Random Geometric Graphs and the Initialization Problem for Wireless Networks

Christian Lavault, Stephan Olariu, Vlady Ravelomanana

Random Geometric Graphs and the Initialization Problem for Wireless Networks

Christian LAVAULT [a] Stefan OLARIU [b] and Vlady RAVELOMANANA [c],
[a] *LIPN – UMR CNRS 7030, Institut Galilée, Université Paris 13*
99, Avenue Clément 93430 Villetaneuse, France; E-mail: lavault@lipn.univ-paris13.fr.
[b] *Department of Computer Science, Old Dominion University*
Norfolk VA 23529-0162, U.S.A; E-mail:olariu@cs.odu.edu.
[c] *LIPN – UMR CNRS 7030, Institut Galilée; E-mail: vlad@lipn.univ-paris13.fr.*

Abstract. The initialization problem, also known as naming, assigns one unique identifier (ranging from 1 to n) to a set of n indistinguishable nodes (stations or processors) in a given wireless network $\mathcal{N}$. $\mathcal{N}$ is composed of n nodes randomly deployed within a square (or a cube) X. We assume the time to be slotted and $\mathcal{N}$ to be synchronous; two nodes are able to communicate if they are within a distance at most r of each other (r is the transmitting/receiving range). Moreover, if two or more neighbors of a processor u transmit concurrently at the same round, u does not receive either messages. After the analysis of various critical transmitting/sensing ranges for connectivity and coverage of randomly deployed sensor networks, we design sub-linear randomized initialization and gossip algorithms achieving $\mathcal{O}\left(n^{1/2}\log(n)^{1/2}\right)$ and $\mathcal{O}\left(n^{1/3}\log(n)^{2/3}\right)$ rounds in the two-dimensional and the three-dimensional cases, respectively. Next, we propose energy-efficient initialization and gossip algorithms running in $\mathcal{O}\left(n^{3/4}\log(n)^{1/4}\right)$ rounds, with no station being awake for more than $\mathcal{O}\left(n^{1/4}\log(n)^{3/4}\right)$ rounds.

Keywords. Coverage, connectivity; hop-diameter; minimum/maximum degrees; transmitting/sensing ranges; analytical methods; energy consumption; topology control; randomized distributed algorithms; fundamental limits of random radio networks.

1. Introduction

Distributed, multi-hop wireless networks, such as ad hoc networks, sensor networks or radio networks, are gaining in importance as subject of research, with very many practical real-life applications [32]. In the paper, wireless networks are a collection of transmitter-receiver devices, referred to as *nodes stations* or *processors*, according to the context.

A wireless network $\mathcal{N}$ consists in a group of nodes that can communicate with each other over a wireless channel. The nodes (or processors) of $\mathcal{N}$ come without ready-made links and without any centralized controller. $\mathcal{N}$ can be modeled by its *reachability graph* G, within which the existence of a directed edge $u \to v$ means that v can be reached from

u. If all transmitters/receivers have the same power, this underlying graph G is symmetric. As opposed to traditional networks, wireless networks are often composed of a number of nodes that can be several orders of magnitude higher than the size of conventional networks [2]. Sensor nodes are often deployed inside a medium. Therefore, the positions of these nodes need not be engineered or pre-determined. This allows random and rapid deployment in inaccessible terrains and suit well the specific needs to disaster-relief, law enforcement, collaborative computing and other special purpose applications.

As customary [3,4,5,9,17,26,27] the time is assumed to be slotted and nodes (processors) can send messages in synchronous *rounds* (or *time slots*). In each round, every node can act either as a *transmitter* or as *receiver*. A node u acting as receiver in a given round gets a message, if and only if, exactly one of its neighbors is transmitting within the same round. If more than two neighbors of u are transmitting simultaneously, u receives nothing. More precisely, such a network $\mathcal{N}$ has no ability to distinguish between the absence of message and at least one collision or conflict. This assumption is motivated by the fact that, in many real-life situations, the (tiny) devices used do not always have the collision detection ability. Moreover, even if such detection mechanism were present, it should be of limited value; especially in the presence of some noisy channels. Therefore, it is highly desirable to design algorithms that work independently of the existence/absence of any collision detection mechanisms.

We consider that the n nodes of $\mathcal{N}$ are initially *homogeneously scattered* in a square X of size $|X|$ (or in a cube X of volume $|X|$). As in several applications, the users of $\mathcal{N}$ can move, and therefore the topology is unstable. For this reason, we wish the algorithms to refrain from assumptions about the topology of $\mathcal{N}$ or about initial information that processors may have concerning the topology. In the present paper, we assume that no processors has any topological knowledge, except the measure (surface or volume) $|X|$ of X, where they are randomly dropped. Besides, observe that even if $|X|$ is known exactly while n is not (viz. exactly, or up to its order of magnitude: $n = \mathcal{O}(|X|)$), an equation such as Eq. (6) in Theorem 2 (see below) allows to handle subtle changes involved between $\mathcal{O}(n)$ and $\mathcal{O}(|X|)$ and occurring in the constants hidden in the "big-Ohs". Moreover, these assumptions are strengthened by the fact that during their deployment some nodes can be faulty with unknown probability.

Methods to achieve *self-configuration* and/or *self-organization* of networking devices appear to be amongst the most important challenges in wireless computing [2]. *Initialization* is part of these methods: before networking, each node must have a *unique* identity (identifier or address) denoted *ID*. A mechanism that allows $\mathcal{N}$ to create a unique identity (ID) automatically for each of its participating nodes is an *address self-configuration* algorithm. In the present paper, nodes are initially *indistinguishable*. This assumption arises naturally, since it may be difficult or impossible to get interface serial numbers while on missions (see also [17,26,27]). Thus, the IDs of such self-configuration algorithms must not rely on the existence of serial numbers.

The problem addressed here is to design and analyze a *fully distributed algorithms* for the initialization problem. As far as we know, the initialization problem was first handled in the seminal papers of Hayashi, Nakano and Olariu [17,26,27] for the case when G is complete. (For the sake of simplicity, we write $\mathcal{N}$ (a wireless network) for G (its underlying reachability graph) when appropriate.)

Note that the transmitting range of each station can be set to some value r ranging from 0 to r_{MAX}. Such a model is commonly used in mobile computing and radio network-

ing [7,19,33]. It is frequently encountered in many domains, from statistical physics to epidemiology (see e.g., [16] for the theory of coverage processes or [23] for percolative ingredients). The random graphs generated in such a way have been first considered in the seminal paper of Gilbert [14] (almost simultaneously, Erdös and Rényi considered the well-known $\mathbb{G}(n,p)$ model [12]). The analysis of many properties of the so called random geometric graphs model $\mathbb{G}(n,r)$, such as connectivity and coverage, have been the subject of intense studies [15,24,28,29,30,31].

Fig. 1 shows devices randomly deployed on some field. The depicted examples suggest that transmission ranges can play a crucial role when setting protocols at least for randomly distributed nodes (stations). Other parameters of importance are the number n of active nodes, the shape of the area X where stations are scattered and the nature of the communications to be established.

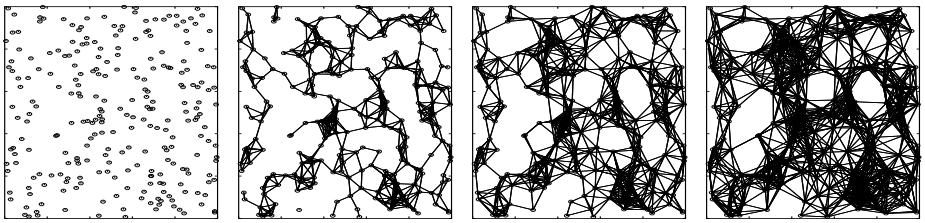


Figure 1. A typical radio network is generated according to the uniform distribution of coordinates of the devices. The transmission ranges of stations are gradually increasing from left to right. The last two pictures show that if the graph obtained has more edges than needed, the number of colliding packets is more difficult to control.

Considering the above observations, the design of efficient algorithms requires to take into account and to exploit the structural properties of $\mathcal{N}$. In our scenario, since none of the nodes knows the number n of stations in $\mathcal{N}$, our first task is to find distributed algorithms that allow a probabilistic counting of these nodes. Then, by setting the transmitting range parameter correctly, $\mathcal{N}$ can be self-initialized with high probability[1]. This is achieved in $\mathcal{O}\left(n^{1/2}\log(n)^{1/2}\right)$ rounds in the two-dimensional case and in $\mathcal{O}\left(n^{1/3}\log(n)^{2/3}\right)$ rounds in the three-dimensional case[2]. As far as we know, this is the first analysis of multi-hop initialization protocols (single-hop protocols are treated in [17,26,27,33]). Our algorithms are shown to take advantage of the fundamental characteristics of $\mathcal{N}$. Such limits are computed with the help of fully distributed algorithms: once known, an initialization algorithm is run to assign each of the n stations (nodes) one distinct ID ranging from 1 to n. Whenever all IDs are assigned, and even though the algorithm is probabilistic, one can check *deterministically* whether each ID is unique (if needed). For the purpose, deterministic linear algorithms (for example) can be used, such as the gossip protocol for symmetric networks in [22, Section 5].

Under the conditions described above, Figures 2 and 3 summarize briefly the input and output of the distributed initialization algorithms presented.

[1] Throughout the paper, an event $\mathcal{E}_n$ is said to occur *asymptotically almost surely* if, and only if, the probability $\mathbb{P}(\mathcal{E}_n)$ tends to 1 as $n \to \infty$. We also say $\mathcal{E}_n$ occurs *with high probability* (w.h.p. for short).

[2] In this paper, $\log$ denotes the natural logarithm.

INPUT:

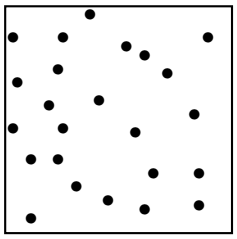


Figure 2. n indistinguishable processors randomly placed in the square X. The only knowledge required is the size $|X|$ of the support.

OUTPUT:

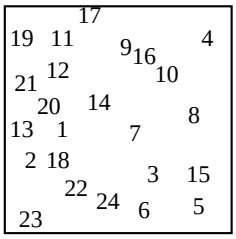


Figure 3. Each of the n processors (stations) is assigned a unique ID ranging from 1 to n. The IDs can serve as IP address (here $n = 24$).

In order to implement the initialization problem, we use a gossip algorithm. Gossiping and broadcasting [8] are fundamental techniques for spreading out information, and they represent naturally the most extensive studied problems in radio networks (see for instance [9,22] and references therein). In the gossiping problem, every station is initially given one distinct message that needs to be sent to all other ones. Under the same assumptions as above, we design a randomized gossiping algorithm that performs its task w.h.p. in $\mathcal{O}\left(n^{1/2}\log(n)^{1/2}\right)$ and $\mathcal{O}\left(n^{1/3}\log(n)^{2/3}\right)$ rounds in the two and three dimension cases, respectively.

Finally, it is shown that both *sub-linear* algorithms (gossiping and initialization) are *asymptotically optimal*, since they achieve (w.h.p.) $\mathcal{O}(D\log n) = \mathcal{O}(D\Delta)$ rounds, where Δ is the maximum degree of $\mathcal{N}$ and D its hop-diameter.

