PAPER Special Section on Discrete Mathematics and Its Applications

An Energy Efficient Leader Election Protocol for Radio Network with a Single Transceiver

Jacir Luiz BORDIM[†], **Yasuaki ITO**[††], *and* **Koji NAKANO**[††a)], ***Members***

SUMMARY In this work we present an energy efficient leader election protocol for anonymous radio network populated with n mobile stations. Previously, Nakano and Olariu have presented a leader election protocol that terminates, with probability exceeding $1 - \frac{1}{f}$ ($f \geq 1$), in $\log\log n + o(\log\log n) + O(\log f)$ time slots [14]. As the above protocol works under the assumption that every station has the ability to transmit and monitor the channel at the same time, it requires every station to be equipped with two transceivers. This assumption, however, is unrealistic for most mobile stations due to constraints in cost, size, and energy dissipation. Our main contribution is to show that it is possible to elect a leader in an anonymous radio network where each station is equipped with a single transceiver. Quite surprisingly, although every station has only one transceiver, our leader election protocol still runs, with probability exceeding $1 - \frac{1}{f}$ ($f \geq 1$), in $\log\log n + o(\log\log n) + O(\log f)$ time slots. Moreover, our leader election protocol needs only expected $O(n)$ total awake time slots, while Nakano and Olariu's protocol needs expected $O(n \log\log n)$ total awake time slots. Since every leader election protocol needs at least $\Omega(n)$ awake time slots, our leader election protocol is optimal in terms of the expected awake time slots.

key words: *adhoc networks, collision detection, distributed algorithms, randomized algorithms*

1. Introduction

A radio network (RN, for short) is a distributed system with no central arbiter, consisting of n radio transceivers, henceforth referred to as *stations*. In a *single-channel* RN the stations communicate over a common radio frequency channel, which is known to all the stations. An RN is said to be *single-hop* when all the stations are within transmission range of each other. In this work we focus on single-channel, single-hop radio networks. Single-hop radio networks are the basic ingredients out of which larger, multi-hop radio networks are built [3], [18]. As customary, time is assumed slotted and all transmissions are edge-triggered, that is, take place at time slot boundaries [3], [5], [8]. In a time slot, a station can transmit or listen to the channel using a transceiver. Note that, a transceiver can perform one of the transmitting and listening operations in a time slot. Should a station need to do both operations at the same time, two transceivers are necessary. However, this assumption is unrealistic as most mobile devices are usually equipped with a single transceiver due to stringent constraints in size and power consumption.

In this work, the stations are assumed to have a local clock that keeps synchronous time, perhaps by interfacing with a Global Positioning System (GPS, for short) [6], [15], [16]. It is well documented that GPS systems using military codes achieve a level of accuracy that is orders of magnitude better than their commercial counterparts [6]. In particular, this allows the stations to detect time slot boundaries and, thus, to synchronize.

We employ the commonly-accepted assumption that when two or more stations are transmitting on a channel at the same time slot, the corresponding packets *collide* and are garbled beyond recognition. It is customary to distinguish among radio networks in terms of their *collision detection* capabilities. In the RN with collision detection the status of a radio channel in a time slot is:

NULL: if no station transmitted in the current time slot,
SINGLE: if exactly one station transmitted in the current time slot,
COLLISION: if two or more stations transmitted the channel in the current time slot.

Note that, if a station has two transceivers, it can send a packet and can detect the status of the channel in the same time slot. However, if a station with a single transceiver sends a packet, it cannot detect the status of the channel.

It is well known that a station expends power while its transceiver is active, that is, while sending or receiving a packet. As mobile stations run on batteries, saving battery power is exceedingly important as recharging them may not be possible while on mission. Thus, in terms of battery consumption, the transmission of a packet and channel monitoring are costly operations. Consequently, we are interested in developing protocols that allow the stations to power their transceiver off (i.e. go to sleep) to the largest extent possible, so as to save energy. We estimate the goodness of a algorithm by the following two yardsticks: *running time slots*: the overall number of time slots required by the algorithm to terminate; *awake time slots*: for each individual station the number of time slots when it has to be *awake* in order to transmit a packet or listening to the channel; and *total awake time slots*: the sum of awake time slots over all stations.

The problem that we address in this work is the classical *leader election* problem which asks to designate one of the stations in the network as *leader*. In other words, after

Manuscript received August 22, 2005.
Manuscript revised November 6, 2005.
Final manuscript received December 15, 2005.

[†]The author is with the Department of Computer Science, University of Brasilia, 70910-900 Brasilia - DF, Brazil.
[††]The authors are with the School of Engineering, Hiroshima University, Higashihiroshima-shi, 739-8527 Japan.
a) E-mail: nakano@hiroshima-u.ac.jp

DOI: 10.1093/ietfec/e89–a.5.1355

executing the leader election protocol, exactly one station learns that it was an elected leader. Historically, the leader election problem was addressed in wired networks [1], [2], [7], [9], [17], where each station can specify a destination station.

