\ldots,c_1 n\}$ to a subset U of stations such that $|U| \geq c_2 n$, for some constants c_1 and c_2.
2. Run „renumeration" procedure on the elements of U. This procedure assigns distinct consecutive ID's, starting with 1, to a large subset of U, say I.
3. Using the partial initialization trick run the initialization algorithm from [11] on the stations that do not belong to I.

Let us describe in detail the first phase:

(A) Split the set of stations into subsets S_0 and S_1. Each station chooses bit b uniformly at random. The stations that set $b = i$ belong to the set S_i.
(B) Each station picks an integer from the set $\{1,\ldots,3n\}$ uniformly at random. Let S_i^j be the set of elements of S_i which choose number j.
(C) `for` $j \leftarrow 1$ `to` n `do`
the stations from S_1^j and S_0^j verify if $|S_1^j| = |S_0^j| = 1$: first all elements of S_1^j send and all elements of S_0^j listen; in the next step all elements of S_1^j listen and each element of S_0^j sends the message just heard; if the message comes through, then $|S_1^j| = |S_0^j| = 1$ and the only element of S_1^j sets ID$\leftarrow j$.

Obviously, at most one station can set its *ID* to j for every $j > 0$.

Lemma 1. *With probability at least $1 - \frac{1}{n^2}$ the number of stations with assigned ID's is bigger than cn, for a constant c independent of n.*

Proof. As we have already seen $n/2 - n^{2/3} \leq |S_b| \leq n/2 + n^{2/3}$ for $b = 0,1$ with probability $1 - e^{-n^{1/3}/6}$. Thus assume that $|S_i| = c_i n$, for $i = 0,1$, where c_0, c_1 are very close to $1/2$, and $c_0 + c_1 = 1$. Let us consider the number of sets S_b^j such that $|S_b^j| = 1$ We call the stations from such S_b^j's successful. Imagine that we pick numbers for stations sequentially. If the jth station of S_b picks a number chosen already by the other station, then we say that a collision occurs. Let p_j be the probability that a collision occurs in the jth step. It depends very much on the previous choices, but every time $p_j \leq \frac{j-1}{3n}$. Let X be the number of collisions. By Lemma 2 from Appendix, we can bound (from above) the probability that $X > \alpha$ (for any α) by the probability that $Y = \sum_{i=1}^{c_b n} Y_i > \alpha$, where Y_i are independent random „0-1" variables such that $\mathbf{P}\left[Y_j = 1\right] = \frac{j}{3n}$. Observe that $\mathbf{E}[Y] = \sum_{j=1}^{c_b n} \frac{j}{3n} = \frac{c_b n(c_b n-1)}{2 \cdot 3n}$. Together with Chernoff Bound (3) this yields

$$\mathbf{P}\left[X > \frac{(c_b n)^2}{3n}\right] \leq \mathbf{P}\left[Y > \frac{(c_b n)^2}{3n}\right] \leq e^{-\frac{c_b n(c_b n-1)}{2\cdot 3\cdot 3n}} = O(n^{-4})$$

for n large enough. A collision may prevent at most two stations from being successful. Thus the number of stations that are not successful is at most $2X$. So the number of successful stations in S_b is at least $c_b n - 2X \geq c_b - \frac{c_b^2 n^2}{3n} = n(c_b - \frac{c_b^2}{3}) \geq c_b' n$ with high probability, where c_b' is a constant very close to $\frac{5}{12}$.

Now let us assume that the set of successful stations in S_b consists of $d_b n$ elements, where $d_b \geq \frac{1}{3}$. (We have just shown that this condition holds with probability $1 - O(n^{-4})$.) Fix $d_0 n$ numbers associated to successful stations from S_0 (each subset of $\{1, \ldots, 3n\}$ of size $d_0 n$ has equal probability). Now imagine that we assign numbers to successful stations from S_1 sequentially. The probability that the $(j+1)$st station is „paired" with a successful station from S_0 is not smaller than $\frac{d_0 n - j}{3n - j} \geq \frac{d_0 n - j}{3n}$. The expected number of „successful pairs" is not smaller than $\sum_{j=1}^{d} \frac{d_0 n - j}{3n} \geq \frac{1}{3n} \cdot \frac{dn(dn-1)}{2} \geq d' n$, where $d = \min(d_0, d_1)$ and d' is a constant. Finally, by Lemma 2 and Chernoff Bound (2), the number of „successful pairs" is bigger than $d'n/2$ with high probability, assuming that the sizes of S_0 and S_1 are close each other and the number of successful stations in them are bigger than $n/3$. But this assumptions are satisfied with high probability, too. ∎

Renumeration Now, we recall the following result from [6]: For no-CD RN there is a deterministic leader election algorithm with $\log^* n$ energy cost[1] and time $O(n)$ that works for an arbitrary set of stations with distinct identifiers in the range $1..n$ such that the number of active stations is at least $c \cdot n$ for any fixed constant $c > 0$. In fact this algorithm works in the weak model. Moreover, it consists of $O(\log^* n)$ phases and the following properties are satisfied in the phase i:

- At least $c_i x_i$ active stations called masters participate in the phase.
- A set of at least $\frac{c}{2} \cdot s_i$ active stations called slaves is assigned to each master. The slaves of the master are numbered by consecutive numbers $1, 2, \ldots$
- Only one master remains after the last phase.