Outline of the paper. The paper is organized as follows. Section 2 presents a randomized distributed algorithm SEND for sending information in our settings. Next, this algorithm is analyzed. In Section 3, we discuss how to set correctly the transmission range of the nodes. Section 3 also provides results on the relationship between the transmission range r, the number of active nodes n, the size of X, the maximum degree Δ and the hop-diameter D of $\mathcal{N}$. These results and the use of the procedure SEND allow to build a broadcasting protocol BROADCAST. The Section ends with the design and analysis of a protocol named SFR (*Search-For-Range*), which serves to find the appropriate transmission range distributively. More precisely, by varying the transmission range, the protocol SFR broadly provides orders of magnitude of the characteristics of $\mathcal{N}$. Section 4 presents the randomized gossiping algorithm specifically intended for random wireless networks. This Section is organized as follows: we first present a randomized algorithm that colors the nodes of $\mathcal{N}$ in such a way that every pair of processors (u, v) within a distance of at most two hops from each other is assigned two distinct colors. Though "greedy", the latter algorithm is shown to color the graph in polylogarithmic rounds (depending on n) using $\mathcal{O}(\Delta) = \mathcal{O}(\log n)$ colors. This efficient coloring algorithm treats the direct and

hidden terminal problems. Once it is obtained, the 2-hop coloration leads to a natural scheduling of the communications to gossip in $\mathcal{O}(D\Delta)$ rounds. In turns, the gossiping algorithm is used to initialize $\mathcal{N}$. This is easily done by means of a simple ranking argument. Section 4 ends with the proofs of correctness and optimality of both algorithms (gossiping and initialization protocols). Finally, Section 5 presents a second initialization algorithm based on an energy-efficient gossip algorithm.

2. Basic protocols for sending information

First, no deterministic algorithm can work correctly in wireless networks when processors are anonymous. This is easily checked: conflict between two indistinguishable nodes can not be solved deterministically. Therefore, this impossibility result implies the use of randomness (see [5]). Since processors do not have identifiers (IDs), the first task is to design a basic protocol for the nodes which compete locally to access the unique channel of communication in order to send a given message. This can be achieved by organizing a flipping coin game between them. Recall also that if the transmission/receiving range is set to a value r, only neighbors within distance less than r are able to communicate in the absence of conflicts. In [30], Penrose proves that there exists a common radius of transmission to achieve the connectivity of the reachability graph.

In the very simple following procedure this parameter as well as the duration T of the trials must be taken into account.

Procedure SEND(msg,T,r)
For i from 0 to T **do**
With probability $1/2^i$ send msg to every neighbor ($\star$ to all processors at distance $\leq r$ $\star$)
end

Note that r is a parameter which can be tuned to a precise value. Again, it is clear that only neighbors within a distance of at most r can receive the message when there is no conflict. Therefore, we have the following definition.

Definition 1 *Given a transmission radius r and a set of n nodes uniformly and independently scattered in a square X of size $|X| = \mathcal{O}(n)$, a random graph is defined by adding edge between any pair of nodes (x, y), such that the Euclidean distance between x and y is less than or equals to r. Denote by r_{CON} the transmission range required to have a connected graph. For a fixed radius of transmission r, let d_v (depending on r, i.e. $d_v \equiv d_v(r)$) be the degree of any given node v.*

Theorem 1 *Let $r \geq r_{\text{CON}}$ be the current transmission range of the processors. Suppose that each of the d_v neighbors of v starts the execution of* SEND(msg, T, r) *in the same round. Let $\mathbb{P}(T, d_v)$ be the probability that v receives the message msg at least once between the time $t = 0$ and the time $t = T$.*
Then, there exists a function $f(T, d_v) = \mathcal{O}(d_v/2^T) + \mathcal{O}(1/\sqrt{d_v})$ such that $\mathbb{P}(T, d_v)$ satisfies

$$.8111 + f(T, d_v) \ \leq \ \mathbb{P}(T, d_v) \ \leq \ .8113 + f(T, d_v). \tag{1}$$

Proof. The assumption that $\mathcal{N}$ is connected ensures that, for any node v, the degree of v is such that $d_v > 0$.
We have $\mathbb{P}(T, d_v) \;=\; 1 \;-\; \prod_{i=0}^{T} \left(1 - \binom{d_v}{1}/2^i \left(1 - 1/2^i\right)^{d_v}\right)$, since only one of the v neighbors can succeed: v and all other $d_v - 1$ nodes are kept silent. For any given i_1 and for all $i \geq i_1$, $\left(1 - d/2^i \left(1 - 1/2^i\right)^d\right) \;\leq\; \left(1 - d/2^i \, \exp\left(-d/2^i \, (1 + 1/2^{i_1})\right)\right)$. So, if $2^t \gg d$, by choosing $i_1 = \lceil 1/2 \log_2 d \rceil$, we obtain after a bit of standard algebra

$$1 - \mathbb{P}(t, d) \;\leq\; \exp\left(-\sum_{m \geq 1} \frac{1}{m} \sum_{i=i_1}^{t} \frac{d^m}{2^{im}} \exp\left(-\frac{dm}{2^i}\left(1 + \mathcal{O}\left(\frac{1}{\sqrt{d}}\right)\right)\right)\right). \quad (2)$$

Now, by Mellin transform asymptotics methods (see [13] and [20, p. 131]), for any $m \geq 1$,

$$\left|\sum_{i=i_1}^{t} \frac{d^m}{2^{im}} \exp\left(-\frac{dm}{2^i}\left(1 + \mathcal{O}\left(\frac{1}{\sqrt{d}}\right)\right)\right) \;-\; \frac{m!}{m^{m+1} \log 2}\right| \;\leq\; \frac{10^{-5}}{m^m \log 2} + \mathcal{O}\left(\frac{1}{\sqrt{d}}\right) \;+\; \mathcal{O}\left(\frac{d^m}{2^{tm}}\right), \quad (3)$$

where the 10^{-5} term is due to small fluctuations: the amplitude of the tiny coefficients of the Fourier series occurring in Mellin transform asymptotics [13].

Next, since $m!/m^{m+2} \leq e^{-m}/m$ when $m \geq 7$,

$$\sum_{m=1}^{6} \frac{m!}{m^{m+2}} \;\leq\; \sum_{m=1}^{\infty} \frac{m!}{m^{m+2}} \;\leq\; \sum_{m=1}^{6} \frac{m!}{m^{m+2}} \;+\; \sum_{m=7}^{\infty} \frac{e^{-m}}{m}$$

and

$$\sum_{m=7}^{\infty} \frac{e^{-m}}{m} \;=\; -\frac{1}{60e^6}\left(60\,e^6 \ln\left(1 - 1/e\right) + 60\,e^5 + 30\,e^4 + 20\,e^3 + 15\,e^2 + 12\,e + 10\right),$$

we derive

$$.18\,869\ldots \leq\; \exp\left(-\sum_{m=1}^{\infty} \frac{m!}{m^{m+2} \ln 2}\right) \;\leq\; .18\,879\ldots \quad (4)$$

Similarly, for any $x \in [0, 1]$ and $d \geq 1$, $(1 - x)^d \leq e^{-dx}$.
Therefore, $\left(1 - d/2^i \, (1 - \frac{1}{2^i})^d\right) \;\geq\; 1 \;-\; d/2^i \, \exp\left(-d/2^i\right)$, and this time we get

$$\exp\left(-\sum_{m \geq 1} \frac{1}{m} \sum_{i=i_1}^{t} \frac{d^m}{2^{im}} \exp\left(-\frac{dm}{2^i}\right)\right) \;\leq\; 1 - \mathbb{P}(t, d). \quad (5)$$

Using the latter inequality yields Eq. (1) after computations similar to Eqs. (3) and (4). ■

In [5], Bar-Yehuda *et al.* designed a randomized procedure called DECAY to send information with probability of success larger than $\frac{1}{2}$ (see for instance [5, p. 108-109]).

In our procedure SEND, the proof of Theorem 1 (see also [13]) shows that, by changing the basis of the coin flipping game, viz. by substituting the probability $1/a^i$ for $1/2^i$ in the algorithm for any constant $a > 1$, the probability of success of the T trials can be made arbitrary close to 1 (also with a logarithmic number of rounds such that $a^T \gg d$).

In the next Section, we turn to the problem of finding suitable values of transmission range whenever the only *a priori* knowledge of processors is $|X|$.

3. Transmission ranges and characteristics of $\mathcal{N}$

The aim of this Section is to provide randomized distributed algorithms that allow the stations of $\mathcal{N}$ to find the required transmission range to achieve at least connectivity of $\mathcal{N}$. To this end, we need to know the relationships between the transmission range r, the number of processors n and the measure $|X|$ of the support. Other characteristics of interest, such as the minimum and maximum degree δ and Δ (resp.) and the hop-diameter D of $\mathcal{N}$, are also fundamental for setting wireless algorithms (see [5]). Moreover, the limits of the randomly generated network $\mathcal{N}$ help when designing such algorithms. We refer here to [14,15,24,31,36,37] for works related to random networks. Two distinct problems are addressed in this Section.

- The first one (Subsection 3.1) concerns the characteristics of the reachability graph G in the superconnectivity regime, i.e. when the radius of transmission of the stations grows much faster than the one required to achieve connectivity of G.
- Subsection 3.2 is devoted to the design and analysis of a distributed protocol SFR, that will allow the nodes to approximate the aforementioned characteristics.

3.1. Fundamental limits of a random graphs in the superconnectivity regime

Following Miles's model [24], a great number n of devices are dropped in some area X. As $n \to \infty$ with $n = \mathcal{O}(|X|)$, the graph generated by the transmitting devices can be well approximated with a Poisson point process (see e.g. [16]). First of all, its extreme *independence* property allows penetrating analysis. Next, Poisson processes remain *invariant* if their points are independently translated (translations being identically distributed with some bivariate distribution: direction and distance). So, the results may take their importance for *moving stations* and therefore, they are well suited to randomly deployed mobile devices. Last, if with probability p such that $p\,n = \mathcal{O}(|X|)$, some nodes are *faulty* or intentionally *asleep* (e.g. for saving batteries in energy-efficient algorithms [27]), our results remain valid. This is due to Poisson processes properties and in the latter scenario, the number of nodes n is simply replaced by $n' = p\,n$.

Among other results, Penrose [30] proved that if $n/|X| = \mathcal{O}(1)$ and X is a two dimensional area, then

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{n}{|X|}\,\pi\,r_{\text{CON}}^2 \;-\; \log(n) \;\leq\; \omega\right) \;=\; \exp\left(-e^{-\omega}\right), \qquad \omega \in \mathbb{R}.$$

Penrose's result asserts that, by letting the radius of transmission range grow as

$$r \;=\; \sqrt{\frac{\log n + \omega(n)}{\pi n}\,|X|}$$

for any arbitrary function $\omega(n)$ tending to infinity with n, the graph obtained is a.a.s. connected.

For our purpose, we need the following results related to the degrees of the nodes according to the successive orders of magnitude of transmission range values.

Theorem 2 *Let r denote the transmission range of the n nodes randomly distributed in the square X of size $|X| = \mathcal{O}(n)$. Then, in the following three regimes, the graph G is connected with high probability:*

(i) For fixed values of k, that is $k = \mathcal{O}(1)$, if $\pi r^2\, n/|X| = \log n + k \log\log n + \omega(n)$, then G has a.a.s. a minimum degree $\delta = k$.

(ii) Let $k \equiv k(n)$ and $1 \ll k \ll \log n/\log\log n$.
If $\pi r^2\, n/|X| \;=\; \log n \,+\, k(n)\, \log\log n$, then the minimum and the maximum degree are a.a.s. $\delta = k(n)$ and $\Delta = e\, \log n$, respectively.