The leader election problem can be studied in the following three scenarios:

Scenario 1: The number n of stations is known in advance;
Scenario 2: The number n of stations is unknown, but an upper bound u on n is known in advance;
Scenario 3: Neither the number of stations nor an upper bound on this number is known in advance.

It is intuitively clear that the task of leader election is the easiest in Scenario 1 and the hardest in Scenario 3, with Scenario 2 being in-between the two.

Several randomized protocols for single-channel, single-hop networks have been presented in the literature. Metcalfe and Boggs [10] presented a leader election protocol for Scenario 1 that is guaranteed to terminate in $O(1)$ expected time slots. Their protocol is very simple: every station keeps transmitting on the channel with probability $\frac{1}{n}$. When the status of channel becomes SINGLE, the unique station that has transmitted is declared the leader. Later, Nakano and Olariu [12] presented two leader election protocols for Scenario 3. The first one terminates, with probability $1 - \frac{1}{n}$, in $O(\log n)$ time slots[†]. Nakano and Olariu [13] also presented a leader election protocol for Scenario 3 terminating with probability at least $1 - \frac{1}{f}$, in $O(\min((\log n)^2 + (\log f)^2, f^{\frac{2}{5}} \log n))$ time slots.

In a landmark paper, Willard [18] presented a leader election protocol for the conditions of Scenario 2 terminating in $\log\log u + O(1)$ expected time slots. Willard's protocol involves two stages: the first stage, using binary search, guesses in $\log\log u$ time slots a number i, $(0 \le i \le \log u)$, satisfying $2^i \le n < 2^{i+1}$. Once this approximation for n is available, the second stage elects a leader in $O(1)$ expected time slots using the protocol of [10]. Thus, the protocol elects a leader in $\log\log u + O(1)$ expected time slots. Willard [18] went on to improve this protocol to run under the conditions of Scenario 3 in $\log\log n + o(\log\log n)$ expected time slots. The first stage of the improved protocol uses the technique presented in Bentley and Yao [4], which finds an integer i satisfying $2^i \le n < 2^{i+1}$, sidestepping the need for a known upper-bound u on n. Nakano and Olariu [14] improved Willard's protocol and proposed a leader election protocol terminating, with probability exceeding $1 - \frac{1}{f}$, in $\log\log n + o(\log\log n) + O(\log f)$ time slots. However, in these protocols every station must be equipped with two transceivers, because a station sending a packet need to detect the status of the channel. Also, since every station must be awake for all the time slots, their protocol is not energy efficient.

Our main contribution is to show a leader election protocol under the assumption that every station has a single transceiver. Hence, it is not possible to transmit a packet and detect the status of the channel at the same time. Quite surprisingly, although every station has only one transceiver our leader election protocol still runs, with probability exceeding $1 - \frac{1}{f}$ $(f \ge 1)$, in $\log\log n + o(\log\log n) + O(\log f)$ time slots. Also, our leader election protocol needs only expected $O(n)$ total awake time slots, while Nakano and Olariu's protocol needs expected $O(n \log\log n)$ total awake time slots. Since every station must be awake for at least one time slot, every leader election protocol with n stations needs $\Omega(n)$ awake time slots. Thus, our leader election protocol is optimal in terms of the expected awake time slots.

The remainder of this paper is organized as follows. Section 2 offers a brief refresher of basic probability theory results. Section 3 shows an idea to implement leader election protocols in radio networks with every station being equipped with a single transceiver. Section 4 develops a leader election protocol that terminates with probability exceeding $1 - \frac{1}{f}$ $(f \ge 1)$, in $\log\log n + o(\log\log n) + O(\log f)$ time slots. We also proved this protocol needs only expected $O(n)$ total awake time slots. Finally, Sect. 5 offers concluding remarks and directions for further investigations.

2. A Brief Refresher of Probability Theory

The main goal of this section is to review elementary probability theory results that are useful for analyzing the performance of our protocols. For a more detailed discussion of background material we refer the reader to [11].

Throughout, $\Pr[A]$ will denote the probability of event A. Let $E_1, E_2, \ldots, E_m$ be arbitrary events over a sample space. It is clear that, by using the well known De Morgan law,

$$\Pr[E_1 \cap E_2 \cap \cdots \cap E_k] \ge 1 - \sum_{i=1}^{m} \Pr[\overline{E_i}]. \tag{1}$$

Notice that (1) holds regardless of whether or not the events E_i are independent.

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 parameter p. It is well known that X has a *binomial distribution* and that for every integer r, $(0 \le r \le n)$,

$$\Pr[X = r] = \binom{n}{r} p^r (1-p)^{n-r}. \tag{2}$$

Thus, the probability that $r = 0$ is at most

$$\Pr[X = 0] = (1-p)^n < e^{-np}. \tag{3}$$

To analyze the tail of the binomial distribution, we shall make use of the following estimates, commonly referred to as *Chernoff bound* [11]:

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

Let X be a random variable assuming only nonnegative values. The following inequality, known as the *Markov*

[†]In this paper, log and ln are used to denote the logarithms to the base 2 and e, respectively.

inequality, will be also used

$$\Pr[X \geq c \cdot E[X]] \leq \frac{1}{c} \qquad \text{for all } c \geq 1. \tag{5}$$

To evaluate the expected value of a random variable, we state the following lemma.

