Energy-Efficient Initialization Protocols for Ad-hoc Radio Networks ⋆

J. L. Bordim[1], J. Cui[1], T. Hayashi[1], K. Nakano[1], and S. Olariu[2]

[1] Department of Electrical and Computer Engineering, Nagoya Institute of Technology, Showa-ku, Nagoya 466-8555, Japan, nakano@elcom.nitech.ac.jp
[2] Department of Computer Science, Old Dominion University, Norfolk, VA 23529, USA, olariu@cs.odu.edu

Abstract. The main contribution of this work is to propose energy-efficient randomized initialization protocols for ad-hoc radio networks (ARN, for short). First, we show that if the number n of stations is known beforehand, the single-channel ARN can be initialized by a protocol that terminates, with high probability, in $O(n)$ time slots with no station being awake for more than $O(\log n)$ time slots. We then go on to address the case where the number n of stations in the ARN is not known beforehand. We begin by discussing, an elegant protocol that provides a tight approximation of n. Interestingly, this protocol terminates, with high probability, in $O((\log n)^2)$ time slots and no station has to be awake for more than $O(\log n)$ time slots. We use this protocol to design an energy-efficient initialization protocol that terminates, with high probability, in $O(n)$ time slots with no station being awake for more than $O(\log n)$ time slots. Finally, we design an energy-efficient initialization protocol for the k-channel ARN that terminates, with high probability, in $O(\frac{n}{k} + \log n)$ time slots, with no station being awake for more than $O(\log n)$ time slots.

1 Introduction

An ad-hoc radio network (ARN, for short) is a distributed system with no central arbiter, consisting of n radio transceivers, henceforth referred to as *stations*. We assume that the stations are identical and cannot be distinguished by serial or manufacturing number. We refer the reader to Figure 1 depicting a 7-station ARN.

As customary, time is assumed to be slotted and all the stations have a local clock that keeps synchronous time, perhaps by interfacing with a GPS system. The stations are assumed to have the computing power of a usual laptop computer; in particular, they all run the same protocol and can generate random bits that provide local data on which the stations may perform computations.

⋆ Work supported in part by ONR grant N00014-91-1-0526 and by Grant-in-Aid for Scientific Research (C) (10680351) from Ministry of Education, Science, Sports, and Culture of Japan.

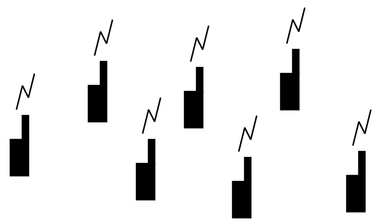

Fig. 1. *Illustrating a 7-station ARN.*

The stations communicate using k radio frequencies (i.e. channels). We assume that in any time slot, a station can tune to one radio channel and/or transmit on at most one, possibly the same, channel. A transmission involves a data packet whose length is such that the transmission can be completed within one time slot.

We employ the commonly-accepted assumption that when two or more stations are transmitting on a channel in the same time slot, the corresponding packets *collide* and are lost. We further assume that the system has *collision detection* capability. Accordingly, at the end of a time slot the status of a radio channel is:
NULL: if no station transmitted on the channel in the current time slot,
SINGLE: if exactly one station transmitted on the channel in the current time slot, and
COLLISION: if two or more stations transmitted on the channel in the current time slot.

We assume that the stations in the ARN run on batteries and saving battery power is exceedingly important as recharging batteries may not be possible while in operation. It is well known that a station expends power while its transceiver is active, that is, while transmitting or receiving a packet. It is perhaps surprising at first that a station expends power even if it receives a packet that is not destined for it. Consequently, we are interested in developing protocols that allow stations to *power off* their transceiver to the largest extent possible. Accordingly, we judge the goodness of a protocol by the following two yardsticks:

- the overall number of time slots required by the protocol to terminate
- for each individual station the total number of time slots when it has to transmit/receive a packet.