(iii) If $\pi r^2\, n/|X| = (1+\ell)\log n$ with $\ell > 0$, then each node v of G has a.a.s. d_v neighbors with

$$-\,\frac{\ell\,\log n}{W_{-1}\left(-\frac{\ell}{e\,(1+\ell)}\right)} + o(\log n) \;\leq\; d_v \;\leq\; -\,\frac{\ell\,\log n}{W_0\left(-\frac{\ell}{e\,(1+\ell)}\right)} + o\left(\log n\right), \quad (6)$$

where W_{-1} and W_0 denote the two branches of the Lambert W function[3] which are detailed in [10]. Moreover, in the case when $\pi r^2\, n/|X| = (1+\ell)\log n$ with $\ell > 0$, each geographical point of the support X is also recovered by $\Theta(\log n)$ disks of transmission.

Sketchproof. For the proof of Theorem 2, we refer to [33], where asymptotic coverage as well as connectivity properties are treated in details for the ranges of transmission considered in Theorem 2.
Observe that in the 3-dimensional case (with a cube instead of a square), similar results hold with the same assumptions as in Theorem 2: every occurrence of the surface (πr^2) being replaced by the volume ($4/3\,\pi r^3$). For example, to have each point of the cube recovered by $\Theta(\log n)$ balls, it is sufficient to set the transmission radius to the value $r \;=\; \sqrt[3]{3\,(1+\ell)\log n\,|X|/4\pi n}$. In this case, w.h.p. the degree d_v of each node v also satisfies Eq. (6).

In the remainder of the paper, we mainly concentrate our attention on results related to the 2-dimensional case, since there exist direct correspondences with the 3-dimensional case, such as the one mentioned above.

Next, we derive an upper-bound of the hop-diameter D in the superconnectivity regime.

[3]The Lambert W function is usually considered as a "special function" and its computation has been implemented in mathematical softwares such as Maple.

Theorem 3 *Let $D \equiv D(r)$ be the hop-diameter of G. Suppose that the transmission range meets the condition $r = \sqrt{3\,(1+\ell)\log n\,|X|/4\pi n}$, with $\ell > 0$. Then*

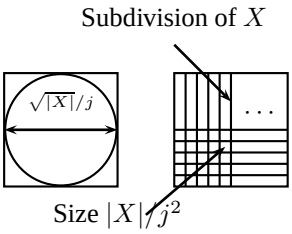
(i) If $\ell > \frac{4-\pi}{\pi-2}$,

$$\lim_{n\to\infty} \mathbb{P}\left(D \leq 3\sqrt{\frac{\pi\, n}{(1+\ell)\,\log n}} + \mathcal{O}(1)\right) \,=\, 1. \tag{7}$$

(ii) If $\ell \leq \frac{4-\pi}{\pi-2}$,

$$\lim_{n\to\infty} \mathbb{P}\left(D \leq 5\sqrt{\frac{\pi\, n}{(1+\ell)\,\log n}} + \mathcal{O}(1)\right) \,=\, 1. \tag{8}$$

Proof. Split the square X into j^2 equal subsquares $S_1, S_2, \ldots, S_{j^2}$. Each of the subsquares has a side $\sqrt{|X|}/j$ and an area $|X|/j^2$. Choose j such that each subsquare S_i can entirely contain a disk of radius r as depicted below.



$\sqrt{|X|}/2j = r = \sqrt{(1+\ell)\log n\,|X|/\pi n}$. So, $j = 1/2\,\sqrt{\pi n\,1/(1+\ell)\log n}$. For the sake of simplicity but w.l.o.g., assume j to be a non negative integer. By Theorem 2 (property *(iii)*), there are $\Theta(\log n)$ nodes inside the disk with high probability.

Any pair of nodes inside the same disk needs *at most* 2 hops to get connected, since they are within a distance of at most $2r$ and since each subgraph inside such a disk is a.a.s. connected.

Lemma 4 *Communications between two adjacent subsquares S_1 and S_2, viz. between any node $a \in S_1$ and any node $b \in S_2$, need at most (w.h.p.)*

a) 6 *hops when $\ell > \frac{4-\pi}{\pi-2} = 0.7519\ldots$ and*
b) 10 *hops when $\ell \leq \frac{4-\pi}{\pi-2}$.*

Proof. Consider adjacent subsquares as depicted in Figs. 4, 5 and 6.
A bit of trigonometry shows that each lens-shaped region such as L_1 (in Fig. 4) has a surface $|L_1| = 1/6\,(4\pi - 3\sqrt{3})r^2$.
L_1 represents the intersection of two disks of equal radius r whose centers are at distance r. Therefore, there is no node inside the lens-shaped region L_1 with probability

$$\begin{aligned}\left(1 - \frac{|L_1|}{|X|}\right)^n &= \left(1 - \frac{1}{6}(4\pi - 3\sqrt{3})\frac{(1+\ell)\log n}{n}\right)^n \\ &\leq\, \exp\left(-\frac{1}{6}(4\pi - 3\sqrt{3})(1+\ell)n\right).\end{aligned}$$

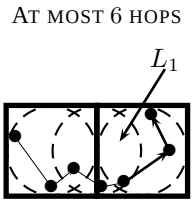


Figure 4. Horizontal transmission.

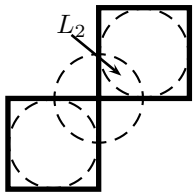


Figure 5. Diagonal transmission.

Figure 6. "Indirect" transmission.

Since each subsquare has at most 4 lenses of size $|L_1|$, none of these regions is empty with probability at least

$$\left(1-\exp\left(-\frac{1}{6}(4\pi-3\sqrt{3})(1+\ell)n\right)\right)^{4j^2} \geq \exp\left(-2\,\frac{\pi n^{1-\frac{1}{6}(4\pi-3\sqrt{3})(1+\ell)}}{(1+\ell)\log n}\right). \tag{9}$$

Hence, with probability tending to 1 as $n\to\infty$, there is at least a node in every lens-shaped region of size $|L_1|$. So, 6 hops at most are needed to transmit a message between two horizontally (or vertically) adjacent subsquares (see Fig. 4), and whence *a)* holds.

To prove *b)*, we consider lenses such as L_2 depicted in Fig. 5. The size of such region is $|L_2| = r^2\,(\pi-2)/2$, which measures the area of the intersection of two equal disks of radius r and at distance $\sqrt{2}\,r$. With arguments similar to Eq. (9), $(1+\ell)(\pi-2)/2>1$ must hold for every lens of size $|L_2|$ to be non-empty (w.h.p.). This condition holds only when $\ell > 2/(\pi-2)-1 = 0.7519\ldots$. For values of $\ell \leq 0.7519\ldots$, transmissions are sent horizontally and then vertically (or vice-versa). Such transmissions can required up to 10 hops (cf. Fig. 6). The proof of the Theorem is now easily completed by simple counting arguments. ■

In the 3-dimensional case, we have the following result.

Theorem 5 *Suppose that n sensor nodes are randomly deployed in a cubic region of volume $|X|$ according to the uniform distribution. If their common transmission range is set to $r = \sqrt[3]{3\,(1+\ell)\,\log n\,|X|/4\pi n}$ with $\ell > 11/5$, then the diameter D of $\mathcal{N}$ satisfies*

$$\lim_{n\to\infty} \mathbb{P}\left[D \leq 12\,\sqrt[3]{\frac{\pi\, n}{6\,(1+\ell)\log n}}\right] = 1. \tag{10}$$

Proof. See [33]. ■

3.2. BROADCAST *and* SFR *(Search-For-Range) protocols*

Subsection 3.1 gives almost sure characteristics of $\mathcal{N}$. Now, we have to verify and to exchange such information by means of distributed algorithms. Two procedures are needed. The first one is the protocol BROADCAST. In this algorithm, some stations (called *sources*) try to scatter a given message to all other nodes in $\mathcal{N}$. It makes several calls of SEND. The second is the protocol SFR (*Search-For-Range*). It is used to adjust the correct transmission range of the nodes, in order to "take control" of the main characteristics of $\mathcal{N}$. SFR works as follows.

Each station starts with the maximum range of transmission. Then, at each step, the transmission range is reduced gradually, till some of the nodes are disconnected. At this stage, all newly isolated nodes readjust their transmission range to get reconnected. Each of them uses the protocol BROADCAST to sped out its "disconnection" message informing all other nodes in $\mathcal{N}$. A node quits the protocol in two cases only: either whenever it broadcasts the "disconnection" message once reconnected after isolation, or after reception of a "disconnection" message containing information about the adequate transmission range.

3.2.1. The broadcasting protocol

The procedure BROADCAST is similar to the one designed in [5] except for the use of SEND to transmit messages.

Procedure BROADCAST(msg,ϵ,Δ,r,N)
$k := 2\lceil \log_2 \Delta \rceil$ ($\star$ Δ is an upper-bound of the maximum degree $\star$)
$\tau := \lceil \log_2 (N/\epsilon) \rceil$ ($\star$ N is an upper-bound of the number of nodes $\star$)
Wait until receiving a message msg
For i from 1 to τ **do**
    Wait until TIME mod$k = 1$ ($\star$ to synchronize $\star$)
    SEND(msg, k, r) ($\star$ attempt to send msg $\star$)
end.

In the above procedure, $\epsilon > 0$ can be made arbitrarily small. Δ is a parameter representing the maximum degree of $\mathcal{N}$ or an upper bound on the maximum degree (according to the regime, Theorem 2 makes it possible to compute Δ for a given value of the transmission range). N is an upper-bound on the number of active nodes. TIME is a protocol which allows any given node to get the current time. Following the proof given in [5, Theorem 4], we have

Theorem 6 ***Bar-Yehuda, Goldreich, Itai [5].***
Suppose that $r \geq r_{\text{CON}}$ *is the actual transmission range of the nodes and let* Δ *be an upper-bound on the maximum degree and N an upper-bound on the number of nodes of* $\mathcal{N}$*. Also assume that some initiators (or sources) start the procedure* BROADCAST*(*msg*,*ϵ*,*Δ*,*r*,N) when* TIME $= 0$*.*
If we let $T = 2D + 5\max\left(\sqrt{D}, \sqrt{\log_2(N/\epsilon)}\right) \times \sqrt{\log_2(N/\epsilon)}$*, then, all the nodes receive the message after* $2\lceil\log_2\Delta\rceil T$ *rounds with probability* $\geq 1 - 2\epsilon$*. Furthermore, with probability* $\geq 1 - 2\epsilon$*, all nodes terminate by time* $2\lceil\log_2\Delta\rceil\,(T + \lceil\log_2(N/\epsilon)\rceil)$*.*

3.2.2. Adjusting the transmitting range: the protocol SFR

The stations need to know bounds of the value of the number n of the nodes. If $p_0 = \lfloor\log_2 n\rfloor$ then $2^{p_0} \leq n < 2^{p_0+1}$.
Thus, by setting $R(2^p) = \sqrt{(\log(2^p) + 2\log 2)\,|X|/\pi 2^p}$, the values of $R(2^p)$ decrease when p increases. In the protocol SFR, we increment the values of p one by one, starting from a value close to the maximal transmission range of the stations. Whenever p passes through $p_0 - 1$, p_0 and $p_0 + 1$, there are some new isolated nodes w.h.p. Actually, it is easily shown that $\sqrt{2\,\log n\,|X|/\pi n} \leq R(2^{p_0-1})$.

We are now ready to present the protocol SFR. Procedure SFR maintains just one variable ϵ representing the tolerance parameter and it is run in parallel by each station.