Lemma 1: Let X be a random variable taking a value smaller than or equal to $T(F)$ with probability at least F, $(0 \leq F \leq 1)$, where T is a non-decreasing function. Then, $E[X] \leq \int_0^1 T(F)dF$.

Proof. Let k be any positive integer. Clearly, X is no more than $T(\frac{i}{k})$ with probability $\frac{i}{k}$ for every i $(1 \leq i \leq k)$. Thus, the expected value of X is bounded by

$$E[X] \leq \sum_{i=1}^{k} \left(\frac{i}{k} - \frac{i-1}{k}\right) T\left(\frac{i}{k}\right) = \sum_{i=1}^{k} \frac{1}{k} T\left(\frac{i}{k}\right).$$

As $k \to \infty$, we have $E[X] \leq \int_0^1 T(F)dF$. □

For later reference, we state the following corollary.

Corollary 2: Let X be a random variable taking a value no more than $\log f$ with probability at least $1 - \frac{1}{f}$. Then, $E[X] \leq 1$.

Proof. Let $F = 1 - \frac{1}{f}$ and apply Lemma 1. we have

$$E[X] \leq \int_0^1 \log \frac{1}{F} dF = [F - F \log F]_0^1 = 1.$$

□

3. Simulating Two Transceivers by One Transceiver

This section is to show an idea to implement a leader election protocol to run in radio networks with each station being equipped with one transceiver.

Let U be a set of two or more stations (i.e. $|U| \geq 2$), S be a subset of U. Note that either S or $U - S$ can be empty.

Suppose that all the stations broadcast on the channel. If the stations have two transceivers, all the stations can listen to the channel and can detect the status of the channel. Thus, all the stations can learn if $|S| = 0$, $|S| = 1$, or $|S| \geq 2$. However, if every station has a single transceiver this is not possible. Only the stations in $U - S$ can listen to the channel and can learn if $|S| = 0$, $|S| = 1$, or $|S| \geq 2$.

In most leader election protocols, it is a key ingredient that all stations in U learn the status of the channel. We are going to show that, in two time slots, all stations in U $(|U| \geq 2)$ can learn if $|S| = 0$, $|S| = 1$, or $|S| \geq 2$. The protocol is as follows:

Protocol `Simulation2(U,S)`

Time Slot 1 Every station in S transmits on the channel and every station in $U - S$ listen to the channel.

Time Slot 2 Every station in $U - S$ transmits on the channel if the status of the channel was SINGLE in Time Slot 1.

Table 1 Possible channel status of the two time slots of `Simulation2(U, S)`.

$\|S\|$	$\|U - S\|$	Time Slot 1	Time Slot 2
0	≥ 2	NULL	NULL
1	1	SINGLE	SINGLE
1	≥ 2	SINGLE	COLLISION
≥ 2	1	COLLISION	NULL
≥ 2	0	COLLISION	NULL
≥ 2	≥ 2	COLLISION	NULL

Clearly, at the end of time slot 1, all the stations in $U-S$ know if $|S| = 0$, $|S| = 1$, or $|S| \geq 2$. Stations in S need to determine if $|S| = 0$, $|S| = 1$, or $|S| \geq 2$ by the status of the channel in time slots 2. Table 1 shows all possible channel status in the two time slots. From the table, it is easy to confirm that, if the status of the channel in time slot 2 is either SINGLE or COLLISION then $|S| = 1$. If it is NULL, then either $|S| = 0$ or $|S| \geq 2$. Clearly, if $|S| = 0$, then S has no station that needs to learn $|S|$. Thus, stations in S can learn that $|S| \geq 2$ if the status of the channel at time slot 2 is NULL.

Thus, we have,

Lemma 3: Let U be a set of two or more stations, and S be a subset of U. All stations in U can learn, in two time slots, if $|S| = 0$, $|S| = 1$, and $|S| \geq 2$.

Note that if $|U| = 1$ it is impossible to learn if $|S| = 0$, $|S| = 1$, and $|S| \geq 2$.

Next, we will show that if we do not have to distinguish $|S| = 1$ and $|S| \geq 2$, then one time slot is sufficient. In other words, we have,

Lemma 4: Let U be a set of one or more stations, and S be a subset of U. All stations in U can learn, in one time slot, if $|S| = 0$ or $|S| \geq 1$.

This can be archived by executing `Simulation1(U,S)` as follows:

Protocol `Simulation1(U,S)`

Time Slot 1 Every station in S transmits on the channel and every station in $U - S$ listen to the channel.

It is obvious that if S has at least one station, then it knows that $|S| \geq 1$. Hence, it is not necessary for the stations in S to learn $|S|$ from $U - S$.

4. Our Leader Election Protocol

The main purpose of this section is to develop a leader election protocol in a radio network with n stations that terminates, with probability exceeding $1 - \frac{1}{f}$, in $\log\log n + o(\log\log n) + O(\log f)$ time slots, where $f \geq 1$ is an arbitrary parameter.