The *initialization* problem is to assign to each of the n stations in the ARN an integer ID number in the range $[1..n]$ such that no two stations are assigned the same ID. The initialization problem is fundamental in both network design and in multiprocessor systems [3, 6].

Recent advances in wireless communications and mobile computing have exacerbated the need for efficient protocols for ARNs. Indeed, a large number of such protocols have been reported in the literature [1, 2, 5, 7, 8]. However, many of these protocols function under the assumption that the n stations in the ARN have been initialized in advance. The highly non-trivial task of assigning the stations distinct ID numbers, i.e. initializing the stations, is often ignored in the literature. It is, therefore, of importance to design efficient initialization protocols for ARNs both in the case where the system has a collision detection capability and for the case where this capability is not present.

Recently, Hayashi et al. [3] have reported an initialization protocol for an n-station, k-channel ARN that terminates with high probability in $O(\frac{n}{k})$ time slots, provided that $k \leq \frac{n}{3 \log n}$. However, the protocol of [3] was not designed to be energy-efficient and the stations have to be awake for the entire duration of the protocol.

Our first main contribution is to propose energy-efficient randomized initialization protocols for ARNs. First, we show that if the number n of stations is known beforehand, the single-channel ARN can be initialized by a protocol terminating, with high probability, in $O(n)$ time slots, with no station being awake for more than $O(\log n)$ time slots.

Next, we consider the case where the number n of stations is not known beforehand. The key insight that leads to an energy-efficient solution in this scenario is an elegant strategy for finding a good approximation for n. Specifically, we propose an energy-efficient approximation protocol for n that terminates, with high probability, in $O((\log n)^2)$ time slots with no station being awake for more than $O(\log n)$ time slots. Using our approximation protocol, we show that even if n is not known the single-channel ARN can be initialized by a protocol terminating, with high probability, in $O(n)$ time slots, with no station being awake for more than $O(\log n)$ time slots.

Finally, we extend this protocol to the k-channel ARN. Specifically, we propose an energy-efficient initialization protocol for an n-station, k-channel ARN that terminates, with high probability in $O(\frac{n}{k} + \log n)$ time slots with no station being awake for more than $O(\log n)$ time slots.

2 Basics

The main goal of this section is to review a number of fundamental results in basic probability theory and to discuss simple prefix sums protocols that will be needed in the remainder of the paper.

2.1 A refresher of basic probability theory

Throughout, $\Pr[A]$ will denote the probability of event A. For a random variable X, $E[X]$ denotes the expected value of X. Let X be a random variable denoting the number of successes in n independent Bernoulli trials with parameters p and $1-p$. It is well known that X has a *binomial distribution* and that for every r, $(0 \leq r \leq n)$,

$$\Pr[X = r] \;=\; \binom{n}{r} p^r (1-p)^{n-r}. \tag{1}$$

Further, the expected value of X is given by

$$E[X] = \sum_{r=0}^{n} r \cdot \Pr[X = r] = np. \tag{2}$$

To analyze the tail of the binomial distribution, we shall make use of the following estimate, commonly referred to as *Chernoff bound* [4]:

$$\Pr[X > (1+\delta)E[X]] < \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{E[X]} \quad (0 \le \delta). \tag{3}$$

We also use the following estimates easily derived from (3):

$$\Pr[X \le (1-\epsilon)E[X]] \le e^{-\frac{\epsilon^2}{2}E[X]} \quad (0 \le \epsilon \le 1). \tag{4}$$

$$\Pr[X \ge (1+\epsilon)E[X]] \le e^{-\frac{\epsilon^2}{3}E[X]} \quad (0 \le \epsilon \le 1). \tag{5}$$

2.2 Prefix sums protocols

Suppose that the ARN has n stations each of which has a unique ID in the range $[1..m]$ $(m \ge n)$. Let S_i denote a station with ID i $(1 \le i \le m)$. Note that for some i, S_i may not exist. Suppose that each station S_i has a real number x_i. The *prefix sum* of x_i is the sum of the real numbers with indices no more than i, that is,

$$\sum \{x_j \mid 1 \le j \le i \text{ and } S_j \text{ exists}\}.$$

The *prefix sums problem* asks to compute all prefix sums. A naive protocol can solve the prefix sums problem in $m-1$ time slots in the one-channel ARN as follows: In each time slot i $(1 \le i \le m-1)$, station S_i transmits x_i on the channel, and every station S_j $(i < j \le n)$ monitors the channel. By summing the real numbers received, each station learns the prefix sum. However, this protocol is not energy efficient, because the last station S_m monitors the channel in all of the $m-1$ time slots.