(L0) **Procedure** SFR(ϵ)
(L1) **BEGIN**
(L2) $R := x \longmapsto \sqrt{\frac{(\log(2^x)+2\log 2)\,|X|}{\pi\,2^x}}$;
(L3) $B := x \longmapsto 24\left\lceil \log x \times \left(\sqrt{\frac{2^x}{x}} + x - \log_2(\epsilon)\right)\right\rceil$;
($\star$ $B(x)$ is the broadcast time. $\star$)
(L4) DISCONNECTED := false;
(L5) $p := \left\lceil \log_2(r_{\text{MAX}})\right\rceil$;
(L6) **REPEAT**
(L7) $counter := 0$;
(L8) $t := 100 \times \left(\left\lceil \log_2(p)\right\rceil + \left\lceil \log_2(2/\epsilon)\right\rceil\right)$;
(L9) **For** i from 1 to t **Do**
(L10) SEND$\langle p, i, R(p)\rangle$;
(L11) Upon reception of a message $\langle p, -, R(p)\rangle$ **Do**
(L12) $counter := counter + 1$;
(L13) **EndFor**
(L14) **If** $counter = 0$ **Then**
(L15) **For** j from 1 to $\left\lceil \log_2\left(\frac{2}{\epsilon}\right)\right\rceil$ **Do**
(L16) BROADCAST("Disconnection p", ϵ,3p,$R(p-1)$,2^{p+1});
(L17) **EndFor**
(L18) DISCONNECTED := true;
(L19) **Else**

(L20) Wait for a message up to $\left\lceil \log_2\left(\frac{2}{\epsilon}\right)\right\rceil \times B(p-1)$ rounds;
(L21) Upon reception of the "disconnection message" **Do**
(L22) Scan the value of p and set DISCONNECTED := true;
(L23) **Else** $p := p+1$;
(L24) **EndIf**
(L25) **UNTIL** DISCONNECTED := true;

When reaching the value p_0, the isolated nodes, whose transmission ranges are now set to $r = R(p_0)$, can increase their transmission range to $R(p_0-1)$ in order to get reconnected. Next, such nodes have to inform all others about the upper-bounds on n, Δ and D, respectively given by

$$2^{p_0} \leq n < 2^{p_0+1}, \qquad \Delta \leq \frac{1}{-W_0(-e^{-1}/2)} \log n < 3\,p_0 \qquad \text{and}$$
$$D \leq 5\sqrt{\frac{\pi\, 2^{p_0}}{(p_0+1)\,\log 2}} < 12\sqrt{\frac{2^{p_0}}{p_0}}, \tag{11}$$

where we use Theorems 2 and 3 for bounding Δ and D, with $\ell = 1$, and the transmission range set to

$$r = \sqrt{\frac{\log\left(2^{p_0-1}\right)|X|}{\left(2^{p_0-1}\right)\pi}}.$$

The message of disconnection can be sent and received correctly by means of multiple calls to the protocol BROADCAST, provided sufficient rounds are given (cf. (L20)) to the broadcasting stations, in order to let all others be aware of the bounds given by Eq. (11). The message broadcasting the above information is a special one, say "*Disconnection* p_0", which contains the correct value of p_0.
Taking Eq. (11) into account, the "broadcast time" given by Theorem 6 is less than $2\lceil\log_2\Delta\rceil\times\left(2D+5\,\max\left(\sqrt{D},\sqrt{\log_2(N/\epsilon)}\right)\times\sqrt{\log_2(N/\epsilon)}+\lceil\log_2(N/\epsilon)\rceil\right)$, with probability greater than $1-2\epsilon$.
This is strictly less than $24\log(p_0)\,\left(\sqrt{2^{p_0}/p_0}+p_0-\log_2(\epsilon)\right)$.

Given these descriptions, the protocol SFR has the following properties.

Theorem 7 *Assume that the network randomly deployed is an instance satisfying Eq. (11). For any $c > 0$ there exists a constant $c_1 > 0$ such that with probability at least $1-1/n^c$, the protocol* SFR*$(1/n^{c_1})$ terminates in at most $\mathcal{O}(D\log n)$ rounds. After this time, every node is aware of the upper-bounds Δ and D on the values of n with probability at least $1-1/n^c$.*

Proof. In lines (L9)-(L13), the inner loop is repeated t times. Consider a node v picked at random. By Theorem 1, for any given node v, as soon as i in line (L9) meets the condition $2^i \gg d_v$, the probability of success of each call of SEND is at least .8... By Theorem 2, and under the assumption that the graph satisfies the almost sure properties of a random network, if $p = p_0 = \lfloor\log_2(n)\rfloor$, $d_v < 3p_0$. Therefore, by setting t as in line (L8), we

ensure that if the node v is still connected, it receives more than one message from its neighbors with probability at least $1 - \epsilon/2$. Similarly, by making the just disconnected nodes repeat sufficient calls of BROADCAST and also have sufficient rounds to send the "disconnection message" to all others, then, w.h.p., all stations of the whole network are allowed to learn the correct upper-bounds on n (and thus Δ and D).
Note that both constants c and c_1 that appear in Theorem 7 do exist, since one can always choose ϵ of the form $\epsilon = 1/n^{c_1}$ in order to get probabilities of failure of order $1/n^c$. ■

According to these results and throughout the remainder of the paper, we have the following definition.

Definition 2 *A random graph G (or a random wireless network $\mathcal{N}$) is said typical if, and only if,*
(i) *Theorem 2 and Theorem 3 hold, and*
(ii) *for any constant $c_1 \geq 1$, after one invocation of protocol* SFR*$(1/n^{c_1})$, every node of G (or $\mathcal{N}$) knows the same value of p_0 satisfying Eq. (11).*

Remark. Let $\mathbb{G}(n,p)$ denote the random binomial graph [12], where each of the $\binom{n}{2}$ edges of the complete graph K_n is present with probability p. The results described above can be compared to the (almost sure) characteristics of the binomial random graph of Erdös-Rényi $\mathbb{G}(n,p)$ [12] as shown in the following table, where $\omega(n)$ is any function tending to infinity with n. In the table, $\rho = n/|X|$ represents the expected number of points per area.

MODELS	Euclidian random graph with intensity $n/\|X\|$ $\mathbb{G}(n,r)$	Erdös-Rényi random graph $\mathbb{G}(n,p)$
PARAMETERS	Radius: $r \equiv r(n,X)$	Edge probability $p \equiv p(n)$
COMMON PROPERTIES	Required value for $r^2\pi n/\|X\|$	Required value for np
Connectivity	$\ln n + \omega(n)$	$\ln n + \omega(n)$
Minimum degree $= 2$	Total coverage of X: $\ln n + \ln\ln n + \omega(n)$	Hamiltonicity: $\ln n + \ln\ln n + \omega(n)$
Minimum degree $= j+1$	Multiple coverage of X and if X is a square j-connectivity: $\ln n + j\ln\ln n + \omega(n)$	j-connectivity: $\ln n + j\ln\ln n + \omega(n)$
Quasi-regularity: all nodes have degree $\sim \omega(n)$	$\omega(n)\ln n$ with $\omega(n) \ll \frac{n}{\ln n}$	$\omega(n)\ln n$ with $\omega(n) \ll \frac{n}{\ln n}$

For instance, one can read from the 4^{th} row of the table that the value required for $r^2\,\pi n/|X|$ is $\ln n + \omega(n)$ to obtain, with high probability, a connected geometric random graph $\mathbb{G}(n,r)$. And similarly for np in the case of a connected random graph $\mathbb{G}(n,p)$) (Erdös-Rényi). In both cases The connectedness property occurs almost surely *iff* $\omega(n) \to \infty$ with n.

4. A first initialization algorithm in random radio networks

Section 3 solved the problem of determining the correct transmission range for the nodes of a random network $\mathcal{N}$. Typically, $\mathcal{N}$ has the characteristics (mainly maximum degree and hop-diameter) dictated by Theorems 2 and 3. Probabilistic upper-bounds on such characteristics can also be established with the protocol SFR. In [4], Bar-Yehuda *et al.* propose algorithms for efficient emulation of a single-hop network with collision detection in multi-hop radio networks,provided the number of stations (nodes), the diameter and the maximum degree of $\mathcal{N}$ (or upper-bounds on them) are known. Combining the results in [4] and [26] with the results in Section 3 leads to a new initialization protocol. More precisely, we can emulate a complete network (with collision detection) using the methods in [4]. Therefore, any broadcasting protocol with the Nakano-Olariu algorithms in [26] makes it is possible to build an initialization protocol in time $\mathcal{O}(nB)$, where B denotes the broadcast time of order $B = \mathcal{O}(D \log n)$ (see for instance [5,9,22]).

Instead, we first color the graph in a specific manner: the two-hop coloration. In this problem, the nodes of $\mathcal{N}$ are colored in such a way that every pair of stations (u, v) within a hop-distance of at most 2 from each other are assigned different colors (codes or "channel assignment"). This specific coloration gives the graph a natural scheduling of the communications which avoids *direct and hidden terminal problems*. Every pair of nodes (u, v) at hop-distance ≤ 2 from each other is assigned one pair of color (code) (c_u, c_v), with $c_u \neq c_v$. When a station u (or v) decides to transmit in the exact round that directly matches its own codes c_u (or c_v, resp.), then it is easily seen that such scheduling is *collision-free*.

4.1. Choice of temporary IDs

Since the stations are supposed to be indistinguishable, the first goal is to allocate them distinct temporary IDs. If an upper-bound N on n is known, it can be done in one pass by assigning each station an integer uniformly picked from the range $\left[1, N^3\right]$.

Procedure TMPIDS(N)
Each node chooses uniformly at random an integer ranging from 1 to N^3
end.

The above very simple procedure has the following property.

Theorem 8 *Suppose that N is an upper-bound on the number of nodes n known by all the stations. After one invocation of* TMPIDS(N)*, with high probability, every station of $\mathcal{N}$ has a unique ID ranging from 1 to N^3 and no pair of stations share the same ID.*

Proof. The proof of this result is a simple application of the balls and bins problem. By throwing n balls (stations) into N^3 bins (temporary IDs) independently and uniformly at random, with probability greater than $\exp\left(-\mathcal{O}(n^2/N^3)\right)$ every bin contains at most one ball. ■

4.2. The two-hop neighbor discovery protocol

Once the temporary IDs are allocated, each node u of $\mathcal{N}$ has to discover all other nodes within distance at most 2 hops. The protocol DISCOVER below allows any given node

u of $\mathcal{N}$ to know the set of its direct and two-hop neighbors (i.e. neighbors of neighbors). This algorithm appears to be extremely useful since the stations are deployed in random fashion and do not have any knowledge of their respective neighborhoods. In the following pseudo-code, N still represent any known upper-bound on the number of nodes n.

Procedure DISCOVER(N)
Begin
For each node u, set $L(u) := \emptyset$;
For k from 1 to $(\log N)^3$ **Do** ($\star$ Discovering direct neighbors $\star$)
With probability $1/\log N$, every node u transmits a message containing its temporary ID: TEMPID$_u$;
Upon reception of a message $\langle$TEMPID$\rangle$, u stores the value TEMPID in a local list:
$L(u) := L(u) \bigcup \{$TEMPID$\}$;
EndFor
For k from 1 to $(\log N)^3$ **Do** ($\star$ Discovering 2-hop neighbors $\star$)
With probability $1/\log N$, every node u sends the list $L(u)$ of its direct neighbors;
EndFor
End.

Theorem 9 *Assume that $\mathcal{N}$ is typical and the transmission range is set to $r = \sqrt{2\log(2^{p_0})|X|/(2^{p_0}\pi)}$ with p_0 satisfying Eq. (11). The running time of* DISCOVER*(2^{p_0+1}) is $\mathcal{O}\left(\log(n)^3\right)$ and with high probability, after one invocation of* DISCOVER*(2^{p_0+1}),*
(i) *Every node u of $\mathcal{N}$ is aware of the list of all its direct and two-hop neighbors.*
(ii) *For each node u, the number of such direct and two-hop neighbors is $\mathcal{O}(\log n)$.*

Proof. The proof of part *(i)* is closely related to the proof in [33, Theorem 7]. For clarity, here are the details. After the first loop of the above algorithm, the proof that every station is aware of the list of all its neighbors relies on two facts.