Our protocol repeatedly performs "Sieve" operation until a leader is elected. Initially, every station is active. In the "Sieve" operation, every station flips a biased coin and belongs to S with a certain probability. If $|S| = 1$, a unique station in S is elected as a leader. If $|S| \geq 2$, then all stations

in S remain active and the other stations go asleep. If $|S| = 0$ then all stations remain active. By repeating this "Sieve" operation, the number of active station decreases, and at some point, a leader is elected. The next section shows the details of our leader election protocol.

4.1 Our Protocol and Its Correctness

Let U be a set of all n stations. We assume that $n \geq 2$, because leader election is not possible if $n = 1$. If $n = 1$, then the channel status is always NULL when the station listen to the channel. Hence, the station cannot distinguish between (1) no station transmits and (2) no station exists.

We begin by presenting two simple protocols, `Sieve2(`p`)` and `Sieve1(`p`)`, ($p \geq 0$). These protocols are the core of our leader election protocol. Let U be the set of current active stations.

Protocol `Sieve2(`p`)`**:** Every active station flips a biased coin and belongs to S with probability $\frac{1}{2^p}$. By using `Simulation2(`U, S`)`, every station learns that $|S| = 0$, $|S| = 1$, or $|S| \geq 2$, in two time slots. If $|S| = 0$, all active stations remain active. If $|S| = 1$, a single station in S is declared as a leader and all the other stations go asleep. If $|S| \geq 2$, all the stations in S remain active, and the other stations go asleep.

Protocol `Sieve1(`p`)`**:** Every active station flips a biased coin and belongs to S with probability $\frac{1}{2^p}$. By using `Simulation1(`U, S`)`, every station learns if $|S| = 0$, or $|S| \geq 1$, in one time slots. If $|S| = 0$, all active stations remain active. If $|S| \geq 1$, all the stations in S remain active, and the other stations go asleep.

Clearly, `Sieve2(`p`)` and `Sieve1(`p`)` run in two and one time slots respectively, and $|U|$ must be ≥ 2 when `Sieve2(`p`)` is called and must be ≥ 1 when `Sieve1(`p`)` is called.

In our leader election protocol, all stations are initially active. At the end of the protocol, exactly one station remain active and is elected as a leader. The other stations are asleep. The protocol proceeds in three phases as follows:

Protocol Leader-Election

Phase 1 `Sieve2(`2^{0^2}`)`, `Sieve2(`2^{1^2}`)`, `Sieve2(`2^{2^2}`)`, ..., `Sieve2(`2^{t^2}`)` are performed until, for the first time, $|S| \leq 1$ in `Sieve2(`2^{t^2}`)`. If $|S| = 1$, then the station in S is declared as a leader. If $|S| = 0$, then let U' be the set of active stations after `Sieve2(`2^{t^2}`)` is called.

Phase 2 `Sieve1(`2^{t^2-1}`)`, `Sieve1(`2^{t^2-2}`)`, `Sieve1(`2^{t^2-3}`)`, ..., `Sieve1(`2^0`)` are performed.

Phase 3 Let S' be the set of current active stations. Stations in U' temporary wake up in two time slots, and `Simulation2(`U', S'`)` is executed to check if $|S'| = 1$. If so, the unique station in S' is declared as a leader. Otherwise, `Sieve2(1)` is repeated until, eventually, $|S| = 1$, at which point a leader has been elected.

Let us see that Leader-Election correctly finds a leader. It should be clear that if $|S| = 1$ in the last call `Sieve2(`2^{t^2}`)` in Phase 1, a leader is elected correctly. If $|S| = 0$ in `Sieve2(`2^{t^2}`)`, then $|S|$ must be ≥ 2 for all calls `Sieve2(`2^{0^2}`)`, ..., `Sieve2(`$2^{(t-1)^2}$`)` in Phase 1. Also, note that if $|S| = 0$ in the call `Sieve2(`2^{t^2}`)`, all the active station before this call remain active after this call. Thus, we have $|U'| \geq 2$. In all calls of `Sieve1` of Phase 2, all active stations remain active if $|S| = 0$, stations in S remain active if $|S| \geq 1$. Hence, $|S'| \geq 1$ when Phase 3 starts. Note that if $|S'| = 1$, `Sieve2` does not work (See Lemma 3). Therefore, at the beginning of Phase 3, `Simulation2(`U', S'`)` is called to determine if $|S'| = 1$ or $|S'| \geq 2$. Also, stations in U' know the value of t, and Phase 2 always runs in t^2 time slot, hence, they know when they must be awake for `Simulation2(`U', S'`)`. If $|S'| = 1$, then a leader is elected. If $|S'| \geq 2$, iterative calls of `Sieve2(1)` can elect a leader. Consequently, Leader-Election always finds a leader.

4.2 The Time Complexity of Our Protocol

We now turn to the task of evaluating the number of time slots it takes the protocol to terminate. Phase 1 terminates when `Sieve2` returns $|S| \leq 1$. Thus, there exist an integer t such that the status of the channel is:

- $|S| \geq 2$ in the calls `Sieve2(`2^{0^2}`)`, `Sieve2(`2^{1^2}`)`, `Sieve2(`2^{2^2}`)`, ..., `Sieve2(`$2^{(t-1)^2}$`)`, and
- $|S| \leq 1$ in the call `Sieve2(`2^{t^2}`)`.