When the protocol terminates the following three conditions are satisfied:

(ps1) Every active station S_i, $(1 \le i \le n)$, stores its prefix sum.
(ps2) The last station, that is, station S_k such that no station S_i with $i > k$ exists, has been identified
(ps3) The protocol takes $2m-2$ time slots and no station is awake for more than $2\log m$ time slots.

If $m = 1$, then S_1 knows x_1 and the above conditions are verified. Now, assume that $m \ge 2$. Partition the n stations into two groups $\mathcal{P}_1 = \{S_i \mid 1 \le i \le \frac{m}{2}\}$ and $\mathcal{P}_2 = \{S_i \mid \frac{m}{2}+1 \le i \le m\}$. Recursively compute the prefix sums in $\mathcal{P}_1$ and $\mathcal{P}_2$. By the induction hypothesis, conditions (ps1)–(ps3) above are satisfied and, therefore, each of the two subproblems can be solved in $m-2$ time slots, with no station being awake for more than $2\log m - 2$ time slots. Let S_j and S_k be the last active stations stations in $\mathcal{P}_1$ and in $\mathcal{P}_2$, respectively. In the next time slot, station S_j transmits the sum $\sum\{x_i \mid 1 \leq i \leq j \text{ and } S_i \text{ exists}\}$ on the channel. Every station in $\mathcal{P}_2$ monitors the channel and updates the value of its prefix sum. In one additional time slot station S_k reveals its identity. The reader should have no difficulty to confirm that the protocol satisfies (ps1)–(ps3) above are satisfied. Thus, we have,

Lemma 1. *The prefix sums problem on the single-channel ARN of n stations with unique ID $[1..m]$ ($m \geq n$) can be solved in $2m-2$ time slots with no station being awake for more than $2\log m$ time slots.*

Next we extend the single-channel prefix sums protocol to a k-channel ARN. We begin by partitioning the stations into k equal-sized subsets $X_1, X_2, \ldots, X_k$ such that $X_i = \{S_j \mid (i-1)\cdot\frac{m}{k}+1 \leq j \leq i\cdot\frac{m}{k} \text{ and } S_j \text{ exists}\}$. By assigning one channel to each subsequence, the prefix sums within each X_i can be computed using the single-channel prefix sums protocol discussed above. This needs $2\frac{m}{k}-2$ time slots with no station being awake for more than $2\log\frac{m}{k}$ time slots. At this point, we have the local sum $\mathrm{sum}(X_i)$ of each X_i and we need to compute the prefix sums of $\mathrm{sum}(X_1), \mathrm{sum}(X_2), \ldots, \mathrm{sum}(X_k)$. This can be done by modifying slightly the single-channel prefix-sums protocol. Recall that in the single-channel protocol, the prefix sums of $\mathcal{P}_1$ is computed recursively, and then that for $\mathcal{P}_2$ is computed recursively. Since k channels are available, the prefix sums of $\mathcal{P}_1$ and $\mathcal{P}_2$ are computed simultaneously using $\frac{k}{2}$ channels each. After that, the overall solution can be obtained in two more time slots. Using this idea, the prefix sums can be computed in $2\frac{m}{k}+2\log k-2$ with no station being awake for more than $2\log\frac{m}{k}+2\log k = 2\log m$ time slots. To summarize, we have proved the following result.