We have $x_1 = n$, $s_1 = 2/c$, $c_1 = c$ and $c_{i+1} = c_i/2$, $x_{i+1} = x_i/2^{s_i}$, $s_{i+1} = c_{i+1} \cdot s_i \cdot 2^{s_i}$ for $i > 1$. So, the number of masters and their slaves at phase i is not smaller than $\frac{c}{2} \cdot s_i c_i x_i$. Observe that $c_i = \frac{c}{2^{i-1}}$, $\frac{c}{2} \cdot c_1 x_1 s_1 = cn$, and $c_{i+1} x_{i+1} s_{i+1} = \frac{c_i}{2} \cdot \frac{x_i}{2^{s_i}} \cdot \frac{c_i}{2} s_i 2^{s_i} = \frac{c}{2^{i+1}} c_i x_i s_i$. Thus $c_{i+1} x_{i+1} s_{i+1} = (c_1 x_1 s_1) \cdot \frac{c}{2^3} \cdot \frac{c}{2^4} \cdot \ldots \cdot \frac{c}{2^{i+1}} = cn \cdot \frac{c^{i-1}}{2^{(i+1)i/2}} \geq (c')^{i^2} n$ for a constant c'. So, for $i = \Theta(\log^* n)$, there are at least $(c')^{\Theta((\log^* n)^2)} n = \Omega\left(\frac{n}{\log^{\langle f \rangle} n}\right)$ stations that participate in phase i, for every $f \in \mathbb{N}$. It means that the algorithm initializes, i.e. labels with consecutive different integers $1, 2, \ldots$, $\Omega\left(\frac{n}{\log\log n}\right)$ stations.

[1] Recall that $\log^* n = o(\log^{\langle s \rangle} n)$ for every $s \in \mathbb{N}$, where $\log^{\langle 1 \rangle} n = \log n$ and $\log^{\langle k+1 \rangle} n = \log(\log^{\langle k \rangle} n)$ for $k > 1$.

It follows from Lemma 1 that at least a linear fraction of stations are assigned in Phase 1 unique ID's from the set $1,\ldots,3n$. In Phase 2 we run the algorithm just described and so get a set of stations I labeled uniquely by consecutive numbers $1,2,\ldots,|I|$ such that $|I|=\Omega(\frac{n}{\log\log n})$.

Phase 3 Paper [11] presents an algorithm for the strong RN that initializes n stations in time $O(n)$ with energy cost $O(\log\log n)$ and probability at least $1-1/n$. It can be easily applied to the case when the approximation of the number of stations up to a constant multiplicative factor is given. Let I be the set of stations that obtained consecutive ID's during renumeration. $|I|=\Omega(\frac{n}{\log\log n})$ with high probability. In Phase 3 we simulate the initialization algorithm from [11] using partial initialization trick (the elements of I with ID's $1,\ldots,\min(|I|,n/2)$ play a role of the *first group* and all other elements belong to the *second group*).

Recall that the initialization algorithm [11] works in time $O(n)$ and has energy cost $O(\log\log n)$. Thus, one can easily check that our algorithm for the weak model works in time $O(n)$ and energy $O(\log\log n)$, too.

7 General simulation

An algorithm for the strong RN can be simulated on a weak RN in the following way: first we run initialization procedure as described in Section 6. Then we run the algorithm based on the partial initialization trick from Section 5: the stations $1,\ldots,n/2$ are responsible for simulating the stations of the strong RN, station i simulating stations $2i-1$ and $2i$, the stations $n/2+1,\ldots,n$ are used for the trick described in Section 5. Simulation of a step j of the strong RN looks as follows: a station simulating stations $2i-1$ and $2i$ of the strong RN sends whatever these stations send, except for the situation when both these stations send – then no message need to be sent by the simulating station. Station $n/2+j$ listens at this step. At the next step it sends the message received and these stations among $1,\ldots,n/2$ listen, which represent the awake stations of the strong RN.