Let $f \geq 1$ be an arbitrary real number and write

$$s = \lceil \sqrt{\log\log(2nf)} \rceil. \tag{6}$$

To motivate the choice of s in (6) we show that with probability exceeding $1 - \frac{1}{4f}$, s provides an upper bound on t. Suppose that we have n' ($\leq n$) active stations when `Sieve2(`2^{s^2}`)` is called. Let X be the random variable denoting the number of stations in S in this call. Clearly, the expected value $E[X]$ of X is at most

$$E[X] < \frac{n'}{2^{2^{s^2}}} \leq \frac{n}{2nf} = \frac{1}{2f}. \tag{7}$$

Using the Markov inequality (5) and (7) combined, we can write

$$\Pr[X \geq 2] < \Pr[X \geq 4fE[X]] \leq \frac{1}{4f}.$$

This implies that with probability exceeding $1 - \frac{1}{4f}$, $|S| \leq 1$ in the call `Sieve2(`2^{s^2}`)` confirming that

$$t \leq s \text{ holds with probability exceeding } 1 - \frac{1}{4f}.$$

Recall that each call of `Sieve2` and `Sieve1` need two and one time slots, respectively. Thus, with probability exceeding $1 - \frac{1}{4f}$, Phase 1 terminates in $2(t+1) \leq 2(s+1) = 2\lceil \sqrt{\log\log(2nf)} \rceil + 2 = 2\sqrt{\log\log n} + O(\log\log f)$ time slots. Since Phase 2 terminates in at most $s^2 = \log\log n + O(\log\log f)$ time slots, we have proved the following result.

Lemma 5: With probability exceeding $1 - \frac{1}{4f}$, Phase 1 and Phase 2 combined take at most $\log\log n + o(\log\log n) + O(\log\log f)$ time slots.

Our next goal is to evaluate the number of active stations at the end of Phase 2. Let `Sieve2(`2^{s^2}`)`, `Sieve1(`2^{s^2-1}`)`, `Sieve1(`2^{s^2-2}`)`, ..., `Sieve1(`2^0`)` be a sequence of the calls from the last call of Phase 1 to the last call of Phase 2. We say that a call of `Sieve2(`2^m`)` or `Sieve1(`2^m`)` ($0 \le m \le s^2$) in this sequence is *failure* if

$$n' > 2^{2^m} \cdot \ln(4(s^2+1)f) \text{ and } |S| = 0,$$

where n' is the number of active stations when the call starts.

Let us show that the probability of the failure is not large. Let Y be the random variable denoting the number of stations in S in the call `Sieve2(`2^m`)` or `Sieve1(`2^m`)`. Suppose that $n' > 2^{2^m} \cdot \ln(4(s^2+1)f)$ holds. Then, $|S| = 0$ with probability at most

$$\begin{aligned}\Pr[Y = 0] &= \left(1 - \frac{1}{2^{2^m}}\right)^{n'} \\ &< e^{-\frac{n'}{2^{2^m}}} \\ &\le e^{-\ln(4(s^2+1)f)} = \frac{1}{4(s^2+1)f}.\end{aligned}$$

Thus, the probability that the call fails is at most $\frac{1}{4(s^2+1)f}$. Importantly, this probability is independent of m. Hence, from (1) we have

Lemma 6: The probability that none of the s^2+1 calls `Sieve2(`2^{s^2}`)`, `Sieve1(`2^{s^2-1}`)`, `Sieve1(`2^{s^2-2}`)`, ..., `Sieve1(`2^0`)` from the last call of Phase 1 to the last call of Phase 2 is at least $1 - \frac{1}{4f}$.

Suppose that all of these s^2+1 calls are success. Let us evaluate the number of active stations at the end of Phase 2. Let m be an integer such that

- the call `Sieve1(`2^m`)` or `Sieve2(`2^m`)` returns $|S| = 0$, and
- all the calls `Sieve1(`2^{m-1}`)`, `Sieve1(`2^{m-2}`)`, ..., `Sieve1(`2^0`)` return $|S| \ge 1$.

Clearly, such m always exists because the first call `Sieve2(`2^{s^2}`)` in the sequence (i.e the last call of Phase 1) returns $|S| = 0$. Let n' be the number of active stations when `Sieve1(`2^{m-1}`)` is called. Since we are assuming that all the s^2+1 calls are not failure, $n' \le 2^{2^m} \cdot \ln(4(s^2+1)f)$ holds. Also, any active station at this moment is still active after the last call `Sieve1(`2^0`)` with probability

$$\frac{1}{2^{2^{m-1}} \cdot 2^{2^{m-2}} \cdots 2^{2^0}} = \frac{1}{2^{2^m-1}}.$$

Let Y' be the random variable denoting the number of active stations when the last call `Sieve1(`2^0`)` is terminated. We have $E[Y'] = \frac{n'}{2^{2^m-1}} \le 2\ln(4(s^2+1)f)$. Hence, using the Markov inequality (5) the probability that $Y' > 8f\ln(4(s^2+1)f)$ is at most

$$\begin{aligned}\Pr[Y' > 8f\ln(4(s^2+1)f)] &\le \Pr[Y' > 4fE[Y']] \\ &< \frac{1}{4f}.\end{aligned}$$

Therefore, we have,

Lemma 7: At the end of Phase 2, the number of active stations is at most $8f\ln(4(s^2+1)f)$ with probability at least $1 - \frac{1}{4f}$ under the assumption that none of the s^2+1 calls from the last call of Phase 1 to the last call of Phase 2 is failure.