Lemma 2. *The prefix sums problem on the k-channel ARN with n stations each having a unique ID $[1..m]$ ($m \geq n$) can be solved in $2\frac{m}{k}+2\log k-2$ time slots with no station being awake for more than $2\log m$ time slots.*

3 An energy-efficient initialization for known n

Suppose that each station knows the number n of stations. The purpose of this section is to show that with probability exceeding $1-O(n^{-2.67})$ the n-station ARN can be initialized in $O(n)$ time slots with no station being awake for more than $O(\log n)$ time slots.

The protocol uses the initializing protocol discussed in [3] which was not designed under energy efficiency in mind.

Lemma 3. *Even if n is not known beforehand, an n-station, single-channel ARN can be initialized in at most $16n$ time slots with probability at least $1-2^{-1.62n}$.*

In paper [3], the constant factor of the time slots is not evaluated explicitly.

The details of our energy efficient initialization protocol are spelled out as follows.

Protocol `Initialization`

Step 1 Each station selects uniformly at random an integer i in the range $[1..\frac{n}{\log n}]$. Let $h(i)$ denote the set of stations that have selected integer i;

Step 2 Initialize each set $h(i)$ individually in $64 \log n$ time slots;

Step 3 Let N_i denote the number of stations in $h(i)$. By computing the prefix sums of $N_1, N_2, \ldots, N_{\frac{n}{\log n}}$ every station determines, in the obvious way, its global ID within the ARN.

Clearly, Step 1 needs no transmission.

Step 2 can be performed in $64n$ time slots using the protocol for Lemma 3 as follows: the stations in group $h(i)$, $(1 \le i \le \frac{n}{\log n})$, are awake for $64 \log n$ time slots from time slot $(i-1) \cdot 64 \log n + 1$ to time slot $i \cdot 64 \log n$. Outside of this number of slots all the stations in group $h(i)$ are asleep and consume no power.

As we are going to show, with high probability, every group $h(i)$ contains at most $4 \log n$ stations. To see that this is the case, observe that the expected number of stations in $h(i)$ is $E[|h(i)|] = n \times \frac{\log n}{n} = \log n$. Now, using the Chernoff bound in (3), we can write

$$\begin{aligned} \Pr[|h(i)| > 4 \log n] &= \Pr[|h(i)| > (1+3)E[|h(i)|]] \\ &< (\frac{e^3}{4^4})^{E[|h(i)|]} \quad \text{(by (3) with } \delta = 3) \\ &< n^{-3.67} \quad (\text{since } \log \tfrac{e^3}{4^4} = -3.67...) \end{aligned}$$

It follows that the probability that group $h(i)$ contains more than $4 \log n$ stations is bounded above by $n^{-3.67}$. Hence, with probability exceeding $1 - n \cdot n^{-3.67} = 1 - n^{-2.67}$, none of the groups $h(1), h(2), \ldots, h(\frac{n}{\log n})$ contains more than $4 \log n$ stations. By Lemma 3, with probability exceeding $1 - 2^{-1.62 \times 4 \log n} = 1 - n^{-6.48}$ the stations in group $h(i)$ can be initialized in $16 \times 4 \log n = 64 \log n$ time slots. Thus, with probability exceeding $1 - n^{-5.48} - n^{-2.67} > 1 - O(n^{-2.67})$ all the groups $h(i)$ will be initialized individually in $64 \log n \times \frac{n}{\log n} = 64n$ time slots, with no station being awake for more than $64 \log n$ time slots.

Let P_i, $(1 \le i \le \frac{n}{\log n})$, denote the last station in group $h(i)$. At the end of Step 2, each station P_i knows the number N_i of stations in $h(i)$. Step 3 can use the stations P_i to compute the prefix sums of $N_1, N_2, \ldots, N_{\frac{n}{\log n}}$. The prefix sums protocol discussed in Section 2 will terminate in $2\frac{n}{\log n} - 2 < 2n$ time slots with no station being awake for more than $2 \log \frac{n}{\log n} < 2 \log n$ time slots. To summarize, we have proved the following result.