First, the main characteristics of the random Euclidean network and second, the number of iterations $\mathcal{O}(\log n)^3$ in this loop are sufficient for each node to send its ID *at least once* to all its neighbors. For the first point, we have seen that if the transmission range is set to $\sqrt{(1+\ell)\log n|X|/\pi n}$ ($\ell > 0$) for any node v of $\mathcal{N}$, then the degree of v meets the condition w.h.p.

$$d_v \;\leq\; -\frac{\ell\,\log n}{W_0\left(-\frac{\ell}{e\,(1+\ell)}\right)} \;+\; o(\log n).$$

Set $N = 2^{p_0+1} \geq n$. In the regime considered in Theorem 9, the maximum degree of $\mathcal{N}$ is bounded by $c\log n$ (w.h.p.), where c is some constant such that e.g., $c \geq 2\,W_0\,(-1/2e)$ (for any constant $\ell > 2$). Using the latter remark, let us complete the proof of part *(i)*. For any distinct pair (i,j) of adjacent nodes and any round $t \in \left[1, \log(N)^3\right]$, define the random variable $X_{i\to j}^{(t)}$ as follows:

$$X_{i\to j}^{(t)} = \begin{cases} 1 & \text{if the node } j \text{ does not receive the ID of } i \text{ at time } t \in \left[1,\log(N)^3\right], \\ 0 & \text{otherwise.} \end{cases}$$

In other terms, the set

$$\left\{X_{i\to j}^{(t)}, i,\, j\neq i,\, t\in\left[1,\log\left(N\right)^3\right]\right\}.$$

denotes a set of random variables that counts the number of arcs $i\to j$ such that j never received the ID of i. Denote by X the r.v.

$$X=\sum_{i\neq j}X_{i\to j},$$

where $X_{i\to j}=1$ *iff* $X_{i\to j}^{(t)}=1$ for all $t\in\left[1,\log\left(N\right)^3\right]$.

Now, the probability that i does not succeed in sending its ID to j at time t is

$$\mathbb{P}\Big(X_{i\to j}^{(t)}=1\Big)\;=\;\left(1-\frac{1}{\log\left(N\right)}\right)\;+\;\frac{1}{\log N}\left(1-\left(1-\frac{1}{\log N}\right)^{d_j}\right).$$

Therefore, considering the whole range $\left[1,\log\left(N\right)^3\right]$ yields

$$\mathbb{P}(X_{i\to j}=1)\;\leq\;\left(1-\mathcal{O}\left(\frac{1}{\log\left(n\right)}\right)\right)^{\log\left(N\right)^3}\;\leq\;\exp\left(-\mathcal{O}(\log n)^2\right),$$

which bounds the probability that i has never sent its ID to j for rounds t in the range $\left[1,\log\left(N\right)^3\right]$.
By linearity the expectation, and since the number of edges is of order $\mathcal{O}(n\log n)$,

$$\mathbb{E}(X)\;\leq\;\mathcal{O}(n\log n)\,\exp\left(-\mathcal{O}(\log n)^2\right). \tag{12}$$

Thus, $\mathbb{E}(X)\ll 1$ as $n\to\infty$, by the first moment method [3], one completes the proof that after the first loop of the procedure, every station is aware of all its direct neighbors.

With similar methods, it is easily seen that the second loop allows the nodes to know one after the other their 2-hop neighbors.

To prove part *(ii)*, observe that if r is the common transmission/receiving range of the stations, all two-hop neighbors of a node u are inside a circle of radius $2r$. Hence, a simple application of the Eq. (6) in Theorem 2 proves assertion *(ii)* of Theorem 9. ■

4.3. A two-hop coloring algorithm

We need some more basic definitions for our coloring algorithms.

Definition 3 $\Gamma(u)\stackrel{def}{=}\{$*neighbors of a fixed node* $u\}$. *Any* $v\in\Gamma(u)$ *is referred to as a direct neighbors of* u.

The set of 2*-hop neighbors is given formally by* $\Gamma_2(u)\stackrel{def}{=}\bigcup_{v\in\Gamma(u)}\Gamma(v)$.

Recall that $\Delta\stackrel{def}{=}\max_u|\Gamma(u)|$. *Similarly, define* Δ_2 *as* $\Delta_2\stackrel{def}{=}\max_u|\Gamma_2(u)|$.

To assign codes (colors) to the nodes of $\mathcal{N}$, let us consider the following simple and intuitive randomized protocol called AssignColor. As defined above, $\Gamma(u) \bigcup \Gamma_2(u)$ is the set of neighbors of u at hop-distance at most 2. At the beginning of the algorithm, each node u stores an initial list of colors $p(u)$ (also referred to as *palette*) of size $|\Gamma(u)| + |\Gamma_2(u)| + 1 = \Delta + \Delta_2 + 1$ and starts uncolored. We can also assume that each node has a distinct ID (this can be effective after one invocation of TmpIDs) and knows its neighbors in $\Gamma(u) \bigcup \Gamma_2(u)$ (by means of Discover).

Then, the protocol AssignColor proceeds in rounds. In each round, each *uncolored node* u, simultaneously and independently picks a color at random, say c, from its palette. Next, the node u attempts to send this information to its direct neighbors in $\Gamma(u)$, and in its turn, each member $v \in \Gamma(u)$ tries to forward the information to every $w \in \Gamma_2(u)$. Trivially, this "two-steps" attempt succeeds *iff* there is no collision with direct neighbors and also "no collision" with 2-hop neighbors. Therefore, before absolutely assigning its color (code) c to u, every member of the set $\Gamma(u) \bigcup \Gamma_2(u)$ has to sent one by one a message of reception.
Note that this can be done *deterministically* as explained in details below. Therefore, u sends a message of *acknowledgement* and every member v of $\Gamma(u) \bigcup \Gamma_2(u)$ can *update* its own palette $p(u)$ and its own set $\Gamma(u) \bigcup \Gamma_2(u)$.

Hence, at the end of such an iteration the new colored node u becomes passive during the remaining of the algorithm. (Note that the protocol AssignColor is simply the "2-hop version" of the coloring algorithm presented in [33, Subsection 5.3].)

Assuming that the upper-bound N on the number of nodes satisfies Eq. (11), a brief description of this procedure follows. Each step below represents a *basic iteration* of the main loop of the algorithm. By allowing $\mathcal{O}(\log n)^3$ iterations, the algorithm is shown to color correctly the graph.

Basic iteration of the main-loop of AssignColor

- **Step 0 :** Every node needs an initial palette of colors of size $|\Gamma(u)| + |\Gamma_2(u)| + 1$ and a set of active direct and 2-hop neighbors. The upper-bound on the number of direct neighbors is set to $\Delta := 3\lceil \log_2(N) \rceil$. Similarly, using Eq. (6) of Theorem 2, an upper-bound on the number of 2-hop neighbors is set to $\Delta_2 = \lceil 8 \log(N) \rceil - \Delta$. Note that the constant 8 reflects the fact that all 2-hop neighbors of u are within an Euclidean distance of at most twice the transmission range from u. Therefore, Eq. (6) yields $-3/W_0(-3e^{-1}/4) \sim 7.14\ldots < 8$.
- **Step 1 :** Every node u picks a color c from its palette and tries to send it to $\Gamma(u)$.
- **Step 2 :** If the previous step succeeds, there is no collision and every node $v \in \Gamma(u)$ receives correctly the message. Since Discover allows u to know its neighbors, u can rank them and in their turn, one after the other accordingly to their relative rank, they have to forward the message to the 2-hop neighbors of u, that is to any $w \in \Gamma_2(u)$. This phase is deterministic and =synchronization requires Δ rounds.
- **Step 3 :** If the previous step works correctly, every member of $\Gamma_2(u)$ receives the message and, in its turn, sends it back. In order to avoid confusion, each message is specifically marked with u (the ID of the initial sender). This step can be performed deterministically, since u can also rank its 2-hop neighbors. Thus, step 3 also requires Δ_2 rounds.
- **Step 4 :** When all their messages are back, all nodes $v \in \Gamma(u)$ need to inform u, one by one and in right order. So, step 4 is performed in Δ rounds.

Step 5 : Upon receiving all messages from all its direct and 2-hop neighbors, the node u has to send them back a message of acknowledgement. Again, this step is deterministic and requires Δ rounds (only the direct neighbors are needed to forward the acknowledgement message). The nodes in $\Gamma(u) \bigcup \Gamma_2(u)$ update their palettes of colors by removing the color c, which is now assigned to u, and u becomes a *passive* node.

The corresponding pseudo-code of the protocol is as follows.

(1) **Protocol** ASSIGNCOLOR(N)
(2) Set $\Delta := 3\lceil \log_2(N) \rceil$ and $\Delta_2 := \lceil 8 \log(N) \rceil - \Delta$; ($\star$ following Eq. (11) $\star$)
(3) Each node u is *active*, with an initial *palette* of colors $p(u) = \{c_1, c_2, \ldots, c_{\Delta+\Delta_2+1}\}$ along with a set of active neighbors in $\Gamma(u)$ and 2-hop neighbors in $\Gamma_2(u)$;
(4) **For** $i := 1$ to $\log(N)^3$ **Do**
(5) For each node u do
(6) • Pick a color c from $p(u)$;
(7) • Send a message containing c with probability $\frac{1}{\Delta+|p(u)|}$;
(8) **If** no collision **Then** ($\star$ 1-hop neighbors $\rightarrow$ forward to $w \in \Gamma_2(u)\star$)
(9) Every station v in $\Gamma(u)$ gets the message properly, one by one in order
(10) Every station v in $\Gamma(u)$ forwards a specific message "$v\ u\ c$";
($\star$ v is the ID of the current node. The step is synchronized by allowing Δ rounds. $\star$)
(11) **EndIf**
(12) **Upon** receiving a message of the form "forward $v\ u\ c$" **Do**
($\star$ 2-hop neighbors $\rightarrow$ just send back twice $\star$)
(13) Every member w of $\Gamma_2(u)$ one by one and in order
(14) sends back a message to the member of $\Gamma(u)$.
(15) Such a message can be of the form "back $w\ u\ c$";
($\star$ This step can be synchronized by always allowing Δ_2 rounds $\star$)
(16) **end**
(17) **Upon** receiving a message of the form "back $w\ u\ c$" **Do**
(18) The node $v \in \Gamma(u)$ sends the message back to u along with its own ID;
($\star$ This step needs Δ rounds of synchronization $\star$)
(19) **end**
(20) **If** u receives all the $|\Gamma_2(u)| + |\Gamma(u)|$ messages **Then**
(21) u sends a message of *acknowledgement* which is also forwarded
(22) by all members of $\Gamma(u)$ to the set $\Gamma_2(u)$; u becomes *passive*;
(23) **EndIf**
(24) **Upon** receiving an *acknowledgement message*
(25) every station in $\Gamma(u) \bigcup \Gamma_2(u)$ removes the color c from its palette;
($\star$ This step is synchronized in Δ_2 rounds $\star$)
(26) **EndFor**
(27) **End.**

Theorem 10 *Assume that the randomly deployed network is typical, with the transmission range set to* $r = \sqrt{2\log(2^{p_0})\,|X|/(2^{p_0}\pi)}$. *Suppose also that the stations have distinct IDs. Then, after the execution of* ASSIGNCOLOR(N)*, with probability tending to* 1 *as* $n \rightarrow \infty$*, every pair of nodes* (u, v) *s.t.* $u \in \Gamma(v) \bigcup \Gamma_2(v)$ *receives two distinct codes (colors). Moreover, the running time of the* ASSIGNCOLOR *is* $\mathcal{O}(\log(n)^4)$ *rounds and the protocol uses* $\mathcal{O}(\log n)$ *colors.*

Proof. Though it is more difficult, the proof of Theorem 10 is very similar to the proof in [33, Theorem 8]. Observe first that the only randomized part of the algorithm is the attempt of u to allocate a color (cf. line 7); after what, all steps are deterministic. So, whenever it is successful, such an attempt can easily be checked by the initiator u, since u maintains the list of nodes in $\Gamma(u)$ and in $\Gamma_2(u)$. Precisely, the algorithm builds a new graph in which each new edge is (virtually) added between every pair of 2-hop neighbors.