Finally, we are interested in getting a handle on the number of time slots involved in Phase 3. We assume that at the end of Phase 2 the number of active stations is $8f\ln(4(s^2+1)f)$. Recall that Phase 3 calls `Simulation2(`U', S'`)` in two time slots, and repeatedly calls `Sieve2(1)`. In each `Sieve2(1)`, each active station belongs to S with probability $\frac{1}{2}$. A particular call `Sieve2(1)` is *failure* if after the call more than $\frac{n'}{2}$ stations remain active, where n' (≥ 2) is the number of active stations just before the call. In other words, the call `Sieve2(1)` is failure if

- $|S| > \frac{n'}{2}$ or
- $|S| = 0$.

Clearly, the probability that $|S| > \frac{n'}{2}$ is at most $\frac{1}{2}$. Since $n' \ge 2$, the probability that $|S| = 0$ is at most $(\frac{1}{2})^{n'} \le \frac{1}{4}$. Thus, a particular call `Sieve2(1)` fails with probability at most $\frac{1}{2} + \frac{1}{4} = \frac{3}{4}$. For simplicity, we assume that `Sieve2(1)` fails with probability $\frac{3}{4}$ and succeeds with probability $\frac{1}{4}$. Since we have at most $8f\ln(4(s^2+1)f)$ active stations when Phase 3 starts, k successful calls of `Sieve2(1)` such that $(\frac{1}{2})^k \cdot 8f\ln(4(s^2+1)f) \le 1$ is sufficient to elect a leader. Hence, we set

$$k = \lceil \log(8f\ln(4(s^2+1)f)) \rceil.$$

Suppose that `Sieve2(1)` is called $8k + 32\ln(4f)$ times, and let Z be the random variable denoting the number of success calls. Since the probability of success is $\frac{1}{4}$, it should be clear that $E[Z] = 2k + 8\ln(4f)$. From the Chernoff Bounds (4), the probability that we have less than k success calls is at most

$$\begin{aligned}\Pr[Z < k] &= \Pr\left[Z < \left(1 - \frac{1}{2}\right)E[Z]\right] \\ &< e^{-\frac{1}{8}\cdot(2k+8\ln(4f))} < \frac{1}{4f}.\end{aligned}$$

Therefore, with probability at least $1 - \frac{1}{4f}$, among $8k + 32\ln(4f)$ calls of `Sieve2(1)` there are at least k successful ones. It follows that if at the end of Phase 2 we have at most $8f\ln(4(s^2+1)f)$ active stations, then Phase 3 calls `Sieve2(1)` at most $8k + 32\ln(4f) = O(\log f) + o(\log\log n)$ times. Since each `Sieve2(1)` takes two time slots, we have

Lemma 8: If at most $8f\ln(4(s^2+1)f)$ stations are active when Phase 3 starts, Phase 3 takes at most $O(\log f) + o(\log\log n)$ times with probability at least $1 - \frac{1}{4f}$,

From Lemmas 5, 6, 7, and 8, and (1) combined, we have

Theorem 9: Protocol `Leader-Election` terminates, with probability at least $1-\frac{1}{f}$, in at most $\log\log n + o(\log\log n) + O(\log f)$ time slots.

4.3 The Total Awake Time Slots of Our Protocol

Let us see the energy efficiency of our leader election protocol. Nakano and Olariu presented a leader election protocol running in $\log\log n + o(\log\log n) + O(\log f)$ time slots with probability $1-\frac{1}{f}$. This protocol is not energy efficient because all the stations must be awake for all the time slots and monitor the channel. On the other hand, in our leader election protocol, most of the stations go asleep and turn off the transceivers. Only the candidate of a leader remain awake. Let us evaluate the total awake time slots, which is the sum of the awake time slots over all stations. Clearly, the total awake time slots of the Nakano and Olariu's leader election protocol is $n\log\log n + o(n\log\log n) + O(n\log f)$ with probability $1-\frac{1}{f}$. Thus, from Corollary 2, the expected total awake time slot is $n\log\log n + O(n)$. We are going to show that the expected total awake time slot is $O(n)$ in our leader election protocol.