Theorem 4. *If the number n of stations is known beforehand, with probability exceeding $1 - O(n^{-2.67})$, an n-station, single-channel ARN can be initialized in $O(n)$ time slots, with no station being awake for more than $O(\log n)$ time slots.*

4 Finding a good approximation of n

At the heart of our energy-efficient initialization protocol of an ARN where the number n of stations is not known beforehand lies a simple and elegant approximation protocol for n. Specifically, this protocol returns an integer I satisfying, with probability at least $1 - O(n^{-1.83})$, the double inequality:

$$\log n - \log\log n - 4 < I < \log n - \log\log n + 1. \tag{6}$$

Once a good approximation of n is available, we can use the protocol discussed in Section 3 to initialize the entire ARN.

The details of our approximation protocol is spelled out as follows:

Protocol `Approximation`
Each station generates uniformly at random a real number x in $(0, 1]$
Let $g(i)$, $(i \geq 1)$, denote the group of stations for which $\frac{1}{2^i} < x \leq \frac{1}{2^{i-1}}$.
for $i \leftarrow 1$ **to** ∞ **do**
  protocol `Initialization` is run on group $g(i)$ for $128i$ time slots;
  if the initialization is complete and if $g(i)$ contains at most $8i$ stations
  then the first station in $g(i)$ transmits an "exit" signal.
end for

Let us first evaluate the number of time slots it takes protocol `Approximation` to terminate. Let I be the value of i when the **for**-loop is exited. One iteration of the for-loop takes $128i + 1$ time slots. Thus, the total number of time slots is at most

$$\sum_{i=1}^{I} 128i + 1 < 64(I + 1)^2.$$

Next, we evaluate for each station the maximum number of time slots during which it has to be awake. Clearly, each station belongs to exactly one of the groups $g(1), g(2), \ldots$, and therefore, every station is awake for $128i \leq 128I$ time slots. Further, all the stations must monitor the channel to check for the "exit" signal. Of course, this takes an additional I time slots. Thus, no station needs to be awake for more than $129I$ time slots.

Our next task is to show that, with high probability, I satisfies (6). For this purpose we rely on the following technical results.

Lemma 5. *If i satisfies $1 \leq i \leq \log n - \log\log n - 4$ then* $\Pr[|g(i)| > 8i] > 1 - n^{-2.88}$.

Proof. Clearly, $i \leq \log n - \log\log n - 4$ implies that

$$2^i \leq \frac{n}{16 \log n} \text{ and similarly } 16i < 16 \log n \leq \frac{n}{2^i}.$$

Since $g(i)$ consists of stations generating a real number x satisfying $\frac{1}{2^i} < x \leq \frac{1}{2^{i-1}}$ the expected number of stations is $E[|g(i)|] = \frac{n}{2^i}$. Using the Chernoff bound in

(4) with $\epsilon = \frac{1}{2}$, we can evaluate the probability $\Pr[|g(i)| < 8i]$ that group $g(i)$ contains strictly less than $8i$ stations as follows:

$$\begin{aligned}\Pr[|g(i)| < 8i] &< \Pr[|g(i)| < (1-\frac{1}{2})\frac{n}{2^i}] \quad (\text{since } 16i < \tfrac{n}{2^i})\\ &< e^{-(\frac{1}{2})^3\frac{n}{2^i}} \quad (\text{by (4) with } \epsilon = \tfrac{1}{2})\\ &< e^{-2\log n} \quad (\text{since } 16\log n \le \tfrac{n}{2^i})\\ &< n^{-2.88} \quad (\text{since } \log(e^{-2}) = -2.88\ldots)\end{aligned}$$

as claimed.

Lemma 6. *If* $i = \lfloor \log n - \log\log n \rfloor + 1$ *then* $\Pr[|g(i)| < 8i] > 1 - n^{-1.83}$.