For any node u, recall that $\Gamma(u) \bigcup \Gamma_2(u)$ represents the set of its direct and 2-hop neighbors and let p_u denote the size of its current palette. Now, define the random variable Y_u as follows,

$$Y_u = \begin{cases} 1 & \text{if the node } u\text{remains uncolored after the } \log N^3 \text{ steps of } \textsc{AssignColor} \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

Let $\Gamma_u^{(t)}$ and $\Gamma_{u,2}^{(t)}$ denote the set of direct and 2-hop *active* neighbors of u (resp.) at any given iteration t of the algorithm. Suppose that we are in such an iteration t. Independently of its previous attempts, u remains uncolored with probability

$$p_{u,t} \;=\; \left(1 - \frac{1}{(\Delta + p_u)}\right) \;+\; \frac{1}{(\Delta + p_u)} \times \left(1 \;-\; \left(1 - \frac{1}{(\Delta + p_v)}\right)^{|\Gamma_u^{(t)}| + |\Gamma_{u,2}^{(t)}|}\right), \tag{14}$$

where there is at least a *direct* collision with a neighbor $v \in \Gamma_u^{(t)}$ or a "2-*hop collision*" with a neighbor $w \in \Gamma_{u,2}^{(t)}$.

For all t, $|\Gamma_u^{(t)}| \leq \Delta$, $|\Gamma_{u,2}^{(t)}| \leq \Delta_2$ and, for all v, $1 \leq |p_v| \leq \Delta + \Delta_2 + 1$. Of more importance, $\Delta = \mathcal{O}(\log n)$ and $\Delta_2 = \mathcal{O}(\log n)$. Thus,

$$p_{u,t} \;\leq\; 1 \;-\; \frac{1}{(\Delta + p_u)} \left(\frac{1}{\Delta}\right)^{|\Gamma_u^{(t)}| + |\Gamma_{u,2}^{(t)}|} \;\leq\; 1 \;-\; \mathcal{O}\left(\frac{1}{\log n}\right).$$

Since there are $\mathcal{O}(\log{(n)}^3)$ iterations, there exists a constant α such that, with probability at most

$$\left(1 - \mathcal{O}\left(\frac{1}{\log n}\right)\right)^{\mathcal{O}(\log n)^3} \;\leq\; \exp\left(-\alpha \log{(n)}^2\right),$$

u remains uncolored during the whole algorithm. The expected number of uncolored vertices at the end of the protocol $\textsc{AssignColor}$ is thus less than

$$\mathbb{E}(Y) = \sum_u \mathbb{E}(Y_u) \;\leq\; n \exp\left(-\mathcal{O}(\log{(n)}^2)\right). \tag{15}$$

Finally, by Markov inequality (cf. e.g. [3]), the proof of Theorem 10 follows. ■

4.4. An optimal gossiping algorithm in $\mathcal{O}(\sqrt{n \log n})$ rounds

The 2-hop coloring process induces a natural scheduling algorithm for gossip. The gossip algorithm is very intuitive: once $\mathcal{N}$ is colored, in each round every station u is allowed to transmit *iff* its color c_u is such that $\textsc{Time} \mod c_u \equiv 0$ (where $\textsc{Time}$ is a function that returns the global current round). Procedure $\textsc{Gossip}$ below starts with a randomly deployed set of stations and uses all the procedures described previously.

Procedure $\textsc{Gossip}$
Step 1: Start estimating the main characteristics of $\mathcal{N}$ with $\textsf{SFR}(1/|X|)$;
Such procedure allows all stations to get estimates on the transmission range r, the maximal degree Δ, the hop diameter D and the number of active nodes n;
Step 2: Allocate temporary IDs to the stations with $\textsc{TmpIDs}(N)$;
($\star$ N denotes a probabilistic upper-bound of n $\star$)
Step 3: Color the graph with $\textsc{AssignColor}(N)$;
Step 4: Use the obtained colors as follows:
Repeat $100 \times \sqrt{N/\log N}$ times
For each node u with color c_u
If $\textsc{Time} \mod c_u = 0$ **Then**
Transmit all known IDs;
EndIf
EndRepeat
End.

Since, $\textsc{AssignColor}$ assigns $\mathcal{O}(\log n)$ distinct colors and the hop-diameter of the $\mathcal{N}$ is given by $D = \mathcal{O}(\sqrt{n/\log n})$, we easily derive the following immediate but important result.

Theorem 11 *If the random plane network is typical, then the procedure* $\textsc{Gossip}$ *requires $\mathcal{O}(\sqrt{n \log n})$ rounds and for every pair of stations (u, v), u receives the temporary ID of v with high probability.*

4.5. An optimal initialization algorithm in $\mathcal{O}(D\Delta)$ rounds

The initialization procedure below is a straightforward consequence of the above algorithm.

Procedure $\textsc{Initialization}(N)$
$\textsc{Gossip}$;
For each station u, sort all received messages;
$\mathrm{ID}(u) :=$ rank of u in the sorted array of temporary IDs;
End.

Theorem 12 *With high probability, the procedure* $\textsc{Initialization}(|X|)$ *assigns each station of $\mathcal{N}$ one unique identity ranging from 1 to n in $\mathcal{O}(\sqrt{n \log n})$ rounds.*

Corollary 13 *The gossiping and initialization algorithms presented above are asymptotically optimal.*

Proof. This is an immediate consequence of the results of Kushilevitz and Mansour in [21]. Since gossiping and initializing are harder than broadcasting, which requires $\Omega(D \log(n/D))$ rounds in graphs such as random plane networks, they both require at least $\Omega(D \log(n/D))$ rounds. Fortunately, $D\Delta$ and $D \log(n/D)$ have a same order of magnitude (w.h.p) in $\mathcal{N}$. ■

Note that all the above results remain valid when $\mathcal{O}\left(n^{1/3}(\log n)^{2/3}\right)$ is substituted for $\mathcal{O}(\sqrt{n \log n})$, inside a cube (instead of a square).

5. Divide-and-conquer algorithms and energy-efficient protocols

In this Section, we present a randomized algorithm running in $\mathcal{O}\left(n^{3/4} \log(n)^{1/4}\right)$ rounds, with no station being awake for more than $\mathcal{O}\left(n^{1/4} \log(n)^{3/4}\right)$ rounds. It is shown that our sublinear and energy-efficient initialization algorithm is at most $\mathcal{O}(\log n / \log\log n)$ far from optimality, with respect to the number of rounds required. In fact, the running time is $\mathcal{O}(D \log n) = \mathcal{O}(D\Delta)$, where Δ is the maximum degree of $\mathcal{N}$ and D denotes its hop-diameter. Indeed, it was shown in [22] that, in the same setting, the easiest broadcasting problem requires $\Omega(D \log\log n)$ rounds.

The main lines of the algorithms are given below.
In order to schedule all communications, the stations of $\mathcal{N}$ are colored in such a way that any pair (u, v) of nodes within distance ≤ 2 are assigned two distinct colors. This coloring algorithm suggests a natural scheduling of all the communications in our protocols. This specific algorithm and some others are called PREPARATION protocols.

Next, the divide-and-conquer principle is applied. We cluster the graph (with CLUSTERING), with a specific node called the *cluster head* in each cluster. Every cluster is then locally initialized (with LOCAL INITIALIZATION).

Finally, the global initialization step (GLOBAL INITIALIZATION) is carried out over the graph of clusters. All communications are realized via the specific paths constructed between neighboring clusters (by swapping over from one path to another). A gossiping algorithm between all cluster heads just followed by a ranking algorithm complete this last step.

In light of the previous Sections, we know that $\mathcal{N}$ satisfies the following main characteristics with high probability.
• There exist two constants c_ℓ and C_ℓ such that the degree d_v of any node v meets the condition

$$c_\ell \log n \leq d_v \leq C_\ell \log n.$$

• $\mathcal{N}$ is $(c_\ell \log n)$-connected.
• The hop-diameter (or diameter) of $\mathcal{N}$ meets the condition $D = \Theta(\log n)$.

5.1. Clustering

The aim of this step is to design a randomized algorithm that partitions the set of nodes into disjoint groups. The hop-diameter of each group ranges between k and $2k$, where k

is a parameter that will be fixed later in the analysis of the algorithm. In each cluster, there is a specific node called the *cluster head* (CH for short). The principle of the clustering algorithm is simple and intuitive.

At first, each node becomes a candidate cluster head with a certain probability. If two or more candidate cluster heads are too close from each other (viz. they are within distance less than k-hops), all of them must be eliminated but one, which is considered the true cluster head. This can be done by choosing the candidate with the biggest TEMPID amongst all others; eliminated candidates become normal nodes.

In the end of the algorithm, we have to collect all *orphan* nodes, that is all the nodes which are not within a k-hop neighborhood of the newly marked out CH. The orphans choose the nearest CH among all the possible cluster heads in their respective $2k$-hop neighborhood. During the clustering protocol, every communication is mainly performed by using the 2-hop coloration algorithm mentioned above.
This partitioning process is a key ingredient of our initialization algorithm. After the execution of the clustering protocol, each cluster can be initialized locally.

5.2. Local initialization

In order to initialize each cluster locally, the protocol `GOSSIP` is used. The idea is very similar to those in [34]. The local initialization protocol is executed distributively by all the stations in all the clusters. Every node in $\mathcal{N}$ transmits its TEMPID to all other stations. The gossiping protocol uses the collision-free scheduler that the coloration algorithm provides and when a station receives a message msg, it appends its TEMPID to msg and transmits this new message to all its neighbors. Since the coloration algorithm uses $\mathcal{O}(\log n)$ colors,after $\mathcal{O}(k \log n)$ rounds, all stations know the TEMPIDs of all other stations in their cluster. Finally, the rank of the TEMPID of a station simply becomes its local ID (denoted LOCALID).
Upon termination of the local initialization step, each station owns two "IDs": its temporary ID (TEMPID) and its local ID (LOCALID), according to the cluster it belongs to.

5.3. Paths between the clusters

In the next step, the paths of communication between each cluster must be constructed. The idea is as follows. During such communications, all stations are intentionally *asleep* to save their batteries, except the stations on such communication paths. To avoid "energy holes" (on the most crowded paths), we have to find out as many disjoint paths as possible and swap over from one path to another.

5.4. Global initialization

The global initialization step is performed by means of a gossiping algorithm between all cluster heads, just followed by a ranking algorithm. All communications are carried out via the disjoint paths between clusters, and by skipping from one path to the other.

Figures 7 and 8 describe and briefly summarize the main steps of the initialization protocol.

Figure 7. Division of the graph into disjoint clusters and local initialization of each cluster. In the figure on the right above, 3 clusters are initialized with integers ranging from 1 to 141, from 1 to 287, and from 1 to 192, respectively.

Figure 8. After the paths construction between clusters (dashed lines), the global initialization is just executed by means of the gossiping algorithm performed via these paths.