We first show that Phase 1 of our protocol runs in $O(n)$ expected awake time slots. Recall that, in Phase 1 `Sieve2(`2^{0^2}`)`, `Sieve2(`2^{1^2}`)`, ... are performed until for the first time, S is empty. Thus, a particular station is in S in the call `Sieve2(`2^{i^2}`)` ($i \geq 1$) with probability

$$\frac{1}{2^{0^2}2^{1^2}\cdots 2^{i^2}}.$$

If this is the case, this station is still active and awake for two time slots in the next call `Sieve2(`$2^{(i+1)^2}$`)`. Thus, the expected number of awake time slots of a particular station is,

$$2 + \frac{2}{2^{0^2}} + \frac{2}{2^{0^2}2^{1^2}} + \cdots = O(1).$$

It follows that the expected total awake time slots in Phase 1 is $n \times O(1) = O(n)$.

Next, we will show that the expected total awake time slots for Phases 2 and 3 is $o(n)$. It is clear that $(j+1)^2 \leq 2^{j^2}$ holds for all $j \geq 1$. Thus, for all $j \geq 1$, we have

$$2^{2^{(j+1)^2}} \leq 2^{2^{2^{j^2}}}. \tag{8}$$

For a fixed n, let j be the integer such that

$$2^{2^{0^2}}2^{2^{1^2}}\cdots 2^{2^{j^2}} \leq \frac{\log n}{2} < 2^{2^{0^2}}2^{2^{1^2}}\cdots 2^{2^{(j+1)^2}}.$$

Clearly, $2^{2^{j^2}} < \frac{\log n}{2}$ holds. From (8), we have $2^{2^{(j+1)^2}} < 2^{\frac{\log n}{2}} = \sqrt{n}$. Thus, for enough large n, we have the double inequality

$$\frac{\log n}{2} < 2^{2^{0^2}}2^{2^{1^2}}\cdots 2^{2^{(j+1)^2}} < \frac{\sqrt{n}\log n}{2}.$$

Hence, for such j, at the end of the call `Sieve2(`$2^{(j+1)^2}$`)` the expected number of active stations is at least $\frac{2n}{\log n}$ and at most $\frac{2\sqrt{n}}{\log n}$. Let us evaluate the probability that, for all the $j+2$ calls `Sieve2(`2^{0^2}`)`, `Sieve2(`2^{1^2}`)`, ..., `Sieve2(`$2^{(j+1)^2}$`)`, S is not empty. The probability that a particular station is in S for all these $j+2$ calls is at least $\frac{1}{2^{0^2}2^{1^2}\cdots 2^{(j+1)^2}} > \frac{2}{\sqrt{n}\log n}$. Thus, from (3), the probability that none of the n stations is not this case is at most

$$\left(1 - \frac{2}{\sqrt{n}\log n}\right)^n < e^{-\frac{2\sqrt{n}}{\log n}}.$$

Hence, with probability at least $1 - e^{-\frac{2\sqrt{n}}{\log n}}$, S is not empty in the call `Sieve2(`$2^{(j+1)^2}$`)` in Phase 1.

Let us consider the following two cases:

Case 1: S is not empty in the call `Sieve2(`$2^{(j+1)^2}$`)` in Phase 1, and
Case 2: Otherwise,

and evaluate the total awake time slots separately.
Case 1: At the end of `Sieve2(`$2^{(j+1)^2}$`)`, the expected number of active station is at most

$$\frac{n}{2^{0^2}2^{1^2}\cdots 2^{2^{(i+1)^2}}} < \frac{2n}{\log n}.$$

Since Phases 2 and 3 runs in $\log\log n + o(\log\log n)$ time slots, The total awake time slots is $(\log\log n + o(\log\log n)) \cdot \frac{2n}{\log n} = o(n)$.
Case 2: Clearly, Case 2 happens with probability at most $e^{-\frac{2\sqrt{n}}{\log n}}$. Let us consider the worst case, that is, at the end of Phase 1, all the n station remain active. Since `Sieve2(1)` is repeatedly performed in Phase 3, it is not difficult to show that Phases 2 and 3 combined takes no more than $O(\log n)$ time slots. Hence, the expected total awake time slots is no more than $O(n\log n)$. Since Case 2 happens with probability $e^{-\frac{2\sqrt{n}}{\log n}}$, Case 2 adds at most $O(n\log n)\cdot e^{-\frac{2\sqrt{n}}{\log n}} < o(n)$ awake time slots to the total awake time slots.

Consequently, we have proved that,

Theorem 10: The expected total awake time slots of Protocol `Leader-Election` is $O(n)$.

5. Conclusions

In this paper, we have presented a leader election protocol in radio network with each station being equipped with a single transceiver. Our protocol elects a leader in $\log\log n + o(\log\log n) + O(\log f)$ time slots with probability at least $1-\frac{1}{f}$ for every $f \geq 1$. The total awake time slots of our leader election protocol is expected $O(n)$.

This protocol improves Nakano and Olariu's leader election protocol in several points: Their leader election protocol requires two transceivers per station. Also, it spends expected $O(n\log\log n)$ total awake time slots.

Acknowledgments

The author would like to thank anonymous referees for their valuable comments. This work is supported in part by JSPS Grant-in-Aid for Scientific Research.