References

1. Y. Azar, A.Z. Broder, A.R. Karlin, U. Upfal, *Balanced Allocations*, SIAM J. Comput. 29(1): 180-200 (1999).
2. J.L. Bordim, J. Cui, T. Hayashi, K. Nakano, S. Olariu, *Energy efficient initialization protocols for ad-hoc radio networks*, ISAAC'99, LNCS 1741, Springer-Verlag, 215–224.
3. W.C. Fifer, F.J. Bruno, *Low Cost Packet Radio*, Proc. of the IEEE, 75 (1987), 33–42.
4. L. Gąsieniec, A. Pelc, D. Peleg, *The wakeup problem in synchronous broadcast systems*, PODC 2000, 113–121. Journal version: SIAM Journal on Computing 14(2) , 207-222 (2001).
5. T. Jurdziński, M. Kutyłowski, J. Zatopiański, *Energy-Efficient Size Approximation for Radio Networks with no Collision Detection*, Technical Report, Technische Universität Chemnitz, Fakultät für Informatik.
6. T. Jurdziński, M. Kutyłowski, J. Zatopiański, *Efficient Algorithms for Leader Election in Radio Networks*, Technical Report, Technische Universität Chemnitz, Fakultät für Informatik.
7. E. Kushilevitz, Y. Mansour, *Computation in Noisy Radio Networks*, Proc. ACM-SIAM SODA '98, 236–243.
8. K. Nakano, S. Olariu, *Randomized $O(\log\log n)$-round leader election protocols in radio networks*, ISAAC'98, LNCS 1533, Springer-Verlag, 209-218.

9. K. Nakano, S. Olariu, *Randomized leader election protocols for ad-hoc networks*, SIROCCO'2000, Carleton Scientific, 253-267.
10. K. Nakano, S. Olariu, *Randomized Leader Election Protocols in Radio Networks with No Collision Detection.* ISAAC'2000, LNCS 1969, Springer-Verlag, 362–373.
11. K. Nakano, S. Olariu, *Energy Efficient Initialization Protocols for Radio Networks with no Collision Detection*, ICPP'2000, IEEE, 263–270.
12. S. Rajasekaran, J. H. Reif, J. D. P. Rolim, (Eds.), *Handbook on Randomized Computing*, Kluwer Academic Publishers, 2001.
13. D.E. Willard, *Log-logarithmic selection resolution protocols in multiple access channel*, SIAM Journal on Computing 15 (1986) , 468-477.
14. IEEE Standard for Information Technology - LAN/MAN: Wireless LAN Medium Access Control (MAC), *http://grouper.ieee.org/groups/802/11/main.html*.

Appendix

Chernoff Bounds Let $X_1, X_2, \ldots, X_n$ be independent Poisson trials with $Pr[X_i = 1] = p_i$, let $X = \sum_{i=1}^{n} X_i$. Then

$$Pr[X < (1-\delta)EX] < e^{-EX\,\delta^2/2} \quad \forall \delta \in (0,1] \tag{2}$$

$$Pr[X > (1+\delta)EX] < e^{-EX\,\delta^2/3} \quad \forall \delta \in [0,1] \tag{3}$$

Sums of dependent versus independent random variables Here, we formulate technical lemmas that are the folklore. They enable to relate probabilities of deviations from the expectations of the sums of dependent variables by the appropriate probabilities for independent variables. (A very similar lemma is formulated in [1].)

Lemma 2. *Let $X_0, X_1, \ldots, X_n$ be independent random variables; for every $i \geq 1$, $X_i \in \{0,1\}$, $\boldsymbol{P}[X_i = 1] = p_i$. ($X_0$ may have arbitrary values)*

Let $Y_0, Y_1, \ldots, Y_n$ be (dependent) random variables, for every $i \geq 1$, $Y_i \in \{0,1\}$, and Y_0 may have arbitrary values. Assume that for every $i \geq 1$, for every v_0 in the range of Y_0 and $v_1, \ldots, v_{i-1} \in \{0,1\}$

$$\boldsymbol{P}[Y_i = 1 \,|\, (Y_0 = v_0, \ldots, Y_{i-1} = v_{i-1})] \;\leq\; p_i.$$

Then for each $c \geq 0$ and v_0 in the range of Y_0:

$$\boldsymbol{P}\left[\sum_{i\geq 1} Y_i \geq c \mid Y_0 = v_0\right] \;\leq\; \boldsymbol{P}\left[\sum_{i\geq 1} X_i \geq c\right].$$

Now, assume that for every $i \geq 1$, for every v_0 in the range of Y_0 and $v_1, \ldots, v_{i-1} \in \{0,1\}$

$$\boldsymbol{P}[Y_i = 1 \mid (Y_0 = v_0, \ldots, Y_{i-1} = v_{i-1})] \;\geq\; p_i.$$

Then for $c \geq 0$ and v_0 in the range of Y_0

$$\boldsymbol{P}\left[\sum_{i\geq 1} Y_i \geq c \mid Y_0 = v_0\right] \geq \boldsymbol{P}\left[\sum_{i\geq 1} X_i \geq c\right]$$