5.5. Detailed Algorithms and Analysis

In this Subsection, we are concerned with three basic algorithms which are frequently used and discussed throughout the remainder of the paper. First, the protocol AssignColor is executed (see the design in [34, Section VI]). AssignColor requires $\mathcal{O}\left(\log(n)^4\right)$ rounds and uses $\mathcal{O}(\log n)$ colors. After the execution of AssignColor, any two stations within 2 hops-distance receive two distinct codes (colors) with high probability. Once it is well-colored, $\mathcal{N}$ is collision-free and we are now ready to design the protocol BROADCAST. (The pseudo-code is in Fig. 9.)

It is easily seen that such an algorithm requires $\mathcal{O}(k \log n)$ rounds, under the condition that the randomized coloring algorithm succeeds with probability 1 (i.e, it errs with probability 0).

One can design a gossip algorithm based upon Broadcast. In the gossiping problem, the task is to sped out the information contained in each station to all others. Such a protocol can be derived from the broadcasting one by changing a few lines, as described in Fig. 10.

Since there are $\mathcal{O}(\log n)$ colors and k steps, the execution time of Gossip(k) is the same as Broadcast(k): $\mathcal{O}(k \log n)$.

```
1  Procedure BROADCAST(msg : message,k : integer )
2  Begin
3   Repeat (100 × k × log n) times
4    For a node u colored with c_u, upon receiving a message of the form ⟨msg, k⟩ Do
5      If (TIME mod c_u) ≡ 0 and k > 0 Then
6       BROADCAST(msg, k − 1) EndIf
7   EndRepeat
8  End
```

Figure 9. The BROADCASTING protocol.

```
1  Procedure GOSSIP(k : integer)
2  Begin
3   Repeat (100 × k × log n) times
4    For each station u with initial message msg(u) and colored c_u,
       upon receiving any message of the form ⟨msg, k⟩ Do
5    If (TIME mod c_u) ≡ 0 and k > 0 Then
6      msg := append(msg, msg(u));
7      TRANSMIT(msg);
8      GOSSIP(k − 1) EndIf;
9   EndRepeat
10 End
```

Figure 10. The GOSSIPING protocol.

5.6. Random clustering

In order to apply a divide-and-conquer algorithm, we design the protocol CLUSTERING, which works as follows.
At first, each station chooses to be a candidate cluster head (CH) with a certain probability (to be specified later in the analysis). The protocol meets the following specifications:

(i) each cluster has a CH;
(ii) each node knows its CH, which is at most within $2k$-hops distance;
(iii) any two CHs are at a distance of at least $k + 1$ hops from each other.

Therefore, there exist randomly chosen candidates in the support area X. In order to satisfy specification *(iii)*, a few candidates which are too close from each other must be eliminated.

By using a broadcasting algorithm at a distance k (with BROADCAST), each candidate CH can detect whether there exist some other candidates in its k-hop neighborhood. The candidate with the biggest TEMPID becomes a true CH and all others are eliminated. Finally, the orphans (nodes without a CH) are collected as follows (with COLLECT). Every CH executes a protocol, with a specific message that enables each orphan to choose the nearest CH in its $2k$-neighborhood.

```
1  Procedure COLLECT(j : integer)
2  Begin
3    For each node u Do
4     If u is a cluster head Then
5       Repeat (100 × j × log n) times
6         If (TIME mod c_u) ≡ 0 Then
7         TRANSMIT(TEMPID,1)
8       EndRepeat
9     Else
10      CLUSTER(u) := NIL, distance := ∞;
11      Upon receiving a message of the form ⟨TEMPID, radius⟩
12    EndIf
13    If distance > radius Then
14    CLUSTER(u) := TEMPID, distance := radius;
15        Repeat (100 × j × log n) times
16          If (TIME mod c_u) ≡ 0 Then
17           TRANSMIT(CLUSTER(u),distance+1);
18        EndRepeat
19    EndIf
20   EndElse
21 End
```

```
1  Procedure CLUSTERING(k : integer,p : float)
2  Begin
3  Step 1: Each station chooses to be a CANDIDATE cluster head with probability
   p;
4  Step 2: For each CANDIDATE station run BROADCAST(TEMPID,k);
5  Step 3: Upon receiving a broadcasting message, eliminate the candidates which
   TEMPID is smaller than the ID(s) of some other(s) candidate(s).
6  Step 4: The remaining candidates are now cluster heads and broadcast their TEM-
   PID by means of COLLECT(TEMPID,2k), to inform the stations at 2k-hop dis-
   tance of their presence and status.
7  Step 5: For each node u, CLUSTER(u) is set to the nearest cluster head among the
   nodes that invocated the protocol COLLECT.
8  End
```

From Section 3, we derive the following result.

Theorem 14 CLUSTERING(k,P) *requires* $\mathcal{O}\left(\max\left(k\log n, \log(n)^4\right)\right)$ *rounds. After the execution of the protocol* CLUSTERING, *w.h.p. any station belongs to a specified cluster and knows its cluster head, which is at a distance of at most* $2k$ *hops.*

Proof. By choosing $P \stackrel{\text{def}}{=} 9/(k^2(1+\ell)\log(n))$, we make sure that the disks with radius kr that are centered at the candidate stations achieve a full coverage of the support area X. More precisely, let ξ be the random variable counting the number of candidate stations. The average number of candidate stations is given by $\mathbb{E}(\xi) = nP$.

By Chernov bounds, we know that, since $\frac{n}{k^2(1+\ell)\log(n)} \to \infty$, $\xi = \Theta\left(\frac{n}{k^2\log(n)}\right)$ w.h.p. Hence, standard calculus yields

$$\mathbb{P}\left(\frac{1}{3}\,\mathbb{E}(\xi) \leq \xi \leq 2\mathbb{E}(\xi)\right) \leq 1 - \exp\Big(-\mathcal{O}(\mathbb{E}(\xi))\Big). \tag{16}$$

Next, by virtue of the result in [25, Thm. 3.2], if $1/3\,\mathbb{E}(\xi)k^2r^2 > 2.83\,|X|$ the disks generated by the candidate stations ensure a full coverage of the support area $|X|$ w.h.p. Then, it is easily seen that the elimination of two "colliding" candidate CHs can be worked out by using the BROADCAST protocol. Similarly, any station which still needs a cluster head is assigned the closest CH in its $2k$-hops neighborhood, by means of the COLLECT protocol.

Finally, CLUSTERING is made of ASSIGNCOLORand BROADCAST, which require $\mathcal{O}\left(\log(n)^4\right)$ (cf. [34]) and $\mathcal{O}(k\log n)$ rounds, respectively. Hence, the time complexity of CLUSTERING is clearly $\mathcal{O}\left(\max\left(k\log n, \log(n)^4\right)\right)$. ■

5.7. Learning the neighborhood and local initialization

The aim of the protocol TOTALKNOWLEDGE is to allow each station to "learn" the topology of its i-hops neighborhood, where i is a parameter of the procedure. In order to construct the adjacency matrix of its neighbors, a given node executes the local procedure APPENDTOADJACENCYMATRIX. It works as follows:

- Every node u (with degree d_u) maintains a local list $L(u)$, initialized to

$$L(u) := \text{TEMPID}(u) \to \text{NIL}.$$

- Upon receiving the number of its direct neighbors $v_1, v_2, \ldots, v_{d_u}$, u updates $L(u)$ to

$$\begin{array}{lll} L(u) := \text{TEMPID}(u) \to v_1 & \cdots v_{d_u} \to \text{NIL} \\ \qquad\qquad\quad\ \downarrow & \cdots \downarrow \\ \qquad\qquad\quad\ \text{NIL} & \cdots \text{NIL}. \end{array}$$

- Next, every neighbor $v_1, \ldots, v_{d_u}$ sends its respective list $L(v_1), \ldots, L(v_{d_u})$ that u appends to its current list and constructs its own neighborhood adjacency matrix.

Clearly, after i steps each participating node can build its own $i \times i$ adjacency matrix, which represents its i-hops neighborhood.

Procedure TOTALKNOWLEDGE, which runs APPENDTOADJACENCYMATRIX is as follows.

```
1  Procedure TOTALKNOWLEDGE(i : integer)
2  Begin
3   Each node u with TEMPID(u) and colored c_u
    maintains a list L(u) := TEMPID(u);
4   t := i;
5     Repeat (100 × i × log n) times
6       If t > 0 and (TIME mod c_u) ≡ 0 Then
7       TRANSMIT(L(u), t);
8       t := t − 1;
9       EndIf
10     Upon receiving a list L Do L(u) :=APPENDTOADJACENCYMATRIX(L);
11    EndRepeat
12 End
```

The local initialization protocol LOCALINIT is the combination of the two protocols CLUSTERING and TOTALKNOWLEDGE

```
1  Procedure LOCALINIT(i : integer)
2  Begin
3     TOTALKNOWLEDGE(i);
4     For each node u belonging to CLUSTER(u) LocID(u) := rank of u
      in the sorted array of IDs of all nodes in CLUSTER(u);
5  End
```

5.8. Paths construction between clusters

Each station u runs TOTALKNOWLEDGE$(4k)$ independently. After what, u owns the adjacency matrix of all its $4k$-hops neighbors. With this information, and the knowledge of all its neighboring clusters, Bellman-Ford algorithm is executed. Every node can thus deterministically build the same routing table between two neighboring clusters; more precisely, between any given pair (s, t) of stations, each within its respective cluster, as described in Fig. 11.

Therefore, the fact that all involved stations do have the same choice of the pair (s, t) is important of course. For example, the first pair of stations (s, t) between two adjacent clusters may be taken as the two smallest stations LOCALIDs in both ones. If such an (s, t)-path exists, the Bellman-Ford algorithm executed by the participating stations finds it.

Observe that the latter algorithm is not run distributively. Besides, the choice of the parameter $4k$ ensures that all these stations have the right required adjacency submatrix (which size is at most $2k \times 2k$).

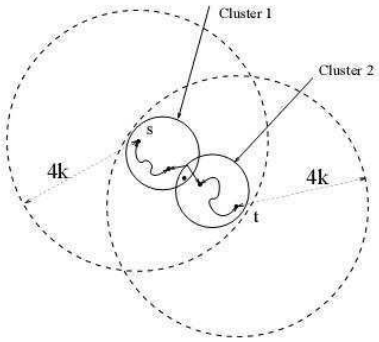


Figure 11. The choice of $i = 4k$ as parameter of TOTALKNOWLEDGE allows the stations to construct all paths deterministically, one after the other, between any two neighboring clusters.

5.9. Gossiping over clusters and main results

Finally, a gossiping algorithm over disjoint paths of length at most $4k$, is performed over the graph of clusters.

As shown in Fig. 12, the communication process between two neighboring clusters is then worked out along the constructed disjoint paths. In order to synchronize the communication between adjacent clusters, we cut up the time into "phases" that are $4k$ rounds long. Each such phase is actually made of an $\mathcal{O}(k)$ communication delay time: it serves as a kind of frame in the swap-over process from a given path to a next disjoint one. The gossiping algorithm is therefore deterministic, and in each round every station knows exactly whether sleeping or communicating.

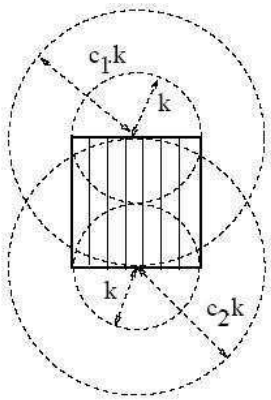


Figure 12. The square of surface $\mathcal{O}\left(k^2r^2\right)$ and its m regular strips frame, that link two half-covered neighboring clusters for swapping over between the disjoint paths

Lemma 15 *Let* CLUSTER(u) *and* CLUSTER(v) *be any two adjacent clusters. W.h.p., there exist at least* $\mathcal{O}(k^2)$ *disjoint paths between* CLUSTER(u) *and* CLUSTER(v).