References

[1] H. Abu-Amara, "Fault-tolerant distributed algorithms for election in complete networks," IEEE Trans. Comput., vol.37, no.4, pp.449–453, 1988.
[2] Y. Afek and E. Gafni, "Time and message bounds for election in synchronous and asynchronous complete networks," SIAM J. Comput., vol.20, pp.376–394, 1991.
[3] 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," Distrib. Comput., vol.5, pp.67–71, 1991.
[4] J. Bentley and A. Yao, "An almost optimal algorithm for unbounded search," Inf. Process. Lett., vol.5, pp.82–87, 1976.
[5] D. Bertzekas and R. Gallager, Data Networks, Second Edition, Prentice-Hall, 1992.
[6] E.D. Kaplan, Understanding GPS: Principles and applications, Artech House, Boston, 1996.
[7] E. Korach, S. Moran, and S. Zaks, "Optimal lower bounds for some distributed algorithms for a complete network of processors," Theor. Comput. Sci., vol.64, pp.125–132, 1989.
[8] M. Kutylowski and W. Rutkowski, "Adversary immune leader election in ad hoc radio networks," Proc. European Symposium on Algorithms, pp.397–408, 2003.
[9] M.C. Loui, T.A. Matsushita, and D.B. West, "Election in complete networks with a sense of direction," Inf. Process. Lett., vol.22, pp.185–187, 1986.
[10] R.M. Metcalfe and D.R. Boggs, "Ethernet: Distributed packet switching for local computer networks," Commun. ACM, vol.19, pp.395–404, 1976.
[11] R. Motwani and P. Raghavan, Randomized Algorithms, Cambridge University Press, 1995.
[12] K. Nakano and S. Olariu, "Randomized $O(\log \log n)$-round leader election protocols in radio networks," Proc. International Symposium on Algorithms and Computation, LNCS 1533, pp.209–218, 1998.
[13] K. Nakano and S. Olariu, "Randomized leader election protocols in radio networks with no collision detection," Proc. International Symposium on Algorithms and Computation, pp,362–373, 2000.
[14] K. Nakano and S. Olariu, "Uniform leader election protocols for radio networks," IEEE Trans. Parallel Distrib. Syst., vol.13, no.5, pp.516–526, 2002.
[15] M. Joa-Ng and I.-T. Lu, "A peer-to-peer zone-based two-level link state routing for mobile ad-hoc networks," IEEE J. Sel. Areas Commun., vol.17, pp.1415–1425, 1999.
[16] B. Parkinson and S. Gilbert, "NAVSTAR: Global positioning system—Ten years later," Proc. IEEE, 1983, pp.1177–1186, 1983.
[17] G. Singh, "Leader election in complete networks," Proc. ACM Symposium on Principles of Distributed Computing, 1992, pp.179–190, 1992.
[18] D.E. Willard, "Log-logarithmic selection resolution protocols in a multiple access channel," SIAM J. Comput., vol.15, pp.468–477, 1986.

Jacir Luiz Bordim received B.E. degree from Passo Fundo University (Brazil) in 1994, M.E. degree from Nagoya Institute of Technology (Japan) in 2000 and PhD degree from Japan Advanced Institute of Science and Technology in 2003. From 2003 to 2005 he worked as a researcher at ATR-ACR Labs (Japan). Since March 2005, he has joined the Department of Computer Science at University of Brasilia (Brasil), where is an Associate Professor. His interests include mobile computing, sensor networks, computer networks, distributed systems, and graph theory.

Yasuaki Ito received B.E. degree from Nagoya Institute of Technology (Japan) and M.S. degree from Japan Advanced Institute of Science and Technology in 2003. He is currently a Research Associate with Department of Artificial Complex Systems Engineering at Hiroshima University, (Japan). His research interests include reconfigurable architectures, computational complexity and image processing.

Koji Nakano received the B.E., M.E. and Ph.D. degrees from Department of Computer Science, Osaka University, Japan in 1987, 1989, and 1992 respectively. In 1992–1995, he was a Research Scientist at Advanced Research Laboratory. Hitachi Ltd. In 1995, he joined Department of Electrical and Computer Engineering, Nagoya Institute of Technology. In 2001, he moved to School of Information Science, Japan Advanced Institute of Science and Technology, where he was an associate professor. He is currently a full professor at School of Engineering, Hiroshima University. He has published extensively in journals, conference proceedings, and book chapters. He has served on the editorial board of journals including IEEE Transactions on Parallel and Distributed Systems, IEICE Transactions on Information and Systems, and International Journal of Foundations on Computer Science. He has also guest-edited several special issues including IEEE TPDS Special issue on Wireless Networks and Mobile Computing, IJFCS special issue on Graph Algorithms and Applications, and IEICE Transactions special issue on Foundations of Computer Science. He has organized conferences and workshops including International Conference on Parallel and Distributed Computing, Applications and Technologies, IPDPS Workshop on Advances in Parallel and Distributed Computational Models, and ICPP Workshop on Wireless Networks and Mobile Computing. His research interests includes mobile computing, power-aware communication protocols, hardware algorithms, FPGA-based reconfigurable computing, parallel computing, algorithms and architectures.