Proof. If $i = \lfloor \log n - \log\log n \rfloor + 1$ then clearly,

$$\log n - \log\log n < i \le \log n - \log\log n + 1.$$

This also implies that

$$\frac{n}{\log n} < 2^i \le \frac{2n}{\log n} \text{ and that } \frac{i}{2} \le \frac{\log n}{2} \le \frac{n}{2^i} < \log n < i + \log\log n < 2i.$$

Using these relations, we can evaluate the probability $\Pr[|g(i)| > 8i]$ that group $g(i)$ contains strictly more than $8i$ stations as follows:

$$\begin{aligned}\Pr[|g(i)| > 8i] &< \Pr[|g(i)| > 4\frac{n}{2^i}] \quad (\text{since } \tfrac{n}{2^i} < 2i)\\ &< (\frac{e^3}{4^4})^{\frac{n}{2^i}} \quad (\text{from (3) with } \delta = 3)\\ &\le (\frac{e^3}{4^4})^{\frac{\log n}{2}} \quad (\text{from } \tfrac{\log n}{2} \le \tfrac{n}{2^i})\\ &< n^{-1.83}. \quad (\text{from } \log(\tfrac{e^3}{4^4})^{\frac{1}{2}} = -1.83\ldots)\end{aligned}$$

This completes the proof.

Lemma 5 and 6 combined, yield the following result.

Lemma 7. *The value I of i when the* **for**-*loop is exited satisfies, with probability at least* $1 - O(n^{-1.83})$, *condition (6).*

Thus, we have proved the following important result.

Theorem 8. *Protocol* `Approximation` *returns an integer I satisfying, with probability at least* $1 - O(n^{-1.83})$, *condition (6). Moreover, the protocol terminates in* $64(\log n)^2$ *time slots with no station is awake for more than* $129\log n$ *time slots.*

5 An energy-efficient initialization for unknown n

The main goal of this section is to present an energy-efficient initialization protocol in the case where the number of stations in the ARN is not known.

Recall that protocol `Initialization`, partitions the n stations into $\frac{n}{\log n}$ groups and initializes each group independently. Since n is unknown, this partitioning cannot be done. Instead, using protocol `Approximation`, we find an integer I satisfying condition (6) with probability at least $1 - O(n^{-1.83})$. After that, execute `Initialization` with 2^{I+4} groups $h(1), h(2), \ldots, h(2^{I+4})$. Observe that condition (6) is satisfied. Then, we can write

$$\frac{\log n}{32} < \frac{n}{2^{I+4}} < \log n. \tag{7}$$

In this case by Theorem 4, with probability $1 - O(n^{-3.67})$, $64 \log n$ time slots are sufficient to initialize a particular $h(i)$. However, since n is unknown, we cannot arrange $64 \log n$ time slots to each group. Instead, we assign $128(I + 4)$ ($> 64 \log n$) time slots. As a result, with probability at least $1 - O(2^{I+4} \cdot n^{-3.67}) > 1 - O(n^{-2.67})$, all of the groups can be initialized locally in $2^{I+4} \cdot 128(I+4) = O(n)$ time slots with no station being awake for more than $128(I+4) = O(\log n)$ time slots. After that, the prefix sums are computed in $2 \cdot 2^{I+4} - 2 = O(\frac{n}{\log n})$ time slots with no station being awake for more than $2 \log(2^{I+4}) = O(\log n)$ time slots. Since condition (6) is satisfied with probability at least $1 - O(n^{-1.83})$, we have

Theorem 9. *Even if the number n of stations is not known beforehand, with probability exceeding $1 - O(n^{-1.83})$, an n-station, single-channel ARN can be initialized in $O(n)$ time slots, with no station being awake for more than $O(\log n)$ time slots.*

6 An energy-efficient initialization for the k-channel ARN and unknown n

The main purpose of this section is to present an energy-efficient initialization protocol for the n-station k-channel ARN, when n is not known beforehand. Let $\mathrm{C}(1), \mathrm{C}(2), \ldots, \mathrm{C}(k)$ denote the k channels available in the ARN.