Proof. (Sketch)
Let $c_1 k$ and $c_2 k$ be the hop-radius of two neighboring clusters. Clearly $1 \leq c_1 \leq 2$ and $1 \leq c_2 \leq 2$. As shown in Fig. 12, there exists a square S of surface $|S| = \mathcal{O}\left(k^2 r^2\right)$ covering half of each two clusters. Split S into m regular (rectangular) strips S_i of equal size $|S_i| = \mathcal{O}\Big(\frac{k^2 r^2}{m}\Big)$ $(i \in [1, m])$, and let N_i be the number of stations within each strip S_i. If $k^2 r^2/m \gg 1$, $\mathbb{E}(N_i) = \mathcal{O}\left(k^2 r^2/m\right) \gg 1$.

Now, by Chernov bounds, we know that there exist two constants ν_i and μ_i for each $1 \leq i \leq m$, such that

$$\mathbb{P}\left(\nu_i \, \frac{k^2 r^2}{m} \leq N_i \leq \mu_i \, \frac{k^2 r^2}{m}\right) \; \geq \; 1 - \exp\!\left(-\mathcal{O}\left(\frac{k^2 r^2}{m}\right)\right).$$

Next, fix $i \in [1, m]$ and denote r_{CON}, the transmission range required to have a connected graph inside the strip S_i. Among other results, Penrose proved in [30] that if $N_i/|S_i| = \mathcal{O}(1)$,

$$\lim_{N_i \to \infty} \mathbb{P}\left(\pi \, \frac{N_i}{|S_i|} \, r_{\text{CON}}^2 \; - \; \log\left(N_i\right) \; \leq \; \omega\right) \; = \; \exp\Big(-e^{-\omega}\Big).$$

Finally, if we let $m = \mathcal{O}\Big(k^2\Big)$, then

$$\frac{\log\left(N_i\right)|S_i|}{N_i} \; = \; \mathcal{O}\left(\log\left(\frac{k^2 r^2}{m}\right)\right) \; = \; \mathcal{O}(\log\log n). \tag{17}$$

In the present case, the transmission radius is such that $r^2 = \mathcal{O}(\log n)$, and therefore, any subgraph within S_i is connected with probability greater than $\exp\left(-n^{\Theta(1)}\right)$. Since the number m of strips is at most polynomial in n, it is growing much slower than the above probability of any subgraph in S_i to be connected; and this holds for all $i = 1, 2, \ldots, m$. Hence, w.h.p. the number of disjoint paths between $\textsc{Cluster}(u)$ and $\textsc{Cluster}(v)$ is at least $\mathcal{O}(k^2)$, and we are done. ■

As an immediate consequence, we have the following main Theorem 16.

Theorem 16 *Let n stations be randomly deployed on a support area X with a linear size $|X| = \mathcal{O}(n)$ and assume the radius of transmission of each station to be $r = \sqrt{(1+\ell)\log(n)|X|/(\pi n)}$.*
For any $k \ll \sqrt{n/\log n}$, the initialization of the stations requires $\mathcal{O}(k\sqrt{n \log n})$ rounds, with no station being awake for more than $\mathcal{O}\left(\max\left(\sqrt{n\log n}/k, k\log n, \log(n)^4\right)\right)$ rounds.

Proof. If each cluster is considered as a graph node, the running time of the initialization protocol is $\mathcal{O}(kD\log n) = \mathcal{O}(k\sqrt{n\log n})$ rounds (where D denotes the hop-diameter of the graph). Swapping over from (disjoint) path to path between adjacent clusters requires that each station is used only every $\mathcal{O}(k^2)$ rounds, and the result follows. ■

Corollary 17 *Under the assumptions of Theorem 16, there exists a randomized initialization protocol running in* $\mathcal{O}\left(n^{3/4}\log(n)^{1/4}\right)$ *rounds, with no station being awake for more than* $\mathcal{O}\left(n^{1/4}\log(n)^{3/4}\right)$ *rounds.*

6. Conclusion

In the present paper, two performing algorithms for solving the initialization problem are designed and analyzed. The random geometric graph model $\mathbb{G}(n,r)$ is used in both algorithms as underlying reachability graph of the wireless network $\mathcal{N}$. $\mathbb{G}(n,r)$ enjoys plenty of fruitful properties for the design and analysis of such algorithm.

The second initialization algorithm presented is time and also energy efficient. Its running time, as well as the awake time per station are both broadly sublinear. More precisely, the time complexity of our algorithm achieves $\mathcal{O}\left(n^{3/4}\log(n)^{1/4}\right)$ rounds, with no station being awake for more than $\mathcal{O}\left(n^{1/4}\log(n)^{3/4}\right)$ rounds.

It is also worth to emphasize the fact that choosing $k = \mathcal{O}(1)$ yields an almost time optimal algorithm. In such a case indeed, the running time shrinks to $\mathcal{O}\left(\sqrt{n\log n}\right)$, whereas the easier broadcast problem requires at most $\Omega\left(\sqrt{\frac{n}{\log n}}\log\log n\right)$ rounds [9, 22]. Hence, our result is at most $\mathcal{O}\left(\frac{\log n}{\log\log n}\right)$ far from optimality.

Finding the lower-bound on the awake time per station for the initializing stations in a random radio network is an open challenging problem. Furthermore, an even more challenging open problem remains of course the design and analysis of an initialization algorithm which could reach the latter lower-bound while keeping a nearly optimal time complexity.

References

[1] M. Abramowitz and I. Stegun, *Handbook of Mathematical Functions*, Dover Publications, 1974.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, *Wireless sensor networks: a survey*, Computer Networks **38** (2002), 393–422.

[3] 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.

[4] R. Bar-Yehuda, O. Goldreich and A. Itai, *Efficient Emulation of Single-Hop Radio Network with Collision Detection on Multi-Hop Radio Network with no Collision Detection*, Distributed Computing **5** (1991), 67–71.

[5] R. Bar-Yehuda, O. Goldreich and A. Itai, *On the Time-Complexity of Broadcast in Multi-Hop Radio Networks: An Exponential Gap between Determinism and Randomization*, Journal of Comp. and Sys. Sciences **45** (1992), 104–126.

[6] A. Cayley, *A Theorem on Trees*, Quart. J. Math. Oxford Ser. **23** (1889), 376–378.

[7] Y.-C. Cheng and T. G. Robertazzi, *Critical connectivity phenomena in multihop radio models*, IEEE Trans. on Communications **36** (1989), 770–777.

[8] B. Chlebus, *Randomized Communication in Radio Networks*, Chapter in "*Handbook of Randomized Computing*," Panos M. Pardalos, Sanguthevar Rajasekaran, John H. Reif, and Jose D.P. Rolim (Eds.), Kluwer Academic, New York, 2001, vol. I, pp. 401-456.

[9] M. Chrobak, L. Gasieniec and W. Rytter, *Fast Broadcasting and Gossiping in Radio Networks*, Proceedings of IEEE Foundations of Computer Sciences (FOCS), 2000.
[10] R. M. Corless, G. H. Gonnet,D. E. G. Hare, D. J. Jeffrey and D. E. Knuth, *On the Lambert W Function*, Advances in Computational Mathematics **5** (1996) 329–359.
[11] N. G. De Bruijn, *Asymptotic Methods in Analysis*, Dover, New-York, 1981.
[12] P. Erdös and A. Rényi *On the evolution of random graphs*, Publ. Math. Inst. Hung. Acad. Sci. **5** (1960), 17–61.
[13] P. Flajolet and R. Sedgewick *Analytic Combinatorics*, Book in preparation. Chapters are available as INRIA research reports. See http://algo.inria.fr/flajolet/books.
[14] E. N. Gilbert, *Random Plane Networks*, Journal of the Society for Industrial and Applied Math **9** (1961), 533–543.
[15] P. Gupta and P. R. Kumar, *Critical power for asymptotic connectivity in wireless networks*, Stochastic Analysis, Control, Optimization and Applications: a volume in honor of W. H. Fleming, W. M. McEneaney, G. Yin and Q. Zhang, Birkhauser, Boston, 1998.
[16] P. Hall, *Introduction to the Theory of Coverage Processes*, Birkhäuser, Boston, 1988.
[17] T. Hayashi, K. Nakano and S. Olariu, *Randomized Initialization Protocols for Packet Radio Networks*, in S. Rajasekaran, P. Pardalos, B. Badrinath, and F. Hsu, Eds., Discrete Mathematics and Theoretical Computer Science, SIAM Press (2000), 221–235.
[18] S. Janson, D. E. Knuth,T. Luczak and B. Pittel, *The birth of the giant component*, Random Structures & Algorithms **4** (1993), 233–358.
[19] E-S Jung and N. Vaidya *A Power Control MAC Protocol for Ad Hoc Networks*, Proceedings of ACM Mobicom'02 (2002), 36–47.
[20] D. E. Knuth, *The Art of Computer Programming - Sorting and Searching, vol 3*. Addison-Wesley, 1973
[21] 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.
[22] D. Liu and M. Prabhakaran, *On Randomized Broadcasting and Gossiping in Radio Networks*, in Proceedings of COCOON'02, Lecture Notes in Computer Sciences, Vol. 2387 (2002), 340–349.
[23] R. Meester and R. Roy *Continuum Percolation*. Cambridge University Press, Cambridge, 1996.
[24] R. E. Miles, *On the Homogenous Planar Poisson Point Process*, Math. Biosciences **6** (1970), 85–127.
[25] S. Muthukrishman and G. Pandurangan, *The Bin-Covering Technique for Thresholding Random Geometric*. Proc. of ACM-SIAM'06 symposium on Disccrete algorithms (2005) 989–998.
[26] K. Nakano and S. Olariu, *Randomized Initialization Protocols for Ad-hoc Networks*, IEEE Transactions on Parallel and Distributed Systems **11** (2000), 749–759.
[27] K. Nakano and S. Olariu, *Energy-Efficient Initialization Protocols for Radio Networks with no Collision Detection*, IEEE Transactions on Parallel and Distributed Systems **11** (2000), 851–863.
[28] M. D. Penrose, *The longest edge of the random minimal spanning tree*, Annals of Applied Probability **7** (1997), 340–361.
[29] M. D. Penrose, *A strong law for the largest nearest-neighbour link between random points*, Journal of the London Mathematical Society **60** (1999), 951–960.
[30] M. D. Penrose, *On k-connectivity for a geometric random graph*, Random Structures & Algorithms **15** (1999), 145–164.
[31] M. D. Penrose, *Random Geometric Graphs*. Oxford Studies in Probability, 2003.
[32] C. E. Perkins, *Ad Hoc Networking*. Addison-Wesley, 2001.
[33] V. Ravelomanana, *Extremal Properties of Three Dimensional Sensor Networks with Applications*, IEEE Trans. on Mobile Computing **3** (2004), 246–257.
[34] V. Ravelomanana, *Optimal Initialization and Gossiping Algorithms for Random Radio Networks*, IEEE Trans. Parallel and Distributed Systems **18** (2007), 1–12.
[35] L. Santalo, *Integral Geometry and Geometric Probability*. Cambridge University Press, 2nd edition, 2003.
[36] P. Santi and D. M. Blough, *The Critical Transmitting Range for Connectivity in Sparse Wireless Ad Hoc Networks*, IEEE Trans. Mob. Comp. **2** (2003), 1–15.
[37] S. Shakkottai, R. Srikant and N. Shroff, *Unreliable Sensor Grids: Coverage, Connectivity and Diameter*, Ad hoc Networks journal **3** (2005), 702–716.
[38] F. Xue and P. R. Kumar, *The number of neighbors needed for connectivity of wireless networks*, Wireless Networks **10** (2004), 169–181.