The idea is to parallelize the protocol in Section 5 to take advantage of the k channels. Recall that this initialization protocol first executes protocol `Approximation` and then execute protocol `Initialization`.

We first extend protocol `Approximation` for the case where k channels are available. Having determined groups $g(1), g(2), \ldots, g(I)$, we allocate groups to channels. For every i, $(1 \leq i \leq k)$, we allocate group $g(i)$ to channel $\mathrm{C}(i)$ and attempt to initialize group $g(i)$ in $128i$ time slots. If none of these attempts is successful, we allocate the next set of k groups $g(k + 1), g(k + 2), \ldots, g(2k)$ to the k channels in a similar fashion. This is then continued, as described, until eventually one of the groups is successfully initialized. We now estimate the number of time slots required to get the desired value of I. Let c be the integer satisfying $ck+1 \leq I \leq (c+1)k$. In other words, in the c-th iteration, I is found. If $c \geq 1$, then the total number of time slots is

$$128k + 128(2k) + 128(3k) + \cdots + 128(ck) = O(c^2 k) = O\left(\frac{I^2}{k}\right) = O\left(\frac{(\log n)^2}{k}\right).$$

If $c = 0$, then the total number of time slots is $128(I + 4) = O(\log n)$. Therefore, the k channel version of `Approximation` terminates, with high probability, in $O\left(\frac{(\log n)^2}{k} + \log n\right)$ time slots, with no station being awake for more than $O(\log n)$ time slots.

Similarly, we can extend protocol `Initialization` for the k-channel case as follows: Recall that we need to initialize each of the groups $h(1), h(2), \ldots, h(2^{I+4})$. This task can be extended as follows: Since we have k channels, $\frac{2^{I+4}}{k}$ groups can be assigned to each channel and be initialized efficiently. Since each group $h(j)$ can be initialized in $128(I+4)$ time slots, all of the 2^{I+4} groups can be initialized in $128(I + 4) \cdot \frac{2^I}{k} = O(\frac{n}{k})$ time slots, with no station being awake no more than $O(\log n)$ time slots. After that we use the k-channel version of the prefix sums protocol discussed in Section 2. This takes $2\frac{2^{I+4}}{k} + 2\log k - 2 < \frac{64n}{k \log n} + 2\log k$ time slots, with no station being awake for more than $2\log\frac{2^I}{k} + 2\log k$ time slots. Therefore, we have the following result:

Theorem 10. *Even if the number n of stations is unknown, with probability exceeding $1-O(n^{-1.83})$, an n-station, k-channel ARN can be initialized in $O(\frac{n}{k} + \log n)$ time slots, with no station being awake for more than $O(\log n)$ time slots.*

References

1. N. Alon, A. Bar-Noy, N. Linial, and D. Peleg, Single-round simulation on radio networks, *Journal of Algorithms*, 13, (1992), 188–210.
2. R. Dechter and L. Kleinrock, Broadcast communication and distributed algorithms, *IEEE Transactions on Computers*, C-35, (1986), 210–219.
3. T. Hayashi, K. Nakano, and S. Olariu, Randomized initialization protocols for Packet Radio Networks, *Proc. 13th International Parallel Processing Symposium*, (1999), 544–548.
4. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
5. S. Murthy and J. J. Garcia-Luna-Aceves, A routing protocol for packet radio networks, *Proc. First Annual ACM Conference on Mobile Computing and Networking*, Berkeley, California, 1995, 86–95.
6. K. Nakano, Optimal initializing algorithms for a reconfigurable mesh, *Journal of Parallel and Distributed Computing*, 24, (1995), 218-223.
7. K. Nakano, Optimal sorting algorithms on bus-connected processor arrays, *IEICE Transactions Fundamentals*, E-76A, 11, (1994), 2008–2015.
8. K. Nakano, S. Olariu, and J. L. Schwing, Broadcast-efficient protocols for ad-hoc radio networks with few channels, *IEEE Transactions on parallel and Distributed Systems*, to appear.